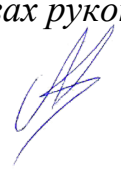
fМОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М.В. ЛОМОНОСОВА
МЕХАНИКО-МАТЕМАТИЧЕСКИЙ ФАКУЛЬТЕТ

На правах рукописи



Леонов Александр Георгиевич

Интеграционная методология поэтапного формирования алгоритмического мышления при обучении информатике и программированию

5.8.2 Теория и методика обучения и воспитания
(информатика, информатика и вычислительная техника)

ДИССЕРТАЦИЯ

на соискание ученой степени

доктора педагогических наук

Научный консультант:
член-корреспондент РАО,
доктор технических наук,
профессор Григорьев С.Г.

Москва – 2024

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность темы исследования.

Обеспечение технологического, научного, производственного и кадрового суверенитета России является важнейшей задачей, решение которой невозможно без цифровой трансформации системы образования, способной обеспечить подготовку высококвалифицированных специалистов. Реалии современности требуют ускоренной подготовки большого числа кадров, обладающих широкими компетенциями в области информатизации всех сфер деятельности. Каждый человек должен не только освоить компетенции, необходимые для утилитарного использования компьютеров в повседневной жизни, но и обладать необходимым в информационном обществе алгоритмическим *стилем мышления (особенностями организации мыслительного процесса)*, чтобы иметь возможность реализовать себя в выбранной профессии в современном мире, наполненным множеством новых специальностей, связанных с информатизацией, цифровыми технологиями и программированием.

Решение данной задачи требует интенсификации процессов подготовки будущих специалистов с упором на освоение компетенций в области информационных технологий. Одновременно встает вопрос о снижении возраста начала знакомства с основами алгоритмизации и программирования с целью формирования алгоритмического мышления у каждого члена общества, создании инновационных образовательных курсов и специализированных педагогических программных систем, ориентированных на массовое систематическое обучение не только студентов и школьников, но и дошкольников.

Существующая система современного образования не решает поставленные задачи, не учитывает должным образом интересы всех социальных групп учащихся. В результате имеют место различные уровни подготовки учащихся, особенно поступающих в высшие учебные заведения, что снижает эффективность образовательного процесса в вузах. Многие школьники и студенты не сумели

освоить основы алгоритмизации и программирования, доступные для изучения в начальной и основной школе. Так, педагогический опыт показал, что студенты начальных курсов университетов, вчерашние школьники, имеют дефекты алгоритмического стиля мышления, не позволяющие им решать даже простейшие задачи на составление алгоритмов.

Вместе с тем необходимо отметить, что пропедевтические курсы по основам программирования для студентов, школьников и дошкольников должны иметь сходную понятийную базу и набор практических заданий.

Освоение в раннем возрасте основ алгоритмизации и программирования позволит начиная с дошкольного и младшего школьного возраста сформировать определенный стиль мышления, называемый далее *алгоритмическим*, что будет способствовать в соответствующем возрасте повышению общего интеллектуального потенциала, комфортному развитию индивидуума в условиях информационного общества и будущей успешной профессиональной деятельности.

Для решения проблемы унификации процессов формирования алгоритмического мышления у обучаемых различного возраста, изначально обладающих разными уровнями подготовки в области алгоритмизации и ИКТ-компетентности, нужно сформировать системный подход для решения поставленной задачи, который здесь назван *интеграционной методологией*. Это означает, что нужно определить наполнение компонентов *методической системы обучения* (*цели обучения, содержание обучения, методы обучения, формы и средства обучения*), целью которой является формирование алгоритмического мышления при обучении информатике и программированию с возможным понижением границ сензитивного периода.

Степень разработанности темы исследования.

Исследования и разработки в области освоения основ программирования были начаты в СССР еще в начале 1960-х годов. Усилиями С.И. Шварцбурда в

Москве и А.П. Ершова в Новосибирске были созданы первые учебные курсы, на которых старшеклассники обучались программированию, теории информации и знакомились с устройством компьютеров.

Доклад А.П. Ершова «Программирование – вторая грамотность» в 1981 году предопределило цифровую трансформацию системы образования. Школьный предмет «Основы информатики и вычислительной техники» введен в СССР в 1985 году, как инновационный курс информатики в 9-10-х классах школы «безмашинного» изучения основных алгоритмов на текстовом учебном *школьном алгоритмическом языке*, так как отсутствовали какие-либо компьютерные средства для написания учениками программ на этом языке программирования. Еще до всесоюзного введения курса информатики в новосибирском университете и сибирском отделении АН СССР Г.А. Звенигородским был разработан другой, более сложный для новичков текстовый язык программирования Рапира, имеющий «машинную» реализацию в составе учебной системы программирования Школьница. В 1985 году в МГУ имени М.В. Ломоносова под руководством А.Г. Кушниренко и В.Б. Бетелина была разработана «машинная» реализация школьного алгоритмический языка – учебная среда программирования «Е-практикум», затем переименованная в КуМир – «Комплект Учебных МИРов». Цифровая образовательная среда (ЦОС) КуМир стала педагогическим программным средством, использование которой в учебном процессе позволило существенно понизить возраст первоначального знакомства с основами алгоритмизации и программирования, перенеся предмет Информатика из старшей школы в основную. Школьный алгоритмический язык, также называемый *языком записи алгоритмов,* получил широкое распространение в СССР, что послужило следствием его использования в школах России и в настоящее время.

Первых успехов в разработке и во внедрении отечественного курса информатики добились: Е.П. Велихов, В.Б. Бетелин, С.А. Бешенков, Н.Я. Виленкин, А.Г. Гейн, Ю.М. Горвиц, С.Г. Григорьев, А.А. Дуванов, В.Г. Житомирский, Я.Н. Зайдельман, Г.А. Звенигородский, А.П. Ершов,

В.А. Каймин, А.А. Кузнецов, А.Г. Кушниренко, Г.В. Лебедев, В.С. Леднев, В.М. Монахов, Ю.А. Первин, А.Л. Семенов, С.И. Шварцбурд и ряд других исследователей.

Дальнейшее развитие учебного курса информатики было сконцентрировано на изучении основ применения информационных технологий. На протяжении двух десятилетий было разработано, апробировано и внедрено несколько учебников по предмету Информатика и ИКТ. Над курсом работали Л.Л. Босова, А.Г. Гейн, Ю.А. Первин, С.Г. Григорьев, В.В. Гриншкун, А.Г. Кушниренко, Г.В. Лебедев, И.В. Левченко, К.Ю. Поляков, А.Л. Семенов, Т.А. Рудченко, Я.Н. Зайдельман, Е.К. Хеннер и ряд других исследователей. В учебниках вопросы изучения принципов и методов обработки информации даны в соответствии с принятыми стандартами содержания и сконцентрированы в основной и старшей школе.

Необходимо отметить работы по созданию систематических непрерывных курсов по информатике в школе на основе идей Жана Пиаже и Н.Я. Виленкина. По инициативе Ю.А. Первина был создан сквозной непрерывный курс «Информационная культура» 1–11 класс, реализованный в учебных заведениях в г. Самара, основанный на *принципах программного управления исполнителями,* сопровождаемый поддерживающими педагогическими программными продуктами. *Принцип программного управления* утверждает, что любую работу, которую человек может выполнить, командуя неким объектом (*исполнителем*), можно передать компьютеру, составив программу выполнения того действия (или последовательности действий), которую исполнителю надлежит выполнить. В учебниках «Информатика 1-4» для начальной школы, созданных Т.А. Рудченко и А.Л. Семеновым, также рассматриваются вопросы программного управления исполнителями.

Образовательная реформа, состоящая в введении в 1985 году в школьную программу предмета Информатика, была необходима для формирования у подрастающего поколения *операционного стиля мышления*. А.П. Ершов использовал программиста в качестве прототипа носителя операционного стиля

мышления, как обладающего необходимыми знаниями, умениями и навыками. Следуя А.П. Ершову, под *операционным стилем мышления* понимают следующие умения и навыки:

- Планирование, – умение планировать структуру собственных действий;
- Моделирование, – умение строить информационные модели;
- Поиск, – умение организовать поиск информации;
- Навыки структурированного взаимодействия;
- Навыки своевременного использования компьютера.

В операционном стиле мышления подчеркивается *алгоритмическая направленность* этого способа мышления, объединенная с *ИКТ-компетенциями*, то есть способность использовать информационные и коммуникационные технологии для получения, обработки, поиска, оценки, производства и распространения информации для успешной работы и жизнедеятельности в информационном обществе.

Опираясь на работы С.Л. Рубинштейна и О.К. Тихомирова, мышление можно рассматривать как ментальную целенаправленную деятельность для решения некоторой конкретной задачи. С этой позиции *алгоритмическое мышление* суть мыслительная деятельность, основанная на использовании алгоритмов – последовательностей шагов для достижения определённой цели. Здесь прослеживается аналогия между процессами, необходимыми для решения *основной задачи программирования*, – конструированием одного формального объекта (исполнителя) на базе другого и умственной деятельностью, относящейся к алгоритмическому мышлению. По этой аналогии можно сказать, что *алгоритмическое мышление* включает в себя следующие компоненты*: моделирование, абстрагирование, декомпозицию, разработку алгоритма, оптимизацию, тестирование, отладку* и направлено на решение целевой задачи. Алгоритмическое мышление необходимо любому члену общества и востребовано не только в программировании, так как помогает систематизировать подход к

решению задачи, но и также способствует развитию критического мышления, навыков и умений анализа и планирования.

Если трактовка термина *программа* зависит от области применимости и контекста, например, как изложение основных принципов, понятий и целей, то в информационной культуре под термином *программа* понимают план будущих действий.

Следуя А.Г. Кушниренко и Г.В. Лебедеву, для формирования *алгоритмического мышления,* как самостоятельной культурной ценности, необходимо определить замкнутый набор понятий, что не отрицает познавательное значение школьного курса Информатики. При этом по В.Б. Бетелину принцип программного управления может быть понят и осознан учеником только после усвоения достаточно сложной системы научных понятий. Согласно Л.С. Выготскому, осознание любого общего принципа требует комплексного освоения ребенком некоторой системы научных понятий. Принцип программного управления позволяет сформировать такой понятийный аппарат в терминах и образах, понятных ученику вне зависимости от возрастной категории. Последнее позволяет указать возраст дошкольника как сензитивный период, когда ребенок может осваивать основы алгоритмизации и программирования для начала формирования алгоритмического мышления.

По Ж. Пиаже, запоминание ребенком системы научных понятий недостаточно для формирования (алгоритмического) мышления. Необходим логико-математический опыт, направленный на действия и операции, совершаемые ребенком с реальными предметами.

Кроме того, система научных понятий не может быть усвоена в короткий временной промежуток. Согласно теории поэтапного формирования умственных действий и понятий П.Я. Гальперина, в качестве основы для осуществления действий с самого начала используются не только теоретические концепции, но и практические операции. Знания формируются не через предварительное запоминание, а в процессе применения знаний к решению задач. Это говорит о

возможности сформировать набор задач, соответствующих возрастным особенностям учеников, успешное решение которых позволит поэтапно, в широком смысле слова, включая длительные временные периоды, усвоить научные понятия и сформировать алгоритмическое мышление.

Поскольку алгоритмы прежде всего знакомы из математики, то перенесение опыта решения математических задач в программирование приводит к следующей схеме образовательного процесса по Л.Н. Ланда, от обучения алгоритмам, поиска алгоритмов, от общих методов, правил, наборов действий и выбора применимых операций для решения до постановки перед учеником проблемной ситуации, предполагая, что обучаемым будет самостоятельно придумываться алгоритм, правда последнее может находиться в области актуально недоступного.

Вместе с тем, педагогические программные средства позволяют более полно использовать средства вычислительной техники для обучения. Например, возможно применение задач и тестовых материалов для автоматизированной проверки уровня сформированной компетенции при использовании цифровых образовательных сред программирования, включающих в себя все средства поддержки изучаемых языков программирования. Интеграционная методология – системный подход для решения задачи формирования алгоритмического мышления у обучаемых, требует создания *методической системы обучения* с вариативным содержанием обучения, ориентированным на различный возрастной контингент учащихся, которая включает практические методы обучения с большим объемом самостоятельной работы и использует цифровые и предметно-цифровые ИКТ-насыщенные средства обучения и различные формы обучения, ориентированные на возраст и начальный уровень компетенции учеников. Такая методическая система обучения позволила бы учащемуся и педагогу в рамках цифровой трансформации образовательного процесса получать доступ к необходимым методическим средствам, содержанию обучения, различным средствам, поддерживающим вариативность форм обучения, и цифровым

средствам обучения и ресурсам на любом уровне обучения и для любого возраста учащихся.

Вышесказанное определяет актуальность заявленной темы исследования **«Интеграционная методология поэтапного формирования алгоритмического мышления при обучении информатике и программированию»**.

Практические проблемы исследования состоят в

- необходимости определения сензитивных границ возраста формирования алгоритмического мышления;
- необходимости определить этапность формирования алгоритмического мышления по возрастным группам;
- определить содержание и средства методической системы обучения, целью которой является формирование алгоритмического мышления, что осложняется различным уровнем компетенции в области алгоритмизации и программирования у обучающихся разных возрастных категорий различных учебных ступеней и уровней.

Теоретические проблемы исследования состоят в

- необходимости концептуализации алгоритмического мышления;
- необходимости определения универсального набора задач и понятий для формирования алгоритмического мышления у обучающихся на любом уровне обучения и в любом возрасте;
- необходимости спроектировать методическую систему обучения для форсированного обучения информатике и программированию, реализующую процесс поэтапного формирования алгоритмического мышления и с определенными нижними границами сензитивного периода первичного знакомства с основами алгоритмизации и программирования.

Объект исследования. Процесс формирования алгоритмического мышления для учащихся разных возрастных категорий при обучении информатике и программированию.

Предмет исследования. Методическая система поэтапного формирования алгоритмического мышления при обучении информатике и программированию.

Цель исследования. Разработка и экспериментальная проверка в педагогической практике методической системы обучения информатике и программированию, направленной на формирование алгоритмического мышления у учащихся разных возрастных групп.

Достижение цели исследования состоит в решении поставленных проблем:

- разработка системного подхода формирования алгоритмического мышления для учащихся широкого возрастного охвата при обучении информатике и программированию;
- разработка методической системы обучения поэтапного формирования алгоритмического мышления при обучении информатики и программирования с высоким уровнем автоматизации образовательного процесса;
- определение границ сензитивного периода первичного знакомства с основами программирования;
- разработка цифровых средств обучения информатике и программированию, ориентированных на разные возрастные группы и соответствующих этапам формирования алгоритмического мышления.

Гипотеза исследования.

1. Как содержание методической системы обучения, существует такой набор заданий и понятий, успешное поэтапное освоение которых приводит к формированию алгоритмического мышления у обучаемых при условии использования в качестве средств обучения ИКТ-насыщенные цифровые образовательные среды и платформы.
2. Такой набор заданий и понятий является универсальным для успешного формирования основ алгоритмического мышления на всех ступенях обучения (у студентов вузов, включая будущих учителей информатики педуниверситетов,

школьников и у дошкольников). Сформированное алгоритмическое мышление едино для учащихся всех возрастов.

3. Возможно понижение границ сензитивного периода первичного знакомства с основами программирования при использовании методической системы обучения с предметно-цифровыми образовательными средами и платформами для обучения информатике и программированию в качестве средств и специальных форм обучения.

Решение поставленных проблем осуществляется в ходе выполнения следующих **задач исследования:**

1. Определить состав алгоритмического мышления и оценить форсированность обучения;

2. Проанализировать закономерности в обучении информатике и программированию в образовательных организациях разных уровней (школа, вуз, дошкольные учреждения), выявить ключевые подходы и методы, способствующие эффективному формированию алгоритмического мышления и навыков программирования у учащихся;

3. Определить набор заданий и понятий, необходимых для поэтапного формирования алгоритмического мышления у учащихся широкой возрастной группы от дошкольников до школьников и студентов;

4. Выявить особенности содержания и форм обучения и разработать методику обучения для понижения границ сензитивного возраста первичного знакомства учеников с элементами алгоритмизации и программирования в методической системе обучения;

5. Разработать предметно-цифровые образовательные среды и платформы в качестве средств и специальных форм обучения информатике и программированию для всех уровней образования – от детских садов до вузов;

6. Осуществить опытно-экспериментальную работу, показывающую возможность понижения границ сензитивного возраста формирования основ алгоритмического мышления при использовании методической системы обучения.

Методология и методы исследования. Проектирование методической системы обучения, включающей содержание, методы, формы и, в качестве средств обучения, цифровые образовательные среды и платформы высокого уровня интеграции для обучения информатике и программированию, которые, как практические приложения, составили основу исследования.

Методологическую основу исследования составили:

теория педагогики (Ж.-Ж. Руссо, К.Д. Ушинский, Я.А. Коменский, И.Г. Песталоцци и др.)

теория деятельности и применения деятельностного подхода в образовании (Л.С. Выготский, П.Я. Гальперин, А.В. Запорожец, Д.Б. Эльконин и др.);

формирование общеучебных умений (Л.Н. Ланда, И.В. Роберт, Р.С. Немов, А.Г. Асмолов, А.М. Пышкало, Н.Н. Поддьяков, И.В. Роберт, А.Н. Поддьяков и др.);

информатизация образования (С.И. Шварцбурд, В.Б. Бетелин, Е.П. Велихов, А.Л. Семенов, С.Г. Григорьев, А.П. Ершов, С.Д. Каракозов, В.В. Гриншкун, А.А. Кузнецов, Т.А. Бороненко, Т.Н. Носкова, А.Ю. Уваров и др.);

теория и методика обучения информатике и ИКТ (В.В. Гриншкун, А.А. Кузнецов, С.Г. Григорьев, С.А. Бешенков, Л.Л. Босова, А.Г. Гейн, А.П. Ершов, В.С. Леднев, А.Г. Кушниренко, Г.В. Лебедев, Ю.А. Первин, И.Г. Семакин, А.Л. Семенов, С.К. Ландо, А.И. Сенокосов, Т.Н. Носкова, Т.А. Бороненко, И.В. Левченко и др.).

На философском уровне методологическую основу исследования составили: системный подход к выявлению основных составляющих алгоритмического мышления, создание методики понижения возраста первичного знакомства с элементами алгоритмизации и программирования, создание методики и

дидактических приемов для обучения информатике и программированию, которые могут быть использованы для различного возраста и уровней компетенций обучаемых.

На общенаучном уровне были определены сущностные признаки объекта исследования, позволившие определить границы сензитивного периода для начала формирования алгоритмического мышления.

На конкретно-научном уровне исследования были задействованы методы цифровой дидактики.

На технологическом уровне исследования – методы, методики и технологии применения цифровых образовательных платформ и сред, как основы цифровой трансформации образовательного процесса в Российской Федерации.

В ходе выполнения исследовательских процедур были также задействованы методы наблюдения, изучения и обобщения педагогического опыта воспитателей дошкольных образовательных организаций, учителей и методистов школьных и дополнительных занятий, педагогические инновации в педагогической литературе, ресурсы интернета, материалы конференций и семинаров по теме исследования, а также опыт экспериментальной работы с учащимися разного возраста.

Научная новизна результатов.

1. Выявлены основные составляющие алгоритмического мышления и доказана возможность его формирования при использовании практики освоения набора заданий и понятий.

2. Предложена и проверена методика понижения границ сензитивного возраста первичного знакомства с основами алгоритмизации и программирования, способствующая освоению основных научных понятий процедурного программирования дошкольниками для поэтапного формирования алгоритмического мышления.

3. Спроектирована и экспериментально проверена в педагогической практике методическая система обучения с вариативным содержанием, ориентированным на различный возрастной контингент учащихся, включающая практические методы с большим объемом самостоятельной работы, использующая цифровые и предметно-цифровые ИКТ-насыщенные средства обучения и различные формы, ориентированные на возраст и начальный уровень компетенции учеников, существенно повышающая эффективность систематического освоения информатики и программирования.

4. Разработаны и экспериментально проверены в педагогической практике автоматизированные цифровые и предметно-цифровые образовательные среды, как средства обучения методической системы обучения поэтапного формирования алгоритмического мышления студентов вузов, школьников и дошкольников, начиная с 4-ого года жизни.

5. Проведен массовый анализ результатов раннего знакомства с основами алгоритмизации и программирования в дошкольных образовательных организациях, показывающий существенный прогресс в области алгоритмических навыков и умений. Дети в своей массе (78%) показали высокий познавательный интерес, улучшились навыки пространственной ориентации (60% имеют высокий уровень), большинство детей старших групп (90%) умеют решать предложенный набор задач в ЦОС ПиктоМир. Дошкольники показали метапредметные результаты: умение планировать последовательность шагов алгоритма для достижения цели, осуществлять итоговый и пошаговый контроль по результату, вносить коррективы в действия, ориентироваться в разнообразии способов решения задач, строить логические рассуждения и др.

6. При помощи искусственных нейронных сетей проанализированы результаты автоматизированного обследования различных категорий обучающихся: дошкольников, школьников, студентов вузов, включая педагогические университеты, которые использованы для оценки результатов и корректировки педагогической практики спроектированной методической

системы обучения. Экспериментальные группы студентов педагогических университетов при увеличении общего объема заданий одинакового уровня сложности на 20-25% показали двукратный рост успеваемости и достигли теоретического максимума усвоения материала.

Теоретическая значимость. Полученные результаты послужили основанием для понимания состава алгоритмического мышления, возможностью определить компоновку универсального набора задач и понятий, необходимых для поэтапного формирования такого мышления при использовании в качестве средств обучения предметно-цифровых и цифровых образовательных сред, начиная с раннего возраста дошкольников.

Универсальный набор задач и понятий, использованный в педагогической практике в содержании образования методической системы обучения, имеет общее ядро, применимое в обучении от детского сада до общеобразовательных школ и вузов.

Спроектирован исследовательский инструментарий, позволяющий автоматизировано оценивать уровень достижений учащихся в процессе обучения информатике и программированию, индивидуализировать образовательный процесс обучаемых, используя элементы искусственного интеллекта давать опережающие оценки будущих уровней компетенций учащихся. Исследовательский инструментарий, как элемент средств обучения методической системы обучения, может быть использован в цифровой трансформации образовательного процесса не только в области преподавания информатики и программирования, но и для естественно-научных и гуманитарных дисциплин, а также в межпредметных курсах.

Практическая значимость.

Сформулированы основные научные понятия, освоение которых необходимо для формирования алгоритмического мышления.

Сформированы наборы практических заданий, выполнение которых способствует эффективному формированию алгоритмического мышления. Успешное поэтапное освоение такого набора заданий возможно с раннего дошкольного возраста.

Создана методическая система обучения информатике и программированию, которая может быть использована для обучения учеников различного возраста и уровней компетенций обучаемых.

Разработаны и экспериментально проверены в педагогической практике средства обучения – цифровые образовательные платформы и среды, как компоненты методической системы обучения, позволяющие с высоким уровнем автоматизации обучать информатике и программированию, эффективно и успешно формировать алгоритмическое мышление у обучаемых.

Этапы и экспериментальная база исследования.

На этапе сбора материалов и постановки проблемы исследования (1996 – 2006 гг.) изучалась степень проработанности проблемы в отечественных и зарубежных материалах исследований, изучался отечественный и зарубежный опыт информатизации образования, анализировались основные направления информатизации учебно-воспитательного процесса в школе и дошкольных учреждениях, в том числе подходы к раннему обучению дошкольников и младших школьников в рамках непрерывного курса по информатике, изучалось современное состояние педагогических программных продуктов, их применимость для оценки компетенций и формирования портфеля достижений учащихся в области информатики и ИКТ. Изучалась возможность формирования мотивационных личностных ресурсов и алгоритмического мышления в процессе пропедевтического обучения школьников информатике и ИКТ. Проводился анализ педагогической, психологической и культурологической литературы. Был проанализирован опыт подготовки учителей основного и среднего образования, работающих в классах с уклоном в области информатики в рамках непрерывного

школьного курса. Изучались основные направления развития методической системы обучения информатике, программированию и информационным технологиям дошкольников, школьников и студентов вузов. Разрабатывалась и реализовывалась цифровая образовательная среда КуМир, как средство обучения основам программирования школьников, изучались возрастные границы применения педагогического программного средства.

Этап формирования задачи и подбора методик (2007–2016 гг.) характеризуется созданием методических материалов, когда уточнялись основные теоретические положения формирования алгоритмического мышления в рамках раннего обучения дошкольников и школьников информатике и программированию, разрабатывалась программа пропедевтического курса по алгоритмике для дошкольных учреждений и для 1-4 классов начальной школы, содержание учебных материалов и методических пособий, разрабатывались и реализовывались средства обучения, как компоненты методической системы обучения, для дошкольников и младших школьников: цифровая образовательная пиктографическая среда ПиктоМир, модифицировалось педагогическое средство для освоения основ программирования школьниками – цифровая образовательная среда КуМир, цифровые образовательные ресурсы для оценки компетенций и накопления информации по достижениям учащихся, писались учебники и методические пособия, разрабатывались методические и дидактические подходы к преподаванию пропедевтического курса информатики и программирования в дошкольных образовательных организациях, организациях общего образования. Результаты отражены в методических пособиях и ряде методических статей. Подготовлены программы повышения квалификации по методике преподавания информатики и программирования с использованием средств обучения ЦОС КуМир и ЦОС ПиктоМир в дошкольных организациях и организациях общего образования для учителей информатики, математики, физики и других предметов.

Этап экспериментальной работы состоял в перенесении результатов исследования в практику основного и дополнительного образования 2017 – 2022

гг. и сопровождался практической исследовательской деятельностью и публикациями результатов практических работ по вопросам формирования алгоритмического мышления, формирования набора заданий, формулирования основных понятий программирования для раннего знакомства с основами алгоритмизации и программирования. На этом этапе были спроектированы и реализованы средства обучения, как компоненты методической системы обучения: блочная цифровая образовательная среда ПиктоМир-К, как элемент ступенчатой поэтапной методологии обучения для формирования алгоритмического мышления у студентов педагогических вузов и школьников, цифровая образовательная платформа Мирера, как средство обучения с применением вариационных форм для обучения информатике, программированию и другим естественно-научным и гуманитарным предметам студентов вузов и старших школьников. Цифровая образовательная платформа Мирера, как компонент методической системы обучения, позволяет использовать методики и формы обучения, включающие индивидуализацию образовательного процесса, используя автоматизированный контроль решения задач различного типа и предметов, формирование цифрового следа обучаемого с последующим анализом образовательного процесса при помощи элементов искусственного интеллекта. Разработанные цифровые образовательные платформы и среды явились частями единой методической системы обучения.

Этап оформления и подготовки к защите результатов проведенного исследования 2023 – 2024 гг. освещен в теоретических публикациях, проводились теоретическое осмысление, систематизация и обобщение полученных результатов, оформление работы.

Результаты экспериментальной работы. Эксперименты проходили в более чем 750 различных дошкольных образовательных организациях и общеобразовательных организациях в рамках программы работы сетевых инновационных площадок ФГУ ФНЦ НИИСИ РАН по теме «Апробация и

внедрение основ алгоритмизации и программирования для дошкольников и младших школьников в цифровой образовательной среде ПиктоМир».

Методическая система обучения в педагогической практике (более 12 тысяч дошкольников ежегодно осваивают основы информатики и программирования) показала, что наполнение содержания обучения заданиями и использование методики обучения, основанной на построении схемы связки алгоритмических учебных языков с производственными языками программирования, даёт положительные результаты. В рамках этой методики учащиеся начинают с изучения пиктографического языка в цифровой образовательной среде «ПиктоМир», затем, соответственно возрасту, переходят к текстовому школьному алгоритмическому языку в «КуМире», и в итоге осваивают производственный язык Python в профильном обучении. Такой подход позволяет постепенно наращивать сложность заданий и поэтапно формировать алгоритмическое мышление. Применение цифровых образовательных сред «ПиктоМир» и «КуМир», интегрированных в цифровую образовательную платформу «Мирера», которая использует искусственные нейронные сети и предлагает вариативные формы обучения в виде индивидуальных образовательных траекторий, способствует повышению качества и ускорению освоения предмета. Оценка процесса решения задач и целенаправленная подготовка будущих учителей информатики с применением этой методики обучения показала двукратный рост успеваемости, которая достигла теоретического максимума усвоения материала.Эксперименты проходили в учебном процессе следующих образовательных организаций: в Московском государственном университете имени М.В. Ломоносова на механико-математическом факультете, в Московском педагогическом госуниверситете (МПГУ), в Государственном университете управления, в НИЦ «Курчатовский институт» с использованием цифровой образовательной платформы Мирера, а также в работе кафедры ДПО ФГУ ФНЦ НИИСИ РАН при переподготовке специалистов системы дошкольного и школьного и вузовского образования.

Подготовлены десятки электронных курсов для педагогических университетов, вузов, курсов повышения квалификации в рамках образовательной программы кафедры дополнительного профессионального образования ФГУ ФНЦ НИИСИ РАН. Методические рекомендации по использованию разделов курса информатики по основам алгоритмизации и программирования были использованы в системе основного и дополнительного образования. Основные положения диссертационного исследования были представлены: в публикациях, отражающих комплексную реализацию результатов в дошкольных образовательных организациях, основной и начальной школы, в педагогических вузах и вузах естественно-научной ориентации; в практической исследовательской деятельности в виде результатов грантов:

Фонда президентских грантов № 22-1-008068 «Дошкольное воспитание: новые ориентиры для педагогов и родителей», 2022-2023 гг.

Фонда президентских грантов № 23-2-003889 Просветительский марафон "В авангарде детства": передача традиционных российских духовно-нравственных ценностей от поколения к поколению, 2023-2024 гг.

РФФИ 18-07-00901 «Исследование и разработка системы распознавания элементов рукотворного интерьера на базе нейронных сетей для построения дополненной реальности и выработки алгоритмов взаимодействия управляемых объектов с реально-виртуальным окружением», 2018-2020 гг.

РФФИ 19-29-14057 «Применение машинного обучения и нейронных сетей для построения динамических персональных траекторий обучаемых и автоматической верификации правильности выполнения заданий в цифровых образовательных системах», 2019-2023 гг.

Результаты диссертационного исследования докладывались и обсуждались более чем на 90 международных, всероссийских, региональных конференциях, форумах, симпозиумах, семинарах и экспертных совещаниях разного уровня, включая:

Всероссийский съезд учителей математики в МГУ имени М.В. Ломоносова, г. Москва, 2010, 2021, 2023.

Международная научно-практическая конференция "Информационные технологии в образовании XXI века, г. Москва, 2015.

ИТО-РОИ, Большой московский семинар по методике раннего обучения информатике, г. Москва, 2007, 2009, 2012, 2013.

Всероссийская конференция «Свободное программное обеспечение в высшей школе», г. Переяславль-Залесский, 2008, 2009, 2010, 2011, 2012, 2016, 2017, 2018, 2019, 2022.

Открытая Всероссийская конференция «Преподавание информационных технологий в Российской Федерации», г. Петрозаводск, 2010.

Международная конференция «Разработка ПО/SECR», г. Санкт-Петербург, г. Москва, 2017, 2018,

Международная конференция "Воспитание и обучение детей младшего возраста" (ECCE), г. Москва, 2018, 2019, 2020.

Международная научная конференция "Информатизация образования и методика электронного обучения: цифровые технологии в образовании", г. Красноярск, 2020, 2021, 2022, 2023.

Российско-китайская конференция исследователей образования «Цифровая трансформация образования и искусственный интеллект», г. Москва, 2019.

Международная научно-практическая конференция "Smart Nations: глобальные тенденции цифровой экономики", г. Москва, 2021.

Московский международный Салон образования, г. Москва, 2021, 2022, 2023, 2024.

Computational Methods in Systems and Software 2022 (CoMeSySo2022) conference «Вычислительные методы в системах и программном обеспечении», г. Прага (Чехия), 2022, 2023.

Международный форум по математическому образованию, IFME, ПФУ, г. Казань, 2023, 2024.

Всероссийский образовательный форум "Невская Образовательная Ассамблея", г. Санкт-Петербург, 2023, 2024.

Международная ежегодная научно-практическая конференция "Новые образовательные стратегии в открытом цифровом пространстве, г. Санкт-Петербург, 2022, 2023, 2024.

II Всероссийская научно-практическая конференция «Актуальные проблемы подготовки современного учителя начальных классов», г. Москва, 2024.

Всероссийский педагогический съезд «Моя страна», г. Санкт-Петербург, 2022, 2023, 2024.

Всероссийский научно-практический семинар «Инновационные подходы в естественно-научном образовании. STEAM-образование: от дошкольника до выпускника вуза», г. Нижний Тагил, 2022, 2023, 2024.

Наука и образование в обеспечении устойчивого развития человеческого потенциала в условиях перехода к цифровой экономике, г. Пермь, 2024

Результаты исследования отражены в 110 научных публикациях, в том числе: в 31 статьях в рецензируемых журналах, рекомендованных ВАК при Минобрнауки России, из них в 17 статьях, рекомендованных МГУ имени М.В. Ломоносова, 1 монографии, 10 учебных пособиях и методических рекомендациях.

Опытно-экспериментальной базой исследования являлись площадки всероссийского уровня в дошкольных образовательных организациях, организациях общего образования под эгидой ФГУ ФНЦ НИИСИ РАН, МГУ имени М.В. Ломоносова, а также институт Детства Московского педагогического государственного университета.

Достоверность результатов обеспечивается результатами проведенных экспериментов при обучении информатике и программированию дошкольников (более 12000 в течении одного года), школьников и студентов вузов, обсуждения полученных результатов на семинарах и конференциях, а также их публикации.

Положения, выносимые на защиту:

1. Основные компоненты алгоритмического мышления, направленные на процесс решения целевой задачи, которые включают моделирование, абстрагирование, декомпозицию, разработку алгоритма, оптимизацию, тестирование и отладку. Эти компоненты могут быть поэтапно сформированы с использованием универсального набора заданий и понятий, подходящих для различных категорий обучающихся, начиная от дошкольников и школьников и заканчивая студентами вузов, включая педагогические университеты.
2. Достижение цели формирования алгоритмического мышления при обучении информатике и программированию включает разработку следующих компонентов методической системы: содержание, методика, формы и средства. Методическая система обучения в качестве содержания включает универсальный набор задач на конструкции структурного программирования, а также базис понятий принципов программного управления. Методическая система обучения использует цифровые образовательные среды и платформы в качестве средств обучения, с вариативными методами и формами обучения, дифференцированными по применениям в детском саду, в общеобразовательных учебных заведениях или в профильных вузах, но имеющие общие универсальные составляющие.
3. Применение текстовых языков программирования с национальной лексикой, близкой к естественному языку, в пропедевтике обучения информатике и программированию для учеников, являющихся носителями национального языка, даёт возможность начать обучение одновременно с изучением чтения и письма, ещё до того, как начнётся освоение иностранного языка. Такой национальный учебный алгоритмический язык интегрирован в качестве содержания и средства обучения методической системы обучения в цифровую образовательную среду, допускающую адаптацию под языки народов России.
4. Использование в методической системе обучения универсального набора задач и понятий (содержание обучения), предметно-цифровой образовательной

пиктографической среды ПиктоМир (средство обучения), бескомпьютерной формы обучения и методики обучения с элементами искусственного интеллекта позволяет понизить границы сензитивного периода при пропедевтике информатики и программирования до возраста детского сада начиная с 4-ого года жизни.

5. Применение в методической системе обучения выделенных универсальных наборов задач и понятий в сочетании с цифровыми образовательными средами «ПиктоМир», «ПиктоМир-К» и «КуМир» значительно сокращает время на освоение фундаментальных понятий информатики и программирования, а также эффективно формирует алгоритмическое мышление у учащихся всех возрастов – от подготовительных групп детских садов до студентов вузов, включая педагогические университеты.

Структура диссертации. Диссертация состоит из введения, 5 глав, заключения, списка литературы и приложений.

ГЛАВА 1. ТЕОРЕТИКО-МЕТОДОЛОГИЧЕСКИЕ ОСНОВАНИЯ ФОРМИРОВАНИЯ АЛГОРИТМИЧЕСКОГО МЫШЛЕНИЯ

1.1 Методологические подходы к автоматизации образовательного процесса

Одним из важных вопросов является изменение или сохранение методологии классической педагогики в современном обществе с высоким уровнем насыщения информационно-коммуникационными технологиями. Что остается применимым и должно быть сохранено, а что требуется изменить, какие аспекты в основании построения современного образовательного процесса отсутствуют и, если этих аспектов будет не хватать, то потребуется ли создавать новые или искать альтернативные решения.

Современное образование претерпевает цифровую трансформацию и переходит на новый уровень методологии и дидактики [104; 254; 223; 251; 14; 81; 182; 133; 46; 221; 158; 240; 306; 371]. Этот процесс поддерживается государством. Президент России В.В. Путин в выступлении на основной дискуссии международной конференции по искусственному интеллекту и машинному обучению Artificial Intelligence Journey 2022, в частности, сказал: «…нужно идти дальше – внедрять элементы изучения искусственного интеллекта в школьные программы математики и информатики, наращивать уровень преподавания этих предметов, повышать квалификацию учителей» [93].

Специфика момента состоит в том, что субъекты современного образования находятся в окружении стремительно развивающихся цифровых технологий, использование которых приводит к коренной перестройке образовательного процесса, особенно в области информационного взаимодействия между обучающимся и педагогом, функции которого может заменить некая автоматизированная обучающая система. Непрерывное развитие цифрового мира приводит к необходимости постоянного совершенствования подобных систем, рассматривая их в разрезе как помощника педагога-человека. Совершенствование педагогических программных продуктов требует дополнительной отдачи от

педагога-человека для перепроектирования интерактивной педагогической системы на базе новых цифровых информационных технологий. Таким образом, общая нагрузка на педагога, как учителя и педагога-разработчика таких систем суммарно не уменьшается, а возрастает, хотя целью создания подобных педагогических автоматизированных систем было снизить рутинную нагрузку на преподавателя. Во многом ни педагогическое сообщество, ни общественность еще не высказывают оптимизма о сроках осуществления полного цифрового перехода в образовании [153; 211; 212].

При этом цифровое образование должно продолжить выполнять свою основную задачу. Л.Н. Толстой писал: «Образование в обширном смысле, по нашему убеждению, составляет совокупность всех тех влияний, которые развивают человека, дают ему более обширное миросозерцание, дают ему новые сведения. Детские игры, страдания, наказания родителей, книги, работы, учение насильственное и свободное, искусства, науки, жизнь — все образовывает» [238].

Все люди разные, дети также рождаются разными, и в задачу образования не входит сделать их ментально и физически одинаковыми. Напротив, «образование - единый целенаправленный процесс воспитания и обучения, являющийся общественно значимым благом и осуществляемый в интересах человека, семьи, общества и государства, а также совокупность приобретаемых знаний, умений, навыков, ценностных установок, опыта деятельности и компетенции определенных объема и сложности в целях интеллектуального, духовно-нравственного, творческого, физического и (или) профессионального развития человека, удовлетворения его образовательных потребностей и интересов» [161], как записано в законе «Об образовании в Российской Федерации». То есть к целям образования прежде всего относится развитие личности, а полная автоматизация этого процесса как преобразование личности требует формально описания, какую личность к какой требуется преобразовать.

Успехи обучения в том или ином предмете часто зависят от физических возможностей ученика и от ментальных и когнитивных характеристик личности,

часто отождествляемых со способностями. Подтверждая тезис, что сложность в построении универсального образовательного метода связана с различиями в учениках, можно сослаться на высказывания чешского педагога Яна Коменского: «…у одних способности острые, у других — тупые, у одних — гибкие и податливые, у других — твердые и упрямые, одни стремятся к знаниям ради знания, другие увлекаются скорее механической работой. Из этого трижды двойного рода способностей возникает шестикратное сочетание их» [91]. Исходя из градаций способностей можно предположить, что это и есть классы особенностей детей, позволяющие сегодня вводить автоматизацию образовательного процесса. Однако ниже Ян Коменский приводит слова Плутарха «Какими дети рождаются, это ни от кого не зависит, но, чтобы они путем правильного воспитания сделались хорошими — это в нашей власти» [91]. Приведенное высказывание говорит об условности подразделения детей на ограниченное число классов, что связано в основном с ограниченностью числа педагогов. Ян Коменский предлагает родителям до 6-летнего возраста готовить своих детей к будущей школе в качестве первых педагогов в жизни ребенка, назвав эту подготовку «материнской школой». Дети должны были воспитываться и обучаться базовым навыкам, включая развитие речи и начальные математические представления. Таким образом, материнские школы Яна Коменского являлись выравнивающим образовательным институтом, позволяющим детям легче адаптироваться к настоящей школе. В своей работе «Великая дидактика» автор предлагал набор определенных универсальных методик, не зависящих от обучаемого. Такой подход дает определенную надежду на автоматизацию образовательного процесса, и сам Ян Коменский предлагал упрощение предметов для их более успешного освоения учениками [91].

Универсальность дидактического подхода обучения конкретного ученика не всегда может гарантировать успешный результат. Снижая рамку уровня образования, все больше учеников становятся способны сформировать требуемые навыки. Однако тем самым из образовательного процесса исключается та часть

обучающихся, которые изначально были готовы быстрее освоить навыки и сформировать умения. Гармоничное развитие индивидуума может помочь решить эту проблему, когда на уроке для учителя все дети обладают равной восприимчивостью и способностью к обучению. Такой идеи придерживался И.Г. Песталоцци, выступая за равномерное развитие личности [183]. При этом И.Г. Песталоцци также поддерживал идею соответствия учебных задач уровню ученика, предлагая практику постепенного поступательного перехода от простого к сложному, от лёгкого к трудному и т.п.

Особо надо отметить, что И.Г. Песталоцци считал важным раннее обучение ребенка, включая формирование логического мышления: «Руководство, ведущее к достижению цели обучения, которое я до сих пор принимал во внимание, все же является только утончением чувственного хода природы для достижения моей цели. Однако к ней может вести еще более высокое средство, еще большее завершение этого утонченного чувственного хода природы — это логическое мышление, развитие логического мышления… В качестве средства для развития во всем их объеме наших сил и задатков мой метод не содержит учение об истинах, а является обучением истине; он не борец против заблуждений, а представляет собой духовное развитие нравственных и умственных сил, которые противостоят заблуждениям; он развивает способность распознавать истину и заблуждение. Сущность его стремлений состоит единственно в том, чтобы обеспечить правильное с точки зрения психологии развитие этой способности и обеспечить его в той мере, как это необходимо. Обучение должно вести к развитию логического мышления» [184]. И.Г. Песталоцци предлагает детализированную методику обучения ребенка, однако автоматизация этой методики достаточно сложна из-за неформальности изложения и невозможности дифференцированно оценить с надлежащей дискретизацией достижения обучаемого.

Ж.-Ж. Руссо особо выделял индивидуальный подход в обучении. Образовательный процесс должен быть построен педагогом так, чтобы интерес к обучению исходил от ребенка, а педагог в режиме диалога раскрывал ученику

истины, свойства предметов и окружающего мира, который последний мог увидеть, услышать, осязать и т.п., то есть воспринять своими органами чувств. Фактически речь шла о предметной учебной среде, которую должен наблюдать и исследовать ребенок. Эта окружающая среда по завершении каждого цикла успешного освоения учеником должна расширяться, дополняясь новыми объектами и явлениями, готовая для нового уровня обучения. Ключевой особенностью обучения Ж.-Ж. Руссо считал соотнесение уровня знаний к уже приобретенному опыту ребенка, подводя ученика к задаче и подталкивая его к самостоятельному решению. Методы обучения выбирались в зависимости от конкретного образовательного контекста, дабы поддерживать заинтересованность и формировать собственную определённую точку зрения на окружающую действительность.

О формировании мышления Ж.-Ж. Руссо писал: «Способ образования идей и придаёт особый характер человеческому уму. Ум, формирующий свои идеи, только по действительными отношениям, есть ум основательный; кто видит отношения такими, каковы они на самом деле, у того верный ум; кто плохо их оценивает, у того ложный ум; кто выискивает отношения воображаемые, не соответствующие ни действительности, ни внешности, тот сумасшедший; кто не умеет вовсе сравниваться, тот слабоумный. От большей или меньшей способности сравнивать идеи и находить отношения и зависит в людях большая или меньшая степень ума, и т. д.» [214].

Важность индивидуализации учебного процесса поддерживал К.Д. Ушинский, придавая особое место в образовательном процессе диалогу педагога с учеником, считая, что беседа с учителем не просто вербальная передача знаний, а развитие мышления ребенка [79]. В качестве важного аспекта образования К.Д. Ушинский выделял родной язык. Язык как элемент национальной культуры инкорпорирует национальные образы, воззрения и чувства, напрямую ассоциированные с национальным сознанием. Изучение родного языка педагог выделял как самостоятельную задачу образования, тесно

связанную с культурой, народным творчеством, историей родины, географией, природой и религией своей страны. Воспитанию как части образования К.Д. Ушинский уделял особое место:

«1) Общей системы народного воспитания для всех народов не существует не только на практике, но и в теории, и германская педагогика не более, как теория немецкого воспитания.

2) У каждого народа своя особенная национальная система воспитания; а потому заимствование одним народом у другого воспитательных систем является невозможным.

3) Опыт других народов в деле воспитания есть драгоценное наследие для всех, но точно в том же смысле, в котором опыты всемирной истории принадлежат всем народам. Как нельзя жить по образцу другого народа, как бы заманчив ни был этот образец, точно так же нельзя воспитываться по чужой педагогической системе, как бы ни была она стройна и хорошо обдумана. Каждый народ в этом отношении должен пытать собственные свои силы» [242].

Последние тезисы приводят к мысли, что не только образование должно вестись на родном языке, но и современные автоматизированные обучающие системы должны базироваться на национальных языках.

Все предыдущие рассуждения были связаны с вопросами сохранения основы образовательных процессов при подходе к вопросу автоматизации обучения, то есть, что можно извлечь из классической педагогики для ответа на вопрос, что и почему должно быть сохранено, что, безусловно приведет к необходимости введения новаций в образование. Вопросы влияния вычислительной техники на образовательный процесс были в фокусе внимания еще в 1960-х годах, прежде всего в области автоматизации труда педагога. Автоматизация деятельности или части деятельности человека часто может быть выражена вербально в виде записи мыслей, приемов, планов, алгоритмов и т.п. Уровень формализации таких записей и рассуждений не всегда возможен, в противном случае в информационном веке можно было бы перепоручить обучение и воспитание автоматическому устройству

в предположении, что такое устройство сумеет достичь гарантированных результатов при выполнении формализованных учебных действий [237].

Вот что пишет Л.Н. Ланда: «Задачи, которые человек должен уметь решать в ходе своей деятельности, крайне многообразны. Научить в школе решению всех задач, которые могут встретиться в жизни, невозможно: их количество практически необозримо. Тем не менее школа должна подготовить учащихся к тому, чтобы в будущем они умели решать самые разнообразные задачи. Сделать это можно единственным способом: обучая учащихся решению конкретных задач, формировать у них достаточно общие методы мышления и вообще деятельности, общие способы подхода к любой задаче, умение искать решение в любой новой ситуации. Формирование таких достаточно общих методов мышления и деятельности является поэтому одной из важнейших задач школы» [112].

Привнести процесс алгоритмизации в педагогику, изобрести алгоритм обучения молодых учителей, это одна из задач, которую обсуждает Л.Н. Ланда. Он вводит наряду с формальным алгоритмом в математике, понятие предписания, которое может быть формальным алгоритмом, но может и не быть, то есть предписание – суть рекомендации по выполнению некоторых действий. При этом самим алгоритмам, как сущности, надо учиться: «Основная задача обучения алгоритмам — это владение ими, т. е. формирование алгоритмических процессов. Знание же алгоритма — лишь средство достижения этой цели. Но средство это крайне важное» [112].

Автор говорит об автоматизации учебного процесса как о программированном обучении: «Важнейшими принципами программированного обучения, как это подчеркивают почти все авторы, пишущие по этой проблеме, являются значительное увеличение удельного веса самостоятельной работы и индивидуализация обучения. Но чтобы учащиеся могли эффективно усваивать знания в процессе самостоятельной работы, необходимо ею особым образом управлять, нужны определенные условия и специальная организация обучения…

Обеспечить условия для эффективного усвоения знаний путем самостоятельной работы, расширить сферу ее применения в учебном процессе и индивидуализировать обучение можно только в том случае, если будут созданы и использованы пособия и устройства, позволяющие гибко приспосабливать ход обучения к динамике усвоения знаний, умений и навыков каждым учеником, автоматически (или полуавтоматически) управляя процессом учения и обучения и автоматически (или полуавтоматически) его регулируя. Такими пособиями и устройствами и являются программированные учебники и обучающие машины» [112]. То есть Л.Н. Ланда говорит об использовании автоматизированных педагогических образовательных систем, которые реализуют декларированную выше индивидуализацию и автоматический контроль учебного процесса. Роль педагога в такой связке меняется, и фактически учитель в цифровой образовательной системе становится наблюдающим за автоматизированным учебным процессом и управляющим им.

Изучая алгоритмы и решая задачи, ученик формирует новый тип мышления, который предварительно можно называть алгоритмическим. Если необходимость в автоматизированных обучающих средствах и системах уже зафиксирована, то вопрос поэтапного формирования алгоритмического мышления остается открытым.

Вот что пишет по поводу поэтапного формирования умственных действий и понятий П.Я. Гальперин «…в ориентировочную основу действия с самого начала включаются не только теоретические знания, но и такие операции, которые позволяют найти искомые величины в конкретном материале. Знания формируются без предварительного заучивания, в процессе применения к решению задач; в подборе этих задач должны учитываться все типы материала, и от нас зависит подобрать его так, чтобы “практика” была представлена во всех желаемых формах» [32]. То есть для формирования алгоритмического мышления необходимо использовать определенный набор заданий, которые покрывают все вопросы практики и поэтапно декапсулируют понятия и теоретические знания.

Л.Н. Ланда предлагает следующий последовательно усложняющийся набор приемов, как научить решать алгоритмические задачи разного уровня формализации и детерминированности:

«1) обучать алгоритмам решения;

2) обучать алгоритмам поиска алгоритмов;

3) обучать общим методам поиска неалгоритмического характера;

4) обучать отдельным правилам действий, указывая учащимся, какие операции можно применять в процессе решения;

5) не обучать ни алгоритмам, ни методам неалгоритмического характера, ни правилам, а ставить человека в проблемную ситуацию, сталкивать его с задачами, рассчитывая на совершенно самостоятельное нахождение (открытие) алгоритмического процесса (алгоритмической процедуры) в ходе самонаучения» [112]. Л.Н. Ланда высказывает определенный скептицизм относительно 5 пункта вышеперечисленных способов обучения решения алгоритмов. Не каждый ученик способен самостоятельно открыть алгоритм, более того, если не использовать предварительно в практике обучения приемы 1-4, то такое количество не научившихся учеников будет достаточно велико.

Нет ничего плохого в обучении алгоритмам [237], потому что, например, многим в жизни придется четко следовать определенным алгоритмам в своей работе. Правильный образовательный процесс требует, чтобы ученик самостоятельно изобрел как можно больше алгоритмов для решения тех или иных задач. Формализация этих задач позволит перепоручить проверку ученических решений автоматизированной системе вне зависимости, о каком из предметов здесь идет речь.

1.2 Методологические предпосылки возникновения информатики как самостоятельного предмета в школе

Чтобы сформировать концепцию развития алгоритмического мышления, необходимо провести исследование, изучив в историческом контексте задачи,

которые решает дисциплина «Информатика» в рамках школьной программы, а также навыки и компетенции, которые она развивает у учащихся.

В период, начавшийся с началом 1960-х годов и продолжавшийся до 1985 года, информатика не была самостоятельным предметом в средней школе СССР. Однако советскими педагогами-новаторами, такими как, например, А.П. Ершов, С.И. Шварцбурд, В.С. Леднев и др. было организовано преподавание экспериментальных курсов, на которых изучались основы программирования, представление информации в ЭВМ, алгоритмы, логическое преобразование информации и т.п., которые фактически явились предтечей предмета информатика [116].

Проблемы информатизации образования, как главного направления информатизации общества, определяют специфическую предметную область. Отмечая требования этой области, А.П. Ершов в статье [60], предваряющей представление реформы 1985 года, которая вводила курс «Основы информатики и вычислительной техники» в советскую общеобразовательную школу, писал:

«Отбор содержания и методов обучения новому предмету оказался очень нелегким делом, главным образом, потому что значительная часть факторов, влияющих на конкретные решения, носила противоречивый характер. Укажем на основные противопоставления:

- стабильность и общепринятость научного багажа общего образования наряду с динамичным и становящимся характером информатики;
- стратегическая необходимость компьютерной грамотности и недостаточная подготовленность общественного сознания, в частности, в учительской среде;
- разные облики программирования: математическая деятельность и сумма приемов работы с ЭВМ, интровертивное и экстравертивное программирование, системное и прикладное программирование, трудная специфическая профессия и массовая человеческая практика;
- разнообразие языковой практики программирования и единство учебного процесса в школе;

- работа в вычислительном кабинете с неограниченным доступом к ЭВМ и традиционная форма школьного урока;
- необходимость единого нормативного начала и необходимость разнообразного и поискового эксперимента.

Надо отметить, что в одном эти разноречивые факторы действовали совместно: они властно диктовали, требовали жестко ограничить учебное пособие по объему материала и сделать его максимально доступным, прежде всего, для учителей – математиков и физиков, которым будет поручено преподавание основ информатики и вычислительной техники после сравнительно короткой курсовой подготовки» [236].

Еще в 1979 году А.П. Ершов вместе с группой специалистов ВЦ Сибирского отделения АН СССР доказал актуальность нового школьного предмета [75], дав конкретную формулировку целей информатизации школьного образования и обосновал необходимость формирования нового стиля мышления у подрастающего поколения. Этот стиль мышления должен соответствовать требованиям информационного общества [81], когда по мнению А.П. Ершова, общество становится информационным с того момента, когда стоимость одного книжного типографского знака начинает превышать цену хранения одного печатного символа в памяти компьютера.

По определению Р.С. Немова «Мышление – это движение идей, раскрывающее суть вещей. Его итогом является не образ, а некоторая мысль, идея... Мышление – это особого рода теоретическая и практическая деятельность, предполагающая систему включённых в неё действий и операций ориентировочно-исследовательского, преобразовательного и познавательного характера» [156]. Системные связи между законами и сущностями изучаемых вещей и явлений выстраиваются в результате целенаправленной познавательной деятельности человека в некую интеллектуальную сеть способом, который выбирает сам человек. Этот способ ментальных построений и операций, производимых над

идеями, приводящий к постижению и (или) созданию понятий, законов и сущностей, называется стилем мышления.

А.П. Ершов поставил задачу построения модели выпускника эпохи информационного общества, понимая под моделью совокупность знаний, умений и навыков, которыми должен владеть выходящий из школы современный молодой человек. При этом выбор в качестве прототипа такой модели пал на программиста, как человека, который существенно в большей степени, чем любой другой специалист, всей своей повседневной как в рутинной, так и в творческой деятельности формировал в себе умения и навыки, помогавшие ему наиболее эффективно использовать вычислительную технику. Об этом А.П. Ершов писал в своей философской статье: «Программист обязан обладать способностью первоклассного математика к абстракции и логическому мышлению в сочетании с эдисоновским талантом сооружать все, что угодно из нуля и единицы, он должен соединять в себе аккуратность бухгалтера с проницательностью разведчика, фантазию автора детективных романов с трезвой практичностью бизнесмена, а кроме того, иметь вкус к коллективному труду, быть лояльным к организатору работ и т.д... Программист – солдат второй промышленной революции и как таковой должен обладать революционным мышлением и мужеством» [66].

Таким образом, обоснование места школьной информатики в народном образовании возможно начать только с рассмотрения перечня основных знаний, умений и навыков, которые необходимы программисту как прототипу модели, эффективно использующей современный, новый для большинства жителей планеты в 1980-х годах универсальный дидактический инструмент – компьютер:

- Умение планировать структуру собственных действий: программист обязан предвидеть в деталях еще нереализованное действие описываемой им программы, используя для этих целей команды – элементарные стандартизованные языковые средства.
- Умение строить информационные модели: программист убежден, что хотя компьютер и способен распознавать типы и структуры

используемых в программе данных, но тем не менее возможность предварительно передать компьютеру дополнительную информацию о данных существенно повышает эффективность программирования.

- Умение организовать поиск информации: актуальность этого умения близка программисту, который ощущает, что операции вычислений и обработки информации вообще выполняются многократно быстрее, чем поиск используемых операндов, и тем более, если размещаются они в непредсказуемых областях запоминающих устройств с различным быстродействием выборки. Следовательно, для суммарного быстродействия создаваемой программы необходимо ясно представлять, где, как и какую информацию следует хранить и искать, и предусмотреть эти действия в программе.
- Дисциплина и структурированность общения: необходимые строгость и формализмы общения человека с компьютером обязывают программиста внимательно учитывать и осознавать уровень программных средств интерфейса - от использования ЭВМ, не снабженных практически никаким программным обеспечением, до разнообразных информационно-технологических инструментов, когда программисту достаточно уметь собирать программу из готовых модулей, не зная подчас алгоритмы и языки программирования реализации таких блоков.
- Навыки своевременного использования компьютера, которые необходимо использовать всякий раз, когда возникает потребность в автоматизированных процессах обработки информации, то есть компетентности применения различных, ранее не знакомых и не излучавшихся приемах работы с информацией.

Если важность вышеперечисленных знаний, умений и навыков выглядит закономерной, тем не менее необходимо показать связь этих категорий с умениями

и навыками существенно более важными и широкими, общечеловеческими, общекультурными и социальными.

Востребованное сегодня умение планирования своей деятельности актуально практически в каждой профессии [87], включая новые, такие как системный инженер композитных материалов, разработчик систем энергопотребления, проектировщик промышленной робототехники и т.п. Однако не ограничиваясь вышеперечисленными, к ним можно добавить также руководителя промышленного производства, спортивного тренера, агронома, военного стратега и т.д., то есть людей различных сфер труда. В практической ежедневной деятельности специалисты различных профессий сталкиваются с процессом планирования и эти навыки нужны им в той же мере, как и программисту, создающему компьютерные программы. Важно отметить, что в большей степени навыки планирования востребованы учительством: учитель планирует и урок, и неделю, и учебную четверть, и учебный год.

Если обратиться к моделированию, то это умение необходимо каждой творческой личности, такой как архитектор, авиаконструктор и др. Информационное моделирование, которым постоянно занимается программист, описывая структуру данных в программируемой задаче, – это частный случай моделирования. Для учителя модель успевающего ученика состоит в формально записываемых требованиях к знаниям и умениям учащихся.

Независимо от профессии, каждый творчески работающий человек испытывает необходимость в методах организации и хранения информации. При этом переводчик, традиционно обращающийся к книжной версии словаря, использует метод бинарного поиска, историк предпочтет хронологическую последовательность представления информации, библиотекарь организует свои каталоги в алфавитном порядке и т.д. Для учителя поиск информации имеет принципиальное значение, так как никаким экстенсивным способом обучения невозможно решить проблему передачи школьнику за время обучения в школе той суммы знаний, которую человечество накапливало тысячелетиями. Подчас важнее

не знать некоторую информацию, а уметь ее найти за кратчайшее время, используя информационно-коммуникационные технологии [260]. Причиной этого послужил информационный взрыв последних десятилетий, который показал стратегический путь обучения – научить ребенка искать информацию, необходимую ему для решения поставленной задачи в тот момент, когда в ней возникает необходимость или, другими словами, надо научить школьника учиться.

Проблемы коммуникации в информационном обществе существенно повышают свою важность, потому что постоянно растут не только требования к объему используемой информации, но и к скорости ее передачи по каналам связи. Такие весьма заметные количества и скорости передачи информации приводят, во всяком случае, в организации учебно-воспитательного процесса на всех уровнях образования и во всех его формах к инновационным, качественным его изменениям [239]. В приведенной ниже схеме отмечен в числе принципиальных переход от дисциплины общения и структурирования сообщений к умениям и навыкам общения в широком общекультурном смысле слова, Рисунок 1.

Инструментирование как часть операционного стиля мышления появилось только со становлением информационного общества, когда была практически решена проблема использования всевозможных механических, электрических и электронных устройств в качестве помощников человека, облегчающих механизировать и автоматизировать физический труд. Достижения в сфере поисковых систем, искусственного интеллекта, анализа естественных языков и др. делают актуальным инструментарий автоматизации различных сфер интеллектуальной деятельности человека, ранее не автоматизировавшихся.

Объединяемая воедино совокупность знаний, умений и навыков, составляющих схему, и называется *операционным стилем мышления человека* информационного общества.

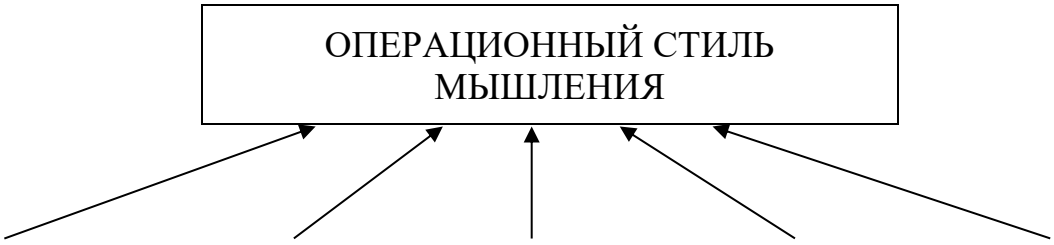


Рисунок 1 – Операционный стиль мышления

Стиль мышления, обоснованный в качестве образовательной цели школьной информатики и описанный в работах [80; 236; 65; 62; 63; 66; 67; 36], изначально был назван «программистским». Только несколько лет спустя А.П. Ершов предложил использовать более точный термин – операционный стиль мышления, в котором подчеркивалась алгоритмическая направленность этого способа мышления, позволяющая и сегодня употреблять термин *алгоритмический стиль мышления* в случаях, когда речь идет о важнейшем педагогическом назначении школьной информатики. Важным аспектом является операциональная сущность, благодаря которой представления о прикладных аспектах школьной информатики совмещают с понятием информационных технологий. Вместе с тем ясно, что операционный стиль мышления представляет собою дидактическую категорию, более широкую, чем алгоритмика и информационные технологии, рассматриваемые по отдельности.

Можно сделать выводы из актуальности проблемы, которая является, по существу, социальным заказом общества к системе образования. Такие выводы по определению касаются общества в целом, а не отдельных личностей, проблема формирования стиля мышления должна быть задачей некоторого социальный института, единого для всех членов общества.

С одной стороны, задача формирования мышления молодого человека должна ставиться достаточно рано: в начальной школе или даже в дошкольном образовании. Запоздание с формированием стиля мышления, с формированием

личности опасно, потому что это приведет в будущем к ломке уже устоявшегося стиля мышления, что и сложнее, и, по оценкам психологов, опаснее. С другой стороны, с обучением школьников использовать конкретные инструментальные средства современных универсальных или специализированных компьютеров не следует торопиться потому, что курс должен быть построен на фундаментальных основах дисциплины, чтобы быть необходимым не только сегодняшним людям, но и тем, для кого современные ЭВМ будут морально устаревшими, кто будет в профессиональной деятельности использовать компьютеры более поздних поколений.

Отсюда вывод: социальным заказом общества является необходимость некоторого непрерывного курса, охватывающего широкий диапазон обучения от младшего школьника до студента вуза, который начинается еще задолго до порога школы, в дошкольном образовании. Очевидно, что такой протяженный по времени курс длиной в десяток-другой учебных лет должен быть построен по принципу дидактической спирали [181], по которой двигаясь вверх, необходимо поэтапно подниматься каждый раз на новый понятийный уровень.

Поставленная задача должна быть в генерализованном виде решена в рамках общеобразовательной средней школы в курсе информатики. Важно отметить, что задача обладает полным концептуальным набором (множеством элементов, понятий, операций, отношений), с помощью которых формируются навыки и умения фундаментального нижнего уровня – навыки эффективного использования компьютерной техники, а следовательно, и общекультурные умения и навыки, формирующих операционный стиль мышления.

Для того, чтобы формировать навыки планирования, весьма полезными оказываются управляющие структуры – серии, ветвления, циклы – из которых, как доказано в теоретическом программировании, может быть построено описание любого процесса, в том числе мыслительного процесса.

Многообразные структуры данных, определяемые в информатике – различные типы чисел, символы, строки и тексты, массивы, записи, файлы разных

форматов – предлагают конструктивный материл для проектирования информационных моделей и в целом для моделирования в разных сферах интеллектуальной творческой деятельности.

Поисковые механизмы информатики от простого перебора до сложных способов поиска в системах управления базами данных и в сложнейших современных сетевых поисковиках обеспечивают функционирование многочисленных прикладных программных систем [119; 120; 121].

Общение (как человеческое, так и машинное) становится эффективным только тогда, когда оно опирается на доведенные до автоматизма навыки использования процедур функций, и, наконец, инструментирование (автоматизация и информатизация) всевозможных интеллектуальных видов деятельности становится возможным с помощью компьютерных программ и систем.

Приведенная схема логически замыкает цепочку ментальных объектов от азов информатики до сформированного стиля мышления, Рисунок 2.

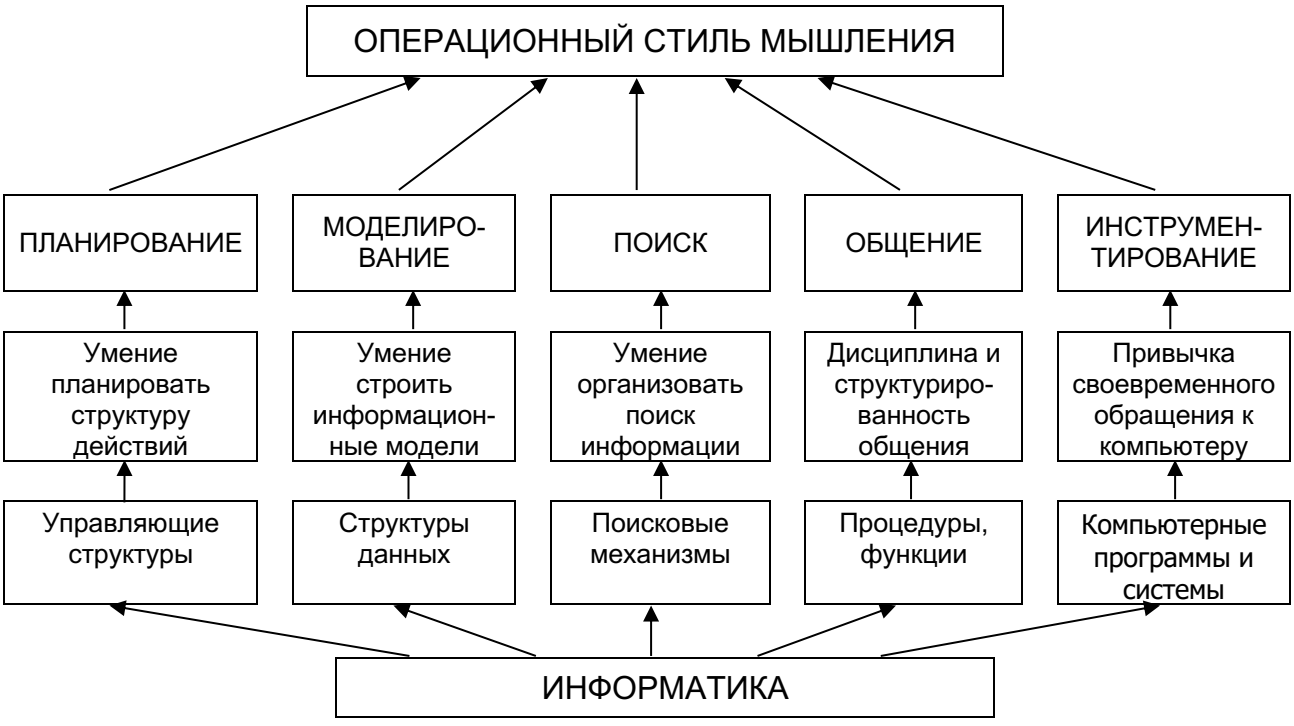


Рисунок 2 – Место информатики в формировании операционного стиля мышления

В советской педагогической школе было принято при анализе учебного процесса оперировать понятием методической системы обучения, введенной А.М. Пышкало. Она представляла собой «структуру, компонентами которой являются цели обучения, содержание обучения, методы обучения, формы и средства обучения» [205]. Такую пятикомпонентную педагогическую систему часто используют и нынешние исследователи учебного процесса. Важно оценить, в какой мере выводы из обоснования школьного курса информатики повлияли на компоненты методической системы обучения. Цели обучения, как было показано выше, сливаются с очерченными выше проблемами формирования операционного стиля мышления.

Прослеживая все остальные компоненты курса информатики, приходится постоянно сталкиваться с социальными стереотипами, возникающими на пути информатики как школьного курса и тормозящими его развитие. Одним из первых таких стереотипов стало установление возрастного порога информатики в школе. Действительно, первые государственные учебники по общеобразовательному школьному курсу информатики согласно реформе 1985 года [64; 175; 80] были адресованы учащимся двух выпускных классов и учителям информатики. Отношение к информатике как курсу, доступному только старшим школьникам, долго господствовало в общественном мнении. Пришлось приложить немало усилий и времени, чтобы сломать этот стереотип, несмотря на отмеченные выше толкования целей информатизации образования, вытекающие из них выводы и постоянно расширяющийся педагогический эксперимент энтузиастов информатики и ее методов в учебный процесс начальной школы [179; 222].

Содержание обучения в информатизированной школе в том виде, в каком оно виделось авторам концепции непрерывного курса информатики [75], показано на таблице, Таблица 1.

Таблица 1 – Схема содержания обучения информатике в школе

Совокупность фундаментальных навыков, знаний, умений, понятий и представлений, необходимых для формирования операционного стиля мышления	Совокупность прикладных навыков, необходимых для применения идей и методов информатики в других отраслях человеческой деятельности	Система основных положений информатики как науки в соответствии с ее местом в современной системе научных знаний	Комплекс знаний, необходимых для общей ориентации в возможностях современной и перспективной техники и прикладных систем информатики
1-5 классы	3-8 классы	9-10 классы	11 класс

В содержании обучения выделяются четыре модуля с нечеткими возрастными границами. Первый модуль относится к пропедевтическому курсу, начинающемуся в первом классе, и включает в себя фундаментальные навыки, знания, умения, понятия и представления, необходимые для формирования операционного стиля мышления. Второй модуль – это информатизированные инновационные межпредметные связи, насыщающие информатикой другие школьные дисциплины. Это наиболее подходящее место показать ученикам, что компьютеры их встретят, не только покидая школу, а сейчас, в окружающей сегодняшней деятельности – учебной деятельности по каждому школьному предмету.

Начинающееся вторым модулем базовое обучение информатике завершается в третьем модуле, где школьник, обобщая собственный опыт, накопленный учебной деятельностью в двух предыдущих модулях, умея строить абстракции из частных проявлений процессов или явлений, уже способен увидеть роль информатики в системе научных знаний.

Наконец, последний, четвертый модуль заканчивает курс. Он непосредственно примыкает, завершает модуль 3 и полностью поглощает профильное обучение. Он изучается только теми учениками, которые выбрали для себя физико-математические или информационно-технологические профили обучения. Здесь в год завершения школы выпускников следует знакомить с современными компьютерами и современными информационно-программными системами и технологиями.

В этой схеме, где пропедевтические курсы относятся к начальной школе, нет полного совпадения со ФГОС среднего общего образования [232] и, что естественно, со ФГОС дошкольного образования [53]. При этом схема носит слишком общий характер, чтобы отразить в содержании образования принципиальный тезис о месте программирования в курсе информатики, и базируется на идеях, высказанных А.П. Ершовым, Г.А. Звенигородским и Ю.А. Первиным в 1980-х годах [137]. Исторически школьному курсу информатики сравнительно не долго удавалось сохранять статус курса, ориентированного на обучение программированию и формированию алгоритмического мышления. Программирование как содержание курса фактически вытиснилось из курса с названием «Информатика», и к 2004 году, когда в государственном образовательном стандарте появился курс «Информатика и информационно-коммуникационные технологии», объем программирования был сокращен, что стало еще одним стереотипом школьной информатики [180; 247; 115].

Большее значение предавалось формированию основ научного мировоззрения, развитию общекультурных навыков работы с информацией, овладению информационными и коммуникационными технологиями [99; 100; 117]. Тем не менее учебники, ориентированные в том числе на формирования умений составлять алгоритмы, широко представлены для школьного образования. К ним относятся, например, учебники «Информатика и информационные технологии» авторства А.Г. Гейн, А.И. Сенокосов и др. [34], «Алгоритмика», С.К. Ландо, А.Л. Семенов и др. [113] и пр. Внесение элементов информатики в

учебный процесс начальной школы в рамках стандарта общего образования по технологии оставляет много вопросов к содержанию обучения, хотя подготовлены учебники и учебные пособия [83; 189; 256; 9; 148]. Также были попытки принести знакомство с основами информационных технологий в дошкольное образование [157]. Анализ, проведенный В.В. Гриншкуном и И.В. Левченко, в частности, показывает, «…что отсутствует универсальный учебник, который мог бы служить основой для изучения всех тем непрерывного курса информатики в средней школе. Основные выявленные проблемы включают несовпадение учебных материалов с возрастными особенностями детей, а также избыточность или нехватку содержания учебных пособий» [51].

Один из основных методов обучения информатике и, в частности, формирования алгоритмического мышления, который можно продекларировать как самостоятельную задачу, состоит в систематическом обучении в рамках не только школьного, но и дошкольного образования.

1.3 Подходы к формированию вычислительного и алгоритмического мышления

Необходимо определить содержание типа мышления, который формируется благодаря средствам информатики, поскольку разнообразие дескриптивных, а не конструктивных подходов к описанию мышления среди различных исследователей выявляет пробел, нуждающийся в заполнении.

В современном мире часто пытаются сопоставлять вычислительное и алгоритмическое мышление, алгоритмическую и вычислительную культуру [358; 321; 315; 182; 401; 311]

Принято считать, что впервые термин вычислительное мышление (англ. computational thinking) появился у Симора Пейперта [364; 177]. При обсуждении вопросов формирования вычислительного мышления у школьников также часто ссылаются на более позднюю работу, в которой С. Пейперт подверг критике классическую методику обучения школьной математике [363]. Как и в случае с

алгоритмическим мышлением, не существует единой формулировки для термина вычислительное мышление [310; 286]. Так, David Moursund, основатель Information Age Education, предполагал, что вычислительное мышление близко по содержанию с понятием процедурного мышления, предложенным С. Пейпертом [364], понимая под процедурным мышлением – процесс придумывания, создания, тестирования и отладку процедур, где, по аналогии с программами для ЭВМ, процедура состоит из списка инструкций, которые могут быть механически (автоматически) распознаны (интерпретированы) и выполнены определенным исполнителем, таким как компьютер или другое автоматическое устройство [375]. Gerald Sussman (MIT, США) определяет содержание термина вычислительное мышление как осознанный подход к решению задач, который включает в себя анализ методов их выполнения. Хотя вычислительное мышление тесно связано с математическим мышлением, но это не одно и то же. Оба типа мышления основываются на абстракции и использовании упрощенных моделей в рассуждениях [375].

Как видно из предыдущего абзаца, вычислительное мышление достаточно близко к математическому мышлению и алгоритмическому мышлению, если не полностью совпадает с последним.

Этимология вычислительного мышления связывает определение с электронно-вычислительными машинами, а так как общение с последними требует некоторого языка взаимодействия, компьютерного языка, языка программирования, некоторые исследователи связывают понятие вычислительного мышления с языком и его компонентами: синтаксисом, семантикой, прагматикой. Gerald Sussman использует метафору «вычислительного мышления как базового языка», сопоставляя в качестве примера процесс написания стихов с решением задачи в вычислительном мышлении. По его мнению, поэт соединяет в последовательности известные ему эмоциональные блоки, выраженные на языке, чтобы решить задачу, создать более крупный блок, вызывающий у читателя требуемые эмоции [375]. Эдгар Аллен По в своем эссе также описывает алгоритм процесса сочинения стихов [190].

Можно утверждать, что ребенок не рождается со сформированным вычислительным мышлением так же, как со сформированным алгоритмическим мышлением, и нет шансов на простое самоформирование вычислительного мышления подобному тому, как дети учатся говорить, Таблица 2.

Таблица 2 – Определения вычислительного мышления [400; 399; 312; 383; 321; 309; 102; 311]

Определение вычислительного мышления	Автор
...мыслительные процессы, используемые при формулировании проблем и их решений так, чтобы решения были представлены в форме, которая может быть легко реализована компьютером или другим автоматическим устройством обработки информации (которые компьютер может эффективно применять);	J. Wing (Жаннетт Винг)
…ментальные процессы абстрагирования проблем и создания автоматизированных решений;	A. Yadav и др.
…это процесс распознавания вычислений в окружающем нас мире и применения инструментов и методов из информатики для понимания и рассуждения о естественных и искусственных системах и процессах;	S. Furber)
…имеет долгую историю в информатике. В 1950-х и 1960-х годах известное как «алгоритмическое мышление», оно означает ментальную ориентацию на формулирование проблем как преобразование некоторого ввода в вывод и поиск алгоритмов для выполнения преобразований. Сегодня этот термин был расширен, чтобы включить мышление со многими уровнями абстракций, использование математики для разработки алгоритмов и изучение того, насколько хорошо решение масштабируется;	Peter Denning
…заключается в том, чтобы научить мыслить, как экономист, физик, художник и понимать, как использовать вычисления для решения проблем, творить и открывать новые вопросы, которые можно плодотворно исследовать;	S. Humpreys
…это мыслительные процессы, участвующие в постановке проблем и представлении их решения в форме, которая может быть эффективно реализована с помощью человека или компьютера. Они тесно связаны с проблемами искусственного интеллекта, которые очень сложны;	А.Ч. Курмангалиев
не существует четкого определения вычислительного мышления, и основная сложность в попытке раскрыть термин связано с определением основных компетенций вычислительного мышления по сравнению с менее значимыми компетенциями. Утверждается, что для целей концептуализации вычислительного мышления и его интеграции в образование не надо пытаться дать окончательное определение вычислительного мышления, а скорее попытаемся найти сходства и взаимосвязи в обсуждениях вычислительного мышления.	J. Voogt

Алан Кей не поддерживает метафору «вычислительного мышления как языка», так как, несмотря на врожденную способность каждого человека к устному языку, такой же универсальной одаренности в отношении письменного языка нет, как нет и в отношении математики и других наук [375].

Одной из первых деклараций содержания вычислительного мышления принято считать статью Жаннетт Винг [398], университет Карнеги-Меллон (США). Следуя автору, можно перечислить, что входит в вычислительное мышление:

- решение проблем, проектирование систем и понимание поведения людей, опираясь на фундаментальные концепции информатики;
- понимание ограничений вычислительных ресурсов для решения задачи, упрощение задачи путем сокращения, преобразования или моделирования;
- рекурсивное мышление и параллельная обработка;
- манипулирование с программным кодом не только для эффективности и корректности, но и для эстетики и дизайна;
- абстракция и декомпозиция проблемы, модификация системы, основанная на объектно-ориентированной и функциональной парадигме;
- априорное предотвращение наихудших сценариев, учет рисков и резервирование;
- эвристические рассуждения для поиска решений;
- поиск решений в условиях неопределенности;
- словарь понятий и терминов как общеупотребимые слова для современного языка.

Как видно из перечисленного набора, если вычислительное мышление и не является полноценным синонимом алгоритмического мышления, то во многом оно базируется на алгоритмическом мышлении, которое, в свою очередь, как результат познавательного и созидательного образовательного процесса индивидуума, обеспечивает его интериоризацию, фактически осуществляя приращение культуры последнего. Таким образом, вопрос о соотношении алгоритмического мышления и

алгоритмической культуры может рассматриваться как последовательность целенаправленного формирования алгоритмического мышления, изменение внутренних структур человеческой психики, в том числе посредством усвоения внешней социальной деятельности. Вышеперечисленное приводит к созданию устойчивого нового культурного опыта, то есть, в частности, к культурному содержанию – алгоритмической культуре. То же верно и для вычислительного мышления и вычислительной культуры. Задача оценивания уровня алгоритмической и вычислительной культуры достаточно сложна, так как не дает явной дискретной оценки. Один из завершающих статью тезисов Ж.Винг звучит так: «Профессора информатики должны преподавать курс под названием «Способы думать, как компьютерный ученый» первокурсникам колледжей, делая его доступным для неспециалистов, а не только для специалистов по информатике» [398].

Еще в 2003 году А.Г. Кушниренко опубликовал статью «Ученый компьютерных наук» [86], в которой раскрыл содержание профессии, фактически ответив на вопрос Жаннет Винг, который был задан лишь три года спустя.

Следующие черты определения вычислительного мышления даются в учебниках школьного образования США (K12) [358]:

- постановка проблемы в форме доступной для решения на компьютере или других автоматизированных инструментов;
- логическая организация и анализ используемых данных;
- абстрагирование при моделировании и симуляции;
- программирование решения (последовательность действий, план, программа) с использованием алгоритмического мышления;
- реализация, анализ и выборка эффективных решений на основании баланса времени и ресурсов;
- систематизация и закрепление процесса решения с целью расширения возможного круга задач.

Как видно из вышеприведенных характеристик вычислительного мышления, одна из них явно, а остальные перефразированно используют алгоритмическое мышление, то есть, даже уходя от формальной дискуссии соотношений терминов, можно определенно сказать, что вычислительное мышление в любой его самой узкой формулировке не может быть сформировано без фундамента, которым является алгоритмическое мышление в самой простой формулировке – как, например, методика решения задач. Важно отметить, что практика такого мышления основана на использовании алгоритмов, последовательностей шагов для достижения определённой цели, включающих, но не ограничивающихся процессами основной задачи программирования, конструированием одного формального объекта на базе другого [110]: моделирование, абстрагирование, декомпозиция, разработка алгоритма, оптимизация, тестирование и отладка. Однако алгоритмическое мышление полезно не только в программировании, так как помогает систематизировать подход к решению задачи, а также способствует развитию критического мышления, навыков и умений анализа и планирования.

Е.К. Хеннер [254] предлагает связать формирование вычислительного мышления с формированием ИКТ-компетентности и информационной культуры, так как обладающий сформированным вычислительным мышлением индивидуум направлен не только на использование средств ИКТ при решении задач, но и на мышление соответствующими категориями, что является важным личностным и метапредметным инструментом и результатом образования.

В работе А.Н. Стась и Н.Ф. Долгановой приводится мнение о соотношении понятий алгоритмического мышления, логического и операционного, что алгоритмическое мышление шире, чем два последних: «…алгоритмическое мышление предполагает понимание сути базовых алгоритмических конструкций, таких как следование, ветвление, цикл, переход, вызов, а также умение грамотно и эффективно использовать эти структуры при составлении простых алгоритмов на основе ограниченного набора элементарных математических операций и строить сложные алгоритмы на основе простых. Наличие развитого алгоритмического

мышления является необходимым условием способности к составлению программ для ЭВМ. Если обучаемый не обладает таким мышлением, то даже знание им одного или множества языковых средств (языков программирования) будет практически бесполезным» [234].

Важным является вопрос о методике формирования алгоритмического и(или) вычислительного мышления и возрастные границы практической методологии.

По мнению Л.Л. Босовой [14], программирование следует рассматривать в рамках развивающего и социального аспектов обучения школьников: развитие мышления, формирование новых ценностей, понимание правил поведения в цифровой окружающей среде. Л.Л. Босова пишет: «…программирование — мощный инструмент развития у обучающихся вычислительного мышления (Computational Thinking), под которым понимается способность человека распознавать и оценивать проблемы, встречающиеся в реальном мире, разрабатывать алгоритмические решения этих проблемы с последующей их реализацией на компьютере» [14]. И далее «…по сути своей вычислительное мышление достаточно близко к используемому в нашей стране понятию алгоритмического мышления: в конечном итоге, и в том, и в другом случае все сводится к признанию необходимости формирования набора тех знаний и умений, которые нужны человеку для полноценной жизни в современном информационном обществе» [14]. Далее автор предлагает набор последовательно усложняющихся связанных задач, обеспечивающий школьнику возможность развития способностей перечисленных выше процессов основной задачи программирования [110].

Таким образом, можно заметить, что важнейшая методика формирования алгоритмического мышления представляет собой практическое программирование дидактически подобранного набора задач, которое по универсальной модели решения задачи Д. Пойа [199] позволяет гарантировать формирование требуемых способностей у ученика.

Аналогичный подход наблюдается и в других работах. С.О. Алтухова, З.А. Кононова [4], соблюдая принцип «от простого к сложному», в целях формирования вычислительного мышления у студентов вуза в выравнивающих курсах «Алгоритмические основы программирования», «Алгоритмы и структуры данных» предлагают последовательный набор лабораторных работ. Данный набор включает в себя практику по последовательному программированию, алгоритмам, содержащим конструкции ветвления, конструкции цикла, алгоритмы обработки одномерных и двумерных массивов, программирование с подпрограммами и пр.

Формирование алгоритмического мышления, а также вычислительного мышления сегодня невозможно рассматривать в отрыве от процесса цифровой трансформации преподавания всех дисциплин: математики, физики, химии и других естественно-научных, а также гуманитарных предметов. Особо надо выделить значимость алгоритмического мышления для практических инженерных STEM-предметов, естественных наук, технологии, инженерии и математики, для которых программирование является естественным объединяющим ингредиентом этих дисциплин. В рамках разнообразия предметов, для которых важным фактором является сформированное алгоритмическое мышление, можно говорить об изменениях в дидактике, возможности создания новых интеграционных методик, которые выводят на новый уровень эффективности образовательный процесс благодаря возможностям, предоставляемым цифровыми технологиями, включая коммуникационное взаимодействие через сеть Интернет, искусственный интеллект и пр.

Можно говорить о новых явлениях в педагогике 21 века: цифровая дидактика, цифровая методика, цифровая дисциплина и т.п. Так, цифровой курс может включать в себя учебные материалы любых цифровых форм и форматов: перевернутый цифровой класс, практические задания с автоматизированным контрольным функционалом, тесты, цифровые практикумы без необходимости устанавливать дополнительное программное обеспечение, интегрирующий web-фреймворк (цифровую образовательную платформу [111]) с поддержкой

принципов гибкого обучения BYOD-Learning (англ. Learning at Any Time, at Any Place via any Device), когда доступ ученику предоставлен с любого устройства ко всем материалам курса. Это все и возможность участия в видеоконференциях, автоматизацию управления индивидуальной образовательной траекторией, систему интеллектуального сопровождения обучающегося с чат-ботами, цифровыми двойниками, консультации с преподавателем, в том числе через социальные сети, сохранение цифрового следа студента и т.д.

Если вопросы автоматической верификации заданий из курсов по программированию, информатики или близким к ним предметам достаточно легко решаются, то проверка заданий по другим естественно-научным и гуманитарным дисциплинам требует дополнительного внимания для формирования детерминированных критериев оценки. При этом цифровой мир может как способствовать образовательному процессу (в виде цифровых образовательных платформ), так и предоставлять учащемуся ресурсы фальсификации контроля достигнутых компетенций. Так, решения задач математических курсов студент может позаимствовать из доступных в Интернете систем [402], тем самым становится ничтожным не только проверка правильности выполненного задания по ответу к заданию, но и даже по решению.

Таким образом, цифровизация повседневной жизни, с одной стороны, создает новые запросы к системе образования, когда вне зависимости от будущей или текущей специальности человеку необходимо формировать алгоритмическое и(или) вычислительное мышление. Однако происходит и обратное давление, когда для эффективных полноценных курсов, предметов в вузах, школах и даже детских садах необходима новая методология, новая цифровая дидактика. Необходимы и новые цифровые методики для проведения цифровой трансформации образования, формирования новых и обновления текущих цифровых дисциплин и курсов, чтобы использовать все богатство окружающего индивидуума цифрового мира. Однако цифровые технологии необходимо использовать в образовании с особенной осторожностью, так далее будет

показана возможность формирования алгоритмического мышления в раннем возрасте.

1.4 Основные понятия программирования как фундамента формирования алгоритмического мышления

Чтобы добиться успехов в развитии способностей к алгоритмическому мышлению, нужно сделать функциональный аспект деятельности в области программирования ключевым элементом образовательного процесса.

Педагогический опыт, приобретенный обучением основам программирования школьников и студентов вузов [135; 204; 82; 140; 134], показал, что несмотря на то, что некоторые слушатели ранее заканчивали курсы по производственным языкам программирования, они тем не менее, как правило, не обладали сформированным алгоритмическим мышлением. Простое запоминание кода превалировало у них над самостоятельным решением задания, часто учебные программы не были результатом изложения придуманного или знакомого слушателю алгоритма на производственном языке программирования, а представляли собой просто запомненную «слово в слово» программу. Тем не менее студенты и школьники имели определенный положительный опыт в пошаговом управлении (с использованием пульта) различных электронных роботизированных изделий. Используя *непосредственное управление* [103], они решали конкретную задачу. Подобными навыками также владеют дошкольники начиная с четвертого года жизни. Легкость освоения непосредственного управления объясняется локальным характером принятия решения, когда нужно сделать выбор на основании предыдущего наблюдаемого результата, Рисунок 3.

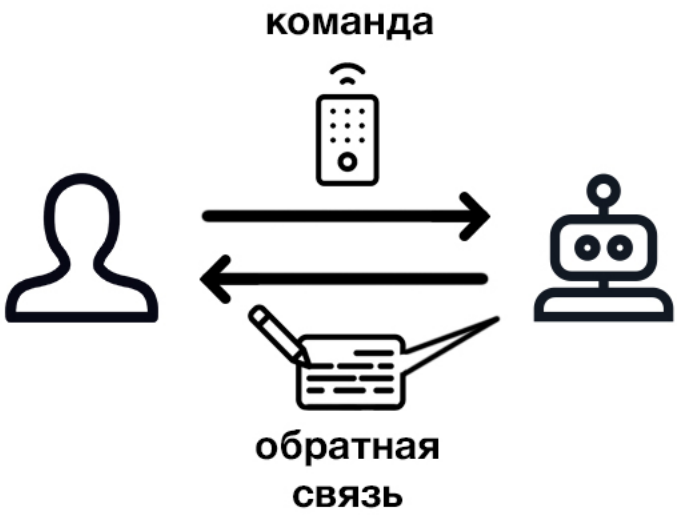


Рисунок 3 – Схема непосредственного управления с обратной связью.

При этом более сложная задача−*непосредственное управление с обратной связью,* схема которого была показана выше на рисунке, также локальна по принятию решения и не вызывает сложности у управляющего субъекта.

Кажущаяся сложность поведения управляемого объекта может быть разделена на отдельные практические независимые процессы, поэтому, например, управление автомобилем можно доверить программе [165]. На начальном этапе освоения основ программирования можно посвятить непосредственному управлению простыми объектами. Однако хотя непосредственное управление имеет отношение к формированию алгоритмического мышления, оно не покрывает его, являясь своеобразными воротами в алгоритмизацию.

Если трактовка термина программа зависит от области применимости и контекста, например, как изложение основных принципов, понятий и целей, то в информационной культуре понятие *программа* – это план будущих действий. Обычно программирование, составление программ связывают с компьютером, универсальным устройством, как правило, используемым для управления другими разнотипными и разнообразными устройствами (объектами). Если программа составлена для выполнения на компьютере, то ее называют *компьютерной*

программой. При выполнении программы компьютер чаще всего не только занимается переработкой информации внутри компьютера, но и командует одним или несколькими автоматическими устройствами, связанными с этим компьютером.

Как и при составлении плана, так и при составлении программы как плана будущих действий основные сложности для автора программы заключаются в том, чтобы попытаться умозрительно представить себе управляемые устройства, «пропустить» выполнение программы через себя и мысленно проверить достижение поставленной цели выполнения программы.

В учебнике А.Г. Кушниренко и Г.В. Лебелева [174; 108] алгоритмом называют соответствующую конструкцию школьного алгоритмического языка от **алг** до **кон**, в отличие от первых учебников А.П. Ершова, где программирование считалось некоторым техническим действием, таким как и *кодирование* [173]. Авторы учебника считали, что алгоритм – это программа на алгоритмическом языке, также ставя знак равенства между алгоритмизацией и программированием. Не обсуждая вариативность определений, *программирование* и *алгоритмизация,* важно подчеркнуть, что оба термина означают создание программ и(или) описание алгоритмов на некоторых языках программирования и(или) алгоритмических языках и(или) другим способом излагая описание плана будущих действий.

При этом одной из важнейших целей курса информатики ставилось «развитие алгоритмического стиля мышления как самостоятельной культурной ценности, независимой в каком-то смысле от компьютеров и всего прочего», формируя при этом «адекватное представление о современной информационной реальности. Это означает некоторую замкнутость, законченность, достаточность набора понятий курса» [222]. Это не отрицает более широкое познавательное значение курса информатики в школе. Так, по А.Л. Семенову [221], одной из частей курса информатики может быть «освоение информационной картины мира». Принцип программного управления позволяет сформировать такую сущность в терминах и образах, понятных ученику вне зависимости от возрастной категории.

При этом замкнутый набор понятий, необходимый для освоения основ программирования, можно сформировать таким образом, чтобы освоение его было доступно даже дошкольникам. Анализ достигнутых результатов в изучении основ программирования и практический опыт работы как со школьниками, так и со студентами университетов помог выявить основные понятия императивного программирования, которые также были использованы при обучении детей и могут быть освоены даже дошкольниками в деятельностно-игровой форме [287; 12; 46; 122].

Таким образом, для успешного формирования основ алгоритмического мышления необходимо сформировать осваиваемый для новичка базис понятий и принципов. *Принцип программного управления* имеет непосредственное отношение к указанному выше подходу.

Необходимо универсальным способом определить объект, которым управляет компьютер. Ниже приведены его характерные свойства:

- фиксированный и известный компьютеру и программисту набор команд;
- ничего не знает ни о программах, ни о языках программирования;
- способен неограниченно долго в пределах своего набора системы команд выполнять их по приказу человека или компьютера, а также других подобных объектов;
- при невозможности по объективным причинам выполнения команды отказывается ее исполнить;
- в отличие от программиста, не умеет составлять программу;
- в отличие от компьютера, не умеет выполнять программы.

Этот объект называется *исполнителем.* Примерами исполнителя являются робот, механический прибор, человек, компьютерная программа и т.п. с фиксированной функциональностью, с возможностью внешнего управления работой в некоторой обстановке. Исполнитель умеет выполнять *команды*, входящие в его функциональность, и не интересуется тем, кто им управляет.

Принцип программного управления говорит о том, что любую работу, которую человек может выполнить, командуя исполнителем, можно передать компьютеру, составив программу выполнения того действия (или последовательности действий), которую исполнителю надлежит выполнить.

С помощью принципов программного управления можно передать компьютеру любую работу, которую человек может выполнить благодаря некоторому устройству или самостоятельно. В результате компьютер получит программу, содержащую последовательность действий, которые необходимо выполнить исполнителю для достижения цели программы.

Оба сценария управления с обратной связью и без нее охватываются принципом программного управления. При этом само разделение сценариев привносит этапность обучения: на первом этапе рассматривается только управление без обратной связи, которое, естественно, более простое для освоения новичками, когда можно полностью сосредоточиться на сути принципа, не отвлекаясь на разнообразие двустороннего общения с исполнителем, отнеся это на второй этап при использовании обратной связи [46; 376]. В процессе работы с детьми выяснилось, что именно этот этап является наиболее сложным и требует больше усилий, чем второй этап, где происходит введение обратной связи [210].

Принцип программного управления без обратной связи может быть реализован следующим образом:

Программа, представляя собой план будущей деятельности, в процессе которой один объект (*компьютер*) управляет другим объектом (*исполнителем*). Эта программа заранее подготовлена человеком (*программистом*) по заранее известным правилам составления программ (*язык программирования*). Процесс выполнения программы компьютером состоит в том, что последний, следуя программе некоторым заранее согласованным способом, выдает управляемому исполнителю последовательность команд, которые и выполняет исполнитель, сообщая компьютеру об окончании выполнения каждой команды и готовности к

приему следующей команды. Чтобы компьютер мог выполнить программу, она должна быть ему предварительно сообщена (*загружена в память компьютера*).

Это описание принципа явно вводит вышеперечисленные 12 понятий: 6 объектов, 1 субъект и 5 взаимодействий между объектами и субъектом.

Объекты:

- программа;
- компьютер;
- память компьютера;
- исполнитель;
- правила составления программ (язык программирования);
- команда.

Субъект:

- программист.

Взаимодействия:

- программист составляет программу;
- компьютер загружает в свою память составленную программу;
- компьютер выполняет программу;
- выполняя программу, компьютер дает исполнителю команды;
- получив команду, исполнитель ее выполняет и готов к поступлению следующей команды;

Взаимодействия и участники представлены ниже, Таблица 3.

Таблица 3 – Участники и взаимодействия между ними.

	Взаимодействие	Участники взаимодействия
1	программист составляет программу;	программист, программа
2	компьютер загружает в свою память составленную программу;	память компьютера, программа
3	компьютер выполняет программу	компьютер, программа
4	выполняя программу, компьютер дает исполнителю команды;	компьютер, программа команда, робот
5	получив команду, исполнитель ее исполняет и готов к поступлению следующей команды**;**	команда, робот

Схема программного управления представляет собой процесс дискретный по времени: между первым пунктом взаимодействия и остальными на составление программы и выполнение программы, Рисунок 4.

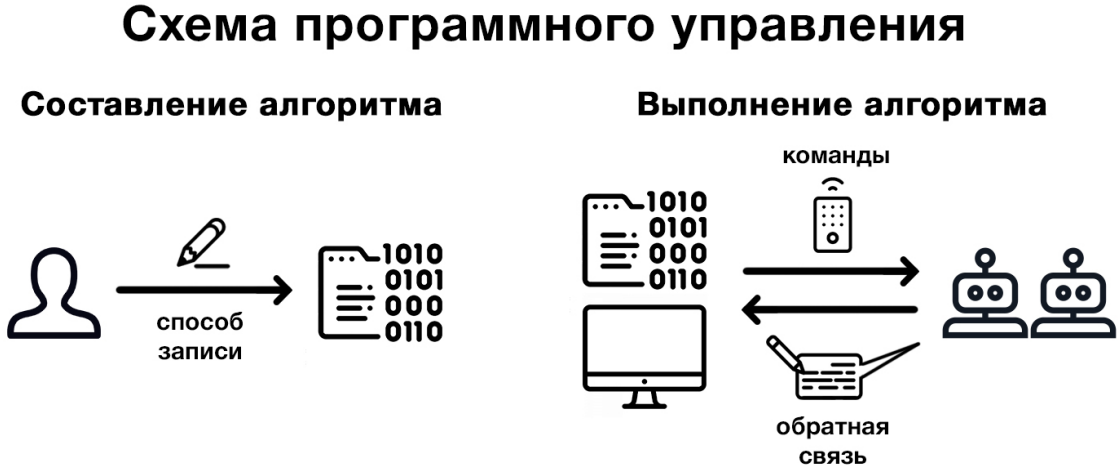


Рисунок 4 – Схема программного управления.

На практике процесс составления алгоритма является циклическим: если выполнение программы не приводит к требуемому результату, то программист производит модификацию своей программы (находит в ней ошибки и устраняет их) и последовательно проверяет ее, снова выполняя на компьютере. Специфичность программирования и состоит в том, что, как правило, сложно с первого раза составить программу, которая решает поставленную задачу, при этом к намеченному результату программист идет через серии неудач, что приучает его впоследствии не бояться ошибок.

Предполагая широкий возрастной охват для освоения принципов программного управления, понижения возраста до младшего школьного и даже дошкольного (средней, старшей и подготовительной групп детского сада), надо очень аккуратно отнестись к именованию понятий. Естественно, что для дошкольника будет сложно сразу освоить полную терминологию, особенно это относится к сложному для понимания понятию исполнитель. Необходимо подобрать адекватную замену, проще осваиваемую и(или) уже знакомую дошкольнику и младшему школьнику.

Понятийно к термину исполнитель наиболее близок *робот* как некое автоматическое устройство, предназначенное для осуществления различного рода действий, обычно выполняемых человеком. Конечно, при этом несколько сужается класс объектов, которые можно отнести к термину исполнитель. Так, человек не может быть роботом, хотя может имитировать его. Но если рассматривать набор понятий, доступный для учащихся различного возраста, то такая подмена методически будет весьма полезной [132; 109].

В измененном составе понятий принцип программного управления без обратной связи реализуется следующим образом:

Программа – это план действий, в котором один из объектов (компьютер) выступает в качестве управляющего для другого объекта (робота), который также является объектом программы, составленной заранее человеком (программистом) по заранее известным правилам составления программ (язык программирования).

Программа, которую выполняет компьютер, представляет собой последовательность действий, которые должны исполняться роботом. В процессе выполнения программы компьютер выдает роботу последовательность команд, которые последний должен выполнять. Затем робот сообщает компьютеру о том, что он завершил исполнение всех предыдущих команд и готов к приему следующей команды. Программа, которую необходимо выполнить компьютеру для того, чтобы начать выполнять свою работу, должна быть ему предварительно сообщена (загружена в память компьютера).

При этом модифицированный набор понятий выглядит так:

Объекты:

- программа;
- компьютер;
- память компьютера;
- робот;
- правила составления программ (язык программирования);
- команда.

Субъект:

- программист.

Программист составляет программу, компьютер загружает ее в память и выполняет, командуя роботом, а робот, в свою очередь, исполняет команду и ждет следующую, Рисунок 5.

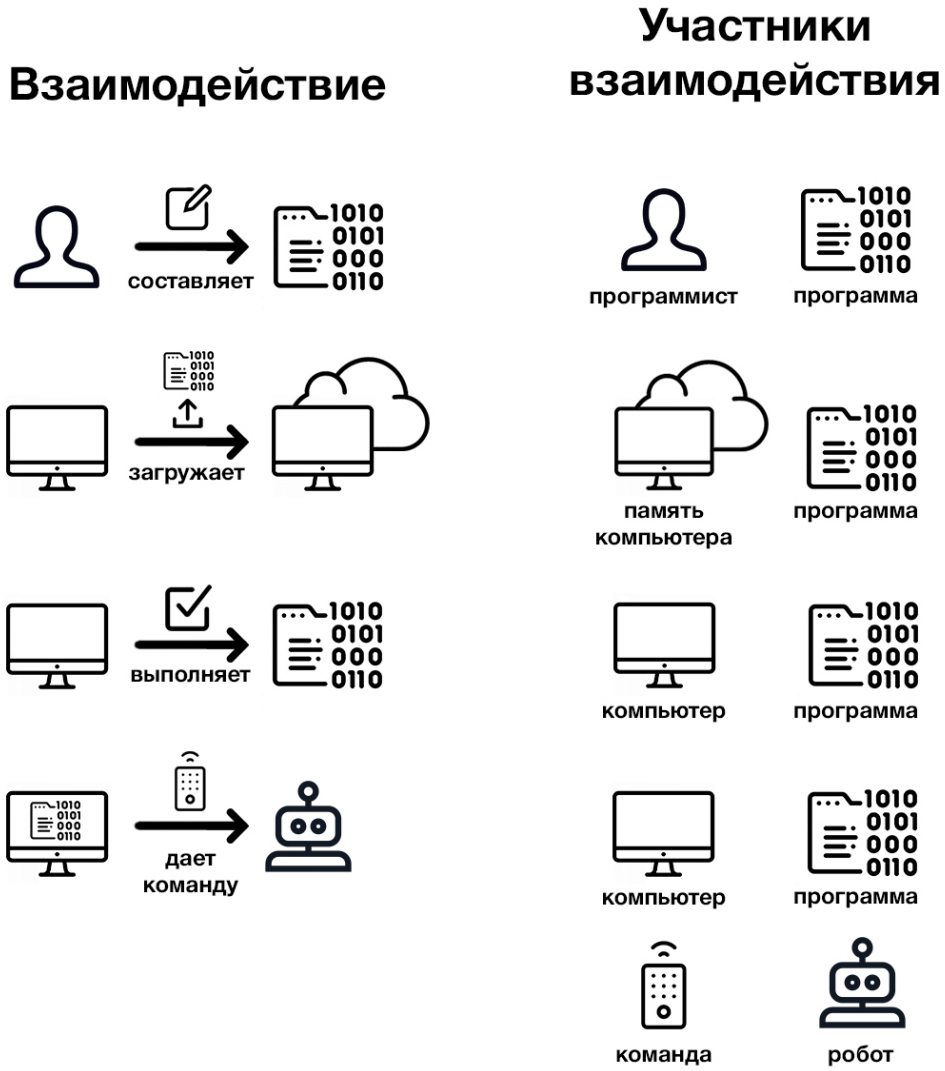


Рисунок 5 – Инфографика: участники и взаимодействия между ними.

При отсутствии обратной связи робот становится достаточно простым объектом для понимания даже для ребенка четвертого года жизни. Дети, наблюдая игрушечных роботов с пультовым управлением и освоив непосредственное управление таким роботом с помощью пульта, в состоянии перейти к программному управлению роботом. При исполнении команды роботом важно так же, как и описано в принципах программного управления, чтобы робот сообщал об успешном или неуспешном исполнении команды. Неуспешное исполнение команды, то есть *отказ* от ее исполнения, приводит компьютер к остановке выполнения всей программы, означая, что задача не решена и цель не достигнута.

Тогда программисту требуется вернуться к обдумыванию программы и поиску ошибок в ней. Отказ от исполнения, невозможность исполнения команды в реальном мире служит сильным средством предотвращения аварий при работе роботов по программе. Осознание этого факта очень важно для ученика, так как оно показывает ему, что ничего фатального не случилось и эту ошибку можно исправить.

Теоретически возможен не только представленный набор понятий и взаимодействия, но и другие термины, главное, чтобы и они могли пояснять принцип программного управления. В пропедевтическом курсе программирования уже на первых занятиях нужно добиться от учеников умения интуитивно понимать предложенную систему понятий, одновременно расширить свой словарный запас, что является не только основной целью данного процесса, но и дополнительной, так как детям необходимо научиться использовать эти термины в речи для выражения новых понятий.

Педагогическая практика показала, что предложенный педагогический набор из 12 понятий позволяет как учителю начальных классов, так и воспитателю детского сада, знакомящему детей с основами программирования, успешно справиться со всеми вышеперечисленными задачами при тридцатиминутных занятиях в группе и достичь твердого усвоения системы понятий к концу первого года обучения.

Описанная выше замкнутая система понятий принципа программного управления для ребенка-дошкольника, только начинающего осваивать алгоритмизацию, может стать первой в его жизни изученной системой научных понятий. В таком случае процесс обучения должен быть организован таким образом, чтобы система понятий была понятна абсолютно каждому ребенку.

Согласно Л.С. Выготскому, осознание любого общего принципа требует комплексного освоения ребенком некоторой системы научных понятий: «Научные понятия являются воротами, через которые осознанность входит в царство детских понятий» [30].

По Ж. Пиаже [186], для формирования у ребенка подлинно научного мышления недостаточно проведения физического эксперимента с запоминанием полученных результатов. Необходим логико-математический опыт, направленный на действия и операции, совершаемые ребенком с реальными предметами.

В.Б. Бетелин справедливо указывает, что принцип программного управления может быть понят и осознан ребенком только после усвоения изложенной выше достаточно сложной системы из 12 научных понятий [12].

Н.Н. Поддьяков в своих работах [195; 197] также уделят много внимания понятийному виду мышления ребенка, где важны понятия и логические конструкции. Н.Н. Поддьяков пишет: «Умение мысленно включать объект в различные системы объектов как принципиально новый способ экспериментирования с предметами (как реального, так и мысленного) значительно расширяло возможности познавательной деятельности дошкольников и оказывало существенное влияние на развитие их поисково-исследовательской деятельности» [196].

Осознанное усвоение этих понятий детьми станет возможным, только если будут предложены активности, позволяющие в деятельностно-игровой форме «вжиться» во все эти 12 понятий, «пропустить их через себя». Предложенный набор основных понятий был специально подобран так, чтобы максимально облегчить дошкольнику или младшему школьнику, вообще говоря, любому новичку освоение каждого из этих понятий и целиком всей системы понятий в деятельностно-игровой форме. На практике первые занятия курса были построены таким образом, чтобы каждый ребенок в группе смог проимитировать выполнение всех взаимодействий, выполняя поочередно роль робота, компьютера и программиста. Это позволило им поработать с материальным воплощением программы, составляя ее и загружая в память компьютера, а также столкнуться с естественными трудностями в случае нарушения правил составления программы. Перенос представления на реальные объекты является мощнейшей составляющей

формирования как алгоритмического мышления, так и ИКТ-компетенций для учеников разного возраста.

Интересно, что для старших школьников и студентов вузов изложение понятий в чисто вербальной форме оказывается менее запоминающимся и, соответственно, менее осознанным в противовес использованию элементов деятельностно-игровой формы аналогично подходу при обучении дошкольников. Таким образом, имеет место универсальный, доступный любому возрасту, подход к изложению основных принципов программного управления. Основные понятия программирования при этом являются необходимым фундаментом при формировании алгоритмического мышления.

Обращаясь к компетентностной модели человека 21 века, можно сказать, что в модели описанной А.Ю. Уваровым [241] алгоритмическое мышление, естественно, попадает в свой домен, связанный с развитием, занимая позицию в группе критического мышления, решения проблем, принятия решений, но являясь базой для других групп: творчества, инноваций и способности учиться, сокращая путь формирования и усиливая каждую из этих составляющих компетенций. При этом можно утверждать, что алгоритмическое мышление может послужить инструментом переноса освоенных знаний, навыков и умений между различными доменами, от мышления к работе, рабочим инструментам, жизни в современном мире и ответной, с обратной рефлексией. Подобное утверждение можно обосновать, ссылаясь на составные части алгоритмического мышления, такие как, например, планирование, декомпозиция проблемы, моделирование, построение процесса решения, целеустремленность в достижении успеха и пр.

Формирование алгоритмического мышления крайне важно и для критического, осознанно управляемого человеком мышления, так, например, процесс оптимизации решения задачи, безусловно, относится к критическому мышлению [240]. Это потребует построения и внедрения компетентностно-ориентированного индивидуализированного образовательного процесса, что

невозможно без систематического использования ИКТ-насыщенных цифровых образовательных сред.

Выводы главы 1

Исследования подходов методологии автоматизации образовательного процесса приводят к выводам, что безотносительно, какую учебную дисциплину планируется автоматизировать, образовательный процесс должен сохранить индивидуализацию обучения, доступность материала, управляемость учебного процесса, опору на родной язык и культуру, развитие смежных навыков и умений и т.п.

Выделены основные составные части алгоритмического мышления, как исторического наследника операционного мышления. Указано соотношение вычислительного и алгоритмического мышления. Предложена методика формирования алгоритмического мышления, как практики решения определенного набора алгоритмических задач.

Сформулированы основные понятия программирования, которые являются фундаментом формирования алгоритмического мышления в пропедевтических курсах по информатике и программированию. Постижение выделенных концепций возможно для детей дошкольного возраста и тех, кто только начинает изучение программировании, что предполагает возможность снижения возрастного порога для начала обучения в этой области.

ГЛАВА 2. МЕТОДОЛОГИЯ ПРЕПОДАВАНИЯ ОСНОВ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ С ИСПОЛЬЗОВАНИЕМ УЧЕБНЫХ ЯЗЫКОВ

2.1 Использование текстовых учебных языков программирования в преподавании основ алгоритмизации

Каковы сходства и различия между учебными языками программирования, профессиональными языками, живыми языками и родным языком с точки зрения их структуры, синтаксиса и семантики, а также какова их роль в формировании алгоритмического мышления и обучении программированию?

Программированию на машинных языках, языках низкого уровня, языках ассемблера обучаются на специальных курсах в университетах, на специальных практикумах [387; 140]. При этом несмотря на то, что ассемблер возник раньше многих других языков программирования [332], для его освоения предварительно требуется глубокое погружение в практику с языками программирования высокого уровня, например, C++, Java, Python и т.п. С другой стороны, в существующих парадигмах программирования начало изучения необходимо начинать с процедурных языков программирования, в том числе и для формирования алгоритмического мышления.

При отсутствии возможности изучать основы программирования с адекватным объемом практики, в том числе и из-за ограниченности доступа учеников и студентов к вычислительной технике [143], тем не менее производственные языки программирования, указанные выше, являются достаточно сложными для пропедевтики программирования. Стандартный подход построения курсов по программированию, свойственный образовательным программам вузов, предполагал выбор некоторой производственной системы программирования на профессиональном языке программирования высокого уровня, включающей изучение, описание этого языка и среду компиляции. При этом для освоения основ программирования подобные подходы были

неприемлемы для школы и тем более для знакомства дошкольников с программированием.

Естественным направлением движения для создания курсов для новичков является создание или выбор готовых учебных систем программирования, основанных на учебном или учебно-ориентированным языке программирования [63]. К числу подобных языков часто относят появившийся в шестидесятых годах прошлого столетия язык Лого (англ. Logo) с рисующей Черепашкой [364], учебный язык Паскаль (англ. Pascal), задуманный изначально его автором Никлаусом Виртом как чисто учебный язык программирования [23], но превратившийся впоследствии в производственный [356] и т.д.

Разумно предположить, что учебный язык программирования — это выбор одного из существующих диалектов какого-либо естественного языка в качестве учебного программирования. Однако естественный язык имеет ряд недостатков, главный из которых заключен в недостаточной формализации, что исключает простые решения при конструировании компиляторов, необходимых для перевода с языка программирования в машинный код.

Исследования [370; 372] выделяют основные различия между языком программирования и естественным человеческим языком. Последнему свойственны не только неоднозначность, но и выразительность, что позволяет человеку понять высказывание на естественном языке в контексте его опыта. Путь, пройденный от машинного кода и языков ассемблера к объектно-ориентированным языкам программирования, как постепенное приближение накопленного опыта программирования к человеческому пониманию, сохраняет управляемую однозначность перевода на машинный язык. При этом возможно двигаться в направлении создания формально-детерминированного диалекта некоего естественного языка.

Но даже при выборе профессиональным программистом языка, на котором будет решаться поставленная перед ним производственная задача, нужно сформулировать критерии выбора и подойти к данному вопросу с особым

вниманием. Так, в [320] опубликовано исследование, показывающее связь между производительностью (объемом кода, временем на разработку и т.п.) и используемым языком программирования. Более того, высокоуровневые объектно-ориентированные языки программирования требуют большего времени на разработку, чем процедурные языки, что замедляет процесс разработки и даже увеличивает количество ошибок в коде.

L.T.F. Gamut [326] утверждает, что естественный язык — это данность, в отличие от языков программирования, что позволяет разработчикам языков программирования проявлять большую гибкость, хотя они и могут (должны) включать определенные обороты естественного языка. Тем менее известны попытки создания идеального языка программирования, который был бы ближе к математическому, более формальному, нежели естественному языку [370; 386].

В исследовании, проведенном A. Stefik и S. Siebert в 2013 году (США), указывается на влияние синтаксиса текстового языка программирования на скорость его изучения новичками. В качестве теста использовались языки с традиционным C-подобным синтаксисом (Perl и Java), язык со случайно выбранными ключевыми словами (названный авторами Randomo) и Python (а также Ruby). К удивлению исследователей, языки с традиционным C-синтаксисом не вырвались вперед перед Randomo, напротив, Python и Ruby были более легко осваиваемыми новичками. Можно предположить, что языки программирования, близкие к естественным языкам, являются интуитивно-понятными для начинающих программистов. Хотя исследования проводились в англоязычной стране, но можно предположить, что близость к национальному языку будет существенным подспорьем для изучения программирования, что, естественно, должно учитываться при разработке национальных языков программирования.

Надо отметить, что попытки максимизации использования естественности в языках программирования были предприняты еще в прошлом веке [349], до активно внедряемых сейчас систем с генеративным искусственным интеллектом. Так, в SQL-запросах в системе ELFS отображался результат на естественном языке,

однако, многократное использование отрицания в ответах вносило путаницу. Подобные проблемы характерны и для русского языка, если в запросе чрезмерно использовать частицу «не». Однако современные решения с успехом обходят эту проблему, например, при анализе текстового ответа студента [2].

В английском языке, как в естественном, достаточно легко можно ввести ограничения, которые позволили в 1980 году без использования нейросетевых технологий выбирать некое подмножество языка в системе программирования «The Natural Language Computer» для ведения диалога с пользователем. Так, например, команды можно вводить только в повелительном наклонении и т.п. [293]. Столь небольшие ограничения создавали иллюзию общения на естественном языке.

Национальные расширения лексики языков программирования, такие как, например, ключевые слова, диагностика и т.д. являются важной вехой в преодолении барьеров к распространению языков программирования [344].

Вполне закономерно, что использование элементов естественного языка в интерфейсе прикладных программных продуктов было привлекательным для разработчиков. Проводились тестирования людей не знакомых с программированием на различных программных интерфейсах, чтобы оценить, например, возможность использования неформальных выражений языка. Попытки проводить аналогии с мышлением человека были использованы, например, в системе «Человеко-ориентированные достижения для начинающих разработчиков программного обеспечения» HANDS (англ. Human-centred Advances for the Novice Development of Software) [353; 360]. Визуальная среда HANDS ориентирована на новичков в программировании, но основа вычислений в ней - события. В HANDS вычисления проводит агент по имени Handy, манипулирующий набором карт, содержащих данные, а сами карты видимы, и все данные хранятся на этих картах, которые являются глобальными объектами для программы. Каждая карта имеет уникальное имя и неограниченный набор пар имя-значение, называемых свойствами. Сама программа хранится в «мыслях» Хэнди, который изображён в

виде животного, собаки, которая знает всего несколько команд, что подчеркивает ограниченность интеллекта системы в целом, а не человека или робота, интеллект которого значительно выше Рисунок 6.

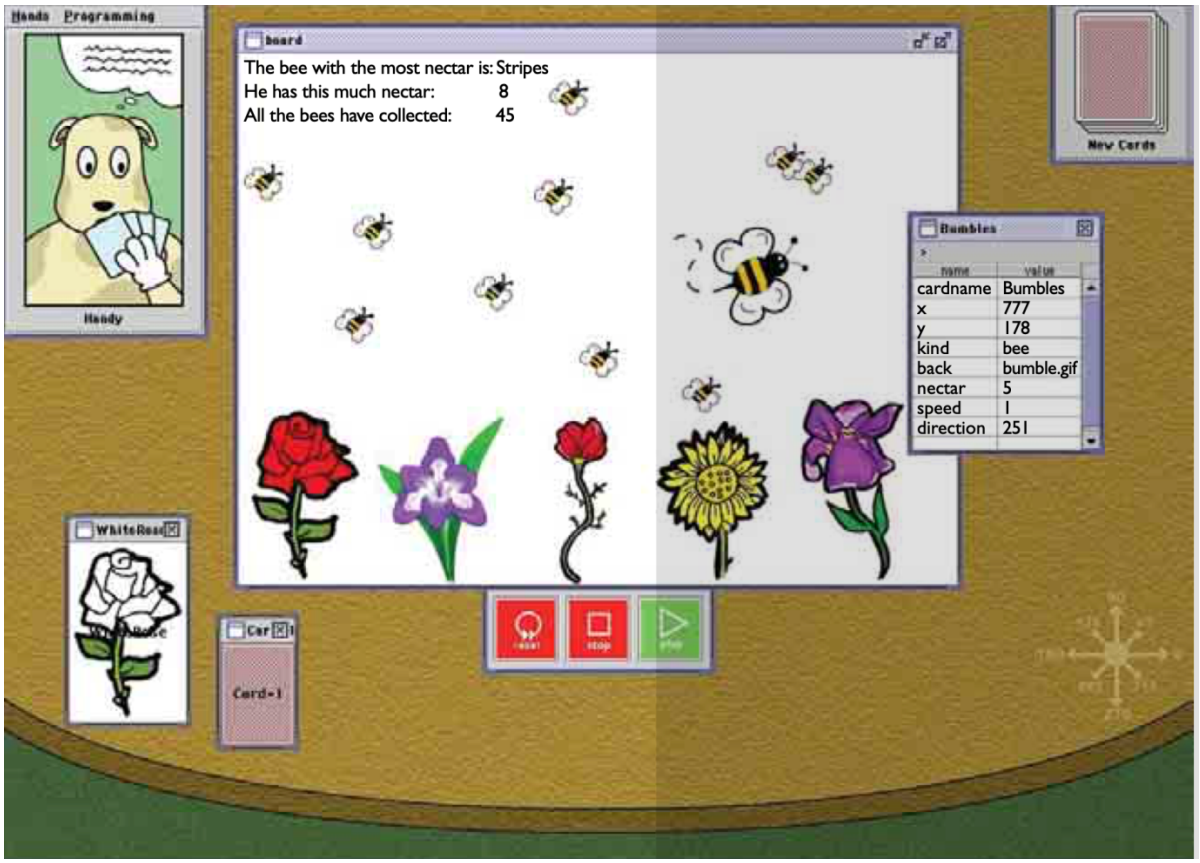


Рисунок 6 – Пример рабочего стола системы HANDS

В HANDS программа осуществляет доступ к данным в стиле баз данных и в котором все операторы могут применяться к одиночным элементам и(или) спискам. При этом запрос представлен в табличной форме, что кажется авторам более естественным, чем текстовый вариант. Язык программирования в системе HANDS поддерживает агрегатные операции, которые могут оперировать с результатами запросов, что в некоторых случаях избавляет от написания кода, содержащего итерации. Авторы пытались «сделать возможным для людей выражать свои идеи так же, как они думают о них» [353].

В ближайшем будущем возможно активное использование программирования на естественных языках. При этом какой язык из 5500 языков мира выбрать для программирования остается открытым. Возможно, выбор будет сделан в пользу шести официальных языков ООН: английский, арабский, испанский, китайский, русский и французский.

Проблема, возможно, является искусственной. Так, изобретение эсперанто [57] дало надежду на появление универсального языка общения, аналогично современной проблеме изобретения универсального языка программирования., Удобный для общения эсперанто, заимствовавший большинство корней из европейских языков, в том числе значительную часть из русского, был апостериорным языком с единообразными грамматическими правилами, что делало его доступным для понимания и лёгким для изучения. Л.М. Заменгоф, изобретая эсперанто как удобный язык для общения, планировал преодолеть отчуждение и разобщённость, царившие в 19 веке в окружавшем его мире. Совершенствуя свой язык, он собирался сделать его средством международного научного общения. Во многом его цель была достигнута. Даже первый в мире аппарат для машинного автоматического перевода был сделан с использованием грамматики эсперанто [334]. Интерес к эсперанто ослабел только с распространением компьютера. Поскольку есть сомневающиеся в успехе эсперанто, метафору поиска универсального языка программирования можно интерпретировать как в положительном, так и в отрицательном смысле, хотя в любом случае этот поиск унифицированного языка программирования имеет сходство с более ранними попытками создания универсального языка [327].

Некоторые зарубежные авторы, учитывая сложившиеся тенденции в глобальной коммуникации в мире, делают предположение, что владение английским языком предоставляет доступ ко «всем богатствам мира», которые могут быть недоступны из-за языковых барьеров [405]. Попытки последних лет разделить мир, вести изоляционистскую политику, проводимую рядом западных стран, не отменяет английский язык как язык международного сотрудничества в

различных областях, включая бизнес и образование. И если решать вопрос о использования единого языка в программировании, то английский язык может оказаться неплохим кандидатом.

Кроме того, английский язык лингвистически экономичен. Большинство глагольных форм идентичны. Любое действие, выражаемое глаголом, должно нести в себе одно из трёх: либо нейтральность, либо подчёркивание процесса, либо завершённость процесса. Существует даже заметное сходство между существительными, глаголами и прилагательными, так называемая «нулевая деривация», ведущая к полному заимствованию [339; 381]. Но лингвистические обоснования исключительной распространенности английского языка не являются однозначными. Так, в японском языке отсутствует спряжение глаголов по числам и лицам, что также говорит о некоторой экономичности японского языка. Если естественный язык использует одну и ту же форму как для настоящего, так и для будущего времени, то он может быть лучшим кандидатом на единственный естественный язык программирования.

Доминируя в программировании, английский язык становится de facto необходимым вторым (иногда третьим, но не менее востребованным) языком после родного языка в стране. Поскольку обучение основам программирования на текстовом языке невозможно без знаний основ английского, это требует для образования в России изучения данного иностранного языка в как можно более раннем возрасте. В противном случае важнейшую часть грамотности - программирование - ученик будет изучать не ранее с момента освоения английского языка. Уравнивая иностранный английский как предмет с предметом, программирование говорит лишь об искусственном языковом барьере, что может негативно сказаться (в том числе и в психологическом плане) на общем образовательном процессе в России как многонациональной стране, имеющей около 150 действующих языков, из них 37 имеют статус государственного.

Проблема заключается не только в английском языке как таковом, но и в ограничениях, которые накладывает его единственность, поскольку

культурологические и социальные барьеры, базирующиеся на английском языке, могут быть проблемой для иностранца даже при использовании Интернета [378].

Если автоматический перевод в 1990-х годах использовался компаниями-разработчиками программного обеспечения для выхода на международные рынки [274], то сейчас такая функция встроена в современные браузеры, и дискриминация по языковому принципу отходит на второй план.

Несмотря на то, что число носителей английского языка не превышает 380 миллионов [391], суммарное количество владеющих английским занимает первое место в мире. Наиболее близко к нему находится мандаринский китайский (включая стандартный, без других диалектов), который занимает твердое первое место по носителям языка. Суммарно английским не владеют более 80% населения Земли. Тем не менее число и распространение неанглоязычных языков программирования существенно уступает указанной выше статистике. Создается впечатление, что в научной среде владение английским языком рассматривается как преимущество, а разработка языков программирования на основе родных языков программиста воспринимается как менее достойное, что, тем не менее, указывает на тенденцию в мире в целом. Неудивительно, что такие популярные языки программирования, как Ruby (Япония) [252] и Pascal (Швейцария) [22], созданные на основе лексики английского языка, были разработаны не носителями английского. Ответ на вопрос, программируют ли в странах на английском вне англоязычного региона или вообще не знающие английского языка на англоязычных языках программирования, безусловно, положительный. При этом программисты в этих странах преодолевают определенные ментальные трудности при программировании и склонны к смешиванию родного и английского языков при написании кода, например, создавая версии языков с возможностью наименовать идентификаторы на родном языке или добавляя к коду пояснения, вновь используя родной язык. Те же, кто не использует национальную лексику в программировании, как говорят, программируют только на «чистом» английском, по-видимому, испытывают гордость своими лингвистическими победами [301].

Отсутствие популярности национальных языков программирования в англоязычной среде связано, по-видимому, с недостатком информации о них в мировом образовательном сообществе. Тем не менее такие языки существуют, и они включены в национальные образовательные программы с целью повышения эффективности учебного процесса, особенно на ранних стадиях знакомства с программированием, делая особый упор на формирование алгоритмического мышления в раннем возрасте.

Так, в Израиле с 2000-х годов идет школьный проект TEVEL, использующий основанный на иврите язык программирования, взявший в качестве основы Logo. Ученики старших классов, в том числе дети с ограниченными возможностями здоровья, начинают свой программистский путь с изучения этого языка [365].

В тестировании результатов школьников в освоении основ программирования в курсе «Разработка приложений в среде программирования» [280] используется web-приложение PAT (англ. Programming Adaptive Testing, Тестирование по Оценке Программирования), которое в том числе проверяет программы, написанные на греческом языке программирования Glossa (греческий перевод языка Pascal). Несмотря на то, что школьники пишут программы в среде программирования, авторы системы пытаются оценить навыки программирования большей частью на текстовых вопросах по знанию предмета (60%), чем по написанию кода (40%).

Отвечая на требования общества еще в 1980-х годах, которое уже тогда именовалось информационным, образовательная реформа 1985 года в СССР ввела в массовую общеобразовательную школу курс информатики. Курс получил официальное название – «Основы информатики и вычислительной техники», а место, занимаемое им в школьной программе, определялось двумя последними годами среднего школьного образования. Более того, курс столь же объективно именовался «безмашинным курсом информатики»: оснащение всей системы образования такой громадной державы, как СССР, было экономически невозможно при всей социальной и политической значимости задачи, поставленной перед

обществом. Жёсткие дискуссии об актуальности спуска нового курса существенно вниз, в начальные классы, показали безальтернативность раннего обучения информатике после появления Концепций информатизации школьного образования [95]. Основной задачей цифровой образовательной среды Е-практикум [19] была программная поддержка на школьных машинах такого безмашинного курса информатики, компьютерная реализация практикума по основам программирования на русскоязычном школьном алгоритмическом языке, который используется в образовании в России и сегодня.

Национальные языки программирования, синтаксис которых не основан на английском, являются репрезентативными в том смысле, что они широко использовались (и даже используются по настоящее время) в обучении. Вне зависимости от принадлежности к той или иной языковой семье естественных языков, от близких к английскому языку до далеких от него, нельзя голословно утверждать, что практика обучения основам программирования на близком к родному алгоритмическому языку или англоязычном языке программирования является негативной. Тем не менее эффект использования национальной лексики в учебных языках программирования вносит позитивную коннотацию для обучаемых на начальном этапе освоения сложного предмета программирования. Для примера будут рассмотрены история создания, использования в обучении, сильные и слабые стороны национальных языков программирования трех стран: Франция, Япония и Россия.

2.2 Преподавание информатики и программирования в Японии и язык программирования Kotodama

Само название языка программирования Котодама отражает культурное значение, которое строго отделяет его от языка с английским синтаксисом, лежащим в его основе, «Squeak» (звукоподражание, имитирующее звук, издаваемый мышью). «Котодама» (япон. 言霊) переводится как «дух языка» [404]

и связывается с ментальным отношением к высказыванию: что человек говорит, то и становится реальностью. Японская культура, имеющая глубокие корни в истории и в литературе, демонстрирует примеры, когда люди скрывали свое имя из страха, что произнесение врагом его имени может дать контроль над личностью. Лингвистически историческим примером важности принципа «Котодама» является тот факт, что во время войны Японии с США (Вторая мировая война) японцы избегали использования английского языка, поскольку считалось, что это дает силу врагу [345].

Использование английского языка Японии датируются еще 17 веком [339]. В современной жизни Японии латинский алфавит часто используется на вывесках и в рекламе, а англоязычная лексика представлена в национальных СМИ. После реформ 1989 и 2002 годов образование, включающее в сферу своих интересов английский язык, стало превалировать Японии. Однако уровень знания и использования английского языка не сильно изменился и до настоящего времени.

Один из трех алфавитов, используемых в современном японском языке, **катакана**, специально зарезервирован для иностранных слов. Большинство из них пришло из английского языка (например, メールアドレス; meeru adores: адрес электронной почты) или в основании их лежит английский язык, хотя налицо модификация, например, посредством сокращения (например, パソコン; pasocom : персональный компьютер). Также любопытно, что в 1870-х годах одним из немногих предложенных решений для урегулирования вопроса всеобщей грамотности в Японии было предложение полностью отменить японский язык и принять вместо него английский [382].

Однако японцы известны как нация, бережно сохраняющая свою культуру. Важно отметить, что с 1980-х годов наблюдается растущий негатив с «интернационализацией» Японии. Так, в 2006 году министр образования, культуры, спорта, науки и технологий Японии Кэндзи Косака в рамках внесения дискуссии о пересмотре Базового закона об образовании сделал заявление, что

английский язык не так важен для образования и что следует вновь сделать упор в обучении на национальный язык и патриотизм [40].

Информатика, являясь одной из важнейших дисциплин в национальном образовании Японии, активно внедряется в университетское и среднее школьное образование [281], становится обязательным в средних школах с 2003 года [357]. Предмет «Информация и компьютеры» содержит в основном темы по навыкам использования текстового редактора, Интернета и межличностных коммуникаций, использующих электронную почту. Лишь в последнее десятилетие начальные и средние школы стали вводить этот предмет, но без реального практического программирования.

Появление языка программирования с японским синтаксисом Kotodama (обычно указывают интегрированную в японскую версию Squeak 2005 года [300] и именуют как «Kotodama на Squeak») было направлено на понижение возраста первичного знакомства учащихся с программированием, а также на подготовку учащихся к возможной карьере в области информатики и вычислительной техники. Созданный группой под руководством Кена Окады (университет Кэйо, Токио), язык Kotodama получил поддержку Министерства экономики, торговли и промышленности Японии [351].

В 6-недельном курсе Kotodama занятие состоит из 10-минутной лекции и 80-ти минутной практики [348], и только после него студенты приступают к изучению производственного языка программирования Java. Таким образом, курс Kotodama являлся пропедевтикой программирования, но при этом вышел за рамки простого освоения навыков и умений и участвовал в формировании основ алгоритмического мышления у студентов и школьников, помогая в освоении абстракций математики и других STEM-дисциплин.

Учебник Мацузавы «Изучение логического мышления и программирования», разработанный специально для поддержки начального курса по Kotodama на Squeak, выделяет особое место для кооперативной работы над заданиями, а также отмечает важность навыков по коллективному принятию

решений. Учебник по форме представляет собой диалог между учеником и учителем, ставя во главу угла общение и коллективный труд.

Использование мультимедийных возможностей языка в Kotodama позволяет ученикам, только знакомящихся с программированием, выполнять сложные творческие задания. Часть заданий напрямую отнесены к культурным традициям Японии, географии и другим особенностям государства.

Гендерные особенности также учтены при составлении задач на Kotodama [281]. Дети разрабатывают проекты, например, музыкальную шкатулку, при этом интерес девочек к таким задачам значительно выше, так как мальчикам больше нравятся задания, например, с автомобилями. Для добавления интереса при использовании геймификации в обучении внимание мужской части класса в проекте удерживается с помощью добавления новых персонажей, например, покемона или игр «со стрельбой» по насекомым. При этом и девочки, и мальчики проявляют одинаковый интерес к игровым ситуациям при решении заданий. Последнее не является специфической чертой японского общества, а имеет равное отношение к большинству наций.

Kotodama на Squeak является свободно-распространяемым программным обеспечением и используется на различных видах вычислительной техники. Squeak представляет собой интерактивный фреймворк с компактной нотацией, поддерживающий параллельные процессы, и относится к классам строго объектно-ориентированных языков как диалект языка Smalltalk. Smalltalk — это, по своей сути, учебный объектно-ориентированный язык программирования с ограниченным набором нотаций, с обширной библиотекой классов и поддержкой графического пользовательского интерфейса [323]. Парадигма Model-View-Controller (MVC, Модель-Представление-Управление), схема разделения данных программы и логики на три отдельных компонента: модель, представление и управление (контроллер), реализованная в объектно-ориентированном языке Smalltalk [297], допускает независимую модификацию каждого компонента тройки. При этом Модель (Model) предоставляет данные вовне и реагирует на

команды Управления, возможно, изменяя своё состояние. Представление (View) отвечает за визуализацию данных, полученных от модели пользователю, то есть реагирует на изменения Модели, а Управление (Controller) интерпретирует действия пользователя, получая от последнего информацию о необходимости изменений Модели. На тех же принципах построен Squeak, первоначально разработанный Аланом Кеем (США), как среда для обучения программированию [283]. Однако отображение программы в английской нотации ограничивает возможности использования среды в Японии. Доработка версии Squeak eToy (названная Kotodama on Squeak) позволяет новичку писать программу фактически на японском языке [369].

По образу игры Маджонг [324] программа на Squeak состоит из последовательности плиток скрипта (*англ.*tile, тайлов), который может быть проинтерпретирован по правилам английского языка, Рисунок 7.

Рисунок 7 – Пример команды Squeak на английском языке

В примере английского скрипта-команды, если читать дословно, объекту Звезде (*англ.* Star) необходимо переместиться вперед (с помощью) пяти точек (на 5 пикселей). В этом выражении по правилам языка подлежащее идет первым, глагол идет следующим, а дополнение находится в конце.

В японском языке по правилам подлежащее идет первым, дополнение идет следующим, а глагол – последним. В «Kotodama on Squeak» английское слово заменяется соответствующим японским словом, но глагол стоит последним, Рисунок 8.

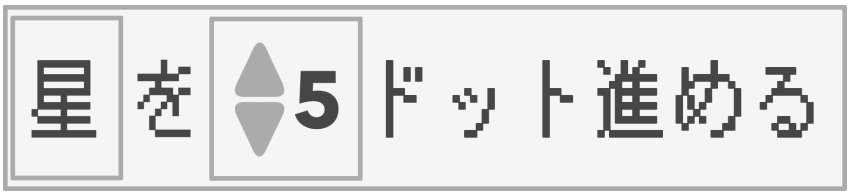


Рисунок 8 – Глагол находится в конце, следуя японскому порядку слов

Порядок слов «дополнение-глагол» такой же, как и в постфиксной нотации, что упрощает интерпретацию стековой машиной, когда действие попадает в стек последним. Как для японского ученика, так и для английского ученика измененная последовательность слов в предложении-команде, нарушающая правила нативного языка, сложна для интерпретации. Человеку, не знакомому с иероглифами, запись программы представляет некий знак или последовательность знаков, которую просто надо запомнить, придавая ей ассоциативный смысл в виде некоего связанного действия. На рисунке показан сценарий, где преподаватель должен объяснить, что его смысл— «Звезда вперед на 5», а не то, что видит ученик на картинке, - «Звезда 5 вперед на». Носитель языка, который умеет интерпретировать знаковую последовательность, будет чувствовать себя странно, увидев глагол, который стоит в конце, Рисунок 9.

Рисунок 9 – Переведенная английская команда рисунка 2 с тем же порядком слов

Это одна из иллюстраций сложности, с которой сталкиваются начинающие программисты, если изучение текстового языка программирования ведется не на национальном языке. M.Araki так формулируют свои требования к языку программирования для начинающих (относительно Kotodama): «Мы считаем, что среды программирования со множеством функций сбивают с толку учеников

начальной школы, впервые столкнувшихся с программированием на компьютере» [281].

Как и русский, японский язык не использует латинский алфавит. Ни один из трех алфавитов японского языка не основан на общем правиле европейских языков «одна фонема — одна морфема», то есть английский и японский языки принадлежат к совершенно разным языковым семьям.

Исследования показывают [305], что носители японского языка и носители английского языка используют различные принципы разбора языка, это означает, что требуется специальная методика обучения английскому языку японцев. Автоматизированный перевод с английского на японский был проблемой до использования нейросетевых технологий. Тем не менее работы по улучшению качества машинного перевода велись постоянно, например, предлагалось изменять порядок слов в предложении до начала перевода (размещение глагола последним, например, «John hit a ball» заменялось на «John a ball hit») [340].

Язык программирования Kotodama соответствовал сразу нескольким важным требованиям для национального проекта. В нем можно было составлять программы на нативном языке, что позволило учащимся сконцентрироваться на алгоритмической сложности задания. Также студенты в дружественной среде знакомились с объектно-ориентированной парадигмой. По мнению Y. Matsuzawa, «учащиеся могут использовать свой родной язык и концентрироваться на логическом мышлении» [351].

Вероятнее всего, отчуждение английского языка заложено глубоко в культурном слое японцев. Один из персонажей учебника по Kotodama на предложение выбрать алфавит выбирает хирагану (один из трех японских алфавитов), сообщая: «Извините, я не очень хорош в английском» [351].

В Kotodama on Squeak используется ряд терминов англоязычного происхождения, записанных алфавитом катакана, например:

- キーワード (kiiwoodo) - ключевое слово,
- アルゴリズム (arugurizumu) - алгоритм,

- ドロップ(doroppu) - капля,
- クリック(kurikku) - щелчок, клик,
- スケッチ(sukechchi) – эскиз.

Из семи ключевых слов, приведенных в первой главе учебника Y. Matsuzawa, шесть заимствованы из английского языка. В словосочетаниях также встречаются заимствования из английского. Так, backspace, кнопка «назад» переводится как バックバットン (bakkubottan) кнопка «назад». Вызывает удивление, что принятие англицизмов в языке программирования не является стимулом у студентов для изучения английской грамматики.

По мнению разработчиков, Kotodama стремится «продемонстрировать» правильность и полноту японского предложения, так как сокращения «уместны для профессионалов», а не для начинающих программистов [357]. Таким образом, составленная учебная программа может быть как правильный (с точки зрения языка) текст, но, с другой стороны, может быть выполнена как алгоритм, написанная на языке программирования. В Kotodama не используются математические символы, такие, например, как «=» и «+». Их место занято глаголами, такими как «заменить» и «добавить» или «увеличить». Из следующего примера видно, насколько близок Kotodama к естественному языку. Ниже приведен примеры программы по управлению Черепахой:

亀を新規作成して、亀太郎と名付ける。	Создайте новую черепаху и назовите ее Каметаро.
jに１を入れる。	Поставьте 1 в j.
j≦4である限り｛	Пока j⩽4 {
亀太郎を長さドット進める。	Продвигайте Каметаро на расстояние вперед.
亀太郎を90度右に回す。	Поверните Каметаро на 90 градусов вправо.
jにj+1を入れる。	Поставьте j+1 в j.
｝を繰り返す。	} Повторяйте.

Важно заметить, что для поддержания соответствия нативному японскому глагол «повторить» завершает цикл пока и находится в тексте программы далеко

от начала цикла. Однако такое расположение не создает путаницы при чтении и не противоречит общей лингвистической корректности языка Kotodama. Можно ожидать критики, что последующее изучение таких языков, как Java, потребует элементарного знания английского, однако программирование на Kotodama позволяет освоить основы алгоритмизации, не тратя драгоценное время на погружение в Java-среду и заучивание английских слов. С другой стороны, в качестве продолжения обучения программированию был бы разумным переход к программированию на Squeak, при котором ученики погружаются в уже знакомую среду программирования и ими освоены алгоритмы, то есть пройден этап чистой алгоритмической сложности. Освоение англоязычного языка программирования позволило естественным образом перейти в будущем к изучению производственных языков программирования. К сожалению, сборка программ из тайлов не позволяет избежать синтаксических ошибок в программе и в этом Kotodama фактически является обычным текстовым языком. Отсутствие математических символов в языке программирования не поможет студенту в изучении смежных дисциплин, таких как математика или другие STEM-предметы, однако позволит сосредоточиться на придумывании алгоритмов для решения заданий.

2.3 Подходы к преподаванию информатики во Франции и язык программирования LSE

Во Франции, как одной из передовых стран Европы в области образования, внедрение информатики в образовательную программу началось в 1970-х годах. Поэтому неудивительно, что термин «информатика» (фр. informatique), придуманный Филиппом Дрейфусом в 1962 году, стал названием дисциплины во Франции, как и в некоторых других странах, включая СССР [374]. Как новый предмет введение информатики в школы Франции предполагалось начать со старших классов, но еще в 1970-х годах высказывались идеи о необходимости начинать изучение этого предмета в 4-м классе школы, выделяя 100 часов

информатики [385]. Эксперимент начался с программы по подготовке педагогов и выделения вычислительной техники для 58 средних школ и лицеев для знакомства с приемами работы с вычислительной техникой и освоения языков программирования. Появились компьютерные клубы, привлекающие молодежь бесплатным доступом к вычислительной технике. При нехватке ЭВМ предмет велся и в безмашинном варианте, где школьники с большим энтузиазмом осваивали основные понятия программирования, составляли алгоритмы [284]. В зависимости от правительственных образовательных программ и финансирования менялось место информатики в школьном образовании. Первый эксперимент по внедрению информатики в школу фактически был прекращен в 1976 году именно из-за ограничения бюджета. Следующая амбициозная программа стартовала в 1978 году с плана подготовки тысяч учителей информатики [385]. Предмет предполагалось вводить начиная с начальной школы. В 1985 году был представлен план массового обучения информатике и компьютерам любого желающего, на успех которого негативно повлияла смена правительства республики. За долгий период содержание предмета неоднократно перерабатывалось, интерес к программированию у молодежи то быстро рос, то падал до минимального значения [285]. Во второй декаде 21 века в университетах Франции в рамках бакалаврских программ студенты могли выбрать изучение предмета, который стал называться Информатика и цифровые науки, ISN (фр. informatique et sciences du numérique) [325].

Во многом успехам первых лет информатики в школе обязаны появлению национального языка программирования. Язык LSE (фр. Langage symbolique d'enseignement, символическому учебному языку) был создан в 1967 году в Центре исчисления Высшей школы электричества и стал основным языком программирования в изучаемых курсах информатики [290]. В своей работе «Le système LSE» F. Bernard определяет годы с 1968 по 1976 как время основных разработок серии языков программирования LSx [288], когда экспериментальные языки LSD, LSG и LST уступили первенство LSD, переименованному

впоследствии в LSE. Однако первый массовый учебник об основах LSE появился только в 1975 году [374].

Для сохранения культуры и самобытности во Франции существует квота на использование национальных продуктов в кино, музыке и т.п., что позволяет, в том числе и образованию, быть отчасти независимым от тенденций, приходящих из-за рубежа (то есть англоязычного мира и культуры). К 1980-х годам росла популярность учебного языка программирования Logo. Появление IBM PC с языком Basic, который в англоязычном мире также пытались использовать в качестве учебного, оказывало давление на образовательную систему Франции. Язык программирования Pascal использовался в университетах республики.

Возможно принять решение, которое состоит в переводе ключевых слов англоязычных языков программирования на французский, но это могло бы привести к отрицательной тенденции выпадения программистских коллективов Франции из более крупного, фактически мирового сообщества пользователей Basic [282].

Если рассматривать язык в практической плоскости, то сначала выделяют его как средство общения между людьми. Программы также пишутся в том числе и для того, чтобы их читали другие программисты. Поэтому уход от национальных языков программирования к ставшим de facto стандартом англоязычных языков программирования является не только уничтожением некоего пласта культуры, но и может сократить размер сообщества ученых, исключив из него начинающих программистов. Сообщество LSE призывало сохранить национальный язык программирования: «Так же, как жители Квебека боролись за свое право говорить по-французски, нормально и для тех, кто выучил язык программирования защищать то, что является для них средством общения» [282].

Язык LSE поддерживался на уровне правительственных программ. В 1980-х Национальным центром педагогической документации (фр. Centre National de Documentation Pédagogique, CNDP) создается Отдел образовательного программного обеспечения (фр. Unité des logiciels éducatifs, ULE). За время,

прошедшее с начала проекта, был наработан большей объем прикладных библиотек, имеющих интерфейс только с LSE. Однако в CNDP также поддерживались разработки для других языков, таких как LISP и LOGO для начальных школ [288].

Проект по внедрению информатики в школы Франции не был только вопросом введения нового предмета в образовательный процесс. Информатика рассматривалась как дисциплина с большим междисциплинарным потенциалом, также формирующим общую и алгоритмическую культуру учащихся в приложении к другим академическим дисциплинам [385]. Информатика была призвана внедряться не как самостоятельный предмет, а как предмет внедряющийся в и через другие учебные дисциплины [284].

В последующие десятилетия были попытки введения некоторых инноваций в преподавание информатики в школах и вузах, однако все работы проводились в области скорее технической (оснащение школ современными компьютерами, освоение учащимися навыков работы с вычислительной техникой) нежели научной. Попытки вернуть информатику в русло формирования алгоритмической культуры введением нового предмета Информатика и цифровые науки предполагаю ввести предмет в школу, организовать преподавание, ориентированное на понимание и освоение информационных технологий, которое выйдет далеко за рамки простого использования компьютеров и программного обеспечения. В начальном образовании необходимо ввести основные понятия компьютерных технологий, разнообразить виды деятельности учеников в рамках не только одного предмета, но и других смежных академических и технологических дисциплин [325].

Язык программирования LSE прост и ориентирован на начинающих программистов. LSE это не только язык программирования, но и некоторая среда создания, редактирования, отладки и выполнения программ, написанных на этом языке. Появившиеся минимашины Mitra 15, а также 8-битные персональные микрокомпьютеры Logabax (LX500) французских ныне несуществующих

производителей позволяли студентам и школьникам составлять свои первые программы на национальном языке программирования [288]. Программа на LSE состояла из пронумерованных строк, в которой требовалось использовать некие договоренности, а именно прибавлять к номеру текущей строки число 10 при написании программы, чтобы упростить дальнейшее редактирование кода. В LSE программисту доступны числовые, логические и строковые типы. Как интерпретируемый язык программирования в среде LSE можно было производить отладку программ путем пошагового построчного выполнения. В язык были встроены ряд стандартных математических и других сервисных функций, такие как, например, квадратный корень, синус, текущая дата и время и т.п.

Для французских школьников и студентов программа, написанная на языке программирования LSE с «французскими» управляющими структурами, выглядела как привычный текст письма, или как элемент диалога [284; 288], то есть ассоциировалась с естественным для ученика языком.

LSE фактически стал «родным» языком программирования для французских учащихся. Педагоги утверждали, что нет необходимости учить английский язык, чтобы начать программировать [289]. Так, G.-L. Baron добавляет, что использование национальной лексики ограждает студентов и школьников от пагубного влияния иностранного языка на родной язык [284]. Так, например, разговорное выражение «je le loade» («я загружаю его»), привнесенное из информационных технологий, как понятно из английского языка, далеко от литературного французского.

В языке программирования LSE есть свой алфавит, словарь, синтаксис и семантика, то есть все атрибуты языка, что позволяет сравнивать его освоение с изучением полноценного иностранного языка. То есть можно достичь результата, когда любой ученик средней школы сможет без труда «говорить» на LSE не хуже, чем на английском и немецком [394]. Однако как язык программирования LSE, содержащий не менее 100 команд, был бы непонятен французу, если бы команды были на английском языке. Значение французскости LSE заключается в том, что он

отражает культурное явление, приписанное определенному контексту в истории французского образования.

Что сказано о родном языке, верно и для национального языка программирования: важно размышлять перед актом говорения, или, как звучит поговорка: «не подумав - не говори» [282]. Для языка программирования и самого процесса составления программы такое сравнение устанавливает связь между человеческим языком и языком программирования, а также добавляет культурное содержание языку программирования LSE.

Простой перевод Basic или Pascal на французский язык как потенциальную замену LSE не сделает их более французскими, более национальными, если вместо «soit» школьники станут писать «let» [282]. Это будет лишь попыткой внести поверхностные изменения, то есть если и имеющие отношение к культуре, то лишь к поверхностной.

Если рассматривать культуру в срезе достижения современных технологий в попытке интернационализировать языки программирования и программное обеспечение, то культура можно представить как вселенную символов, искусственно созданную людьми, наполненную значениями, убеждениями и ценностями, которые в том числе и обуславливают человеческое поведение [343]. При этом культура позволяет интерпретировать человеческие поступки, понять их целеполагание. Интернет, беспроводная связь, компьютеры, роботы, языки программирования и т.д. изменившие бытие человека, являются веским поводом для того, чтобы задаться вопросом, существует ли базовая универсальная основа всех культур мира, существуют ли общие межкультурные законы, архетипы культуры [318].

В современных исследованиях рассматривают целостный подход, когда нет ни универсальной культуры, ни универсальных законов и редукционный подход, который рассматривает культуру как символический дискурс, что есть, существуют некие универсальные законы, позволяющие сформировать универсальную культурную наднациональную картину [274].

Целостный подход различает два уровня культуры: глубинную и поверхностную. Глубинная культура включает в себя убеждения, идеи, язык, правила, знания, процедуры, нормы и проявляется в символах, артефактах и объектах, которые являются элементами поверхностной культуры. Значение символов и артефактов определяется глубинной культурой, и поэтому любой отрыв от последней приводит к потере предполагаемого смысла. Поэтому понимание ценностей, которые лежат в основе любого конкретного культурного проявления, необходимо для интерпретации этого проявления в культуре, в которой оно было создано. Это объясняет невозможность сравнения LSE и французского Basic.

Английский и французский языки принадлежат к различным группам, но обладают общей основой и имеют, в том числе в грамматике и лексике, много общего, то есть многие элементы синтаксиса англоязычного программирования легко переводятся на французский. Например, конструкция выбора если-то-иначе в английском варианте if…then…else переходит во французское выражение пословно si…alors…sinon.

Французские слова, как правило, длиннее английских аналогов. Чтобы исключить синтаксические ошибки при наборе, одновременно подчеркнув, что программа предназначается для выполнения компьютером, в LSE разрешено сокращение ключевых слов, например, bonjour и au revoir (начало и конец программы) можно сократить до однозначной интерпретации: bo* и au*.

Однако и LSE можно было обвинить в недостаточной патриотичности. Десятичная запятая заменилась на «англосаксонскую» точку, то есть вместо 3,14 нужно писать 3.14 [394]. Неправильно пишутся также некоторые ключевые слова языка, которые содержат символы, характерные для французского алфавита, например, chaîne (строка) в LSE заменено на chaine, а jusqu’à (до тех пор, пока) на jusqua.

Имеющий свои особенности французский язык иногда вносит неоднозначность. Например, цикл faire (делать) включает слово pas (шаг) и выглядит так:

faire…pour…pas…jusqua (делать… для… шагом… до тех пор, пока).

В английском варианте это бы выглядело похоже:

do…for…step…until (делать… для… шагом… до тех пор, пока))

Однако **pas** также является частицей отрицания (не) во французском языке. Таким образом, может возникнуть нежелательная ассоциация со структурой **until**, которую на французском LSE можно прочитать как «пока не».

Эти существенные грамматические различия в языках делали сложными программы автоматического перевода с французского на английский или обратно. Появление программ перевода, основанных на искусственном интеллекте, практически полностью сняло большинство проблем.

В 1983 году Жак Арсак представил новую версию языка LSE83, исправив некоторые рудименты времени создания языка программирования LSE [282]:

- нумерацию строк;
- наличие оператора перехода на строку ALLER EN (GO TO);
- смешивание функций и подпрограмм в одной конструкции PROCEDURE;
- возможность прямого входа в процедуру через команду перехода на строку ALLER EN (GO TO);
- однострочная конструкция SI... ALORS... SINON (ЕСЛИ... ТО... ИНАЧЕ).

Естественно, что такой императивный язык неструктурного программирования уже в 1980-х годах требовал или новой, адекватной времени версии, или ведения сложной дисциплины написания программы на LSE, чтобы быть ближе к структурному программному коду. Однако FORTRAN, который на начальном этапе был далек от современных тенденций в технологии программирования и как язык программирования возник значительно раньше LSE, но, адаптируясь к требованиям времени, не потерял свой актуальности и в настоящее время [45].

Язык программирования LSE и его применение сыграли важную роль во французском образовании, особенно в момент первого становления информатики как предмета в школах и университетах Франции. Национальный синтаксис языка программирования LSE для франкоговорящих детей был простым и похожим на естественный язык, на котором они говорили. Технические и(или) лингвистические ограничения показывают несостоятельность пропедевтики программирования на одном из традиционных текстовых императивных языков на основе английского языка. Однако ни автору LSE83 [282], ни его коллегам, усовершенствовавшим версию до LSE-2000 [354], так и не удалось добиться поставленной цели – распространения LSE в системе образования Франции и широкого использования в начальной школе страны франко-лексического языка программирования. Важные вопросы национального образования в области информатики и программирования не решены до настоящего времени. Нет сложившейся устойчивой программы, объединяющей все школы республики. Интерес школьников и студентов к этой дисциплине - информатике - не показывает взрывного роста заинтересованности, мировые тенденции в цифровизации в проекте на Францию также не детерминированы [325]. Хотя в 2022-2023 гг. специальность «Цифровые и компьютерные науки» в вузах республики выбирает все больший процент студентов, однако она остается не столь популярной, как математика или физика [377].

2.4 Преподавание информатики в СССР

Педагогические эксперименты по обучению школьников программированию в СССР начались в 1950х-1960х годах с момента появления вычислительных центров и электронно-вычислительных машин. Знакомство школьников с программированием происходило задолго до введения школьного курса информатики, например, в Институте математики СО АН СССР под руководством А.П. Ершова [69]. Аналогичные тенденции в образовании происходили и в развитых зарубежных странах. Так, хотя в США не было

общенациональных программ, таких как советские физико-математические школы, но у педагогического сообщества США уже появлялся интерес к вычислительной технике [61]. В 1967 г. С. Пейперт, сотрудник Массачусетского технологического института, разработал учебный язык Logo для обучения основам программирования детей школьного возраста [364].

Когда подготавливалась первая в Советском Союзе реформа школьного образования 1985 года [166; 224], предусматривавшая введение школьного курса информатики, среди специалистов этой области – программистов, педагогов, администраторов системы образования – шли дискуссии об обосновании места этой науки в школе. Еще в 1970-х и ранее советские ученые В.М. Глушков, А.И. Берг, А.И. Китов, А.А. Ляпунов и др. предлагали активно развивать и использовать вычислительную технику в народном хозяйстве [39; 10]. Электронно-вычислительные машины, как универсальный инструмент обработки информации, уже тогда был востребован. Но компьютер – эффективная и производительная, но дорогая машина, особенно при отсутствии массового производства ЭВМ, значит, требовалось сделать так, чтобы она была максимально загружена работой и не простаивала, а была эффективно использована [236]. А это означает, что для успешного развития народного хозяйства необходимо производство и повсеместное внедрение компьютеров и, следовательно, опережающий рост числа работников, способных эффективно использовать вычислительную технику, то есть необходима подготовка инженеров, разработчиков и, конечно, программистов, а значит, требуется массово готовить студентов (и школьников) к изучению новой дисциплины.

Эти аргументы, лежавшие на поверхности, не отражали, однако, того растущего глубокого влияния, которое компьютер оказывал на жизнь человека 20 века. Обучение программированию в 1980-х годах велось в безмашинном варианте. И этому были объективные причины, связанные с ограниченностью доступа учеников и студентов к вычислительной технике [63]. С другой стороны, производственные языки программирования были достаточно сложными для

новичков, особенно, когда речь шла об обучении детей основам информатики в школе. А.П. Ершов в 1981 году писал: «… вопрос о том, как учить детей способности планировать свои действия и их последствия, какая операционная обстановка при этом нужна, очень далек от тех методических альтернатив, которые мы обсуждаем, например, при профессиональном обучении программированию» [67].

Высказывание А.П. Ершова касалось массового обучения новичков основам алгоритмизации в качестве базового курса для старшей школы еще в СССР в 80-х годах [60; 62; 75], однако эту мысль можно расширить и для пропедевтических университетских курсов по программированию.

При этом в рамках курса требовалось не только знакомить обучающихся с основами программирования, но и формировать конкретные навыки по использованию вычислительной техники, которая еще не имела массового распространения в СССР. Школьники и студенты прежде всего обучались пользованию ЭВМ для выполнения различных задач, преимущественно вычислительного характера. Также они знакомились с устройствами интерактивного ввода информации, например, с манипулятором мышь и клавиатурой, осваивали основные навыки по редактированию программ и обычных текстов, работы с электронными таблицами и графическими редакторами и пр.

Отсутствие подобных навыков не позволяло эффективно заниматься программированием, так как к сложности придумывания алгоритма и кодирования на некотором языке программирования добавлялось отсутствие навыков ввода информации в ЭВМ. Фактически вместо изучения и придумывания алгоритмов, а также навыков и умений составлять соответствующие им программы, обучаемый основное время тратил на непроизводительные расходы по вводу и редактированию информации. Так, на механико-математическом факультете Московского государственного университета имени М.В. Ломоносова на первых практических занятиях студенты обучались быстрому набору текстов в специальном тренажере – игре «В слова», позже переименованной в Клавир [7]. В

этом интерактивном тренажере-практикуме на экране появлялись слова из набора управляющих конструкций некоего алгоритмического языка, и игрок должен был быстро набирать их, одновременно перемещая курсор по полю, «убегая» от догоняющих его слов. При этом не только формировались навыки работы с клавиатурой компьютера или терминала, но и осуществлялось знакомство с терминами, необходимыми для написания учебных программ.

Механизм реализации учебного курса по программированию должен предусматривать принцип «быстрого старта», когда первую содержательную программу студент или школьник сможет написать на начальных занятиях курса. Сложность профессиональных систем и языков программирования возвращает к подходу к принципу программного управления и исполнителям [124].

Важно отметить, что программные исполнители не очень сложны для понимания новичкам, первые учебные программы, как правило, короткие, поскольку, в отличие от промышленных языков программирования, учебные языки управления программными исполнителями используются с ограниченной целью – познакомить студента и школьника с базовым набором понятий и сформировать конкретные навыки [178].

Г.А. Звенигородский обучение школьников начинал с основных понятий программирования, проводя аналогию с видами предложений русского языка: «…По цели высказывания различаются повествовательные, вопросительные и побудительные предложения. Предложения, в которых мы сообщаем о чем-либо, называются повествовательными; предложения, в которых содержится вопрос, — вопросительными; предложения, в которых мы побуждаем кого-либо к действию, приказываем, просим, — побудительными. На уроках программирования в этой книге мы будем интересоваться видами предложений гораздо чаще, чем интересуются ими в школе на уроках русского языка. Поэтому вместо длинных названий (повествовательное предложение и так далее) будем пользоваться более короткими: утверждение, вопрос и предписание» [74].

При этом для каждого умения или навыка возможно проектирование уникального исполнителя. Так, для освоения понятия цикла в учебной системе «Школьница» Г.А. Звенигородского использовался исполнитель, который назывался «ТОМ_СОЙЕР» [73]. Функционал этого исполнителя включал «покраску» досок забора, при этом он мог «закрасить» не только одну доску, но и целую последовательность рядом расположенных досок, подобно одноименному персонажу Марка Твена, который не красил доски сам, а давал команды своим приятелям.

В другой программно-методической системе «Роботландия» [55] для обучения детей понятиям ветвления и навыкам записи условных команд использовался программный исполнитель «Пиастры», где жадный арабский купец пересчитывает пиастры, чтобы за наименьшее количество взвешиваний на весах обнаружить фальшивую монету среди кучи монет. Другой исполнитель – «Машинист» моделирует движения локомотива на железнодорожных ветках и тупиках сортировочной станции. Методическая задача использования программного исполнителя «Машинист» состоит в ознакомлении учеников с понятием вспомогательного алгоритма (подпрограмма, функция) и обучении их практическому применению этого понятия при решении задач в рамках пропедевтического курса программирования. При этом использование практикумов в обучении программированию дает больший эффект в понимании научных понятий и принципов, нежели рисунок на доске, изображающий механизм работы программного стека, который хоть и может являться некоторой моделью, но все равно не существенно упрощает для ученика понимание. Использование программных исполнителей позволяет «пощупать информатику руками» [54; 213].

Подобный унифицированный подход предполагает формулировку некоего стандарта к инструментарию курсов информатики и программирования, который использован как единый дидактический инструмент на всех этапах непрерывного курса. Как элемент единой цепочки инструментальных педагогических программных средств на этапе пропедевтического курса — это совокупность

исполнителей, каждый из которых отрабатывает некоторые конкретные умения и навыки работы с управляющими структурами и структурами данных [138]. Инвариантные элементы исполнителей, постепенно расширяющие изучаемые объекты и формирующие новые умения и навыки, подводят к необходимости наличия учебного языка управления исполнителями. От этого этапа управления множеством отдельных исполнителей простыми языковыми средствами уже можно делать переход к учебно-ориентированному языку программирования, который в связи с определенными требованиями последующего перехода к производственным языкам программирования также является текстовым [131].

Если национальные разработки учебных языков программирования не всегда получали массового распространения в народном образовании стран-разработчиц, то, напротив, отечественный учебный язык, называемый еще *школьным алгоритмическим языком*, *языком записи алгоритмов*, получил широкое распространение сначала в СССР с введением обязательного предмета Информатика и ИКТ в 1985 году, а потом и в России, перешагнул 40-летний юбилей и вошел в список языков программирования, рекомендованных для прохождения ГИА в основной школе [246]. Этот успех во многом связан с использованием национальной кириллической лексики. Однако этим не исчерпываются его достоинства.

2.5 Цифровая образовательная среда КуМир

Естественным педагогическим программным расширением языка записи алгоритмов является цифровая образовательная среда (ЦОС) КуМир (Комплект Учебных Миров), – практикум по основам программирования, который был создан и поддержан на всех видах вычислительной техники. Она служит компьютерной поддержкой пробного учебника «Основы информатики и вычислительной техники» [64]. В первом массовом учебнике по обязательному школьному курсу «Информатика и ИКТ» для записи алгоритмов использовалась привычная для нашей страны русскоязычная нотация управляющих конструкций.

ЦОС КуМир позволила ученикам не просто записывать программы в тетрадях, но и загружать их в компьютер, отлаживать и получать результат. В 90-х годах прошлого столетия ЦОС КуМир оказалась передовой программной системой, построенной на принципах редактирования-компиляции, однако и в настоящее время ЦОС КуМир не потерял своих учебно-производственных позиций. ЦОС КуМир отслеживает сделанные учеником синтаксические ошибки при редактировании программы, а также следит за корректностью работы со структурами данных в режиме выполнения. Фактически в ЦОС КуМир интегрирован отладчик, который как в непрерывном, так и в пошаговом режиме выполнения визуализирует «на полях» программы изменения значений величин, логических выражений и т.п. Рисунок 10 и Рисунок 11.

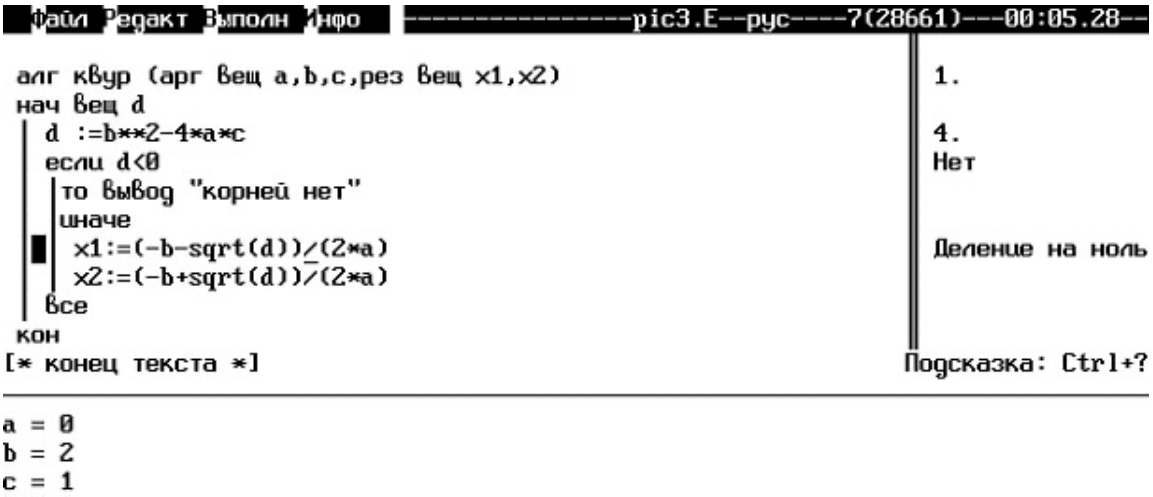


Рисунок 10 – Примеры выполнения алгоритма нахождения корней квадратного уравнения с возникновением ошибки в ранних версиях системы КуМир

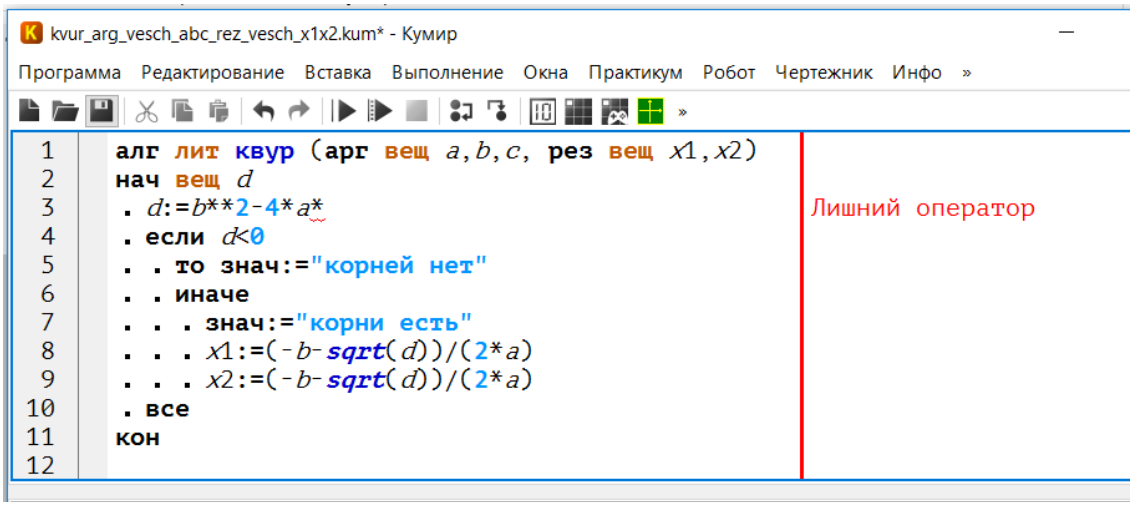


Рисунок 11 – Пример диагностики синтаксической ошибки на полях до начала выполнения алгоритма в процессе набора текста в ЦОС КуМир

При этом в ЦОС КуМир поддержан объектно-ориентированный подход при помощи конструкции Исполнитель. Так, Исполнитель объединяет в себе алгоритмы (процедуры) с общей функциональностью (Исполнитель "Математика", Исполнитель "Графика" и т.д.), скрывает внутренние (служебные) переменные и функции и показывает наружу лишь подмножество пользовательских переменных и функций (интерфейс). Использование внешних программных исполнителей расширяет функциональность учебно-педагогической цифровой программной среды.

Поскольку в школьном алгоритмическом языке служебные слова и управляющие конструкции являются русскоязычными словами или их сокращениями, то такой учебный язык программирования является более легким в освоении для учеников, чем англоязычные языки. Использование кириллицы требует внимательности при составлении программ, поскольку требуется быстрое переключение между языками (русским и английским). В ЦОС КуМир можно использовать не только выбор языка мышью или аккордные нажатия клавиш, но и временную смену языка при удержании языковой клавиши, что обеспечивает временную смену раскладки клавиатуры, например, с латиницы на кириллицу. Это

важно, так как по традиции, как на уроках математики, так и в условиях задач ЕГЭ по программированию, величины (параметры и неизвестные) обозначают латинскими буквами, а также встроенные математические функции имеют латинские названия.

Для ускорения составления программы можно использовать вставку целиком управляющих конструкций. Сокрытие операций ввода-вывода ускоряет отладку вспомогательных алгоритмов, когда при выполнении алгоритма аргументы автоматически запрашиваются, а результат по окончании работы алгоритмов выведен на экран, Рисунок 12.

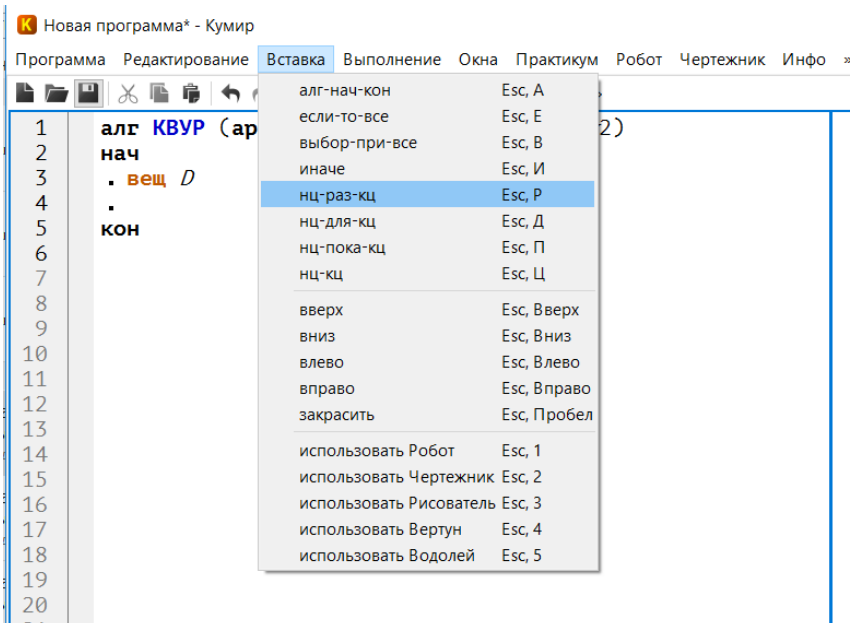


Рисунок 12 – Вставка конструкции целиком в ЦОС КуМир

Важной особенностью языка записи алгоритмов является возможность включать пробелы в имена алгоритмов и величин, а также вставлять отрицание «НЕ» внутрь такого имени. Это легко понять на примере, Рисунок 13.

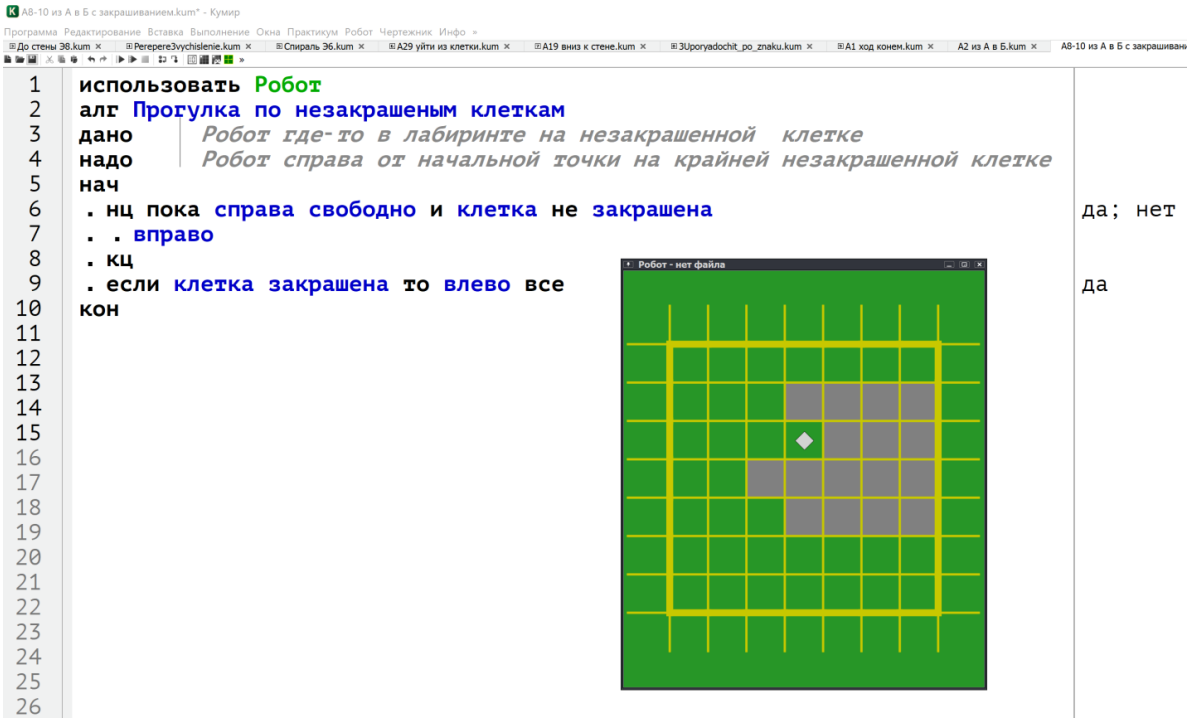


Рисунок 13 – Алгоритм «Прогулка по незакрашенным клеткам» вправо в ЦОС КуМир

В цикле используются составное условие **справа свободно** и отрицание условия **клетка закрашена**. Понимая, что частица **не** отрицание ко всему условию, разумно ожидать, что, как и в абсолютном большинстве производственных языков программирования, отрицание **не** ставится перед вторым условием. Тогда, прочитав “справа свободно и не клетка закрашена”, у учащегося может возникнуть вопрос: а что закрашено, если «не клетка». Для сохранения смыслового значения высказывания на родном языке требуется перенести частицу «не» внутрь названия алгоритма «клетка закрашена». В таком случае не теряется не только логика условия цикла, но и смысл словесного выражения в языке: «справа свободно и клетка не закрашена».

Другая, достаточно редкая для производственных языков программирования возможность синтаксиса школьного алгоритмического языка состоит в использовании составных имен алгоритмов и величин. Обычно требуется, чтобы переменные в языках программирования именовались единым словом, что сильно

ограничивает смысловое содержание для переменных и алгоритмов. Так, алгоритм **Прогулка по незакрашенным клеткам** не сокращается до одного слова **Прогулка**, так как может быть целое множество «прогулок» с различным содержанием, и программист, на мгновение оторвавшись от текста программы, может забыть, «про что» этот алгоритм. Обычно слова, именующие переменные, склеивают в одно, убирая пробелы и заменяя строчные на прописные буквы в начале каждого слова или пробелы на подчеркивание:

1. **ПрогулкаПоНезакрашеннымКлеткам**
2. **Прогулка_по_незакрашенным_клеткам**

Ни один из этих вариантов не соответствует правилам русского языка, хотя WalkingThroughUnpaintedCells на английском языке или Marcher_à_travers des_cellules_non_peintes на французском выглядят вполне читаемыми для русскоговорящего ученика. Это связано с тем, что российские школьники и студенты не являются носителями этих языков и ментально для них это единое выражение, обозначающее некоторый алгоритм, который надо просто запомнить и, вновь увидев, попытаться вспомнить, чему оно соответствует. В этом у национальной цифровой образовательной системы для школьного алгоритмического языка в России большое преимущество.

Создание и широкое распространение ЦОС КуМир вместе с улучшением технического оснащения системы школьного образования и домашних хозяйств России позволяют проводить занятия по программированию в практической форме. Напротив, использование англоязычных и профессиональных систем программирования на уроках по информатике не обеспечивает реального освоения в средней школе элементов программирования, включающего написание и отладку нескольких десятков простейших учебных программ. Основные причины этого кроются в высоких затратах на освоение профессиональных систем программирования новичками. Так, для практического освоения профессиональной системы программирования в курсе информатики основной школы просто недостаточно учебных часов. Кроме того, существует высокая

трудоемкость ручной проверки учебных программ, созданных учениками, то есть отсутствие полноценного контроля классных и домашних работ учеников.

Эти задачи успешно решаются с использованием учебной системы программирования ЦОС КуМир, которая обеспечивает, в том числе и возможность автоматизации проверки практических заданий по программированию.

Для преподавателя существует возможность добавить к программе ученика тестирующий алгоритм, который скрыт от ученика, но может быть выполнен по желанию школьника, Рисунок 14, а сам тестирующий алгоритм приведен в таблице, Таблица 4.

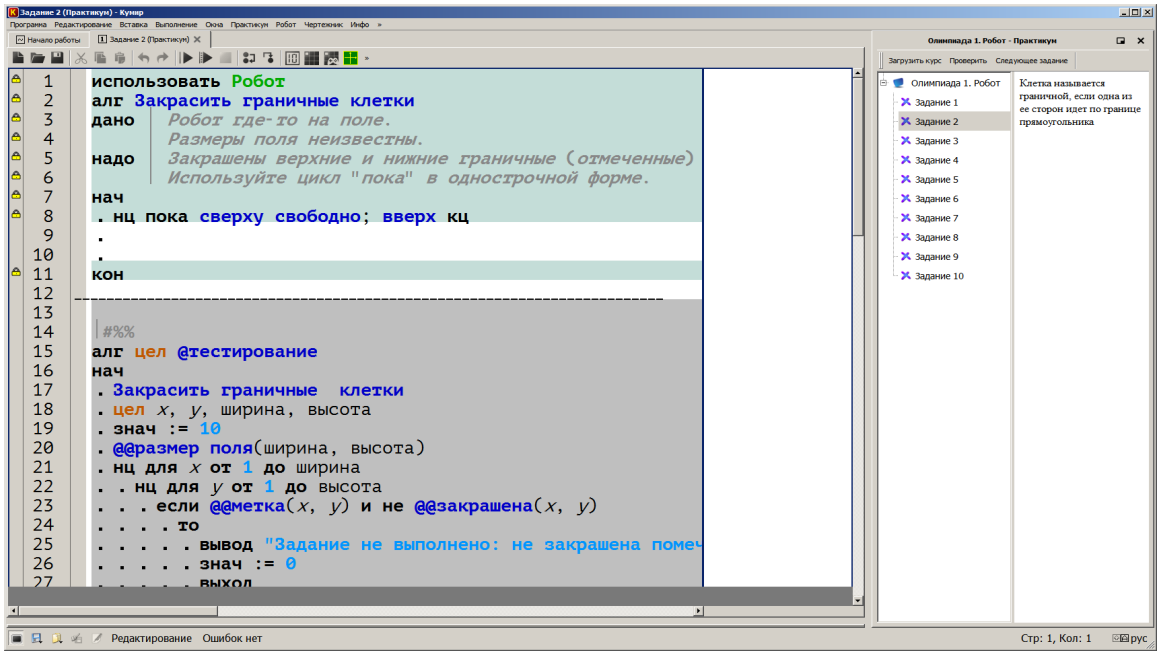


Рисунок 14 – Задача «Закрасить граничные клетки» со скрытым тестирующим алгоритмом

Этот алгоритм имеет уникальное имя, которое не может использовать ученик. Тестирующий алгоритм вызывает программу ученика, а затем проверяет, что все помеченные клетки лабиринта закрашены, а непомеченные клетки, напротив, не закрашены. В случае неуспеха алгоритм-функция **@тестирование** возвращает результат **знач=0**, а в случае прохождения теста **знач** устанавливается

равным 10 и сообщается, что «Задание выполнено успешно». Последнее нужно для оценки заданий, объединенных в практикумы.

Таблица 4 – Вариант тестирующего алгоритма для задачи «Закрасить граничные клетки»

```
алг цел @тестирование
нач
. Закрасить граничные  клетки
. цел x, y, ширина, высота
. знач := 10
. @@размер поля(ширина, высота)
. нц для x от 1 до ширина
. . нц для y от 1 до высота
. . . если @@метка(x, y) и не @@закрашена(x, y)
. . . . то
. . . . . вывод "Задание не выполнено: не закрашена помеченная клетка", нс
. . . . . знач := 0
. . . . . выход
. . . все
. . . если @@закрашена(x, y) и не @@метка(x, y)
. . . . то
. . . . . вывод "Задание не выполнено: закрашена непомеченная клетка", нс
. . . . . знач := 0
. . . . . выход
. . . все
. . кц
. . если знач = 0
. . . то
. . . . выход
. . все
. кц
. если знач = 10 то
. . . вывод "Задание выполнено успешно"
. все
кон
```

Замкнутость и изолированность систем программирования создает определенные трудности при попытке использовать подобные решения совместно с цифровыми платформами, в том числе LMS (англ.Learn Management System) [228]. В таком случае от ЦОС требуется предоставление интерфейса, возможно, консольного, для автоматизации процесса проверки ученических программ сторонними средствами, когда сама ЦОС, ориентированная на взаимодействие с пользователем, не может быть использована для проверки большого количества программ без участия человека-оператора.

Одним из важных свойств ЦОС Кумир является возможность создания и использования специальных практикумов по программированию с автоматической проверкой созданных учащимися программ. Такая проверка возможна как на компьютере ученика, так и на удаленном сервере в рамках цифровой образовательной платформы, которая проверяет и учитывает результат выполнения задания. Педагог — разработчик практикума по теме — подготавливает для ученика набор заданий, каждое из которых представляет собой заготовку КуМир-программы, список необходимых исполнителей и набор обстановок для исполнителей, на которых будет производиться проверка [202].

Учитель для каждого задания готовит проверяющую программу, которая предназначена для проверки правильности выполнения задания и является скрытой частью задания. Ученик работает над полученной из практикума задачей в ЦОС КуМир, как обычно, редактируя программу и отлаживая ее. По готовности задания ученик может проверить его самостоятельно, выполнив тестирование на своем компьютере, а также автоматически передав программу или весь практикум (все задания, которые ученик решил или пробовал решать в данном практикуме, сохраняются в виде единого файла - “Файла рабочей тетради”) в цифровую образовательную платформу на проверку.

Используя ЦОС «КуМир» и встроенные в ЦОС практикумы с автоматической проверкой, в рамках курса информатики в основной школе на практическое программирование можно потратить до 35 часов, то есть около трети всех часов 105-часового годового курса. Это возможно, если педагоги подготовлены и сопровождены методически, а также, если школа оснащена вычислительной техникой. За указанное время можно освоить основы процедурного программирования и благодаря практике научить школьников решать сложные задачи из кодификатора ЕГЭ, а не только простейшие задачи по обработке данных [58; 89].

Наличие доступа у школьников в образовательной организации к учебным или производственным системам программирования на одном из перечисленных в

ФОП ООО России языков программирования: Паскаль (Pascal), Си (C++) [235], Питон (Python) и т.д. не решает задачу, так как необходимо расширение выбранной системы программирования до учебных практикумов, позволяющих при наличии или отсутствии возможности высокоскоростного подключения к сети Интернет проводить автоматизированную проверку правильности выполнения заданий. При этом подобные практикумы должны быть сопровождены не только технически, но и методически, иметь наполнение сотни задач, включая перечисленные в кодификаторе ЕГЭ, и множество более простых заданий, решающих некоторые фрагменты алгоритмически-сложных задач.

Если использовать ЦОС КуМир и школьный алгоритмический язык, то это позволит не только автоматизировать проверку задач, но и иметь автономную систему подготовки практикумов. Надо отметить, что авторы российских учебников по информатике в качестве методического обеспечения часто используют практикумы ЦОС КуМир [200].

В 7-х математических классов школы №179 МИОО с 2008 года курс программирования на производственном языке программирования C++ предварялся вводным годовым курсом алгоритмизации с целью освоения школьниками основных принципов построения алгоритмов и основных алгоритмических конструкций процедурных языков программирования на примере школьного алгоритмического языка в ЦОС КуМир [88]. Двухлетний педагогический опыт показал, что переход в следующем классе к изучению C++ практически не вызывал затруднений учеников. Несмотря на то, что управляющие конструкции языка C++ синтаксически отличны от управляющих конструкций учебного языка программирования с национальной лексикой, но сформированный понятийный аппарат и уже изученные алгоритмы в рамках пропедевтики программирования позволил школьникам освоить производственный язык в сокращенные сроки. Этот успех был обеспечен пропедевтическим курсом с большим количеством заданий (более 200 задач различной сложности).

Благодаря возможности ЦОС КуМир дополнять ученическую задачу учительской программой с автоматической проверкой, курс был построен на вариативности обстановок встроенных исполнителей и автоматизированном тестировании заданий ученика. По количеству заданий курс превысил соответствующие разделы учебников [106; 76]. Такой подход обеспечил непрерывность образовательного процесса, когда ученику доступно не только выполнение домашней работы и работа в классе, но и возможность получить предварительную оценку от автоматизированной ЦОС КуМир. При этом подобный подход не отменял участие педагога в финальной оценке достижений учащегося. Таким образом, при резком увеличении количества заданий на составление программ практикующий педагог потратил свое время лишь при подготовке курса, задач и тестирующих алгоритмов, что положительно повлияло на качество образовательного процесса и снижение рутинной нагрузки на преподавателя.

Причина проведения эксперимента на механико-математическом факультете МГУ в начале 2000-х годов в курсе программирования для студентов начальных курсов обуславливалась различным уровнем компетенций, поступивших в университет в области информатики и ИКТ, когда даже алгоритмически-простейшие задачи вызывали определенную сложность у первокурсников. Фундаментальные традиции факультета предписывали осваивать промышленные языки программирования C и C++. Студентам был предложен выбор: осваивать C традиционными методами, изучая конструкции и синтаксис языка, одновременно алгоритмически решая учебную задачу, или воспользоваться знакомой со школы цифровой образовательной средой КуМир, предоставляющей элементы автоматизированной проверки заданий и ряд других возможностей, существенно упрощающих создание и отладку программ учащимися по сравнению системами программирования на C и C++. Примечательно, что 60-70% тестируемой группы студентов выбрали ЦОС КуМир в качестве основной системы для решения зачетных заданий [118].

Входная оценка компетенций обучаемых показала, что ЦОС КуМир выбирали не только студенты с начальным опытом в программировании, но и слушатели, обладающие продвинутым уровнем компетенции в области алгоритмизации и программирования. Последние выбирали ЦОС КуМир благодаря простому интерфейсу ЦОС КуМир, компактному функционалу алгоритмического языка и комфортным методам редактирования и отладки программ, Рисунок 15.

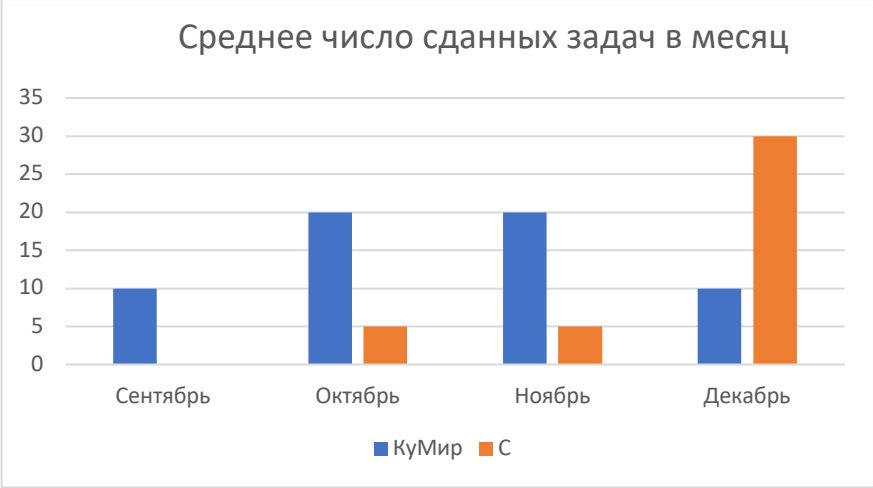


Рисунок 15 – Среднее число задач, выполненное студентами группы.

В результате экспериментальная группа выполнила на 50% больше алгоритмически-эквивалентных заданий. В отличие от программирования на С, студенты, выбравшие ЦОС КуМир, практически сразу приступили к сдаче заданий. Для них основной проблемой была только алгоритмическая сложность задач, а не сложность языка программирования или программного окружения. Создание и отладка программы учащимися в ЦОС КуМир, использующими школьный алгоритмический язык, более проста и удобна [98], чем традиционные среды программирование на производственных языках программирования, таких как С, С++, Python, Java [335; 373; 390]. Указанные производственные языки программирования представлены не только компилятором, но и развитой системой IDE - интегрированной средой разработки (англ. Integrated Development

Environment) — программным окружением, включающим ориентированный на конкретный язык программирования редактор, программу-дизайнер для создания интерфейса, компилятор, отладчик и т.п. – т.е. комплекс средств, используемых профессиональными программистами для разработки программного обеспечения на данном языке программирования. Основная проблема производственных IDE для учащихся состоит в том, что эффективное использование подобной системы программирования требует внимательного ее изучения и освоения, на которое могут уйти многие часы и дни занятий. И школьный алгоритмический язык и ЦОС КуМир можно освоить за 6 академических часов, научившись составлять простейшие программы, включающие многие комбинации основных управляющих конструкций, то есть сформировать основы алгоритмического мышления [130].

Более того, студенты, выбравшие для себя ЦОС КуМир и школьный алгоритмический язык, к последнему месяцу семестра были практически свободны от задач и у них осталось больше времени для подготовки к зачетам по профильным математическим предметам. В весеннем семестре в экспериментальной группе было проведено 4 занятия по языку С. Переход к кодированию на языке С программ, написанных на школьном алгоритмическом языке, происходил почти бесшовно. Все студенты группы, использовавшие в первом семестре, ЦОС КуМир, успешно освоили программирование на С, причем с меньшими временными затратами, чем студенты контрольной группы, изучающих С с первого семестра. Это связано с тем, что алгоритмизация была выделена практически в чистом виде при использовании ЦОС КуМир. Переход от одного учебного языка программирования к производственному не составил для них большого труда, в том числе и потому, что в ЦОС КуМир предусмотрена возможность автоматической трансляции со школьного алгоритмического языка на С++.

2.6 Особенности школьного алгоритмического языка и ЦОС КуМир

Школьный алгоритмический язык программирования, применяя подход программного управления исполнителями, успешно реализует парадигму объектно-ориентированного программирования (ООП), при этом сохраняя все ключевые аспекты структурного программирования, аналогично учебному языку Паскаль. Это играет решающую роль в снижении возраста, когда дети могут впервые познакомиться с основами алгоритмизации и программирования, делая это доступным уже на уровне начальных классов.

Несмотря на кажущуюся простоту школьного алгоритмического языка, так, список команд Робота, Чертежника и конструкции языка программирования помещаются на один разворот школьного учебника [108], Рисунок 16,

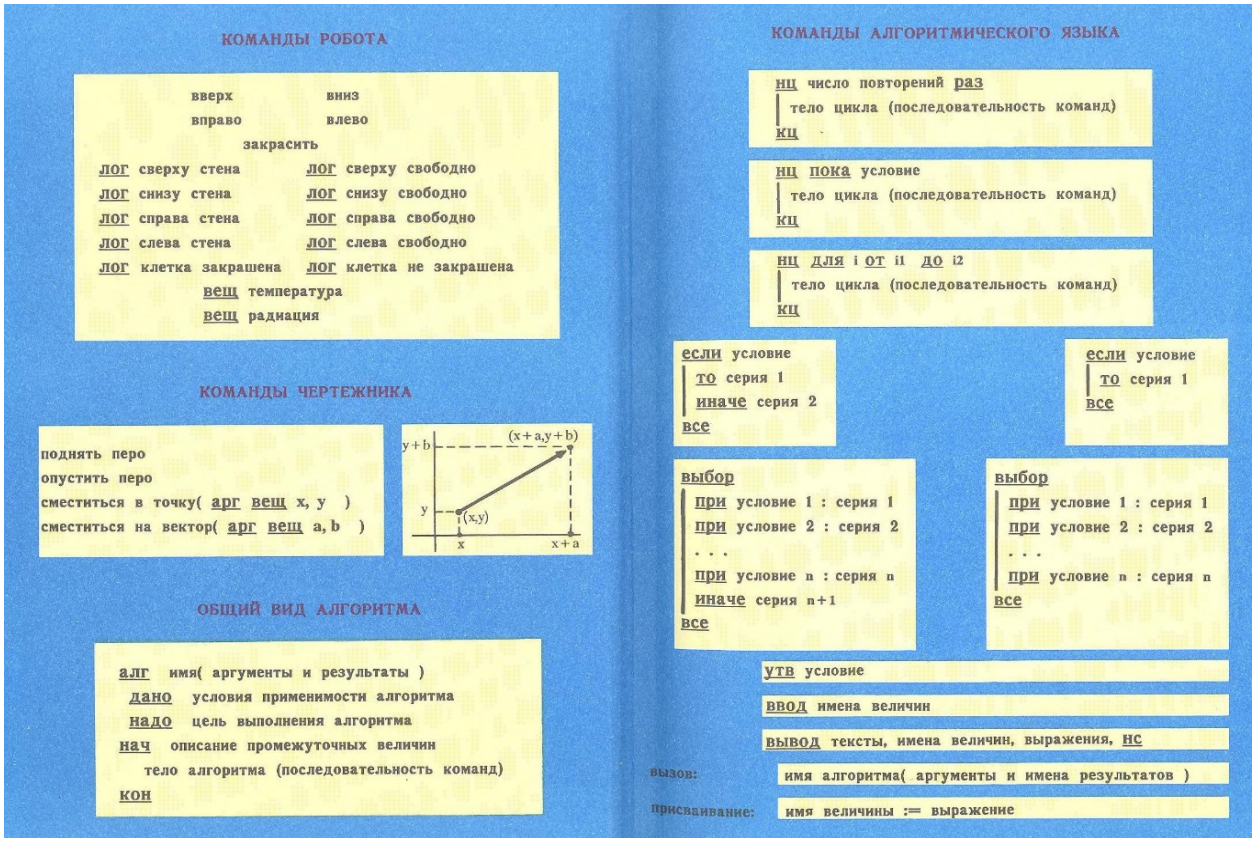


Рисунок 16 – Основные команды школьного алгоритмического языка

сам школьный алгоритмический язык по своим алгоритмическим возможностям не уступает языку Pascal [227]. Он является удачным выбором для освоения основ

алгоритмизации, когда требуется в короткие сроки научиться основам процедурного программирования [130], в том числе и тем учащимся, кто не ставит для себя цель в будущем освоить профессию программиста.

Так, К.Ю. Поляков [201] к достоинствам школьного алгоритмического языка относит то, что русские команды (управляющие конструкции, имена величин и т.п.) обучаемые воспринимают намного легче английских, особенно на начальном этапе обучения программированию. А.Г. Гейн рассматривает обучение алгоритмизации как средство развития мышления и предлагает обучать на естественном языке, позволяющем описывать, в том числе и собственную алгоритмизуемую деятельность [35]. Атрибуты текстовых учебных языков, такие как близость к естественному языку (вплоть до совпадения) и национальный синтаксис, перекликаются с идеями педагогов и разработчиков национальных языков программирования в Японии и Франции.

Однако обучение основам программирования и алгоритмизации в ИКТ-насыщенной современной аудитории не может проводиться без соответствующей цифровой (компьютерной) образовательной среды, комфортной для новичка и избавленной от непроизводительных расходов, которая должна включать не только продуманный набор заданий для формирования основ алгоритмического мышления, но и иметь собственный набор дидактических приемов [130; 129].

Современная тенденция преподавания программирования в вузах предполагает не только развитие алгоритмического стиля мышления [60], но и изучение одного из современных языков, поддерживающего принципы объектно-ориентированного программирования (ООП): такие языки программирования, как C++ [388], C# и Python [142], или Java [267]. Язык Pascal также содержит элементы ООП [298]. Основу языка Pascal составляла тройка понятий, логически выводимая из объектов и действий над объектами:

Действия -> Команды (Циклы) -> Вспомогательные алгоритмы

Объекты -> Величины (Таблицы)

Изначально в языке Pascal не существовало полноценной записи для объектов, структурирующей их по предназначению и способам действий с ними, что, скорее всего, объясняется тем, что не использовались структурированные средства для работы с объектами, хотя еще в конце 1960-х годов в Норвежском Вычислительном центре (Осло) Кристианом Нюгорд и Оле-Йоханом Даль предложен законченный проект объектно-ориентированного языка Simula-67, в котором впервые было введено понятие класса (объекта) и связывание его с допустимыми над ним действиями (методами) [317; 316].

Полностью объектно-ориентированный языком программирования можно считать разработку Алана Кея (Исследовательский центр в Пало-Альто Xerox PARC) [219] Smalltalk-72, а затем и в Smalltalk-80.

В языке Smalltalk присутствуют только объекты и сообщения как способы взаимодействия объектов. При этом строки, целые числа, логические значения, определения классов и т.п. – всё является объектами. Выполнение программы состоит в пересылке сообщений между объектами. Любое сообщение может быть послано любому объекту, при этом объект-получатель определяет, является ли это сообщение правильным и что надо нужно предпринять [328]. По сути, Smalltalk является не только языком программирования, а системой программирования [338]. Так же организован школьный алгоритмический язык в ЦОС КуМир.

Термин ООП подчеркивает первичность объектов над действиями и алгоритмами. Хотя на практике важны и алгоритмы, и объекты: и те, и другие надо уметь структурировать [103] и иметь средство для работы с ними. В различных языках программирования такие понятия имеют различное именование: в языке Ада они названы «пакетом» (package) [218], в языке Modula-2 — «модулем» [24], в C++, C#, Java и Simula—«объектом» и «экземпляром класса» [265], Pascal вводит конструкцию Unit [23], школьный алгоритмический язык (ЦОС КуМир) – «Исполнитель» [123]. В учебнике А.Г. Кушниренко и др. [84] соответствующее понятие называют еще «информационной моделью исполнителя».

При этом само понятие *исполнитель* более широко по своему значению. Под исполнителем понимается не только конструкция школьного алгоритмического языка ЦОС КуМир, но и человек, автомат или прочее устройство или группа устройств, связанных общими свойствами и имеющих навсегда фиксированную систему команд. К важным свойствам исполнителя является его «незнание» об управляющей им системе, что в ООП именуется абстрагированием.

Важно отметить, что в ЦОС КуМир исполнитель является не только внутренней структурой алгоритмического языка, но и одновременно, подобно исполнителям из реальной жизни, может существовать отдельно от системы программирования. Так, для человека простейшим исполнителем (с минимальной системой команд-предписаний) можно назвать электрическую лампочку освещения комнаты, с которой ежедневно приходится сталкиваться каждому. Заходя в темную комнату, человек «включает свет», а покидая ее, «выключает», используя кнопку-выключатель. В таком случае принято говорить, что исполнитель «лампочка» имеет кнопочный режим управления.

Существует множество более сложных исполнителей, имеющих кнопочное управление: видеоплеер, телефон, автомобиль и, наконец, компьютер. Не трудно заметить, что ежедневное использование исполнителей и кнопочное управление ими не создает непреодолимой методической трудности введения педагогом нового для учащихся понятия. Несомненно, понятие класса в C++ потребует больших усилий как от обучаемого, так и от учителя [16].

Вполне закономерным является изучение в школьном курсе информатики (учебник А.Г. Кушниренко [108]) исполнителя Робота, и, как следствие, закономерным является желание учащихся управлять роботом не только в кнопочном режиме, но и программируя его – то есть составляя план будущих действий робота:

Попутно можно заметить, что, изучая исполнители ЦОС КуМир, учащийся не видит внутренней (скрытой) сущности и деталей реализации того или иного исполнителя, хотя полностью известен функционал изучаемого объекта. Это не

означает отсутствие обратной связи у исполнителей в школьном алгоритмическом языке ЦОС КуМир, но относится к еще одному принципу ООП, именуемому инкапсуляцией.

Таким образом, в школьном алгоритмическом языке в ЦОС КуМир вводится четыре фундаментальных понятия информатики и программирования:

Действия -> Команды (Циклы) -> Вспомогательные алгоритмы

Объекты -> Величины (Таблицы) -> Исполнители

Первые два понятия отражают методы записи действий и объектов (в частности, большого количества действий и большого количества объектов). Вторые – отражают фундаментальные приемы структуризации, которые человечество выработало за последние годы. Эти понятия просты и доступны школьникам, могут быть поняты и освоены в процессе решения задач, и все вместе образуют фундамент, на котором можно развивать и внутренние способности человека к алгоритмическому мышлению, и понимание реальностей окружающего мира. Освоив основные понятия современной информационной культуры, можно развиваться в разных направлениях: от изучения способов конструирования структур данных и новых языков программирования до решения более сложных прикладных задач.

ЦОС КуМир позволяет не только использовать готовых исполнителей и обучаться программированию, составляя алгоритмы управления ими, но и создавать новых внутренних исполнителей в программе, имеющих свои локальные и глобальные величины, и предоставляющих доступ к алгоритмам исполнителя другим исполнителям.

К числу трех принципов ООП относят еще наследование и полиморфизм. В версии ЦОС КуМир, нашедшей широкое распространение в 1990-х годах, была доступна возможность переопределения операций, что было продемонстрировано примером исполнителя «Комплексные числа». Вполне закономерно, что и современная многоплатформенная версия системы имеет аналогичные возможности. Однако в производственных языках программирования доступны и

столь экзотические методы ООП, как множественное наследование. Смысл таких приемов, особенно при обучении, вызывает вполне закономерное сомнение в его надобности. И здесь не уместны аналоги с рекурсией как метода в программировании и замене его итерацией.

Подводя промежуточный итог можно утверждать, что ЦОС КуМир может занимать вполне заслуженное место в профильном обучении, в начальном курсе ООП. При этом использование ЦОС КуМир позволяет существенно упростить процесс обучения, сократить число затрачиваемых на темы часов и, что наиболее важно, показать учащимся закономерность возникновения ООП и продемонстрировать его положительные, фундаментально философские качества – инкапсуляцию, полиморфизм и наследование.

Особенности школьного алгоритмического языка, заложенные А.П. Ершовым и позволяющие ему быть востребованным в школьных и вузовских пропедевтических курсах программирования, можно свести к 4-м главным:

1) изначальная нацеленность на учебный характер языка и на достижение низкого порога вхождения,

2) русскоязычная лексика,

3) поддержка внешних исполнителей,

4) консервативность синтаксиса и семантики, близость к проверенному учебному языку Паскаль (Pascal).

Одной из трудностей выбора языка для записи алгоритмов в школе А.П. Ершов называл противоречие между разнообразием языковой практики программирования и единством учебного процесса в школе. Использование производственных языков программирования в обязательном курсе Информатики и ИКТ в 7-9 классах и ранее не представляется разумным, так как в обязательных курсах 7-9 класса не ставится задача подготовки профессиональных программистов.

Использование учебных языков, школьного алгоритмического уменьшает технические затраты на освоение собственно языка и оставляет больше времени на решение чисто алгоритмических проблем. В легкости освоения школьный алгоритмический язык выигрывает у других текстовых учебных языков, в частности у Pascal, так как ориентирован на русскоязычных учеников. Таким образом, на освоение основных конструкций школьного языка ученик тратит в разы меньше времени, чем пришлось бы потратить при изучении одного из производственных языков. Проведенные исследования показывают, что изучить алгоритмический язык (ЦОС КуМир) и освоить основы алгоритмизации можно всего за 6 часов академических часов [129]. Для курсов на основе производственных языков программирования подобный результат недостижим. Решающим фактором такого минимального срока освоения учебного языка и ЦОС является национальная русскоязычная лексика в школьном алгоритмическом языке.

Освоение начал программирования проще начинать на примерах наглядных алгоритмов управления, а не на примерах абстрактных алгоритмов вычисления или преобразования текстов. А.П. Ершов активно пропагандировал этот подход, и возможность управления внешними исполнителями была предусмотрена еще в языке Робик [68]. Использование простых и наглядных исполнителей Робот и Чертежник во многом обеспечило успех учебников информатики [73; 77].

Несмотря на то, что к 1985 году в языках Робик и Рапира был предложен ряд оригинальных синтаксических и архитектурных решений, в школьном алгоритмическом языке были использованы только консервативные решения, что способствовало популярности школьного языка среди школьных учителей [105].

Естественность алгоритмического языка еще и в том, что семантика управляющих конструкций частично наследуется из естественного языка. В частности, конструкция ветвления «если-то-иначе-все» школьниками воспринимается легко, как пришедшая из повседневной жизни, в которой детям приходится выслушивать фразы типа: «ЕСЛИ ты не сделаешь уроки, ТО гулять не

пойдешь». Как видно, запись конструкции ветвления не сильно отличается от того, что знакомо любому школьнику с детства.

Доводы в пользу использования английских слов в записи конструкций парируются предложением ввести эти слова в повседневную жизнь: «IF ты не сделаешь уроки THEN не пойдешь гулять». Но как в языке, на котором написана программа, передаваемая на выполнение компьютеру, в алгоритмическом языке присутствует определённая алгебраичность и формализм. Так, в конструкции цикла используются короткие ключевые слова-сокращения «нц-кц» вместо длинных словосочетаний «начало цикла – конец цикла». Практика показывает, что дети мгновенно запоминают и легко осваивают подобные сокращения, понимая их необходимость.

Еще одна особенность школьного алгоритмического языка, это возможность использования принятой в математике формы записи сложных неравенств типа 0<x<1 вместо несуразной записи 0<x **и** x<1. В производственных языках программирования первым языком, где была решена эта проблема, был Python (Питон).

Особенность ЦОС КуМир, сближающая ее с языком программирования Python, сведение к нулю синтаксической значимости закрывающих операторных скобок. В ЦОС КуМир удаление в программе на школьном алгоритмическом языке закрывающих операторных скобок не приведет к потере информации о структуре программы. Такой результат достигается путем автоматической генерации вертикальной разметки и отступов в редакторе программ ЦОС КуМир.

Также легко исключить ошибки во вводе управляющих конструкций, выбирая из меню макрокоманды, вставляющие шаблоны конструкций целиком.

Особенностью ЦОС КуМир является проверка корректности глобальной структуры программы в процессе ее редактирования. Полная диагностика всей программы, включая обнаружение неопределенных или повторно определенных имен, выполняется при любом покидании строки.

Очень важно, что ЦОС КуМир подробно и в большинстве случаев точно диагностирует синтаксические ошибки, сигнализируя об ошибке как можно ближе к месту ее возникновения, выводя диагностику на поля программы и расцвечивая ошибочные строки. Это обеспечивается тем, что формальная грамматика школьного алгоритмического языка в ЦОС КуМир охватывает не только правильные программы, но и часть программ, которые могут быть превращены в правильные исправлением единичной ошибки.

Ниже приведены дидактические особенности интерпретатора ЦОС КуМир.

Как и в большинстве производственных языков программирования, объявление переменных не приводит к автоматической инициализации величины. Все одиночные величины и элементы массивов в момент их создания имеют специальное значение «не определено». Попытка использования неопределенной переменной или элемента массива диагностируется на этапе выполнения. Выходы индексов массива за допустимые границы также диагностируется.

В отличие от производственных языков программирования, выход за допустимый диапазон значений целочисленных переменных и элементов массивов в школьном алгоритмическом языке ЦОС КуМир диагностируется на этапе выполнения. Поэтому попытка вычисления 31 степени двойки в ЦОС КуМир приводит к диагностированному переполнению, в то время как, например, в Pascal ABC это вычисление дает отрицательное число.

В ЦОС КуМир нет режима командной строки, поэтому выполнить определенный фрагмент программы нельзя. Однако в ЦОС КуМир можно запустить на выполнение вспомогательный алгоритм с аргументами и результатами, не тратя время на составление основного алгоритма. При таком запуске аргументы будут запрошены до начала выполнения, а результаты будут выведены на экран по завершении выполнения. Эта особенность экономит много времени учеников при отладке несложных вспомогательных алгоритмов в заданиях.

Отладочный режим ЦОС КуМир, при котором результаты всех присваиваний и проверок условий выводятся на поля, во многих случаях избавляет новичка от использования отладочных печатей.

ЦОС КуМир снабжен минимальным набором отладочных возможностей: есть точки останова, операторы стоп и пауза, пошаговый режим исполнения, изображение в отладочном режиме результатов присваиваний на полях программы, режим отображения всех локальных и глобальных переменных и стека вызовов вспомогательных алгоритмов. Из этого минимального набора новичок, как правило, использует при отладке только один удобный режим - пошаговое выполнение с выводом результатов присваиваний на поля.

ЦОС КуМир не помогает придумать алгоритм, но помогает записать его на школьном алгоритмическом языке и отладить за кратчайшее время.

Начиная осваивать программирование, ученик должен сосредоточиться на продумывании алгоритма задания и использовании учебного языка программирования (школьного алгоритмического языка), интегрированного в простейшую, дружественную к обучаемому среду исполнения (ЦОС КуМир), которая предоставляет средства для эффективной отладки и проверки правильности выполненного задания. То есть ученик придумывает алгоритм, минимально преодолевая сложности, связанные с записью алгоритма на еще не освоенном языке программирования в неосвоенной среде программирования, и запускает программы в этой среде программирования, получая адекватную информацию о правильности выполнения задания с возможностью оперативного исправления ошибок.

Трудность перевода (кодирования) на том или ином языке программирования в первую очередь связана со сложностью выбранного языка программирования. При этом чем сложнее язык программирования, тем сложнее новичку без ошибок выразить (закодировать) на нем даже учебный алгоритм. Достижение уровня профессионального программирования, накопление опыта, при котором у ученика не возникает сложностей на этом этапе составления

программы, невозможно за отведённые ФОП ООО часы [246]. Стать определенным успешным этапом в формировании тройки знать-уметь-владеть основами алгоритмизации и программирования, преодолеть эти трудности сможет национальная лексика учебного языка программирования, приближенная к естественному языку, простота школьного алгоритмического языка, интеграция редактора и компилятора и общая ориентация ЦОС КуМира на новичков.

Для ученика практически невозможно использование учебного языка программирования, не интегрированного в цифровую образовательную среду. Потому что новичок должен видеть результаты работы его программы, то есть визуализировать только результаты выполнения учебной задачи недостаточно, необходимо наблюдать работу программы, как, например, в ЦОС КуМир, где поддерживается специализированный отладочный режим, в котором не требуется предпринять специальных действий для пошагового или непрерывного выполнения с выводом результатов на поля программы.

2.7 Вопросы образования на национальных языках

Обучение программированию на национальных языках является важным шагом в направлении развития IT-образования, сохранения культурного наследия и стимулирования экономического роста в странах бывшего СССР и России. Школьный алгоритмический язык, интегрированный в цифровую образовательную платформу КуМир с возможностью автоматизированной проверки учебных заданий и поддерживающий, как и императивные производственные языки программирования, парадигмы процедурного и объектно-ориентированного программирования, сыграл и играет определенную существенную роль в образовании России. В 1980-1990-х годах в СССР учебник [174] был переведён на молдавский язык, в том числе и школьный алгоритмический язык и ЦОС КуМир. Однако этот опыт был единичным, и в настоящее время школьный

алгоритмический язык поддерживает фактически единственный государственный язык России – русский, который изучают абсолютно все школьники страны.

Тем не менее в 1990-х годах в России был положительный опыт не только по адаптации англоязычного программного обеспечения, но и созданию новых педагогических программных продуктов, основой которого является производственный язык программирования, имеющий англоязычную основу. Речь идет о разработке Пролог-Д (автор С.Г. Григорьев [48; 49]). Система логического программирования Пролог-Д предназначалась в том числе и для поддержки курса информатики, поэтому разрабатывалась как интегрированная система программирования, ориентированная на использование русского языка. Это не помещало впоследствии ее адаптации для казахского и болгарского языков, при которой потребовалось учитывать особенности языков и национальные традиции. Как русский вариант системы Пролог-Д, так и ее казахская и болгарская версии были портированы на все типы отечественной и зарубежной учебной вычислительной техники [47; 50].

Важность использования национальной лексики можно проследить еще при разработке первых языков программирования. В 1940-х годах немецкий инженер и конструктор серии компьютеров Конрад Цузе придумал язык программирования Plancalkul (Описание планов) [406], программы на котором писались фактически по-немецки, хотя не ставилось никаких педагогических задач обучения детей программированию.

Согласно Всероссийской переписи населения 2020 года, Россия насчитывает 155 живых языков [230]. Тем не менее в такой многонациональной стране существует проблема обучения молодого поколения на родном (но не русском) языке. При этом небольшая численность той или иной степени малочисленной народности не упрощает проблему, а подчас делает её более острой.

Игнорировать национальное достояние, культуру, традиции любой малой народности недопустимо. Это прямой путь к сегрегации – политике принудительного отделения какой-либо группы населения. Сегрегация

практически исчезла после мировых успехов, достигнутых движением за права человека во второй половине XX века, и ныне встречается крайне редко [144]. Однако в большинстве западных стран, где некогда существовала административная сегрегация, она до сих пор может фактически существовать в скрытом или явном виде [41; 70]. Без сомнения, дети должны иметь возможности для обучения в начальной школе на родном языке своей народности, что указано в «Концепции государственной языковой политики Российской Федерации» [94]. В частности, в Концепции сказано, что одним из «принципов государственной языковой политики РФ является «… воспитание уважительного отношения и популяризация позитивного образа семей, использующих двуязычие и многоязычие, … повышение роли носителей языков народов Российской Федерации в деятельности по сохранению языкового многообразия ...» [94]. В разделе же VI Концепции сказано, что «по итогам реализации настоящей Концепции предусматривается достижение следующих результатов: … обеспечена возможность для использования языков народов Российской Федерации в сфере образования и культуры» [94].

Статистика российского образования ведёт две категории учёта национальных начальных школ России, поскольку приходится подсчитывать количество школьников, изучающих родной язык в начальной школе, и количество детей, которым преподаются предметные дисциплины на родном языке. В обоих видах статистического учёта речь идёт именно об обучении в начальных классах. Многие малые народности России, в том числе получившие свою языковую письменность на базе латиницы, перешли на кириллицу ещё в 30-е годы прошлого века. Некоторые малочисленные народности обрели письменность во второй половине XX века сразу с опорой на базе кириллического алфавита. Эта алфавитная база, несомненно, сыграла свою положительную роль в создании педагогических перспектив национального обучения в начальных школах малочисленных народностей [25].

В учебные планы большинства национальных начальных школ включаются несколько одновременно изучаемых языков: реже – два, чаще – три (родной, русский, иностранный). ФГОС НОО [154], ФГОС ООО [170], ФГОС СОО [232]- обеспечивают сохранение и развитие культурного разнообразия и языкового наследия народов России, реализацию права на изучение родного языка, возможности получения образования на родном языке, овладение духовными ценностями и культурой многонационального народа Российской Федерации.

Выводы главы 2

Исследование методики преподавания основ алгоритмизации и программирования в различных неанглоязычных странах показывает, что использование текстовых учебных языков программирования с национальной лексикой положительно влияет на эффективность образовательного процесса, на формирование алгоритмического мышления в рамках национальных культур и ценностей народов, что обеспечивается в том числе близостью учебного языка к естественному языку.

Преподавание информатики и программирования с использованием национального педагогического программного обеспечения в качестве средства обучения (цифровых образовательных сред с интегрированным учебным языком программирования) играет решающую роль в снижении возраста, когда дети могут впервые познакомиться с основами алгоритмизации и программирования, делая это доступным ученикам начальных классов. Фундамент современных пропедевтических разделов курса информатики по основам алгоритмизации и программирования с автоматизированными средствами обучения самостоятельного контроля освоенной компетенции учащимися и преподавателями был заложен еще при преподавании информатики и ИКТ для основной школы с использованием ЦОС КуМир.

Подобный подход к освоению основ информатики и программирования в целях формирования алгоритмического мышления, изначально нацеленный на

школьников, и разработанное для него содержание обучения оказались эффективным средством для новичков самых разных возрастов, от школьников до студентов педагогических университетов, воспитателей детских садов, школьных учителей и т.д.

Методика обучения информатике и программированию, основанная на использовании школьного алгоритмического языка программирования, который реализует подход программного управления исполнителями, соответствует парадигме объектно-ориентированного программирования (ООП), при этом сохраняя все ключевые аспекты структурного программирования.

ГЛАВА 3. МЕТОДОЛОГИЯ ПОНИЖЕНИЯ ВОЗРАСТА ПЕРВИЧНОГО ЗНАКОМСТВА С ОСНОВАМИ ПРОГРАММИРОВАНИЯ

3.1 Игровая методология понижения возраста первичного знакомства детей с основами алгоритмизации и программирования

Усвоение набора сложных понятий, необходимое для формирования основ алгоритмического мышления, как содержания обучения информатике и программированию, даже для детей не будет представлять особых трудностей, если обучение проводить игровой форме в пропедевтической методической системе обучения, так как ведущей деятельностью дошкольников является игра.

Методика преподавания арифметики уже достаточно устоялась в России и во всем мире. Несколько столетий тому назад арифметику преподавали в университетах Европы [26], а сейчас изучение, например, признаков делимости отнесено ФОП ООО [246] к 7 классу основной школы. В начальной же школе третьеклассникам знакомы целые числа в пределах 1000, в том числе представление в виде суммы разрядных слагаемых, увеличение или уменьшение числа в несколько раз и т.п. [245], то есть каждый выпускник начальной школы на сегодня владеет азами арифметики. При этом, хотя систематически арифметика изучается в начальной школе, но знакомство с цифрами, числами и счетом начинается в дошкольном возрасте [244].

Алгоритмика и программирование гораздо моложе арифметики и математики. Базовые элементы алгоритмики, подобно базовым элементам арифметики – десятичным цифрам, записи чисел с помощью цифр и операциям сложения и вычитания, могут быть освоены в раннем детстве, если только детям будет предоставлена цифровая предметная среда, позволяющая освоить эти элементы в деятельностно-игровой форме. Эта материальная среда оказывается более сложной, чем среда для освоения элементов счета. Эта сложность вызвана объективными причинами – описание процессов составления программы человеком и последующего выполнения программы автоматическим устройством

требует введения не менее дюжины базовых понятий [12; 287], в то время как для описаний процессов счета и использования его результатов можно обойтись гораздо меньшим количеством понятий. Если рассматривать внедисциплинарный подход по формированию дидактических приемов понижения возраста обучаемого, то тут наблюдается схожая методика в образовательных процессах.

Методика М. Монтессори [151; 149; 152; 150] основана на гуманистических идеях о свободном развитии личности, обращении к уникальным врожденным способностям ребенка, когда, по мнению М. Монтессори, в раннем возрасте до 7 лет [52; 42] формируется до 80% интеллекта. В рамках содействия естественному развитию в подготовленной педагогом предметно-пространственной среде, в том числе с привлекательными открытыми материалами, ребенок самостоятельно занимается, когда учитель лишь наблюдает за его развитием, оказывая помощь в необходимый момент. Несмотря на некоторый детский эгоцентризм [15; 185], дети Монтессори побуждаются к сотрудничеству, взаимопомощи, активно обучают друг друга [396]. Подходы М. Монтессори во многом спорны, но методика, подтвержденная результатами, не может не вызывать восхищения [31].

Школа Монтессори не ставила, да и не могла ставить задачу формирования у дошкольников и младших школьников основ алгоритмического мышления, но методология М. Монтессори, позволяющая понизить возраст детей, например, при освоении письма, базировалась на наборе дидактических приемов, который в рамках проведённых ею работ можно назвать *дидактическим окружением* Монтессори. В это окружение также включена предметно-пространственная дидактическая среда, направляющая развитие ребенка, непрерываемый образовательный процесс, в котором участвуют и педагоги-воспитатели, и родители учеников, кооперативная деятельность, объединяющая малышей, и, конечно же, игра. Надо отметить, что дидактическое окружение Монтессори достаточно легко распространяется и на образовательный процесс в вузах [43], где потребность в гибких формах обучения демонстрирует рост в последнее время [396; 278].

Значение игры в жизни человека признано психологами и педагогами по всему миру, особенно в образовательной сфере [322; 392; 269; 270; 29; 28]. В 1961 году была основана Международная ассоциация защиты прав ребенка на игру (IPA). Этой организацией был разработан документ «Декларация о правах ребенка на игру», который подчеркивает, что игра необходима для физического и психического здоровья ребенка, и игра является частью образования [336].

Игра активно используется как в дошкольном, так и в начальном общем образовании России [53; 154]. Для дошкольников в ФГОС ДОО прямо указано: «реализация Программы в формах, специфических для детей данной возрастной группы, прежде всего в форме игры, познавательной и исследовательской деятельности, в форме творческой активности, обеспечивающей художественно-эстетическое развитие ребенка» [53].

Некоторые педагоги критикуют конкретное применение ФГОС ДОО в различных дошкольных образовательных организациях. Так, Т.П. Авдулова ратует за свободную игру малышей, а не замену творческих занятий обучением дошкольников и организацией занятий только дидактического характера.

Также писал об игре А.В. Запорожец, указывая, что развивающий потенциал игры детей может реализоваться только в форме «детской самодеятельности» [71], хотя А.В. Запорожец не отрицал ценности дидактических игр [72; 78].

С 1960-х годов дидактические игры следовали в фарватере народного школьного образования, когда навыки и умения дошкольников формировались и закреплялись на последовательности усложняющихся игр, например, путем усложнения сюжетов как способа развития детской игры [176; 43; 268; 1]. Д.Б. Эльконин пишет: «Должна быть создана программа детских игр, в которой должны быть указаны не только сюжеты для различных возрастов, но и круги содержаний по возрастам. … Это и есть настоящая программа по игре, которая должна быть связана и с образовательной работой детского сада» [268]. Последовательное усложнение игр соответствует введенному Л.С. Выготским понятию зоны ближайшего развития: «Отношение игры к развитию следует сравнивать с

отношением обучения к развитию. За игрой стоят изменения потребностей, изменения сознания более общего характера. Игра — источник развития и создает зону ближайшего развития. Действие в воображаемом поле, в мнимой ситуации, создание произвольного намерения, образование жизненного плана, волевых мотивов — все это возникает в игре и ставит ее на высший уровень развития» [28].

При этом важно соблюсти баланс между игрой как средством развлечения, как способом самостоятельного познания окружающего мира ребенком и игрой как дидактическим приемом обучения и воспитания под руководством педагога или воспитателя.

Компьютерные игры, являясь практически непродуктивным времяпровождением досуга, также относятся к играм, и методы их разработки могут быть использованы при создании педагогических программных продуктов. Так, обычно компьютерные игры имеют настолько удобный и привычный интерфейс [208] с пользователем (игроком), базирующийся на накопленных игроком знаниях и навыках, что процесс обучения, вхождения в игру становится не нужным, и человек сразу приступает к решению поставленной перед ним задачи: победить в сражении, разгадать загадку, посадить самолет и т.п. Подобные решения в области интерфейса, мотивации и т.д. нужно использовать в образовательном процессе для его интенсификации, а также следует внедрять геймифицированное обучение основам программирования и другим STEM-предметам для детей младшего школьного возраста и даже дошкольников [264].

Так, группа исследователей из Инженерно-технологического университета штата Флориды, США провели анализ различных образовательных методик [313], включая игры и интерактивные симуляции. Было обнаружено, что игровые методики в результате дают более ощутимое развитие когнитивного уровня учащихся, чем классические методы обучения. При предоставлении самостоятельного выбора студентам технологии обучения предпочтение отдавалось играм и интерактивным симуляциям. Также были выявлены и

гендерные предпочтения в методике обучения. Однако при обучении под руководством ментора столь очевидных предпочтений не было выявлено.

Ж. Пиаже не связывал свою теорию напрямую с образованием, хотя более поздние исследователи объяснили, как особенности теории Пиаже могут быть применены к преподаванию и обучению [359]. Пиаже оказал огромное влияние на разработку образовательной политики и педагогической практики. В трудах Пиаже игра связана с возникновением и развитием символической функции мышления у ребенка, что позволяет ему выполнять подмену реального объекта знаком [366]. Этот факт дает возможность существенно снизить возраст знакомства детей с основами программирования, о чем пойдет речь в следующем параграфе.

Ke Fengfeng из Университета Флориды (США) в 2009 году проанализировала около сотни исследований в области использования учебного игрового программного обеспечения в образовательном процессе, разделив по 5 классам, большинство из которых относились к первому классу - влияние компьютерных игр на обучение [342]. Эмпирические исследования установили значительный положительный эффект от компьютерных игр, четверть не подтвердили особых результатов игрового обучения и около 20% не смогли установить разницу между классическим образовательным процессом и инновационной технологией использования компьютерных игр. Это означает, что, несмотря на явные признаки эффективности игрового компьютерного обучения, необходимо проводить дальнейшие исследования, в том числе дидактики игрового образовательного процесса, влияния на учеников игрового обучения, влияния ментальных возможностей и ограничений, процессов дизайна интерфейса игрового учебного программного обеспечения и т.п.

Если рассматривать учебные компьютерные игры как усложненные учебные среды, в которых ученикам требуется постоянная поддержка педагога и предобучение до начала использования педагогического программного продукта, то исследования показывают повышение эффективности при такой поддержке учителя. При этом наибольший эффект наблюдается при получении новой или

вводной информации про игровую среду, а также при проведении тренинга под наблюдением ментора для освоения основных навыков игрового программного обеспечения [403].

Программирование сложно для начинающих, составление плана будущей деятельности компьютера и (или) исполнителя требует решений и описаний способами, которые не являются ни знакомыми, ни естественными. Известны ряд исследований, которые направлены на изучение уровня естественности языков программирования и способов решения проблем непрограммистами. При этом изучается понятийная база учебного курса и языка программирования и тезаурус новичка в срезе решаемой задачи. Например, детей 7-8 лет просят объяснить, как, по их мнению, устроена и как «работает» популярная игра Pacman [295], их тезаурус сохраняется и анализируется с целью помочь разработчикам педагогического программного продукта предвосхитить терминологические и интерфейсные трудности, предусмотреть альтернативные подходы, более соответствующие образу мышления людей [361].

3.2 Пиктографическое программирование как пропедевтика алгоритмического языка

Пиктографическое программирование представляет собой методику обучения (пропедевтику) алгоритмического языка, которая использует пиктограммы вместо текстовых команд для создания программ. Этот подход имеет такие ключевые аспекты, как универсальность для обучения всех возрастов, простота и доступность для освоения дошкольниками, визуальная привлекательность, что делает его научно обоснованным и эффективным средством обучения основам алгоритмизации и программирования. Пропедевтическая методическая система обучения, использующая в качестве средства обучения пиктографические цифровые образовательные среды, позволяет ученикам освоить непосредственное управление роботом-исполнителем и,

впоследствии, решать набор заданий, составляющих содержание обучения, компонуя из пиктограмм простейшие последовательные программы.

Основная цель начального курса программирования – развитие алгоритмического стиля мышления, что является для новичка самостоятельной и достаточно сложной задачей. Как было сказано ранее, важно, чтобы в начале курса для ученика выделялись чисто алгоритмические проблемы, свободные от сопровождающих их сложностей смежных предметов, например, математики. Однако также не менее важно, чтобы иные технические или лингвистические трудности также были за пределами занятий. То есть максимально очистить и упростить понятийную базу, доступную для освоения школьникам, дошкольникам, новичкам. Генезис исполнителя Робот является следствием предыдущего тезиса, потому что для понимания и осознания Робота и, естественно, управления им не требуется дополнительных затрат на изучение объекта, так как он находится в зоне актуального развития. То есть сам робот чрезвычайно прост, и изучать его практически нечего точно так же, как нечего изучать описания большинства игр, где игровая обстановка сама, по сути, является полноценным описанием задания играющему.

Известно, что детская игра представляет собой модель взрослой деятельности, которая доступна в том числе не только человеку, но и животным, которые в игре, например, отрабатывают навыки борьбы, охоты и т.п. Обычная кукла как некое визуальное подобие человека, возможно, даже имитирующая человеческую деятельность, как, например, говорящие, плачущие, шагающие куклы и даже роботы, имеющие лишь отдаленное сходство с людьми, все они моделируют не только человека как объект, но и элементы его поведения, что говорит о некоем знакомом ребенку из жизни функциональном подобии [26]. Робот может вести себя как человек, а значит, выполнять работу, которую может или даже не может выполнить последний. При этом и кукла, и робот создают игровое пространство, виртуальный мир как модель действительного мира. Игра является важнейшей школой детской жизни. Л.С. Выготский писал: «В игре ребёнок всегда

выше своего среднего возраста, выше своего повседневного поведения; он в игре как бы на голову выше самого себя. Игра содержит в себе, как в фокусе увеличительного стекла, все тенденции развития; ребёнок в игре как бы пытается сделать прыжок над уровнем своего обычного поведения» [27]. Место игры в образовательном процессе предоставляет ребенку возможность в наглядно-действенной форме моделировать эмоционально притягательные для него образы.

Одна из серьезных проблем понижения возраста знакомства детей с основами алгоритмизации и программирования для формирования алгоритмического мышления состоит в том, что новичок должен обладать сформированными навыками чтения и письма для того, чтобы составлять программы на текстовых языках программирования. Конструкции учебного языка, пусть даже использующего нативную лексику, надо суметь прочитать, осознать и только потом найти нужное выражение, составить его на учебном языке программирования и затем загрузить в компьютер для выполнения программы. Блочные режимы редактирования, когда конструкция языка вставляется, целиком мало помогают в этом процессе, так как решают лишь одну из нескольких частей задания, составляющих написание программы.

Таким образом дошкольники лишаются возможности осваивать в деятельностной форме основы алгоритмизации. Тем более это касается национальных дошкольных образовательных организаций России, где дети осваивают не только общегосударственный русский язык, но и родной язык национального образования. Проблема, которую необходимо решить, состоит в создании бестекстового учебного языка, интегрированного в цифровую образовательную среду, дидактических приемов и материалов, позволяющих осваивать основы алгоритмизации и программирования младшими школьниками и дошкольниками.

Знаки классифицируются по типам отношений, основанным на фактической смежности индексов или указателей, например, стрелка-указатель или указательный жест [215]. Знаки, основанные на визуальном сходстве с источником

обозначения, Чарльз Сандерс Пирс назвал иконическими или иконами [188]. Так, иконическое изображение кисти–типичный знак-икона. Поскольку знак как модель представляет собой способ организации знания, то отрыв от внешнего подобия создаёт возможность более свободного оперирования знаками, более совершенный способ организации информации.

Пиктограмма как знак визуально повторяет либо черты, либо объект, либо явления (действия), которые этот знак подразумевает. Пиктограммы как элементы человеко-машинного интерфейса, или, как их еще называют, иконки, известны и используются с 1970-х годов [341].

Таким образом, добавляя элементы игры в образовательный процесс, объединяя с простотой управления и минимизируя время на освоение педагогического программного продукта, естественно сделать вывод, что в начальной школе и в дошкольных образовательных организациях обучение программированию рекомендуется начинать не с текстового программирования на алгоритмическом языке [137], а с пиктографического программирования, когда программа «собирается» учеником из пиктограмм, обозначающих те или иные действия Робота, а не из последовательности синтаксически-корректных предложений на учебном языке программирования.

Безусловно, встает существенный вопрос, с какого минимального возраста можно начинать занятия по основам алгоритмизации и программированию, например, программного управления простейшими роботами-исполнителями? Педагогический опыт показывает, что современное поколение можно начинать знакомить еще до освоения азбуки, если предоставить ребенку интересную игрушку или посадить его за управление красочным и интересным компьютерным персонажем в игры, то в возрасте 4-6 лет дети вполне справляются с процессом управления, мысленно составляя программу. При этом ребенок вполне способен объяснить, как надо решать поставленную в игре задачу, в какой последовательности нажимать кнопки на пульте – т.е. у ребенка к началу управления сформирована программа действий персонажа в игре или управляемой

игрушки. Отличие такого специфического программирования в младшем возрасте от программирования в старшем возрасте в том, что, не владея сколь угодно формальными методами записи программы, ребенок не может облечь свой план в письменную форму, так как не умеет еще читать и писать, однако может рассказать и нарисовать. Еще совсем недавно такой барьер был бы непреодолим: сначала потребовалось бы обучить ребенка элементарной грамоте и формальному языку пусть учебного программирования, а уже затем он получил бы возможность самостоятельно составлять программы на некотором формальном языке. Сейчас этот барьер успешно преодолевается.

Существует масса компьютерные игр, отвечающих вышеуказанным требованиям, но не ставящих своей целью систематическое освоение программирования, что, естественно, не позволяет в полной мере сформировать у ребенка алгоритмическое мышление.

Можно выделить программу-игрушку Lightbot где в центре внимания находится забавный персонаж-робот с антенной на голове, который умеет выполнять лишь простейшие команды: перемещение на одну клетку, поворот, зажигание лампочки, прыжок на ступеньку вверх. Робот попадает на клетчатое поле-лабиринт из проходов между стенками-кирпичами, на некоторые из которых робот может запрыгнуть. В клетках, отмеченных цветом на поле, надо зажечь лампочку. Цель робота– зажечь лампочки во всех выделенных цветом полях. Основные достижения этого программного продукта в том, что программу робота ребенок составляет, набирая ее из готового списка, изображенного в виде характерных пиктограмм. Интерфейс продукта вполне прост и ясен для ребенка возраста 5-7 лет – (англ. drag & drop) перетаскивать команды в программу [347].

Задача, поставленная перед играющим, усложняется еще и тем, что размер программы ограничен всего 12 командами, но дополнительно предлагается два контейнера на 8 команд каждый для вспомогательных программ f1 и f2, Рисунок 17.

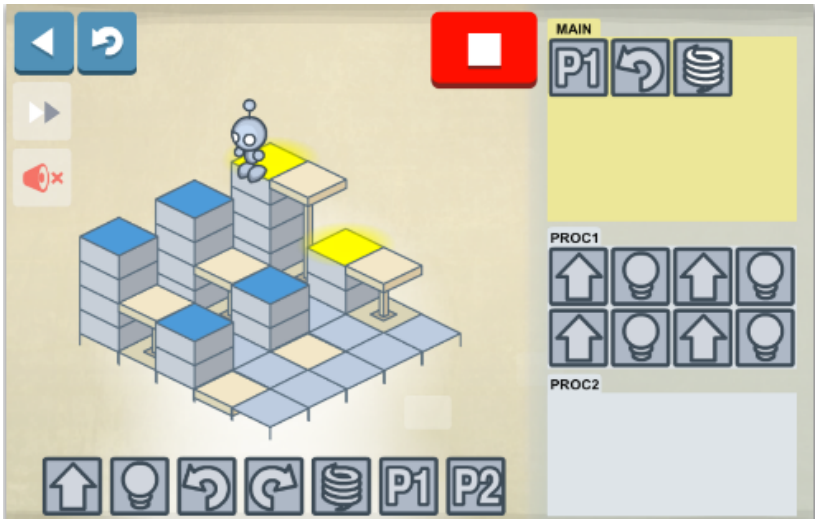


Рисунок 17 – Экран бестекстовой среды программирования Lightbot

В задаче требуется составить программу, по которой робот, прыгая по блокам, зажжет все блоки, помеченные синим цветом. На данном уровне игры задан шаблон программы, состоящий из главной части программы (main) и двух подпрограмм (proc1 и proc2). Дидактическая задача, которую решает программа-игрушка Lightbot, состоит в освоении ребенком основы составления линейных программ, включая вспомогательные программы. Система команд робота, однако, не охватывает все доступные дошкольникам базовые концепции программирования. Существенный недостаток программы состоит в том, что у робота отсутствует пульт управления, используя который ребенок при непосредственным управлением мог бы освоить основные действия робота. Виртуальная среда, в которой «живет» Фонарщик, сложна, и попытки ее реализации в реальном мире не проводились.

Scratch Junior – на сегодняшний момент одна из популярных сред бестекстового программирования [379], Рисунок 18.

Рисунок 18 – Экран бестекстовой среды программирования Scratch Junior. В нижней строке показаны пиктограммы команд персонажа Котенок

Персонажи, события и процессы, моделируемые в среде Scratch Junior и показываемые на экране, носят игрушечный характер, развиваются в условных двумерных обстановках, и потому задача материального воплощения этих обстановок разработчиками Scratch Junior не ставилась. Дети, начинающие знакомиться с программированием в среде Scratch Junior, с первых шагов погружаются в воображаемый виртуальный мир и, как правило, остаются в нем в течение всего периода обучения. Как исключение, в среде Scrath Junior можно составлять программы, управляющие несложными роботами-игрушками, но без взаимодействия этого робота с виртуальными обстановками и персонажами Scratch Junior. Составление программ управления реальными роботами-игрушками в Scratch Junior радикально отличается от программирования воображаемых двумерных персонажей, ради чего и создавался Scratch Junior [291].

Исследования [352; 56; 376] показывают, что для визуального программирования удобен не Робот из ЦОС КуМир, а упрощенная модель с ограниченной системой команд (всего четыре) – Вертун из ЦОС ПиктоМир, находящийся на космической платформе робот, к задачам которого относится не только передвижение по космодрому, но и ремонт поля после разрушений, оставленных взлетающими космическими кораблями, Рисунок 19.

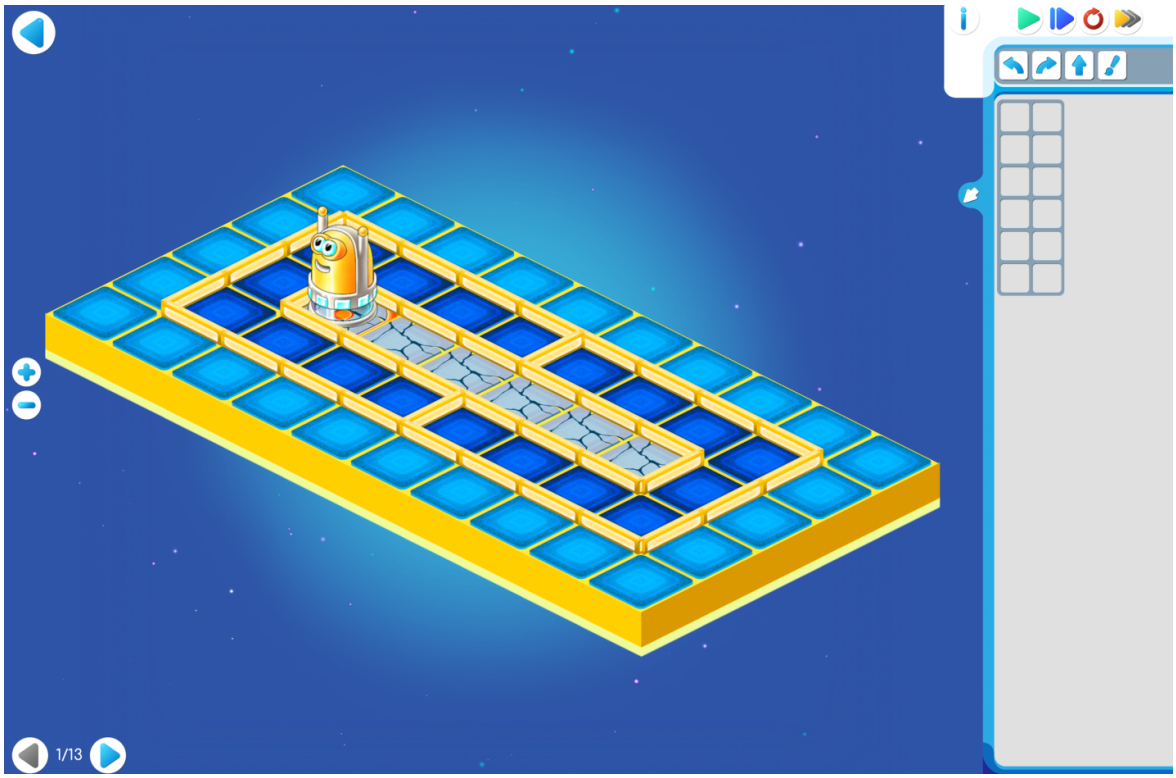


Рисунок 19 – Пример задания ЦОС ПиктоМир с роботом Вертун

Поэтому в набор команд Вертуна включены не только команды перемещения по клеткам поля, но и команда восстановления клетки, - **Закрасить**, которая знакома ребенку с детства и изображена иконкой – кисточкой. Кисточкой с краской пользуются, когда хотят нарисовать картину или сделать ремонт в комнате. Вертун очень экономный робот, если дважды **Закрасить** одну и ту же клетку, то робот потратит всего одну порцию краски, понимая, что покрашенное не надо красить.

Вертун ходит только вперед, в сторону, куда обращены его глаза-камеры. Вертун назван так потому, что для изменения направления движения ему нужно повернуться в нужном направлении по часовой стрелке **Повернуть направо** или **Повернуть налево** (против часовой стрелки). Для начинающего программиста робот прост для понимания. Единственная сложность, которая трудна для новичков, – это пространственное восприятие мира Вертуна, когда для выбора направления нужно изначально повернуться в нужную сторону, что не всегда дается детям с первого раза.

Процесс составления алгоритмов учащимися в ЦОС ПиктоМир достаточно прост и естественен, что гарантируется практически нулевыми затратами на изучение функций робота. Все иконки с управляющими командами находятся на полочке, откуда ученик с помощью указания или перетаскивания может переместить их в поле, где находится сама программа (шаблон программы). Чтобы ученику не нужно было далеко тянуться, команды можно копировать из одной клетки шаблона в другую.

Специально, чтобы не нагружать опорно-двигательный аппарат малышей, имеется два способа наполнения шаблона программы. Первый способ состоит в стандартной операции на планшете - перетаскивании иконок из одного места экрана в другое. Для детей младшего возраста такая операция может быть затруднительной. Поэтому можно использовать метод выделения пиктограммы на полочке, при этом пиктограмма начинает «дышать», увеличиваясь и уменьшаясь в размере, тем самым показывая, что ее выбрали. Далее достаточно указать место, куда должна быть поставлена это пиктограмма, и она переместится в указанное место программы.

Шаблон программы имеет определённый размер, что позволяет заранее просчитать учащемуся, насколько длинную программу он может составить, так как добавление пиктограмм возможно, пока в шаблоне для алгоритма есть свободные позиции.

Чтобы удалить лишнее иконки из программы, нужно просто выделить эту иконку и «выбросить» за пределы шаблона. Конечно, методически правильнее было бы вернуть ненужную пиктограмму на полку, что соответствует принципу «взял-верни на место», что допустимо в ЦОС ПиктоМир. Однако, как правило, выбор ребёнка падает на наиболее простое решение - выбросить пиктограмму за пределы шаблона задачи.

Все действия ученика просты и естественны для любого начинающего программиста и доступны даже дошкольникам, начиная с четырёх - пяти лет.

Важно отметить, что составлению программ предшествует привычное ребенку пультовое управление роботами-исполнителями в ЦОС ПиктоМир. В этом случае команды не записываются в программу, а немедленно исполняются. Ребенок наблюдает за результатом выполнения каждой команды и принимает решение, выбирает следующую команду, точно зная, в какой обстановке и каком положении робота будет выполняться выбранная команда. Для облегчения последующего перехода к составлению программы записывается протокол нажатия кнопок при непосредственном выполнении. Этот дидактический прием основан на введении понятия «копилка» – прозрачный «стакан», где накапливаются все выполненные роботом команды, то есть при непосредственном управлении фактически формируется программа решения поставленного задания. Если учеником совершена ошибка, нажата не та клавиша, то в любой момент можно «откатиться назад», то есть вернуться к обстановке и позиции робота на предыдущем шаге. Такой подход позволяет продемонстрировать один из главных принципов программирования, что к решению задачи можно идти через некорректные промежуточные результаты, что позволят начинающему программисту не бояться «сломать» виртуальных исполнителей, в отличие от манипуляции с реальными объектами. Программа, накопленная в «копилке», может быть перенесена в шаблон программы, что позволяет не только получить визуальное пиктографическую запись программы, но и выполнить ее, проверив, что задача действительно решена, Рисунок 20.

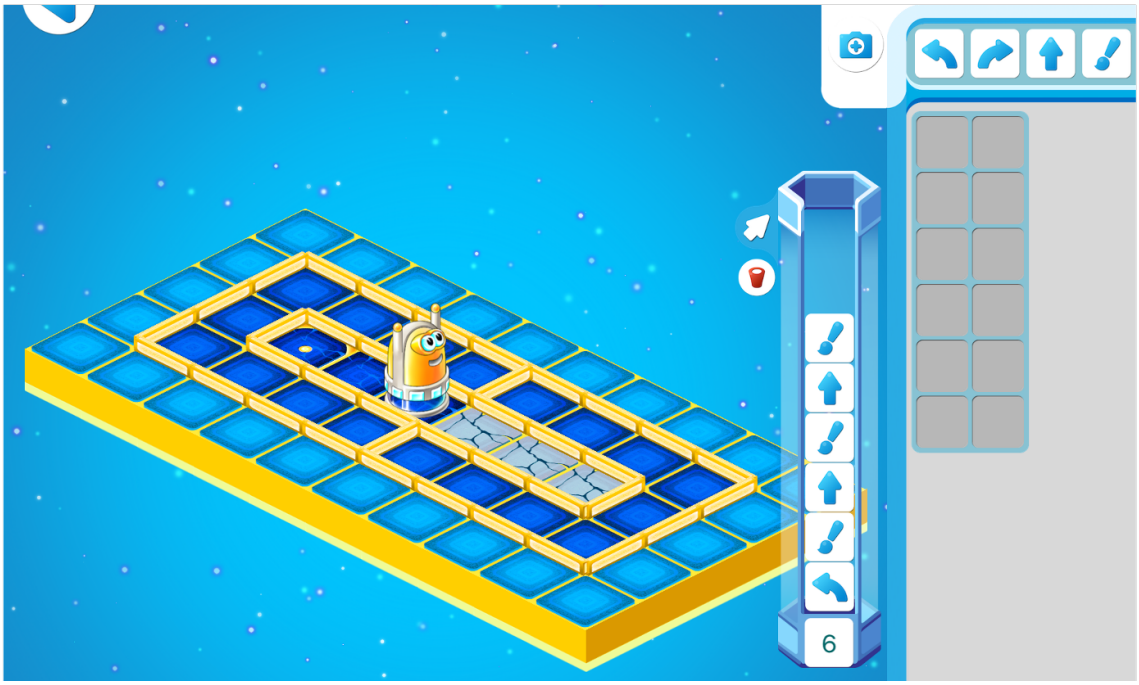


Рисунок 20 – Экран ЦОС ПиктоМир с открытой копилкой

Описанный режим становится важным переходным этапом между непосредственным и программным управлением. Непосредственное управление с пульта просто и понятно ребенку, но оно само по себе не формирует представлений о программировании как о предварительном планировании, не учит заранее думать о следующих действиях. «Чистое» программное управление, когда при составлении программы все следующие действия приходится держать «в уме», оказывается слишком сложным. Копилка позволяет сделать этот переход плавным: управление ведется в непосредственном режиме, но, когда задача решена, команды из копилки можно перенести в программу и увидеть, как робот решает задачу, выполняя все команды одну за другой без вмешательства человека. Так формируется первоначальное представление о программном управлении.

В дальнейшем работу с копилкой удобно использовать для разработки и отладки отдельных фрагментов сложных задач, а также для объяснения новых элементов, например, циклов с повторителем.

Можно провести аналогию между двумя видами речи (устной и письменной) и двумя видами алгоритмических взаимодействий: непосредственное (пультовое) и косвенное (программное). Пультовое управление роботами - аналог говорения в устной речи. Имитация робота - аналог понимания взрослого в устной речи. Имитация выполнения команд-вопросов роботом - аналог диалога. Поскольку используется принцип, что роботы просты, то дети легко осваивают «устное пультовое диалоговое алгоритмическое общение с роботами-исполнителями команд», что готовит их к «письменной алгоритмической речи» - программированию.

Составление программы - аналог письменной речи. Письменная речь и программирование трудны для новичков по причине отложенности и необходимости представить себе ситуацию, в которой участвуют объекты, не присутствующие в момент написания текста или составления программы.

Таким образом, для успешного формирования основ алгоритмического мышления в раннем возрасте нужно не жалеть времени и усилий на твердое освоение устной алгоритмической речи, прежде чем переходить к письменной речи. Если в процессе освоения письма дать ребенку диктофон, то для него сильно облегчится задача генерации письменного текста, поскольку часть из него ребенок задаст в привычной устной форме. Точно также в процессе составления программы нужно дать ребенку способ создать фрагмент программы в режиме устной алгоритмической речи, в режиме пультового управления, то есть вернуться на мгновение в состояние диалоговых взаимодействий. Дидактически это обеспечивается «копилкой» ЦОС ПиктоМир.

Кроме главного алгоритма, с которого всегда начинается выполнение программы, в ЦОС ПиктоМир можно использовать вспомогательные алгоритмы. У простого Вертуна всего четыре простых команды, но в дальнейшем юный программист сможет узнать, что Вертун достаточно умный робот. Так, например, он может выяснить и даже сообщить компьютеру, свободен ли путь в ту клетку, в которую он хочет переместиться, или этот путь загорожен стеной, или здесь

заканчивается космодром. Вертун умеет отвечать на вопросы компьютера и понимать, на какой клетке он стоит на поле. Робот может ответить не только свободна ли клетка перед ним (пиктограмма с зеленой стрелкой), или ответить на вопрос, есть ли перед ним препятствие (пиктограмма с кирпичной стеной), но и стоит ли он на голубой клетке (пиктограмма с голубой клеткой) или стоит ли он на синей клетке (пиктограмма с синей клеткой), а также ответить на вопрос: клетка, на которой стоит Вертун, не голубая (пиктограмма с голубой клеткой, перечеркнутая красной чертой), или клетка, на которой стоит Вертун, не синяя (пиктограмма с синей клеткой, перечеркнутая красной чертой), Рисунок 21.

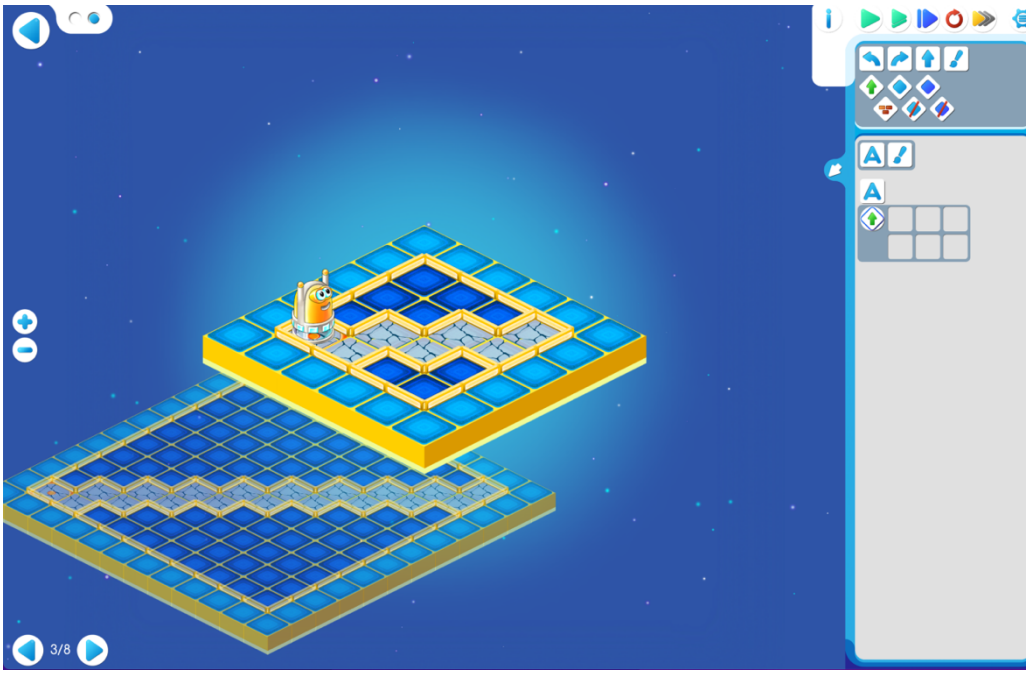


Рисунок 21 – Пример сложного задания ЦОС ПиктоМир с роботом Вертун

При работе с ЦОС ПиктоМир младшеклассник или дошкольник может сосредоточиться на алгоритмических проблемах решения поставленной перед ним задачи, а не на изучении системы программирования и языка программирования. Действительно, пиктографическая система программирования не только не требует от ребенка детального знакомства с новым для него языком, но и воспринимается ребенком вполне естественно, как продолжение игр пультового управления. Пультовое управление компьютерными героями сейчас осваивают гораздо раньше, чем алфавит, и использование уже полученных навыков для

обучения реальному программированию упрощает задачу восприятия нового материала [258].

Виртуальные роботы, которыми непосредственно управляет ученик, а затем переходит к программному управлению, также естественны и могут ассоциироваться со многими героями игр и мультфильмов [105]. После освоения курса пиктографического программирования роботов, в ходе которого учеником были изучены понятия вспомогательного алгоритма, конструкции ветвления и циклы, возникает естественная задача перехода на текстовую систему программирования.

ЦОС КуМир позволяет решать более широкий круг задач, управляя аналогичными виртуальными роботами-исполнителями. Ученику проще перейти к текстовому программированию, если программы будут составляться для тех же Роботов и решаться будут такие же задачи, алгоритм решения которых уже был придуман и реализован учеником для робота в ЦОС ПиктоМир.

Однако удобство пиктографического программирования является демотивирующим фактором в задаче перехода к текстовому программированию. Целостность ЦОС ПиктоМир создает желание расширить возможности пиктографического языка, оттянув как можно позже изучение текстового «взрослого» языка программирования [292].

Как правило, робот из ЦОС ПиктоМир (как и исполнители из миров ЦОС КуМир) лишен материального воплощения. Этот недостаток спокойно воспринимают старшие школьники, не говоря уже о студентах вузов. Но для детей нужно предложить реального исполнителя, которого можно потрогать и, конечно, запрограммировать.

3.3 Использование методики обучения в игровой форме с материальными объектами в пропедевтике программирования для дошкольников

Использование методики обучения в игровой форме с материальными объектами помогает детям в освоении основных понятий последовательного программирования. Игры с материальными объектами являются интерактивными и визуально привлекательными, что значительно повышает мотивацию учеников. Наблюдение за действиями роботов в реальной обстановке, непосредственное управление ими и составление простейших программ помогают ученикам формировать основы алгоритмического мышления. Этот процесс соответствует теории поэтапного формирования умственных действий П.Я. Гальперина, поскольку он предполагает неразрывную связь между психическими процессами и внешними действиями. Ученики, наблюдая за действиями роботов и управляя ими, учатся понимать, как работают алгоритмы и как они реализуются в реальном мире.

Как показал массовый опыт проведения курсов по основам алгоритмизации в дошкольных образовательных организациях, формирование алгоритмического мышления можно с успехом начинать с раннего возраста.

Информатизация дошкольного образования – очень широкая и важная для современного мира тема. Работа над курсом программирования для дошкольников явилась первым этапом разработки 4-х летнего курса программирования для дошкольников и младших школьников [126].

Успешность внедрения курсов во многом была обеспечена использованием уже на первых занятиях определенных методических и технических решений, как использование виртуальных и реальных роботов в процессе знакомства детей с основами алгоритмизации и программирования.

Как было предложено Симуром Пейпертом [177] еще полвека назад, дети работают не только с виртуальными (экранными) роботами, но и с реальными роботами-игрушками, которые перемещаются по полу игровой комнаты, имитируя перемещения виртуальных роботов на экране планшета. Мировой опыт знает

некоторые такие решения, в которых дети изучают элементы программирования на материальных объектах. Ниже приведены некоторые из них.

Робототехнический набор Робомышь [141] позволяет детям освоить некоторые концепции программного управления. На корпусе реального робота-игрушки Робомышь размещены 7 кнопок с командами, которые умеет выполнять робот. Ребенку показывается игровое поле, на котором размещены Робомышь, препятствия и цель - «кусочкек сыра». Ребенок должен придумать последовательность команд, которые нужно дать Робомыши, чтобы она дошла до сыра. Придуманную последовательность команд – программу движения Робомыши – ребенок с помощью карточек с пиктограммами команд выкладывает на столе и затем вручную «загружает» эту программу в память Робомыши, нажимая на кнопки на ее корпусе. Далее Робомышь самостоятельно проходит запланированный ребенком маршрут, Рисунок 22.

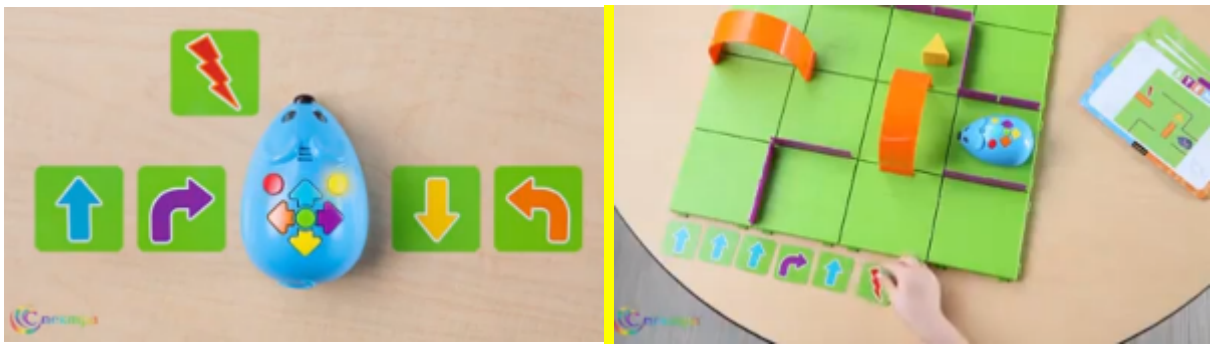


Рисунок 22 – Слева робомышь, на корпусе которой 7 кнопок с командами, справа – игровая обстановка и программа из карточек, которую выкладывает ребенок

Данный робототехнический набор помогает детям освоить общую идею программного управления, но только на простейших примерах. Еще одним недостатком данного набора, затрудняющим освоение детьми идеи программного управления, является тот факт, что «загруженная» в память мыши программа не наблюдается детьми. Наконец, с помощью Робомыши нельзя познакомиться с концепциями «подпрограмма», «ветвление», «цикл». То есть Робомышь полезна и

эффективна, но только на первых шагах знакомства с элементами программирования [366].

Робототехнический набор Matatalab, позволяющий создавать и выполнять программы в материальном мире [308].

На рисунке, Рисунок 23, показаны управляющая башня набора и выложенная на пьедестале башни программа и игровое поле, по которому движется управляемый башней робот («водитель» виден через прозрачный оранжевый купол водительского отсека). Выше игрового поля показан крупным планом фрагмент управляющей программы – подпрограмма с именем f0.

Рисунок 23 – Робототехнический набор для безэкранного программирования

Набор Matatalab позволяет составлять программы управления без компьютера, перемещая в материальном мире карточки, на которых изображены пиктограммы команд. При нажатии на кнопку пуск компьютер управляющей башни «смотрит» на программу, «загружает» ее в свою память и выполняет. Набор позволяет освоить концепции системы команд робота, программного управления и подпрограммы, набрать опыт составления программ. К его недостаткам следует отнести бедный набор поддерживаемых базовых конструкций программирования, что не позволяет с помощью данного набора освоить весь набор конструкций структурного программирования.

Интегрированный робототехнический набор Lego We Do 2.0 включает и компоненты для сборки простейших роботов, и смешанную пиктограммно-текстовую среду программирования собранных роботов [346]. Однако эта среда программирования специализирована, содержит много сложных команд, позволяющих управлять аппаратными компонентами, включенными в набор. Тем самым набор нацелен на изучение детьми функций и взаимосвязей основных компонентов робототехнических систем, а не на освоение основ программирования на примере робототехнических систем. Поэтому набор хорош для изучения азов робототехники детьми возраста 8-12 лет, но для систематического освоения программирования неэффективен.

Робототехнический набор Роббо [307], состоящий из мобильной расширяемой колесной платформы и так называемой Роббо-лаборатории, Рисунок 24.

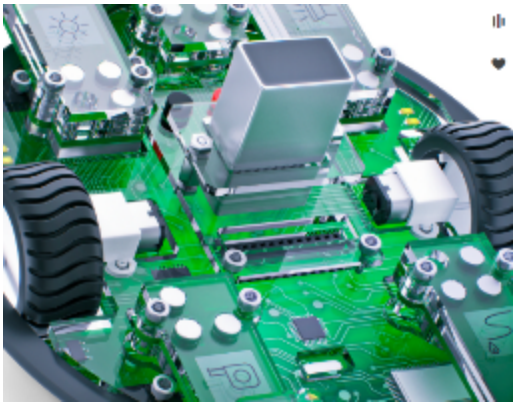


Рисунок 24 – Колесная платформа Роббо

При использовании платформы Роббо дети фактически имеют дело с одним единственным готовым, уже собранным колесным вездеходом. Никакая деятельность по механической сборке движущегося робота, подобная сборке различных роботов Lego из типовых механических элементов, в наборе Роббо не предусмотрена. К готовой колесной платформе ребенок может подключить с помощью быстросъемных магнитных разъемов один, два или более датчиков, включенных в набор его изготовителем. Движением платформы с использованием

информации от установленных на ней сменяемых датчиков можно управлять с помощью программы, созданной в учебной среде программирования Scratch или даже на производственном языке программирования C++. Роббо-лабораторию можно подключить по USB-кабелю к любому компьютеру и превратить в пульт управления мобильной платформой или любыми персонажами программной среды Scratch [380].

В пропедевтическом курсе алгоритмики в цифровой образовательной среде (ЦОС) ПиктоМир, в рамках которого дошкольники и младшие школьники осваивают основные понятия последовательного программирования, упор делается на составление программ для управления реальными и виртуальными роботами-исполнителями. При обучении программированию дошкольников и младших школьников ребенка не только учат составлять простейшие программы управления реальными роботами-игрушками или виртуальными роботами на экране планшета, но и формируют в его голове модель мира, в котором робот, программист, программа и компьютер тесно взаимодействуют между собой по понятным, четко установленным правилам.

На начальном этапе этого освоения важнейшую роль играют материальные объекты - учебные пособия, такие как

- игровое поле, составленное из разноцветных сочленяющихся квадратных мягких ковриков размерами, достаточными, чтобы на одном коврике мог стоять 6-летний ребенок, который используется для настоящего робота, Рисунок 25.

- реальный робот-игрушка, Ползун, перемещающийся по игровому полю, повинуясь командам, подаваемым в форме слышимых детьми звуковых сигналов с помощью звукового пульта или планшета с установленным на нем ЦОС ПиктоМир.

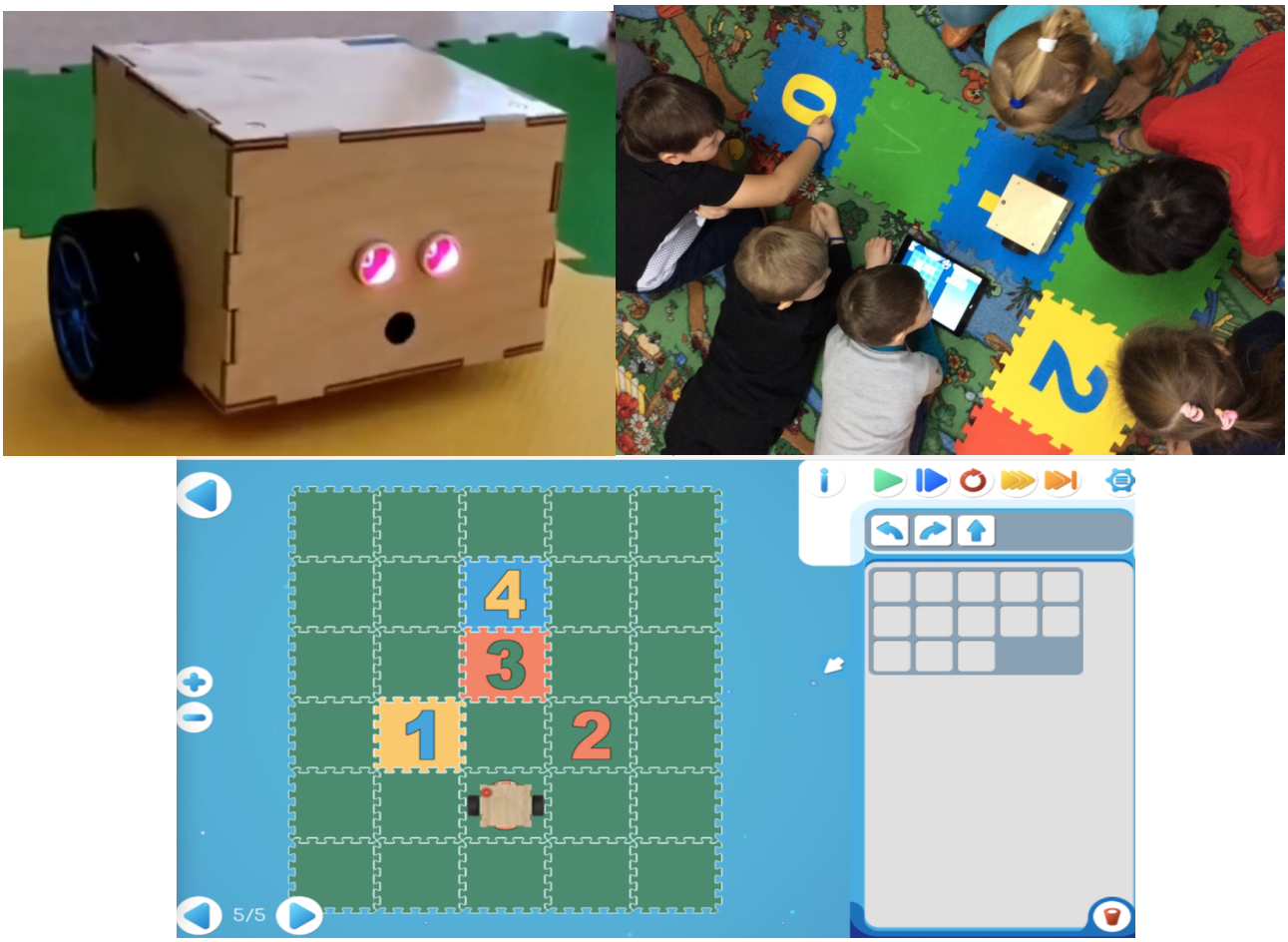


Рисунок 25 – Реальный робот «Ползун» на реальном игровом поле из ковриков

Робот Ползун имеет ограниченное число команд, три команды движения **Вперед**, **Повернуть направо**, **Повернуть налево**. Ползун передвигается по полю, составленному из ковриков и являющимся обстановкой для этого робота. Ползуном можно управлять в непосредственном режиме с помощью пульта. В ЦОС ПиктоМир есть программная модель Ползуна. Можно составить программу и управлять Ползуном на экране так же, как, например, с Вертуном. Дети могут самостоятельно или с помощью преподавателя составлять на планшетах программу управления Ползуном.

При помощи планшета с готовой программой в ЦОС ПиктоМир и реального Ползуна после сопряжения экранного робота Ползун с реальным можно выполнить составленный алгоритм. Компьютер по очереди передает роботу команды

программы, экранный Ползун выполняет их на экране, а реальный робот передвигается на поле из ковриков. При этом детям слышно, как именно компьютер передает команды: каждой команде соответствует свой звук, Ползун тоже «слышит» эти звуки и, подчиняясь им, движется вперед и выполняет повороты. Этот наглядный пример очень важен для мотивации и для лучшего понимания сути программного управления: ребенок видит, как компьютер управляет *реальным* (существующим отдельно от компьютера, не на экране) роботом по заранее составленной программе.

На каких принципах и как функционирует система беспроводной связи, трудно понять не только детям, но и взрослым, поэтому реальные роботы, совместимые с ЦОС ПиктоМир, управляются звуковыми командами. В отличие от ненаблюдаемых команд беспроводного общения, звуки, издаваемые роботом, слышны детям, и им становится понятно, как робот получает команды от компьютера.

Выбирая методику и способ взаимодействия робота и компьютера, для использования в имитации детьми робота и компьютера необходимо предложить способ, которым роботу даются команды. Естественно, что в материальном мире команды роботу должны даваться с помощью каких-либо физических воздействий. Методически требуется, чтобы и ребенок на занятиях в игре мог исполнять роль такого робота, при этом в качестве команд нужно выбрать воздействия, воспринимаемые человеком. Таким образом, в качестве физических процессов, доставляющих команды роботу, не применимы ультразвук, радиоволны и инфракрасные лучи, поскольку человеком они также не воспринимаются. Среди важных органов чувств человека - зрение и слух - поэтому наиболее удобным оказалось подавать роботу команды в звуковой форме.

Современные технологии позволяют снабдить игрушечного робота механизмом распознавания коротких одно-двухсловных команд, то есть можно было бы выбрать для Ползуна звуковые команды в речевой форме. Однако дидактически этот способ неприменим, так как создает путаницу и входит в

противоречие с опытом ребенка. Итак, когда дети играют в игру Робот и Командир, в которой один ребенок (Командир) должен командовать другим ребенком, имитирующим Ползуна, у первого ребенка возникает соблазн давать речевые команды типа «шагай» или «два шага вперед», которые второй ребенок прекрасно понимает и начинает выполнять. В этом случае приходится тратить определенные усилия, чтобы вернуть детей в «игру по правилам» и объяснять, что таких команд у Ползуна нет. Избежать этой ненужной путаницы помогает звуковое кодирование команд робота. Если команды кодируются, то возникают и правила, которые описывают множество из трех команд Ползуна, и в этом перечне команда «шагай» просто не предусмотрена и дать ее Командир не может.

В проведенном исследовательском эксперименте были испробованы два способа кодировки команд робота звуковыми сигналами.

Первый способ – музыкальное кодирование. Для корректного выполнения программы человеком или компьютером использовался «музыкальный» способ с помощью кодирования команд робота Ползуна определенными нотами первой и второй октавами (ми второй октавы, ля первой октавы и ля второй октавы), которые проигрывались одинаковой длительностью:

- Вперед – ми второй октавы;
- Повернуть налево – ля первой октавы;
- Повернуть направо – ля второй октавы.

Дети слышали, как компьютер подавал команду роботу, и видели, что робот ее выполняет. Если же роботом управляет человек, то для подачи роботу музыкальной команды он должен проиграть или пропеть требуемую ноту.

На практике оказалось, что хотя дети и способны различать высоту тона музыкальных нот, но это задача является дополнительной нагрузкой на ребенка и воспитателя.

Второй выбранный на практике способ кодировки похож на азбуку Морзе. Три команды Ползуна кодируются однократным, двукратным или трехкратным

проигрыванием одной и той же ноты, скажем, «камертонной ноты» ля первой октавы (440 герц). Такая кодировка легко понимается и осваивается детьми.

Робот Ползун материален, и поэтому у детей не вызывает затруднений понимание того, что робот кем-то изготовлен и в него заложены возможности выполнения конечного числа команд. Не вызывает затруднений и понимание того, что есть тип робота Ползун, а есть конкретный экземпляр робота данного типа. и понимание того, что два экземпляра Ползуна, услышав команду, начнут ее выполнять параллельно. Хотя Ползун и не понимает речевые команды, в него встроен динамик и речевой генератор. Получив команду, Ползун голосом рапортует, какую именно из трех команд он начинает выполнять, а по завершении выполнения команды произносит «Готово». В дидактических и свободных играх дети начинают мгновенно копировать данное поведение робота. Дети с удовольствием командуют реальным Ползуном и друг другом. В игре компьютер может командовать одновременно и реальным роботом Ползуном, и экранным роботом-Ползуном, и ребенком, имитирующим Ползуна.

3.4 Составление программ из материальных объектов как бескомпьютерная форма методической системы обучения

Составление программы из материальных объектов, как бескомпьютерная форма методической системы обучения, представляет собой инновационный подход к обучению основам программирования, который сочетает в себе принципы наглядности, интерактивности и практико-ориентированного подхода. Этот метод основан на использовании физических объектов – кубиков с нанесенными на них знаками-командами пиктографического языка Пикто средства обучения – ЦОС ПиктоМир для составления программ, имитируя таким образом работу программиста на компьютере без необходимости непосредственного постоянного использования компьютера или других электронных устройств.

Работа детей дошкольных образовательных организаций с электронными средствами обучения ограничены санитарно-эпидемиологическими требованиями

[216]. Для детей средней и даже старшей группы использование планшетов невозможно. Поэтому в результате экспериментов было предложено составлять программы из материальных объектов, – *пиктокубиков*, – деревянных кубиков со стороной 4 см и нанесенными на них пиктограммами команд ЦОС ПиктоМир.

На образовательный процесс также накладываются эргономические ограничения опорно-двигательного аппарата детей. Но подобранный размер кубиков позволяет даже детям трех-четырёх лет складывать программы из пиктокубиков, дополнительно развивая мелкую моторику. Дети выкладывают программу из материальных кубиков для управления виртуальными роботами не на экране компьютера-планшета, а на столе. Для исполнения программы используются технологии распознавания фотографии программ, составленных учащимися, Рисунок 26.

Загрузка программы в память компьютера – аналог процесса запоминания программы ребенком – состоит их двух этапов: сначала ребенок фотографирует составленную программу компьютером (планшетом), Рисунок 27, затем фотография программы из пиктокубиков на столе распознается процессором планшета с помощью технологии искусственного интеллекта.

Рисунок 26 – Распознавание ЦОС ПиктоМир фото программы из кубиков

Рисунок 27 – Ребенок фотографирует составленную программу планшетом

Вычислительные мощности, достигнутые современной микроэлектроникой, позволяют использовать для решения широкого круга задач не только суперкомпьютеры, серверы и десктопы, но и мобильные системы, смартфоны и планшеты. Этап распознавания, хотя и достаточно сложен технологически [155], однако на современных планшетах этот процесс интегрирован в ЦОС ПиктоМир.

Этот метод позволяет детям создавать программы, используя материальные объекты, такие как пиктокубики или карточки с пиктограммами с целью составить

последовательность шагов программы, что помогает визуализировать и понять алгоритмы, не прибегая к использованию компьютера.

Используя пиктокубики, дети могут создавать последовательности команд, которые затем, распознанные компьютером, интерпретируются как программа ЦОС ПиктоМир. Кубики удобнее, чем карточки, так как на одном кубике на 6 сторонах могут располагаться 6 различных пиктограмм-команд робота. Используя один вариант кубика, можно составить различные программы для робота. Такой подход, объединяющий действия с материальными объектами и программирование, помогает развить не только навыки логического и алгоритмического мышления у детей, но и добавить к ним закладку в виде мышечной памяти по работе с пиктокубиками, что является более яркой картинкой деятельности, сохраняющейся в памяти длительное время.

Конечно, есть определенные требования расположения кубиков на столе, когда требуется соблюдать определенный порядок и выравнивание в рядах. Если компьютер не в состоянии «понять» (распознать) предлагаемую программу, то для ребенка дидактически это является эквивалентом плохого подчерка или синтаксической ошибки в предложении и здесь, в программе.

В правила составления программ на начальных занятиях добавляются только три конструкции:

- числовой повторитель,
- вспомогательный алгоритм-подпрограмма с однобуквенным именем,
- команда **Мигнуть** (подождать).

Процесс составления и выполнения программы состоит из нескольких шагов:

1. Ученик получает задачу с шаблоном программы.
2. Ученик собирает программу из кубиков для решения задачи.
3. Воспитатель или ученик (в зависимости от его возраста) показывает программу компьютеру. Распознавание происходит в реальном времени, и на пиктокубиках видны названия распознанных пиктограмм. Если этот этап

выполняет ребенок, который еще не умеет или не любит читать, то успешность распознавания можно понять, если каждый кубик попал в свой квадрат.

4. Воспитатель или ученик фотографирует программу из пиктокубиков.

5. Программа, понятая компьютером, заполняет шаблон (или часть шаблона, если создается фрагмент);

6. Ученик проверяет результат выполнения программы.

Для использования в конкретной программной ЦОС ПиктоМир было проведено обучение нейронных сетей на распознавание используемого в ЦОС ПиктоМир набора из 18 пиктограмм, используемых в двумерных структурах, реализующих две управляющие конструкции ЦОС ПиктоМир: числовой повторитель и подпрограмма с фиксированным однобуквенным именем. В ЦОС ПиктоМир используется Питоно-подобный метод кодирования управляющих конструкций с помощью отступов.

Набор 18 распознаваемых пиктограмм состоял из

- четырех пиктограмм команд робота Вертуна (вперед, повернуть налево, повернуть направо, закрасить);
- девяти пиктограмм числовых повторителей, 1 раз, 2 раза, 3 раза, 4 раза, 5 раз, 6 раз, причем повторители с двумя и тремя точками, а также повторитель шесть представлены 2 подтипами пиктограмм, чтобы при распознавании программы ориентация пиктограммы повторителя не влияла на результат работы;
- трех вспомогательных алгоритмов с именами А, Б и В;
- команды тянуть Тягуна;
- команды мигнуть (подождать).

Программа из деревянных пиктокубиков выкладывается на плоской поверхности, например, на доске или на столе. Дидактически, удобно иметь коробки, повторяющие шаблон задания, либо универсальные, рассчитанные на 3-4 кубика в длину и 5-6 в ширину. Тем самым исключаются «пляски» кубиков один относительно другого, что улучшает качество распознавания.

Распознавание пиктограмм происходит посредством соответствующей обученной нейронной сети. Далее происходит привязка пиктограмм к структуре шаблона, находится комбинаторная структура их взаимного расположения в виде двумерной таблицы с помощью нейротабулятора [128].

Однако может оказаться, что ученик выложил программу (или часть программы) не по шаблону или выложил недостаточное/излишнее количество пиктограмм. Для коррекции результатов распознавания программы таблица склеивается построчно с исключением пустых ячеек из склейки. В итоге получается непрерывная одномерная последовательность пиктограмм, которая затем пытается разместиться в шаблоне.

Для упрощения восприятия ребенком связи реального и виртуального мира роботов требуются материальные игровые компоненты, к которым относятся не только пиктокубики и магниты с изображением пиктограмм ЦОС ПитокМир, но и мягкие игрушки, реальные роботы и среда их обитания – резиновые сочленяемые коврики. Все это является предметно-игровой средой и позволяет построить пропедевтический курс программирования для малышей, начиная с 4-ого года жизни, когда уже на первых занятиях дети играют со сверстниками и материальными учебными пособиями (роботом и набором кубиков с пиктограммами команд) в ролевые или сюжетно-ролевые игры.

При этом компьютер, в соответствии с требованиями санитарных правил, используется только по своему прямому назначению - для выполнения загруженных в его память программ и не используется для других целей: ни для составления программ, ни для генерации изображений виртуальных роботов и виртуальных обстановок на экране. На первых занятиях курса программа, робот и обстановка, в которой робот действует, являются реальными, а не виртуальными объектами. Ребенок может выступать в роли робота, исполняя поступающие звуковые команды от компьютера или от другого ребенка, выступающего в роли компьютера-командира. Он может также работать программистом, самостоятельно

составляя программу, перемещая материальные кубики на столе и переходя позднее к составлению программ путем псевдоматериального перемещения пиктограмм на сенсорном экране планшета. В этом случае он может выступить в роли компьютера, выполняя составленную другим ребенком программу и командуя при этом другим ребенком, играющим роль робота и т. п.

Важно, чтобы по мере того, как основные понятия программирования осваиваются на интуитивном уровне при работе на планшетах, во время бескомпьютерных занятий происходил переход этих интуитивных представлений в вербальную форму. Дети под руководством педагога должны обсуждать значения таких терминов, как программист, робот, программа, значения фраз типа «я выполнил программу, которую составил Коля», «программа составлена из 6 пиктограмм», «робот выполнил 10 команд». Это наполнение словарного запаса детей и развитие навыков монолога и диалога с использованием накопленных «профессиональных» терминов является столь же важной целью курса, как и обретение навыков самостоятельного составления простейших программ в предметно-игровой среде программирования.

При этом ключом к освоению сформулированной системы научных понятий программирования дошкольниками является интенсивная практика. Бóльшую часть курса занимает составление программ управления без обратной связи. Каждая такая программа решает одну конкретную задачу, обеспечивает адекватное поведение робота в одной единственной внешней остановке, предъявляемой ребенку на полу в игровой комнате или в графической форме на экране планшета. Программы без обратной связи составляются с использованием всего двух управляющих конструкций: числового повторителя и подпрограммы (вспомогательного алгоритма) с однобуквенным именем.

Так, в разработанной «поминутной» методичке годового курса «Алгоритмика для дошколят» [107] описаны 30 занятий, еще 4 занятия предусмотрены как резервные. Конструкция повторитель впервые появляется на занятии 10 и вводится как способ сокращения длины программы. Конструкция

подпрограмма впервые появляется на занятии 15 и вводится как способ «шифрования» фрагментов программы. Естественно, что разнообразие программ, которые можно составить с использованием двух указанных управляющих конструкций, невелика, это класс линейных программ. Однако с помощью этих линейных программ можно описать поведение достаточно сложных объектов – роботов.

Педагогическая практика показала, что набора содержательных задач, решаемых с использованием этих двух управляющих конструкций, достаточно для удержания внимания детей в течение года (30 занятий) при условии наличия реального робота и виртуальных представлений роботов. В курсе «Алгоритмика для дошколят» используются 5 виртуальных роботов и 1 реальный робот (Ползун), имитирующий одного из виртуальных.

Вариативность компьютерной части занятий способствует коллективному (кооперативному) программированию. Члены команды сидят за одним столом, но каждый работает на своем планшете, составляя свою программу действий своего робота, решая только ему порученную задачу. Эти программы далее выполняются параллельно, и каждый участник видит результаты этого параллельного выполнения на своем экране. В процессе отладки составленных программ может потребоваться изменение самих программ, написанных разными членами команды, или пересмотр правил «дележки» работы между членами команды.

Несмотря на весьма малую продолжительность компьютерной части каждого занятия курса – от 15 минут в первом полугодии до 20 минут во втором – учитель добивается определенного результата, когда каждый ребенок на каждом занятии выполняет как минимум 4-6 заданий, то есть в годовом курсе каждый дошкольник самостоятельно составляет как минимум 120-150 простейших программ. Самостоятельное выполнение более сотни упражнений является необходимым условием устойчивого освоения теоретического и практического материала курса.

В конце первого года обучения на последних занятиях начинается (но не завершается) переход от управления без обратной связи к управлению с ее использованием:

- в системы команд роботов вводятся команды-вопросы, и в предоставляемые ЦОС ПиктоМир-Конструкции языка программирования добавляются ветвление и повторение;

- в систему основных понятий вводится новый вид команды – команда-вопрос и новый вид взаимодействия: робот отвечает на команду-вопрос компьютера «да» или «нет».

3.5 Соревновательная методика как компонент пропедевтической методической системы обучения информатике и программированию дошкольников и младших школьников

Соревновательная методика, как компонент пропедевтической методической системы обучения информатике и программированию для дошкольников и младшеклассников, представляет собой инновационный подход к формированию алгоритмического мышления у детей. Эта методика обучения программированию для дошкольников и младшеклассников, основанная на использовании кооперативных и соревновательных элементов, представляет собой комплексный подход, направленный на стимулирование интереса и мотивации детей к изучению основ программирования и алгоритмизации, а также на развитие их коммуникационных навыков.

В России традиционной методически проработанной формой соревнований в соответствии с Федеральным законом № 273-ФЗ «Об Образовании в Российской Федерации» являются олимпиады по предметам [160].

Исследования в области педагогики и психологии [206] указывают на то, что коллективные занятия могут иметь значительное преимущество перед индивидуальными занятиями в образовательной среде. Когда учащиеся работают в группах, они могут обмениваться идеями, объяснять друг другу задание, задавать

вопросы, тем самым развивать навыки сотрудничества, кооперативного мышления, что способствует не только более глубокому пониманию осваиваемого материала, но и развитию социальных навыков.

При этом в коллективных занятиях могут использоваться два основных подхода: кооперативное обучение и состязательное обучение. В кооперативном обучении учащиеся работают вместе для достижения общей цели, обмениваясь знаниями и опытом. Этот подход способствует развитию коммуникативных навыков, уважению к точке зрения других и разделению ответственности. С другой стороны, состязательное обучение может стимулировать мотивацию учащихся через соревновательные элементы. Однако при этом важно учитывать индивидуальные особенности каждого ученика, чтобы избежать негативного влияния на самооценку. Таким образом, эффективное обучение часто включает в себя комбинацию кооперативных и состязательных методов, а также учитывает возрастные особенности учащихся для достижения оптимальных результатов. Такие кооперативные олимпиады по программированию называют *алгоритмиадами*.

В зависимости от цели и места включения алгоритмиад в образовательный процесс можно выделить различные их типы:

- алгоритмиады профессиональной ориентации, когда целью является знакомство с профессией программиста или со смежными специальностями для детей, планирующих связать свою жизнь с информационными технологиями в различных областях. Такие алгоритмиады полезны для выявления талантов в профессиональной направленности будущего программиста, а также получения элементов практического опыта решения творческих задач, которые не известно, как решать;
- алгоритмиады в кружковой деятельности в качестве цели ставят мотивацию детей, направленную на более глубокое изучение программирования в процессе коллективных занятий;

- алгоритмиады как часть образовательного процесса имеют целью при использовании элементов состязательной методики проверить сформированные знания, навыки и умения, повысить интерес обучаемых к регулярной учебной деятельности, а также осуществить ментальную подготовку к итоговым экзаменационным испытаниям, когда искать решение нужно в стрессовом состоянии в условиях ограниченности времени на ОГЭ и ЕГЭ.

После введения обратной связи становится возможным давать задачи на составление «универсальных» программ, работающих не в одной, а в нескольких разных остановках. Эти задачи также даются в виде картинки без словесного описания класса обстановок, в которых должна работать программа. На очередном уровне игры требуется составить программу, которая работает не в одной, как раньше, а в двух или трех заданных обстановках, Рисунок 28.

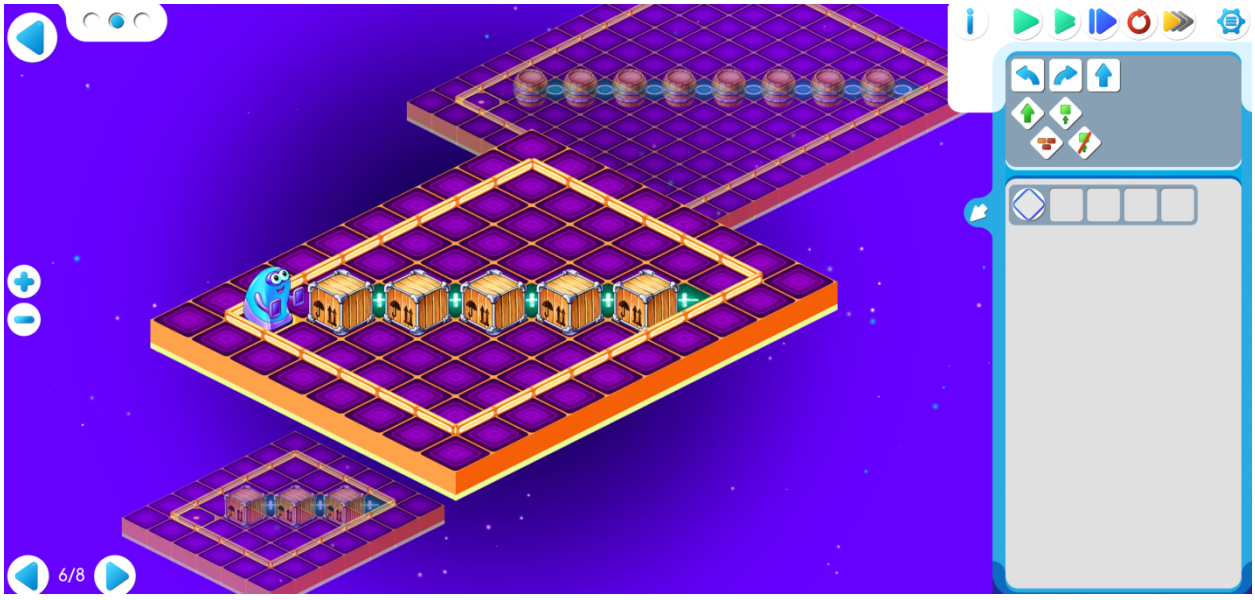


Рисунок 28 – Пример задачи в ЦОС ПиктоМир с роботом Двигун, требующей составления универсального алгоритма, так как предъявлены три различные обстановки

Опыт показал, что дети возраста 6-7 лет без труда и с энтузиазмом осваивают азы программирования с использованием описанного выше подхода и,

приходя в первый класс школы в рамках систематических занятий информатикой и программированием, способны освоить программирование в объеме основной школы. Это позволяет сделать вывод, что такое раннее освоение основ программирования является необходимым для всех без исключения детей.

Практический опыт показывает, что формирование основ алгоритмического мышления можно и нужно начинать в рамках дошкольного развития ребенка, причем возраст детей, включаемых в активности по освоению основ программирования, может варьироваться от подготовительной группы дошкольной образовательной организации до детей 4-ого года жизни [145]. Осторожное (в смысле контроля за здоровьем детей), но тем не менее интенсивное внедрение электронных средств обучения в образовательный процесс, непосредственными участниками которого являются дошкольники или младшие школьники, привело к необходимости созданию серии курсов «Алгоритмика для дошкольников» [258], знакомящих малышей с основами программирования. При этом каждое занятие разделено на компьютерную и бескомпьютерную части в соответствии с возрастными и ментальными ограничениями детей. При этом можно констатировать, что как индивидуальные, так и коллективные активности на занятиях без компьютеров обладают выраженным разнообразием. Это позволяет развивать алгоритмическое мышление у ребенка и использовать процесс этой подготовки не только для ускорения познавательного, речевого, коммуникативного развития, но и для духовно-нравственного воспитания. Схема проведения занятий в ЦОС ПиктоМир на планшете или другом персональном электронном средстве обучения, как правило, одна и та же. На компьютерной части занятия ребенок выполняет практические задания на программирование роботов, обычно индивидуально и изредка в командной форме. Практические задания по форме напоминают игру на компьютере, когда играющему нужно последовательно проходить уровни, собирать монеты или другие призы. Задания в ЦОС ПиктоМир, подготовленные педагогом для данного конкретного занятия, служат для освоения и(или) закрепления навыков и умений составления программ, как очередной шаг в

формировании алгоритмического мышления. Конечно, чтобы разнообразить занятия и повысить интерес ребенка к предмету, можно было бы варьировать сценарий и сами игровые ситуации, в которых действуют роботы ЦОС ПиктоМир, а также добавить действующих лиц, но, как правило, сценарии прохождения этапов игры стандартны и неизменны. В качестве постановки задачи игроку демонстрируется картинка, изображающая робота в конкретной обстановке. Ребенок должен составить и отладить программу управления роботом, чтобы внести в эту обстановку изменения, приводящие к решению поставленной перед учеником задачи.

Для упрощения создания алгоритма для решения задачи, как правило, предлагается шаблон программы, который, с одной стороны, ограничивает возможности по составлению программы, делая задачу выполняемой за счет фиксированного числа команд-пиктограмм, а с другой стороны, является подсказкой, Рисунок 29.

Рисунок 29 – Пример задания для робота Двигун ЦОС ПиткоМир. Нужно поместить ящик и бочку в клетки, отмеченные крестиком и кружочком соответственно. В этом задании можно использовать счетчик «Кувшин»

По такому же сценарию проводятся не только индивидуальные занятия, но и занятия в парах, вне зависимости от того, что это будет: обычный урок с кооперативной деятельностью или это будут соревнования, то есть кооперативно-

соревновательная часть курса. Постановка задачи в ЦОС ПиктоМир для кооперативной игры не отличается от индивидуальной игры, участники видят картинку, изображающую двух роботов на клетчатом поле. Эта картинка исчерпывающе описывает условия задачи: составить две параллельно выполняющиеся программы управления роботами таким образом, чтобы, работая совместно, роботы провели изменения в конкретной обстановке для достижения решения поставленной задачи. Из чего видно, что ЦОС ПиктоМир, базирующаяся на интересе детей к игре, тем не менее накладывает на компьютерные задания однотипную схему их проведения: составление и отладка детерминированной однопоточной или параллельной двухпоточной программы управления роботами, Рисунок 30.

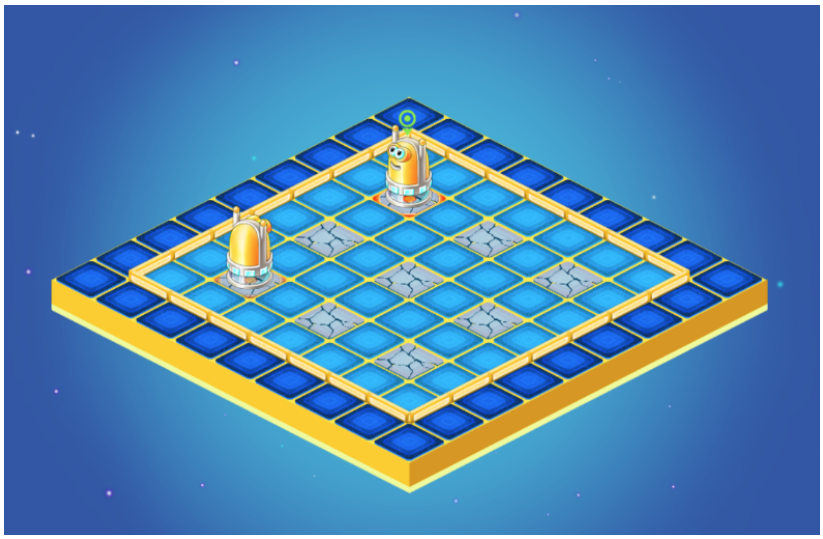


Рисунок 30 – Пример задания ЦОС ПиктоМир для двоих детей: Роботы Вертуны, управляемые каждый своей программой, должны совместно починить (закрасить) нечетное число поврежденных плит-клеток космической платформы

Указанная однообразность не является серьезным недостатком курса. Так, практический опыт показывает высокий уровень мотивации у детей во время компьютерных занятий в ЦОС ПиктоМир. Введение новых форм активности и нового типа задач поможет поддержать этот высокий уровень и вполне вероятно, что мотивация может даже повышена.

Учебные задания в пропедевтических курсах подобраны, как правило, так, чтобы начинающий программист смог справиться с решением задачи за урочное время. Тем не менее систематическое последовательное решение задач курса непосредственно участвует в формировании алгоритмического мышления ученика, прививая в том числе общеупотребимый стиль(и) программирования.

В производственном программировании, как правило, невозможно составить программу без использования вспомогательных алгоритмов, процедур функций. Естественно, что при этом программисту требуется разделить задачу на части.

На вузовских курсах по языкам программирования слушателям настоятельно рекомендуется декомпозировать алгоритм и пользоваться вспомогательными функциями. Но если задание простое, то мотивировать студента можно только декларативно, явно указав в задании, что не выделять подзадачу в отдельную функцию недопустимо. Такой подход вызывает внутреннее сопротивление у слушателя, так как иной мотивации для декомпозиции алгоритма преподавателем не предъявляется. Напротив, в ЦОС ПиктоМир, где задаются ограниченные шаблоны в заданиях, если линейное решение просто не помещается в главный алгоритм, ученик естественным образом мотивируется к разбиению задачи на подзадачи и поиску повторений в алгоритме, Рисунок 31.

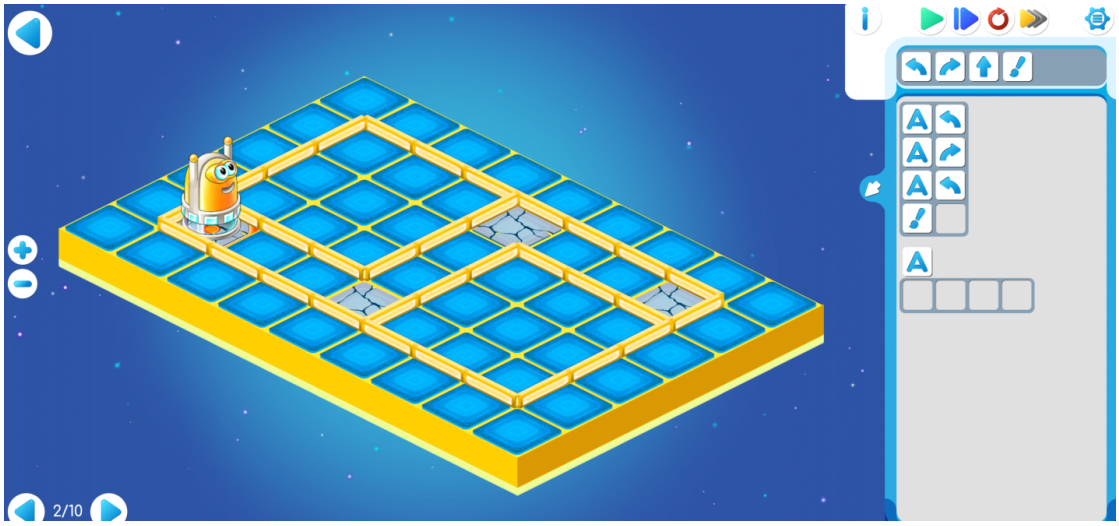


Рисунок 31 – Пример задания ЦОС ПиктоМир, которое возможно решить только с использованием вспомогательного алгоритма

Как видно из предыдущего примера, при создании программы удобно (иногда просто необходимо) поставленную задачу разбивать на подзадачи и составлять главный алгоритм в предположении, что подзадачи уже решены. После этого можно перейти к решению каждой отдельной подзадачи таким же способом. Эти шаги, называемые шагом декомпозиции, нужно продолжать до тех пор, пока не достигнут результат, то есть, когда последняя подзадача, не требующая разбиения на меньшие, будет решена. Решение исходной задачи получается, как объединение всех составленных программ (всех подзадач и главного алгоритма). Чем сложнее исходная задача, тем больше шагов декомпозиции может потребоваться. Такой метод программирования называют *технологией сверху-вниз* [110].

Неформально, эта технология программирования предполагает сначала наметить общее решение (в широком масштабе) и, последовательно декомпозируя задачу (разбивая решение на части), на каждом шаге пытаться приблизиться к её решению. То есть сначала происходит высокоуровневое проектирование решения, а затем постепенное углубление в детали.

Этот подход также часто используется в различных областях, включая индустриальные информационные технологии, инженерные проекты и управление бизнес-процессами. Преимущество такого подхода очевидно: на каждом этапе задача фактически является уже решенной, что практически исключает ошибки проектирования.

Альтернативная технология снизу-вверх предполагает изначально сосредоточиться на деталях разработки и подобрать инструментарий, набор программ-алгоритмов, который предполагается в будущем использовать для решения поставленной задачи. Этот подход более трудоемок, потому что потребуется предвосхитить потребности еще не готового решения.

Для повышения интереса детей к предмету можно создать и включить в курс компьютерные задания нового типа, вносящие вариативность в занятия. В отличие

от большинства задач, выполнение этих заданий требуют предварительной подготовки своих инструментов (вспомогательных алгоритмов), а не обычного составления программы управления роботом.

Эти вспомогательные алгоритмы необходимо составить заранее, а саму задачу ученик будет решать в режиме пультового управления без создания программы управления роботом. Комплекс ЦОС ПиктоМир включает в себя среду «КубоРобот», которая может не только использоваться в курсах по основам программирования, но и иметь самостоятельную ценность как отдельное педагогическое программное средство для формирования элементов алгоритмического мышления у учеников в рамках других курсов. Так, методическая ценность КубоРобота состоит в помощи детям в понимании и усвоении одного из фундаментальных понятий программирования – концепции вспомогательного алгоритма.

В КубоРоботе ученики решают задачу, состоящую в нахождении самого быстрого способа прохождения лабиринта из заданной на карте точки старта к финишу, используя пультовое, как еще его называют, непосредственное управление роботом. Игра состоит из 2-х этапов:

Вначале, на первом этапе задача ученика заключается в составлении двух вспомогательных алгоритмов с именами А и Б, которые он предполагает использовать в конкретной обстановке данной задачи.

На втором этапе ребенок использует уже придуманные им алгоритмы А и Б и управляет роботом Вертуном для максимально быстрого достижения финиша.

В примере, Рисунок 32, игроку в задании необходимо заполнить шаблоны для алгоритмов А и Б так, чтобы за минимальную последовательность, состоящую из шагов: выполнение алгоритма А, Б, команд вперед, повернуть направо, повернуть налево, провести Вертуна из стартовой точки лабиринта к финишу. Предложенный в задаче шаблон алгоритма А состоит из 3-х клеток и шаблон алгоритма Б из 4-х клеток.

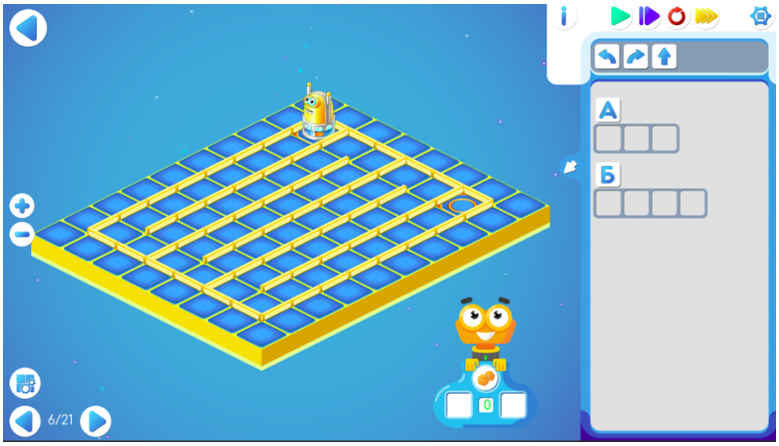


Рисунок 32 – Пример задачи-головоломки в КубоРоботе ЦОС ПиктоМир

3.6 Подходы поэтапного освоения программирования управляемых объектов со сложным поведением как новое содержание пропедевтической методической системы обучения для дошкольников и младших школьников

У дошкольников и также у младших школьников для формирования основ алгоритмического мышления можно использовать такие виды деятельности, как игры-имитации со сложными и простыми управляемыми объектами, например, роботами, которые находятся в некоторой обстановке и выполняют внешние команды по определенным правилам. Основная задача для методологов состоит в компоновке правил, которые определяют наполнение обстановки и управление роботами, а также в подборе сложности и оптимизации (минимизации) соответствующего объема информации, который необходимо освоить дошкольнику.

«Федеральный государственный образовательный стандарт дошкольного образования» предписывает, что в рамках образовательных областей, к которым, в частности, относится социально-коммуникативное, познавательное, речевое и художественно-эстетическое развитие, могут реализовываться в различных видах деятельности: «для детей дошкольного возраста (3 года - 8 лет) – ряд видов деятельности, таких как игровая, включая сюжетно-ролевую игру, игру с правилами и другие виды игры, коммуникативная (общение и взаимодействие со

взрослыми и сверстниками), познавательно-исследовательская (исследования объектов окружающего мира и экспериментирования с ними) …» [53].

Подобную деятельность и набор заданий также можно включать в дополнительную или парциальную программы для освоения детьми команд реальных и виртуальных роботов ЦОС ПиктоМир.

Традиционно при использовании ЦОС ПиктоМир для начинающих осваивать программирование упор делается на минимизацию сложности роботов, несложные обстановки, где функционируют роботы, логически простые для понимания новичками команды. Безусловно, Вертуна, Двигуна, Тягуна и Ползуна – роботов, которые имеют минимальный набор команд, перемещаются по клетчатой обстановке, можно назвать *роботами с простым поведением*.

Все эти роботы с простым поведением на клетчатом прямоугольном игровом поле на каждом шаге занимают ровно одну клетку. Клетки могут быть ограничены стенами с одной или нескольких сторон. Иногда в заданиях требуется не только некоторым образом модифицировать обстановку при помощи робота с простым поведением, но и завершить задание в определенной выделенной клетке поля. Каждый робот с простым поведением умеет выполнять сильно ограниченное число команд. Другие элементы заданий, как и сами клетки, также не могут занимать больше одной клетки.

Все роботы с простым поведением имеют общие три команды изменения их положения на поле: переместиться на клетку вперед, повернуться направо, повернуться налево. Этот набор команд составляет практически минимальный набор для перемещения по клетчатому полю игры.

Отсутствие сложности описания роботов с простым поведением приводит к ограниченности возможных заданий. Кроме того, для роботов с простым поведением характерна локальность поведения:

- локальность действий: робот может выполнять действия только в той клетке, на которой он находится. Так, Вертун может красить только ту клетку, на

которой он стоит. Это ограничивает его возможности по изменению игрового поля. Из последнего следует:

- ограниченность возможности действий: робот не может закрашивать клетки, не меняя свою позицию. Это также упрощает его функциональность.
- независимость от окружения: правила построения обстановки локальны, то есть изменения в одной клетке не влияют на соседние, что упрощает создание игровых сценариев.

Все вышесказанное относится ко всем роботам с простым поведением, в том числе и к обратной связи от роботов, когда ответ робота локален и зависит от состояния текущей и ближайшей клетки:

- Тягун: робот, созданный для работы на космическом складе, расставляя грузы по местам, перемещает только по одной бочке или ящику за один ход. Робот тянет груз за собой, то есть груз должен непосредственно касаться его спины, находясь за ним на соседней клетке. Это также ограничивает возможности робота по транспортировке груза, если непосредственно перед Тягуном находится стена.
- Двигун: робот (родственник Тягуна) также, работая на космическом складе, расставляет грузы, толкая их перед собой. Двигун может сдвинуть один или два груза, которые находятся непосредственно перед ним в соседних клетках по направлению его движения. Также ограничивает функциональность Двигуна случай нахождения стены по направлению движения робота.

Такой подход к обучению действительно может ускорить освоение основных понятий программирования и сделать процесс обучения более доступным для детей средней и старшей групп детского сада. Игра с роботами не только является увлекательной, но и позволяет детям лучше понять принципы функционирования различных устройств.

Освоение правил поведения роботов со сложным поведением может стать важным этапом в развитии познавательных и речевых навыков детей, что представляет самостоятельную педагогическую ценность. Это поможет им лучше понимать причинно-следственные связи и научиться формулировать свои мысли.

Подобная учебная модельная ситуация встречается и в практике профессиональных программистов. Они также сталкиваются с необходимостью изучать сложные системы и правила их функционирования. Однако, в отличие от детей, они уже имеют определённые навыки и знания, которые помогают им быстрее разобраться в новой информации.

Когда ребёнок составляет программы для роботов с простым поведением, он в основном сосредоточен на разработке алгоритма. Хотя технически составить готовый алгоритм из пиктограмм может быть сложно, эта сложность сравнима с управлением интуитивно понятным роботом с очевидным поведением. При составлении программ для роботов со сложным поведением ребёнку приходится приложить больше усилий для понимания сложной обстановки и её структуры, то есть для осмысления всех исходных данных задачи. Однако педагогический опыт показывает, что такие задачи вполне доступны детям 6–7 лет, которые уже овладели азами программирования. Это значит, что подобные задачи находятся в зоне ближайшего развития этих детей.

Освоение роботов со сложным поведением позволит максимально разнообразить знания, умения и навыки ребёнка [53]. В рамках курса дети могут освоить ряд предметных навыков, связанных с комбинаторикой. Н.И. Хохлова пишет: «Комбинаторика составляет основу детского экспериментирования. Характеристики этого процесса качественно меняются по мере получения ребенком все новых сведений об объектах окружающей действительности» [255]. Однако развитие комбинаторных возможностей у детей дошкольного возраста требует пропедевтической работы: «В обычной практике экспериментирование детей с многофакторными объектами носит эпизодический характер и поэтому не оказывает существенного влияния на их комбинаторное мышление. Запланированное, целенаправленное обучение, предполагающее организованное экспериментирование детей со специально подобранным объектом, может обеспечить более эффективное развитие способностей ребенка» 255]. В работе [А.Н. Поддьякова исследуется деятельность ребенка при комбинаторном

экспериментировании: «Комбинаторное (многофакторное) экспериментирование – это особое направление познавательного развития детей. Оно служит предпосылкой становления начальных форм системного подхода к изучению сложных явлений» [194]. А.Н. Поддьяков утверждает важность для ребенка реальных действий с объектами: «…что комбинаторное экспериментирование с многофакторными, системными объектами является скорее "родной", чем "чужой" деятельностью для дошкольников. Дошкольники сензитивны, чувствительны к проявлениям многофакторности, к ситуациям, требующим комбинаторного экспериментирования, легко откликаются на них и демонстрируют неожиданно высокий уровень их понимания. Дошкольный возраст – это сензитивный период для введения детей в мир многофакторных, системных объектов и явлений» [191].

Практическое программирование детьми в возрасте от 6 до 9 лет роботов со сложным поведением представляет собой материализованный этап комбинаторного процесса. В процессе обучения дети знакомятся с многофакторными зависимостями, что способствует развитию их логического и системного мышления, а также формированию навыков анализа сложных систем и процессов. Этот подход позволяет детям на практике освоить принципы работы со сложными системами, научиться анализировать множество факторов и учитывать их при принятии решений. Это особенно важно для развития комбинаторного мышления, которое необходимо для решения сложных задач и проблем.

Действительно, изучение роботов со сложным поведением не оправдано для детей 3–5 лет. Это может добавить лишнюю сложность при решении задач на программирование роботов и снизить интерес к обучению.

Однако если дошкольники или младшие школьники освоили составление простейших алгоритмов с обратной связью, то они получат дополнительный развивающий эффект, если достигнут возраста 6–7 лет. В этом возрасте дети уже готовы к более сложным задачам и могут лучше понять принципы работы сложных механизмов.

В качестве примера можно познакомиться с роботом со сложным поведением Паровозик. Паровозик и его партнёры (вагоны и цистерны) представляют собой расширение возможностей роботов с простым поведением, таких как Двигун и Тягун. Последние были созданы для перемещения одиночных или сдвоенных грузов, например, ящиков и бочек. Паровозик и партнёры фактически являются более сложными версиями этих роботов. Они могут двигаться по полю и взаимодействовать друг с другом, но их состояние может измениться только в результате действий Паровозика. Это связано с тем, что у вагонов и цистерн нет индивидуальных наборов команд, которые позволяли бы им программно управлять своим состоянием. Некоторые задачи не могут быть выполнены только Двигуном и требуют привлечения к работе Тягуна. Так, ящики, стоящие у стены, Двигун не сможет отодвинуть, поэтому здесь предлагается для решения Тягун. Специфика робота со сложным поведением, такого как Паровозик, состоит в том, что он может выполнить работу и Двигуна, и Тягуна, Рисунок 33.

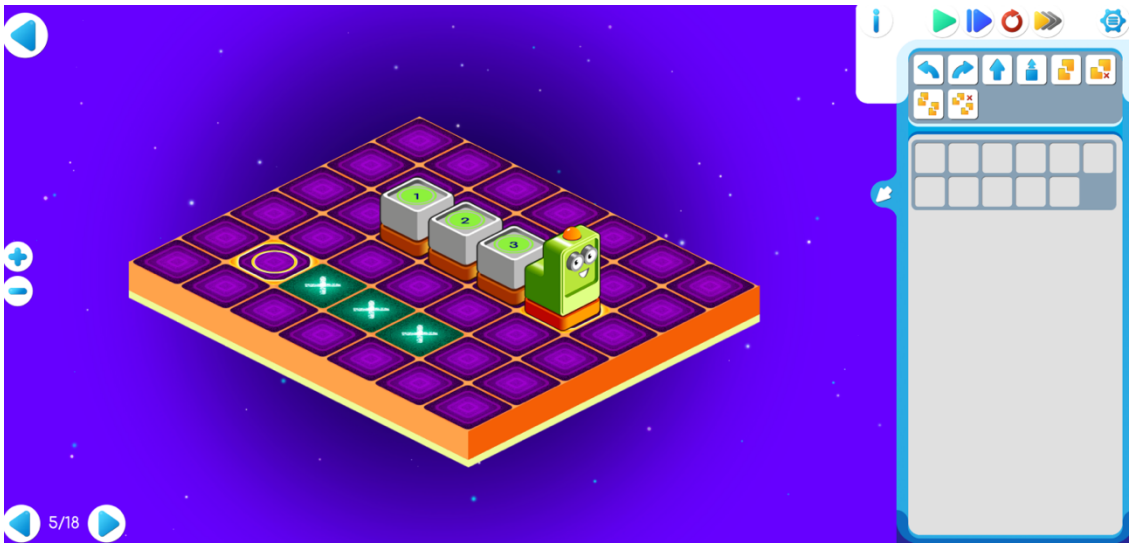


Рисунок 33 – Пример задания для Паровозика ЦОС ПитокМир: Робот должен переместить (отбуксировать) вагоны на клетки-базы, отмеченные крестиком

По легенде, Паровозик используется на тех же космических складах, где работают роботы Тягун и Двигун, но в отличие от них, ему требуется перевозить прицепы: вагоны и цистерны, предварительно собирая их в составы. Хорошей

подготовкой к работе с новым роботом, и это следует из его названия, будет наблюдение за работой маневровых локомотивов на настоящей сортировочной железнодорожной станции. Паровозик ведёт себя подобно маневровому локомотиву на железнодорожных ветках: он перемещается по станции-складу, прицепляя и (или) отцепляя вагоны и цистерны с помощью электромагнитной сцепки. В этом виртуальном мире вагоны и цистерны прикрепляются к Паровозику и между собой электромагнитными сцепками. Процесс присоединения вагона к составу проводится автоматически под управлением компьютера из программы «ЦОС ПиктоМир».

Сцепление между вагонами и между вагоном и Паровозиком в ЦОС ПиктоМир показано изменением состояний вагонов — на каждом из них могут быть включены (активированы) специальные индикаторы (вагон или цистерна подсвечивается и видна сцепка(и)) и одна или две магнитные сцепки. В этой игре внутренние состояния прицепов (вагонов и цистерн) отображаются в виде визуальных индикаторов. Это позволяет ребёнку понять, какие прицепы подсвечены, соединены сцепками друг с другом или с Паровозиком.

Естественно, что если управляют роботом со сложным поведением, то и структура поля, и действия команд сложнее и уже не носят локального характера. Так, например, визуально команда буксировки (передвижение состава на один шаг) может выполняться с некоторой задержкой, при которой наблюдается, что сначала передвигается Паровозик, разрывая связь с вагонами, затем по очереди двигаются вагоны. Этот эффект зависит от длины состава и его расположения и символизирует этапность выполнения команды в цифровом мире.

Кроме того, нелокальность робота со сложным поведением Паровозика проявляется в выполнении команд, когда результат зависит от обстановки и затрагивает не только близкорасположенные клетки, но и далекие от позиции Паровозика. Более того, его влияние распространяется на практически все клетки игрового поля, где расположен состав Паровозика. Аналогичное относится и к командам-вопросам, которые можно задать роботу. В случае с роботами с простым

поведением результат определялся локально, то есть по близко расположенным клеткам. В случае с роботом со сложным поведением ответ на команду-вопрос Паровозика может зависеть от состояния не только от окружающих его, но и большого количества клеток и даже, возможно, всего поля.

Понимание природы указанных сложностей и их обсуждение для ребенка 6-7 лет дает заметный развивающий эффект. С логической точки зрения составы представляют собой цепочки, то есть примеры математической структуры. Практическое освоение этой структуры предусмотрено разделом 12.3. ФГОС НОО [154] «4) умение выполнять устно и письменно арифметические действия с числами и числовыми выражениями, решать текстовые задачи, умение действовать в соответствии с алгоритмом и строить простейшие алгоритмы, исследовать, распознавать и изображать геометрические фигуры, работать с таблицами, схемами, графиками и диаграммами, цепочками, совокупностями, представлять, анализировать и интерпретировать данные;». Программирование Паровозика в ЦОС «ПиктоМир» помогает детям освоить различные типы цепочек.

Как и у роботов с простым поведением, поле Паровозика состоит из клеток. В зависимости от задачи на поле может быть несколько однотипных Паровозиков, которые могут выполнять задания вместе. Кроме Паровозика на поле есть вагоны и цистерны, которые он может сцеплять в составы. На поле также допустимы стены, а некоторые клетки могут быть отмечены в качестве стоянок для вагонов и цистерн.

Описание игры Паровозик ЦОС ПиктоМир вводит для ребенка два десятка новых терминов, например: клетка с базой (стоянкой) для вагона, клетка с базой (стоянкой) для цистерны, клетка с универсальной базой и т.п. Такой уровень сложности обусловлен самой сутью робота со сложным поведением Паровозика, что и дает полное описание сложного поведения робота, которое должен освоить человек и(или) компьютер. Такой подход выбран намеренно. Так, освоение понятий и терминов, описывающих сложное поведение, помогает решать задачу ускорения познавательного и речевого развития детей возраста 6-7 лет. Дети,

прежде всего, должны освоить правила поведения Паровозика и практически научиться предсказывать результаты выполнения команд Паровозика в различных обстановках и использовать это понимание при составлении программ управления роботом со сложным поведением.

В.П. Зинченко в книге «Сознание и творческий акт» пишет: «Так или иначе, человек эффективно использует в поведении, деятельности, мышлении, созерцании построенную им картину мира. Иное дело, насколько он ее осознает и способен ли явить образ мира в слове, в картине, в действии, в поступке, в схеме, в формуле и т. д. Некоторым это удается, но даже в этом случае они не могут вразумительно рассказать, как они этого достигают. А. А. Ухтомский когда-то сказал, что люди сначала научаются ходить, а потом задумываются, как им это удалось» [17].

В терминах В.П. Зинченко практика программирования учит ребенка «явить образ мира в действии» (командуя роботом с помощью пульта) или «явить образ мира в схеме и формуле» (составляя программу будущего поведения робота), но не «в слове». То есть ребенок осваивает управление сложным роботом с пульта, умеет составлять программы управления сложным роботом, то есть успешно программирует робота, не научившись выразить словесно и еще не сформировав представление, какие свойства робота он при этом использовал.

Из это следует, что дети могут успешно учиться программированию роботов (Паровозика и других) без необходимости освоения специфической терминологии, описывающей поведение роботов. Поэтому важно формулировать задачу речевого развития на основе программирования роботов как самостоятельную и самоценную.

Для ускорения речевого развития на предметном материале вводного курса программирования в ЦОС «ПиктоМир» для освоения терминологии роботов со сложным поведением необходимо выделять отдельное время. Однако для полноценного усвоения этой терминологии необходимо разработать методики

обучения, которые бы включали материал на начальных этапах обучения без использования компьютеров.

3.7 Комбинаторная сложность задач составления пиктографических алгоритмов в ЦОС ПиктоМир как критерий подбора заданий для содержания пропедевтической методической системы обучения.

Комбинаторная сложность задач составления пиктографических алгоритмов в цифровой образовательной среде «ПиктоМир» может выступить критерием при подборе заданий для содержания пропедевтической методической системы обучения. Этот критерий отражает уровень сложности, который должен соответствовать текущему уровню подготовки учащихся. Подбор заданий на основе комбинаторной сложности составления пиктографических алгоритмов позволяет достичь оптимального сочетания доступности и сложности материала, что способствует эффективному освоению алгоритмических умений и развитию алгоритмического мышления. Этот подход фактически переводит задачи из недоступной области в зону ближайшего развития, а затем и в область актуального развития, где ребенок может действовать самостоятельно.

Сложность заданного фиксированного алгоритма можно определять с точки зрения процесса его выполнения динамически, как количество операций, которые алгоритм выполнит для успешного решения задачи при наиболее неблагоприятных данных заданной длины. На начальном этапе освоения программирования обсуждение этой динамической сложности не является первоочередной задачей. С точки зрения новичка актуальна сложность процесса «придумывания» алгоритма в процессе решения задачи на составление алгоритма, которую новичок получил извне. Разумеется, какой бы ни была среда программирования, с формальной точки зрения, новичок всегда составляет алгоритм на некотором данном ему алгоритмическом языке, хотя описание этого языка может и не даваться в вербальной форме. Можно было бы определить комбинаторную сложность

формального алгоритмического языка как величину, зависящую от длины $n \in Z_+$ выводимой в данном языке цепочки, равную числу выводимых цепочек длины n в этом языке. Если не накладывать ограничений на способ построения выводимых цепочек, то для любых встречающихся в педагогической практике языков программирования число выводимых цепочек будет быстро расти с ростом n. Однако с точки зрения практики более важно не количество выводимых цепочек, а количество алгоритмов-кандидатов (кандидатов на верное решение поставленной задачи), которые обучаемый может сгенерировать в предложенной ему среде программирования при соблюдении наложенных извне ограничений на процесс составления. Это количество алгоритмов-кандидатов называют *комбинаторной сложностью задачи на составление алгоритма* в условиях выполнения обучаемым заданного способа построения алгоритма.

Пусть существует некий формальный алгоритмический язык, что в соответствии с принятыми правилами может быть описано некоей грамматикой $\Gamma = \{T, N, p, S\}$. Цепочки, выводимые в языке, представляют собой бесконечный набор комбинаций терминалов, выводимых с использованием продукций p. Как уже говорилось, даже если ограничить длину цепочки небольшим числом n, число выводимых цепочек окажется слишком большим для непосредственного рассмотрения. Поэтому в задачах на составление алгоритмов для новичков возникает необходимость накладывать ограничения на способ составления алгоритма так, чтобы уменьшить комбинаторную сложность решаемой новичком задачи.

Эти ограничения ни в коем случае не могут формулироваться в вербальной форме, а должны быть представлены в интуитивно понятной, геометрической форме, что обеспечит их мгновенное понимание обучаемыми, начиная с возраста 3-4 лет.

В ЦОС ПиктоМир предложен новый геометрически интуитивный способ радикального уменьшения комбинаторной сложности задач на составление алгоритмов в пиктограммных средах программирования – метод шаблонов. Метод

шаблонов состоит в предоставлении новичку шаблона-подсказки для составления алгоритма решения задачи. В каждом шаблоне используется геометрическое представление синтаксических и семантических особенностей пиктограмм – атомарных элементов пиктограммного языка программирования, которые можно использовать в составляемом алгоритме.

Фактически шаблон задает и тем самым подсказывает полную структуру алгоритма решения поставленной задачи, что важно для эффективного ознакомления обучаемых с правильным стилем программирования. Однако на ранних стадиях освоениях программирования важно то, что предложенный подход явно задает обучаемому фиксированное количество позиций шаблона, каждую из которых можно заполнять независимо от остальных позиций, что, по меньшей мере с теоретической точки зрения сводит для новичка задачу составления алгоритма к полному перебору. Поэтому возникает вопрос: возможно ли решение заданий по составлению алгоритма полным комбинаторным перебором?

Многие авторы, включая Ж. Пиаже [250], считают, что организация полного систематического многомерного перебора доступна лишь взрослым и подросткам, но не младшеклассникам и тем более дошкольникам. Ж. Пиаже, впрочем, надеялся, что поскольку полный комбинаторный перебор N параметров с точки зрения программиста полностью описывается вложенными циклами глубины N, обучение детей практическому программированию в комфортной учебной среде типа Logo позволит понизить возраст освоения школьниками полного многомерного перебора С. Пейперту [177].

В работах А.Н. Поддьякова [192; 193] выражена более оптимистическая точка зрения на возраст, начиная с которого дети способны освоить полный комбинаторный перебор. В этих работах указывается, что и дети с 7 лет в состоянии освоить счетчик-стратегию, то есть последовательный полный перебор всех значений выбранного элемента при сохранении постоянными значений других элементов и после выбора нового значения следующего элемента, повторение цикла перебора.

Вопрос же о способности детей до 7 лет к многофакторным взаимодействиям можно считать открытым, хотя про варьирование одного фактора при сохранении всех остальных факторов неизменными имеются зафиксированные эксперименты [191].

В предположении, что дети до 7 лет не в состоянии исследовать многофакторные объекты, к которым относятся роботы со сложным поведением, кажется сомнительным утверждение, что комбинаторный метод проведения исследования у детей 5-6 лет может сводиться к счетчик-стратегии перебора элементов составных программах. Интуитивно уже владея элементами сопоставления большого и маленького, например, соотнося к своему размеру («эта ложка маленькая (кукольная), потому что мне трудно держать ее в руке», «тут я точно утону» (ребенок 6 лет, впервые увидевший море) и т.п. ребенок в этом возрасте уже начинает соотносить свои потенциальные возможности с задачами, которые перед ним ставит воспитатель, педагог и т.п. И дети, и тем более взрослый в состоянии понять, что на начальном этапе осваивания программирования ребенок не в состоянии составить программу неопределенно большого размера. Поэтому предоставление шаблона является теми спасительными «перилами», которые позволяют юному программисту идти по «мостику» (составлять программу), делая первые шаги в программировании.

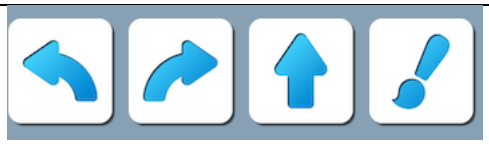
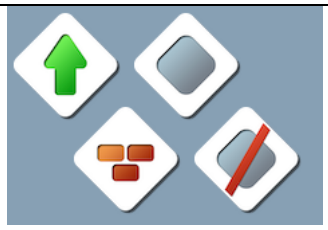
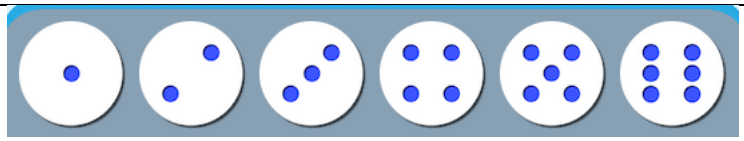
Очевидно, что если прямая счетчик-стратегия приводит к большому количеству шагов, то провести такое экспериментальное составление программы полным комбинаторным перебором просто нереально не только дошкольнику, но и взрослому. Следует, однако, заметить, что даже частично заполненный шаблон, независимо от процента заполнения, является синтаксически правильной выполнимой программой, что позволяет новичку выполнять составляемый алгоритм не только после полного заполнения шаблона, но и на промежуточных стадиях. Во многих задачах это позволяет новичку пошагово составлять жадный алгоритм решения поставленной задачи.

Вопрос о изобретении «жадного алгоритма» составления программы неполным комбинаторным перебором всех пиктограмм в шаблоне остается открытым. Но если задачи подобраны так, что число параметров и шагов полного комбинаторного перебора невелико, то и такой способ составления программы приведет новичка к решению задачи.

Далее несколько слов о доступности предложенного в ЦОС ПиктоМир метода шаблонов. Язык Пикто ЦОС ПиктоМир поддерживает все конструкции структурного программирования, в том числе и вложенные конструкции ветвления и повторения. В языке Пикто имеются пиктограммы для представления этих конструкций, а также пиктограммы команд-действий используемого в данном алгоритме робота.

Естественно, что в языке Пикто ЦОС ПиктоМир циклические конструкции и конструкции выбора по семантике отличаются от команд управления роботом, Таблица 5. Но важнее, что заголовок каждой из этих конструкций задается одной-единственной пиктограммой. Пиктограммы этих заголовков отличаются по геометрической форме друг от друга и от команд-приказов роботов. Именно это исключает возможность построения новичком синтаксически неверных конструкций в среде ПиктоМир. Благодаря включению шаблона в условие задания на составление алгоритма вариативность задания на составление учебного алгоритма резко сокращается, что понижает комбинаторную сложность задач на составление алгоритмов в ЦОС ПиктоМир.

Таблица 5 — Различные геометрические формы пиктограмм в ЦОС ПиктоМир

Команды-действия		Форма-Квадрат
Команды-вопросы		Форма – ромб
Команды-повторители		Форма – круг

Уже в начале предоперационного периода дети в состоянии соотносить формы предметов, которые они осваивали ранее в сенсомоторный период до третьего года жизни. Манипуляции с пиктограммами различных геометрических форм в ПиктоМире фактически продолжают знакомые детям манипуляции с предметами в играх-вкладышах М. Монтессори [150; 151]. В этих играх в рамки-пластины с отверстиями нужно вставлять подходящие по форме и размеру вкладыши. Такие игры не только помогают развивать мелкую моторику, координацию движений, но и знакомят детей с формами предметов. Таким образом, к первой встрече с ЦОС ПиктоМир и к моменту составления первых программ по заданному шаблону на экране ЦОС ПиктоМир ребенок уже считает очевидным, что размещать пиктограмму (аналог предмета) в ячейке шаблона (аналог отверстия) можно только согласно ее форме, то есть ребенку ясно, что круглую пиктограмму повторителя нужно размещать в круглой ячейке шаблона, а квадратную команду-приказ робота нужно размещать в квадратике шаблона и ребенок не удивляется, что при отпускании пиктограммы над ячейкой пиктограммы неправильной формы пиктограмма исчезает (не попадает в отверстие).

В то время как общий вопрос о возможности систематического освоения счетчик-стратегии дошкольниками остается открытым, эксперименты с заданиями на составление алгоритмов в ЦОС ПиктоМир показывают, что в ряде содержательных задач малой комбинаторной сложности дети возраста 5-6 лет справляются с теми или иными вариантами полного комбинаторного перебора.

Первый приведенный ниже пример относится к начальным шагам новичка в освоении программирования. Можно заметить, что на начальном этапе юные программисты знакомятся только с последовательным программированием, составлением алгоритмов как последовательностей прямых команд-предписаний управления роботом-исполнителем.

Пример простейшего задания на программирование для новичков показан на рисунке, Рисунок 34. В этом задании нужно заполнить две клетки пиктограммами так, чтобы Вертун закрасил две клетки (требование прийти Роботом в конкретное место в данном задании отсутствует).

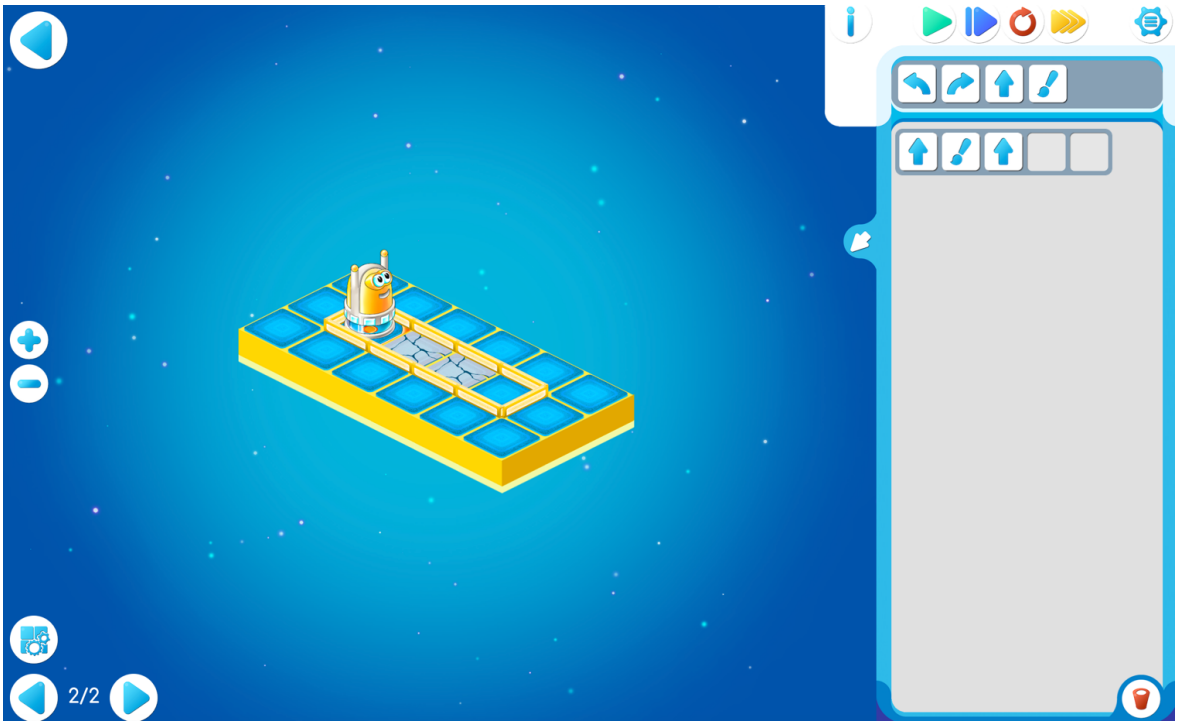


Рисунок 34– В примере задачи ЦОС ПиктоМир нужно заполнить две клетки пиктограммами

В распоряжении программиста полочка с 4 пиктограммами. Именно они, а других в данном задании нет, могут быть помещены в свободные клетки шаблона алгоритма. Так, при комбинаторном подходе нужно поставить одну из 4-х пиктограмм на первый свободный квадратик и одну из 4-х на второе свободное место. Таким образом, комбинаторная сложность данного задания на составление алгоритма, то есть сложность полного комбинаторного перебора при заданном шаблоне здесь будет 4x4=16 вариантов, что количественно вполне доступно и дошкольнику 4-5 лет. Уточнение: если вдуматься в условие задачи, то формально счетчик-стратегия потребует перебрать 25=5x5 вариантов, так как разрешается оставлять клетку шаблона пустой.

Другой фактор, влияющий на успешность выполнения поставленного задания, состоит в том, насколько вероятно, что в последовательности полного перебора ученик найдет ответ до исчерпания всех вариантов. Дело в том, что при счетчик-стратегии сначала будет попытка заполнить первую клетку, и благодаря тому, что каждый созданный ребенком вариант алгоритма сразу можно проверить (заставить робота выполнить программу), есть немалая вероятность того, что решение будет найдено после перебора не более, чем за 4-5 вариантов (ведь в данной задаче есть верные алгоритмы решения, в которых последняя клетка потому что последняя клетка алгоритма не используется, остается пустой. Вообще говоря, такая ситуация – наличие «ненужной» клетки в шаблоне – является исключением и в большинстве заданий в ЦОС ПиктоМир не встречается. Исключения составляют те сложные задания, которые «неизвестно-как-решать» [225; 220], причем в таких заданиях автор порой и сам не может гарантировать, что эталонное авторское решение задачи является оптимальным, то есть не может гарантировать, что нет решений, оставляющих неиспользованными некоторые клетки шаблона.

В заключение можно рассмотреть ситуацию, когда в качестве эксперимента и исследовательского задания детям 4-5 лет дается задача, которая осознанно

может быть решена только в возрасте 6-7 лет, но в силу невысокой комбинаторной сложности может быть решена прямым перебором, Рисунок 35.

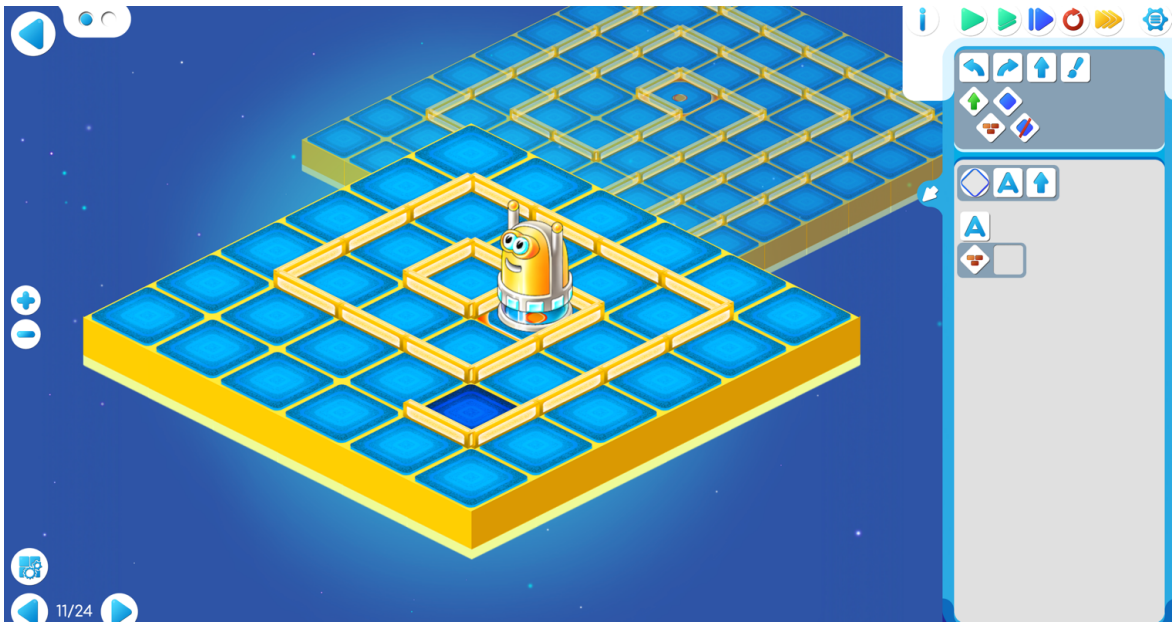


Рисунок 35– «Выход из спирального лабиринта» - задача, которую «неизвестно-как-решать». Для решения нужно разместить в шаблоне одну команду-ромбик и одну команду-квадратик.

Шаблон данного задания требует составления универсальной программы, которая должна работать правильно в двух однотипных обстановках. В шаблоне предусмотрены конструкция *ветвления* и конструкция *цикла с условием*, что требует знакомства с командами-вопросами робота с обратной связью. Понятиями *обратная связь* и *команда-вопрос* дети возраста 6-7 лет легко овладевают примерно за пятьдесят получасовых занятий любого систематического курса Алгоритмики (т.е. в начале четвертой четверти второго года курса). Для детей, освоивших эти понятия, задача на рисунке выше оказывается в зоне ближайшего развития. Но для детей возраста 5-6, которым только что показали на экране работу команд-вопросов робота Вертуна в конструкциях ветвление и повторение, «Выход из спирального лабиринта» – это трудная задача, которую «неизвестно-как-решать». С другой стороны, комбинаторную сложность данной задачи легко вычислить: 4 варианта пиктограмм-ромбиков для первой клетки шаблона и 4 варианта

пиктограмм-квадратиков управления Вертуном для последней клетки шаблона. Всего 16=4x4 вариантов. А практика перебора небольшого количества вариантов (в данном примере 16 вариантов) находится в зоне ближайшего развития детей возраста 5-6 лет и педагогический опыт показывает, что самостоятельно или с небольшой поддержкой педагога дети так или иначе решают эту задачу, которую «неизвестно-как-решать». В ходе этой работы дети приобретают опыт самостоятельного экспериментального исследования и бесценный опыт успешного экспериментального решения задачи, которую «неизвестно-как-решать».

При этом *комбинаторная сложность задачи на составление алгоритма* является адекватной метрикой общей сложности задания и определяет возможность успешного решения задачи ребенком путем полного комбинаторного перебора. Практика решения задач на двухпараметрический полный перебор показывает, что хотя многие дети начинают перебирать пиктограммы произвольным образом, часто повторяясь, действуя стохастически, вне счетчика-стратегии, тем не менее при небольшой комбинаторной сложности задачи составления алгоритма эта задача, как правило, решается всеми детьми.

Так, например, в одном из экспериментов в группе детей возраста 5 лет один ребенок начал экспериментировать бессистемно, но быстро пришел к идее последовательного перебора команд-квадратиков в последней свободной клеточке шаблона и обнаружил, что робот успешно проходит первый поворот только при условии размещения в этой ячейке команды **направо** 

Получив этот результат и зафиксировав эту команду **направо** в последней ячейке шаблона, ребенок начал последовательно перебирать команды-ромбики в первой свободной ячейке шаблона, пока не обнаружил, что если разместить в этой ячейки ромбик  , то робот успешно пройдет обе спирали, остановившись в последней клетке каждой из них.

Важно, что незаполненные клетки шаблона ЦОС ПиктоМир позволяют тем не менее на любом этапе выполнить программу, а их семантика по умолчанию в

ЦОС ПиктоМир подобрана так, чтобы сделать процесс выполнения программы наиболее информативным для ученика. В частности, незаполненный шаблон цикла ПОКА в ПиктоМире выполняется как бесконечный цикл, что позволяет ребенку увидеть результат выполнения тела цикла даже в случае, когда шаблон заголовка еще не заполнен. Именно эта особенность ЦОС ПиктоМир позволила ребенку в описанном выше эксперименте видеть при выполнении программы эффект выбора команды-квадратика в последней ячейке шаблона, оставив незаполненной первую команду шаблона.

Занятия, где дети возраста 4-5 лет получают алгоритмические задачи, которые «неизвестно-как-решать», обладают большим общеразвивающим эффектом. На таких занятиях у детей не только формируются основы алгоритмического мышления, но и развивается преадаптивность, когда игра рассматривается как источник новых возможностей, что учит ребенка мыслить вне шаблона, формируя способность решать задачи, которые «неизвестно-как-решать» [248].

Выводы главы 3

Педагогический опыт и разработка пропедевтической методической системы обучения детей дошкольного возраста и младшего школьного возраста показали, что с использованием пиктограммных языков программирования, как средства и методики обучения, можно осуществлять поэтапное формирование алгоритмического мышления начиная с детей четвертого года жизни, то есть снизить границы сензитивного периода обучения основам информатики и программирования. Бестекстовое программирование позволяет вводить понятия последовательного программирования, как содержания обучения, и успешно решать необходимый набор заданий, в том числе и в национальных дошкольных образовательных организациях и школах.

Методическая система раннего пропедевтического обучения информатике и программированию представляет собой набор компонентов пропедевтики

парадигм программирования, состоящий из уникального дидактического материала, включающего в качестве содержания обучения не только последовательный набор задач и методические подходы, но и, как стержневой элемент, средств обучения в вариативных формах, отечественную предметно-цифровую образовательную среду, в которую можно компоновать указанное содержание для дошкольников и младших школьников на основании сформулированных в настоящей работе критериев для поэтапного формирования основ алгоритмического мышления.

Использование линейки усложняющихся роботов-исполнителей является эффективной методикой для бесшовного перехода между разновозрастными методическими системами обучения информатике и программированию. Такой подход позволяет трансформировать приобретаемые знания, навыки и умения, начиная с пропедевтики до основного курса и заканчивая профильными курсами, обеспечивая непрерывное развитие и углубление компетенций учащихся.

ГЛАВА 4. ИНТЕГРАЦИОННАЯ МЕТОДОЛОГИЯ ФОРМИРОВАНИЯ АЛГОРИТМИЧЕСКОГО МЫШЛЕНИЯ ПРИ ОБУЧЕНИИ ИНФОРМАТИКЕ И ПРОГРАММИРОВАНИЮ С ИСПОЛЬЗОВАНИЕМ РАЗНОВОЗРАСТНЫХ МЕТОДИЧЕСКИХ СИСТЕМ ОБУЧЕНИЯ

4.1 Обоснование выбора цепочек цифровых образовательных сред как цифровых средств обучения информатике и программированию

Интеграционная методология формирования алгоритмического мышления при обучении информатике и программированию с использованием разновозрастных методических систем обучения представляет собой системный подход, объединяющий различные методы и формы обучения, адаптированные под разные возрастные группы. Этот подход позволяет обеспечить эффективное обучение основам информатике и программированию, учитывая индивидуальные потребности и возрастные возможности каждого учащегося. Выбор и комбинация цифровых образовательных сред в качестве средств обучения информатике и программированию основан на едином подходе к формированию алгоритмического мышления у обучающихся, описанном в предыдущих главах настоящей работы. Переход от пиктографического программирования к текстовому через использование блочных цифровых образовательных сред представляет собой эффективный метод обучения, обеспечивающий плавный и бесшовный переход между этими двумя формами представления программ. Такой подход позволяет учащимся постепенно адаптироваться к текстовому программированию, сохраняя при этом высокий уровень интереса и мотивации к обучению.

При проектировании курса по программированию, включая пропедевтические курсы для новичков, одной из основных проблем является достижение баланса между ожидаемыми результатами освоения курса и временем, выделенным на аудиторную и самостоятельную работу ученика. Для предметов, к

которым относятся информатика и программирование, нужно особенно аккуратно соблюдать баланс в тройке (знать, уметь и владеть), чтобы теория для начинающего осваивать программирование не превалировала над практикой.

Например, вполне закономерно, что любой выпускник 9 класса России должен владеть и понятиями, и навыками практического программирования в объеме, предписываемом действующей федеральной программой по информатике базового уровня ФОП ООО [246]. Среди основных целей изучения учебного предмета «Информатика» в средней школе называется освоение основных алгоритмических структур и умение применять эти структуры для построения алгоритмов решения задач, а в качестве второй цели – умения и навыки составления простых программ по построенному алгоритму на одном из языков программирования высокого уровня.

Для вводного курса программирования освоение основных алгоритмических структур и умение их применять – задача более важная и, главное, более трудоемкая для достижения, чем навыки составления простых программ. Для усвоения каждой из тридцати шести (36=6x6) попарных комбинаций этих шести алгоритмических конструкций ФОП ООО Информатика [246] обучаемый должен составить как минимум 5-7 простейших программ. Отсюда следует вывод, что для освоения азов программирования нужно составить не менее двух сотен простейших программ.

Поурочное планирование ФОП ООО отводит на тему программирования не более 30 часов. При использовании для составления программ полнотекстовых сред программирования (IDE - англ. integrated development environment - интегрированная среда разработки, комплекс программных средств, используемый программистами для разработки программного обеспечения) типа ПаскальABC [227], КуМир [98], Visual Studio[335], Eclips [390] или PyCharm [373] выполнение нескольких сотен заданий в отведенные 30 часов совершенно невозможно, даже учитывая самостоятельную внеурочную работу школьника.

Перечисленные выше полнотекстовые среды программирования пришли в школу в основном из промышленности и системы высшего образования и оптимизированы на составление сложных программ, а на простейших программах неэффективны. Как сказано в главе 3, школа должна использовать на начальном этапе освоения программирования методики и инструменты, разработанные для младшеклассников и дошкольников: пиктограммные (бестекстовые) и, как будет показано ниже, блочные (не требующие клавиатурного ввода) среды программирования.

Использование отечественных пиктограммных и блочных сред программирования [187] позволит обучить азам программирования всех выпускников начальной школы с затратами 35-40 учебных часов, что примерно равно числу часов, отводимых сегодня в ФОП ООО [246] на изучение темы «программирование» в 102-часовом курсе информатики основной школы базового уровня.

ЦОС «ПиктоМир» обеспечивает не только понятный и «быстрый старт» для составления элементарных алгоритмов и освоения основных конструкций процедурного программирования, но и гарантирует систематические достижения в знаниях и умениях учеников школ, дошкольников и студентов педагогических университетов в начальных курсах программирования [107].

При использовании ЦОС «ПиктоМир» будь то школьник или студент вуза, оба на старте направлены на решение чисто алгоритмических задач. При этом для ученика ЦОС ПиктоМир представляет собой дидактическую вселенную. При решении задачи для новичка нужно знать не только, как составить алгоритм для задания, то есть изобрести, придумать этот алгоритм, но и уметь записать его на некотором формальном языке программирования (который также надо знать заранее). При этом также необходимо еще владеть навыками успешного взаимодействия (уметь воспользоваться) средой программирования (IDE), чтобы загрузить свою программу в компьютер и выполнить ее. То есть ученик должен

решить для себя не только с чего начать освоение программирование, но и выборочно или одновременно осваивать вышеуказанные направления.

Чтобы сосредоточиться на решении алгоритмической проблемы, требуется, чтобы два других направления – язык программирования и среда программирования – были минимально сложными и осваивались за минимальное время с высоким результатом в тройке знать-уметь-владеть. Как показано в главе 3, в ЦОС «ПиктоМир» эти сложности сведены к минимуму: пиктографический язык программирования прост, объекты, которыми управляет компьютер, знакомы с раннего детства (робот прост и понятен даже дошкольнику). Поэтому при решении задачи можно погрузиться в процесс непосредственного изобретения алгоритма, то есть заниматься чисто алгоритмической сложностью ее решения.

Однако пиктографическая ЦОС «ПиктоМир», сведя языковые сложности и трудности освоения среды программирования практически к нулю, формирует дополнительные сложности для обучаемого при переходе от пиктографического программирования к чисто текстовому по следующим причинам. Так, атомарной единицей пиктографического языка является пиктограмма, и заполнение заранее подготовленного шаблона исключает возможность синтаксически некорректно установить пиктограмму не на свое место, а значит, что программа на пиктографическом языке в ЦОС «ПиктоМир» всегда соответствует синтаксическим правилам алгоритмического языка. Для текстовых языков программирования последнее не выполняется. С другой стороны, сборка программы из ограниченного набора пиктограмм всегда более быстрый и интерфейсно более удобный процесс. Если обратиться к чисто текстовой ЦОС КуМир, то здесь сильно выручают блочные вставки конструкций целиком. Однако, такая функциональность не решает всех проблем перехода от пиктографического языка программирования к текстовому. Так, педагогический опыт показывает, что курс вводного программирования, который читался на протяжении нескольких лет в МПГУ в Институте Детства, очень легко осваивался студентами на начальном этапе, когда алгоритмы составлялись в ЦОС ПиктоМир, и для студентов первые

серьезные трудности начинались при переходе к освоению ЦОС КуМир (текстовый школьный алгоритмический язык) и Python.

Дело в том, что в этом случае нарушался принцип преодоления одной сложности из трех, описанный выше. Хотя задания и исполнители были теми же самыми, но и язык программирования, и среда программирования оказались в новинку для студентов. Также этот процесс можно проиллюстрировать слишком высокими ступеньками лестницы, по которой должен подняться ребенок. Избавление от этой проблемы, добавление промежуточных ступенек в процессе освоения программирования и назван здесь *ступенчатым* подходом.

Расширение блочного подхода, вставления управляющих конструкций целиком, как в ЦОС КуМир, оставаясь в знакомой среде программирования, позволило бы решить проблему синтаксических ошибок при наборе текста. Подобные разработки ведутся не только в России, например, среда Scratch предоставляет такие возможности [379]

Трудности перехода от учебного синтаксически-ориентированного блочного программирования к профессиональному полнотекстовому программированию обсуждаются во многих работах последних лет [368; 397; 331], но эти трудности обсуждаются на методическом уровне. Предложенный подход отличается тем, что в дополнение к методике разработано еще и семейство из трех совместимых по языку снизу-вверх цифровых образовательных сред:

ЦОС ПиктоМир −> ЦОС ПиктоМир-К −> ЦОС КуМир

с общим набором учебных миров и поддерживающих эту методику. Обучение с использованием подобного семейства уменьшает затраты обучаемых на освоение новых интерфейсов.

Цифровая образовательная среда, поставленная как промежуточная ступень освоения процедурного программирования ЦОС ПиктоМир-К, Рисунок 36, не только полностью повторяет набор роботов-исполнителей ЦОС ПиктоМир, но и содержит новых исполнителей, в том числе давно известных исполнителей из ЦОС КуМир, то есть обладает максимально полным набором учебных миров,

необходимых и достаточных в рамках данной методики. В ЦОС ПиктоМир-К доступны следующие роботы-исполнители: Робот, Двигун, Тягун, Вертун, Зажигун, Ползун, Водолей, Кузнечик, Черепаха, Чертежник, Кувшин и другие.

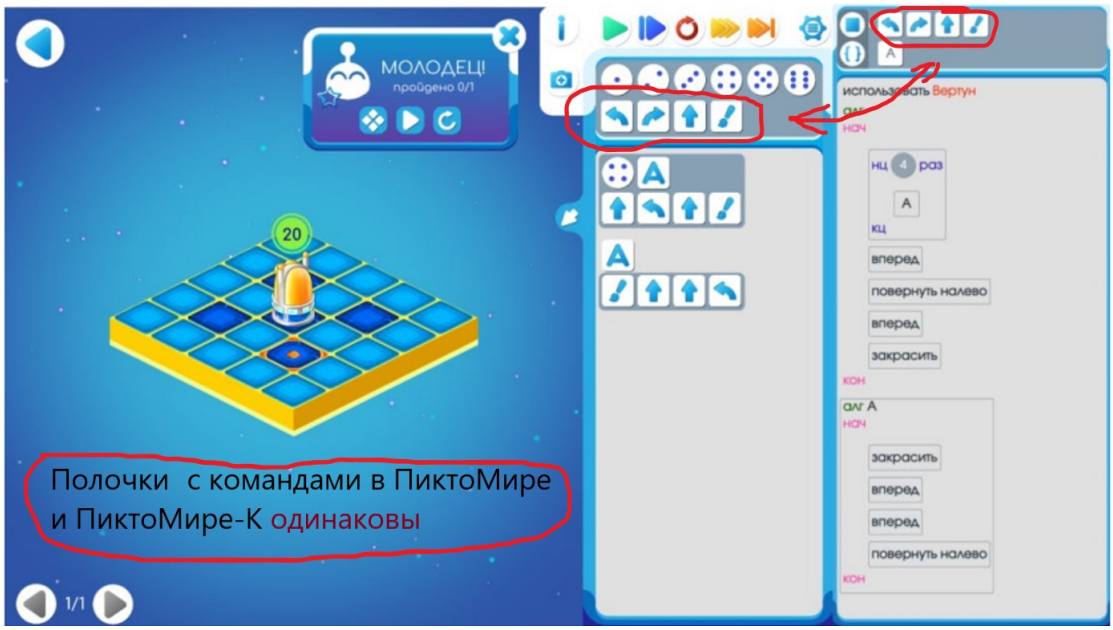


Рисунок 36 – Интерфейс ЦОС ПиктоМир-К и ЦОС ПиктоМир идентичны

Важным расширением, а также пропедевтикой переменных различного типа в алгоритмическом языке в ЦОС ПиктоМир-К можно использовать величины, которые, в отличие от ЦОС КуМир и многих языков программирования, автоматически инициализируется при объявлении нулем, поскольку все величины в ЦОС ПиктоМир-К могут хранить только целые числовые значения. Введение величин в язык Пикто позволяет использовать аргументы во вспомогательных алгоритмах, например, в командах исполнителя Чертежник в ЦОС ПиктоМир-К. Такое ограничение в типизации предопределённых переменных также требует, чтобы команды исполнителя Робот температура и радиация выдавали целые, а не вещественные значения, в отличие от ЦОС КуМир.

ЦОС ПиктоМир-К позволяет составлять программы управления более десятка виртуальных роботов, используя подмножества школьного алгоритмического языка. Процесс подготовки задания педагогом аналогичен

созданию задачи в ЦОС ПиктоМир. Он начинается с выбора роботов, которые могут быть использованы учеником в программе, затем создается шаблон задачи, который фактически определяет меню редактирования ученической программы, необходимой (или оптимальной) для этого конкретного задания. Так, на рисунке, Рисунок 37, представлен шаблон программы для управления робота Вертун.

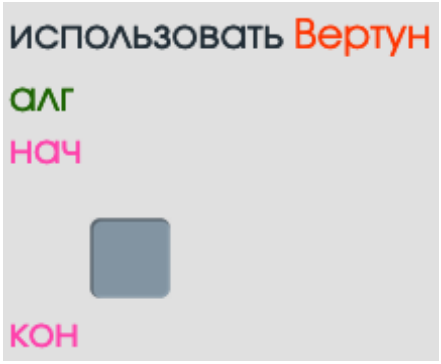


Рисунок 37 – Пример шаблона программы управления роботом Вертун в ЦОС ПиктоМир-К

В первом ряду выпадающего меню ЦОС ПиктоМир-К, изображенном на рисунке, Рисунок 38, находятся команды-действия, предопределенные имена подпрограмм-алгоритмов (в данном примере без параметров) и повторители.

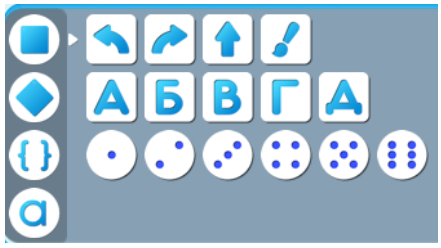


Рисунок 38 – Пример выпадающего меню ЦОС ПиктоМир-К для набора программы по управлению роботом. Поле выбора компонентов языка разделено на вкладки по группам.

Из меню ученик может выбирать ветвления (инструменты выбора, в том числе для сравнения числовых значений), обозначенные ромбиком, циклы (инструменты для создания итерирующих управляющих конструкций),

обозначенные двумя фигурными скобками, или добавить выражение с некоторой величиной (инструменты создания целочисленных и логических переменных и арифметических выражений).

Для присвоения значения величине, если число не превышает 6, то используется представление, аналогичное ЦОС ПиктоМир, где числа от 1 до 6 обозначались гранями игрального кубика. Если число находится вне указанного диапазона, то для ввода целых числовых значений в десятичной записи используется экранная клавиатура, что позволяет проходить курс на планшетах и не использовать клавиатуру компьютера.

При подготовке задачи педагог не только создает шаблон будущего решения, но и выбирает набор инструментов, которые будут доступны ученику для выполнения задания. Тем самым дается подсказка по составлению алгоритма, так как не все возможности алгоритмического языка можно использовать в этой задаче.

При обучении в ЦОС ПиктоМир-К ученик может непосредственно сосредоточиться на освоении расширения алгоритмического языка, так как редакторы ЦОС ПиктоМир-К и ЦОС ПиктоМир практически идентичные и используют однотипный подход, исключающий синтаксические ошибки в программе (форма пиктограмм подсказывает, в каком месте программы можно их использовать). Например, квадратные пиктограммы по-прежнему обозначают действия, круглая форма пиктограммы задает целые значения, а ромбические пиктограммы используются для ветвлений.

Тогда при решении задачи, то есть при наборе программы, ученик производит привычные действия, такие же, как и в ЦОС ПитокМир, перетаскивает пиктограммы на места шаблона (соответственно их форме). Например, вставление конструкции **цикл n раз**, изображенная на полке как знак, обозначающий указанную конструкцию, которая «раскрывается» в шаблоне в текстовое представление. Каждая конструкция языка может быть обозначена или пиктограммой (на полочке), или представлена текстом (в программе). Для новичка в текстовом программировании очень быстро формируются ассоциативные связи

между знаковым и текстовым представлениями алгоритма. Даже для младшеклассников, которые могут не любить читать, такие ассоциативные связи являются воротами в текстовое программирование. Программа в ЦОС ПиктоМир-К может быть составлена на школьном алгоритмическом языке или на языке Python, Рисунок 39.

Рисунок 39 – Программа закрашивания углов и центральной клетки квадрата на алгоритмическом языке в ЦОС ПиктоМир и ЦОС ПиктоМир-К.

Введение выравнивающего курса “Азы программирования” в педагогические университеты является необходимым шагом для подготовки высококвалифицированных специалистов, способных эффективно работать в условиях современного цифрового общества и образования. Курс помогает студентам адаптироваться к новым образовательным реалиям, где активно используются цифровые инструменты и платформы. Овладение основами программирования позволит будущим учителям лучше понимать структуру и логику алгоритмов, что, в свою очередь, улучшит их способность объяснять сложные концепции ученикам. Это повысит качество обучения и мотивацию учащихся.

Сегодня не все молодые педагоги, учителя естественно-научных дисциплин, в том числе и учителя информатики, обладают устойчивыми навыками

процедурного программирования. Предусмотренные обязательной программой ФГОС ООО [246] элементы программирования не осваиваются подавляющим большинством выпускников школы, в том числе и выпускниками, успешно поступающими на естественно-научные и педагогические специальности.

Опрос первокурсников мехмата МГУ имени М.В. Ломоносова за последние годы показывает, что в каждой группе из 20-25 студентов обнаруживается до 20-25% первокурсников, которые не отладили в своей жизни ни одной программы на компьютере (не составили и не проверили алгоритм учебной задачи). При этом 30-40% студентов той же группы не способны за разумное время написать программу нахождения числа максимальных значений в числовом массиве (на выбранном самим студентом языке программирования) или даже подсчитать число разных из трех величин.

Педагогический опыт работы со студентами в московских вузах показывает, что многие из них также испытывают трудности в составлении линейных и циклических программ, управляющих виртуальными исполнителями, такими, например, как Робот, Черепашка, Чертежник.

Чтобы решить эту проблему, для педагогических университетов был создан уникальный выравнивающий короткий курс, в задачи которого входила наработка технических навыков последовательного программирования, которые должны были бы быть освоены в средней школе. Предварительное анкетирование показывает, что значительная часть студентов имеет устойчивые отрицательные ожидания по поводу своих возможностей освоения элементов программирования в данном курсе, и потому должны быть приложены усилия по снятию психологического барьера. Для этого есть три пути.

Во-первых, начальные задания выполняются в игровой форме в бестекстовой ЦОС ПиктоМир, имеющей веб-интерфейс и доступной с любого имеющегося у студента привычного ему мобильного устройства. Практика показала, что проведение начальных занятий в игровой форме полностью снимает проблему

аутсайдеров, обладающих недостаточным или нулевым бэкграундом в области программирования.

Во-вторых, на вводной лекции объявляется, что все обучение в курсе будет проводиться в соревновательно-игровой форме, в частности, каждое выполненное студентом задание немедленно проверяется и оценивается системой, что также снимает тревожные ожидания учащихся через 2-3 занятия.

В-третьих, при переходе от «детской» бестекстовой ЦОС ПиктоМир к текстовой ЦОС КуМир используется промежуточная ЦОС ПиктоМир-К, когда появление сложностей дозируется: сначала простая пиктографическая ЦОС ПиктоМир, где сложностью являются только алгоритмы, затем ЦОС ПиктоМир-К, где при сохранении уже известных алгоритмов добавляется новый язык программирования (школьный алгоритмический язык), затем при фиксации алгоритмов и языка меняется среда программирования в ЦОС КуМир. Введение переменных в ЦОС КуМир также вводится с целью перекодировки ПиктоМир-программ, использующих исполнителя-счетчика Кувшин. Такой подход – перекодировка уже составленных программ в текстовую форму - полностью снимает технические трудности освоения новой для обучаемых ЦОС КуМир.

В курсе программирования для освоения основных алгоритмических конструкций по ФОП ООО [246] обучаемые получают 250 заданий в ЦОС ПиктоМир. Обучение основным алгоритмическим конструкциям проводится на примере программ, управляющих роботами, а составлять такие программы легче всего в пиктограммной форме.

После ЦОС ПиктоМир следует познакомить студентов с текстовым представлением программы и основными алгоритмическими конструкциями. Для этого студенты решают 200 заданий в учебной гибридной многоязыковой ЦОС ПиктоМир-К, где программа представляется на экране в одном из двух вариантов:

- на школьном алгоритмическом языке, с выделением блочной структуры рамочками,
- на подмножестве языка Python, с заданием блочной структуры отступами.

В процессе составления программы обучаемый может менять программу путем перетаскивания пиктограмм с командами роботов и заголовками алгоритмических конструкций в текстовое представление программы.

В ЦОС КуМир, отечественной среде программирования на школьном алгоритмическом языке, дается 100-150 заданий, при выполнении которых достигается цель научиться кодировать на языке высокого уровня два десятка заданий, перечисленных в ФОП ООО Информатика, такие, например, как:

- подсчёт частоты появления символа в строке, или
- нахождение максимального или минимального элемента массива.

Для новичка основную проблему представляет клавиатурный ввод текста программы и возможность возникновения при этом синтаксических ошибок. Поэтому при переходе к ЦОС КуМир для привыкания к текстовому школьному алгоритмическому языку студенты начинают с 30 простейших заданий по управлению Роботом, ранее выполненных в ЦОС ПиктоМир и ЦОС ПиктоМир-К, и только после этого переходят к содержательным заданиям на массивы и строки.

Педагогический опыт показал, что при подготовке будущего учителя информатики следует потратить не менее 10-12 часов на изложение перечисленных в ФОП ООО конструкций и понятий программирования на языке Python. Систематическое знакомство обучаемых с Python происходит на последних занятиях, когда студенты получают более 100 заданий для выполнения на языке программирования Python.

4.2 Методика раннего введения процессов счета и числовых данных при обучении информатике и программированию для младших школьников

Методика раннего введения процессов счета и работы с числовыми данными в обучении информатике и программированию для младших школьников включает несколько ключевых аспектов:

- программный исполнитель-счетчик в пиктографической ЦОС ПиктоМир представляет собой инструмент, который помогает детям

научиться составлять универсальные алгоритмы, зависящие от серий похожих ситуаций, а не от конкретной обстановки;

- введения понятия величина в блочной ЦОС ПиктоМир-К, как расширения исполнителя счетчик в блочном и текстовом и представлении программы. Как и счетчик, величины представляют собой контейнеры для хранения значений, которые могут изменяться в процессе выполнения программы.

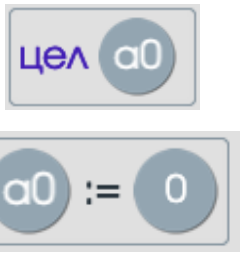
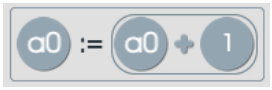
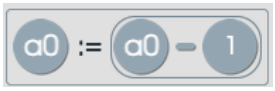
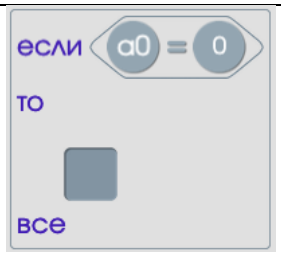
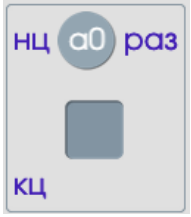
Целые величины в программировании с использованием ЦОС ПиктоМир появляются раньше блочного программирования в ЦОС ПиктоМир-К. Параллельно с введением понятия «обратная связь» в систему основных понятий вводится и понятие числа (неотрицательное целое число). Для «материализации» понятия **числа** вводится виртуальный исполнитель «Волшебный кувшин с камнями», играющий роль *счетчика*. Так, еще до начала текстового программирования, чтобы использовать в программах по управлению роботами элементы счета, дошкольнику нет необходимости расширять понятийную базу и осваивать понятия **переменной** и **команды присваивания**. Ему достаточно освоить семантику команд базового исполнителя «Кувшин» ЦОС ПиктоМир.

Кувшин в ЦОС ПиктоМир можно наполнять камнями по одному или выбрасывать по одному камню за одну команду Кувшина. Несмотря на то, что у Кувшина нет команды, аналогичной присваиванию произвольного положительного целого числа некоей величине, такая операция с успехом имитируется двумя шагами:

- опустошением Кувшина (присвоение величине 0, обнуление);
- **циклом n раз** – где **n** именно то целое число, которое нужно присвоить величине, положить камень в Кувшин.

Таким же способом можно увеличивать или уменьшать значение переменной. Единственная проблема состоит в том, что в программе есть только один целый счетчик, но если задание решается алгоритмом с одним счетчиком, то его можно заменить Кувшином по правилам, указанным в таблице, Таблица 6.

Таблица 6 – Соответствие операций счетчика и Кувшина в ЦОС ПиктоМир и ЦОС ПиктоМир-К

Пиктомир	**ПиктоМир-К**
	
	
	
	
	
	
	

Наличие счетчика позволяет решать интуитивно понятные детям задачи управления роботом типа «дойти до ближайшей стены и вернуться в исходную точку».

Система программирования, в которую включены не один, а два Кувшина (две целочисленные переменные), становится Тьюринг-полной. Это означает, что если дать обучаемому полную свободу в составлении программы, то задача проверки правильности (и даже просто завершаемости) составленной учеником программы станет алгоритмически неразрешимой. С методической точки зрения это означает, что условия заданий должны тщательно подбираться с тем, чтобы обеспечить практически мгновенное понимание воспитателем любой придуманной ребенком программы. Этого можно добиться, задавая достаточно шаблон составляемой ребенком программы.

Разумеется, введение новых понятий – *обратная связь* и *счет* – сопровождается играми. Исполняя роль робота, дети отвечают на команды-вопросы «да» или «нет», исполняя роль Кувшина-счетчика, ребенок по команде добавляет или удаляет камешек из реального кувшина и отвечает на команды-вопросы «кувшин пуст» и «кувшин не пуст». На введение обратной связи в курсе для дошколят требуется не менее 5 занятий. Для устойчивого освоения этих понятий на следующем году обучения потребуется еще 15 занятий.

4.3 Место цифровых образовательных сред и платформ в образовательном процессе и методической системе обучения

Цифровые образовательные среды и платформы играют ключевую роль в обучении информатике и программированию, предоставляя необходимые инструменты и ресурсы для повышения эффективности методической системы обучения и обеспечивая образовательный процесс новыми гибкими формами обучения, а также интерактивность и визуальность при гибридных и дистанционных формах обучения, что в конечном итоге позволяет сделать процесс обучения более наглядным и интересным, а также повышает мотивацию студентов.

В рамках проходящего процесса цифровизации образования в России в 2016 году стартовал проект «Современная цифровая образовательная среда в Российской Федерации», целью которого было предоставление к началу 2021 года 6 млн. школьникам и студентам вузов свободного доступа к обучающим онлайн курсам [229]. К проекту творчески отнеслись общественность, педагоги, ученые. Так, например, С.Д. Каракозов, Л.Р. Пикалова [207] предложили комплекс мер для повышения эффективности имплементации проектов формирования и дальнейшего развития цифровой образовательной среды в субъектах России. Проведенная работа оказала определенную поддержку образования в условиях сложной эпидемиологической обстановки и угрозы заражения обучающихся и преподавателей, когда весной 2020 года по одному самому массовому сценарию проведения занятий по предметам в высших учебных заведениях России было предложено временно перейти на преподавание предметов с использованием только дистанционных технологий [162]. Во время пандемии 2020-2021 гг. важной частью этих образовательных технологий стали не только онлайн лекции (вебинары), но и самостоятельные занятия слушателей в онлайн-курсах. То есть во многом аудиторная учеба и работа в классах была заменена на гибридную очно-дистанционную форму проведения занятий. Так как не все образовательные организации были готовы к такой методике работы, часто занятия заменялись на онлайн-курсы.

Вот, что сказал по поводу онлайн-образования президент России В.В. Путин на ежегодной пресс-конференции 17 декабря 2020 года: «Конечно, онлайн-система, такой формат, не заменит никогда прямого, личного контакта между студентом, учащимся и преподавателем. Во всяком случае, этого ещё очень долго не произойдёт…Тем не менее система, при которой и в школе, и в высшей школе онлайн-образование будет использоваться, – система существует, она, конечно, будет развиваться… И это надо будет делать, это востребовано, это стало частью нашей жизни, и не нужно этого бояться. Но преувеличивать эти возможности тоже не нужно. Это не на всю жизнь, не навсегда – массовое онлайн-образование» [59].

Далее президент высказался по поводу качества образовательного процесса: «Теперь по поводу качества. Конечно, наверное, – я так понимаю, что подвопрос у Вас есть такой скрытый, – конечно, я же сказал, что прямого общения онлайн-система не заменит. И, наверное, в чём-то качество… В чём-то есть плюс, когда есть возможность послушать мировых светил науки, а в чём-то в текущей работе, может быть, и страдает. Поэтому лучше всего, конечно, когда это смешанная система» [59].

Согласно постановлению Правительства Российской Федерации от 07.12.2020 № 2040 "О проведении эксперимента по внедрению цифровой образовательной среды" цифровая образовательная среда– это «совокупность условий для реализации образовательных программ начального общего, основного общего и среднего общего образования с применением электронного обучения, дистанционных образовательных технологий с учетом функционирования электронной информационно-образовательной среды, включающей в себя электронные информационные и образовательные ресурсы и сервисы, цифровой образовательный контент, информационные и телекоммуникационные технологии, технологические средства и обеспечивающей освоение учащимися образовательных программ в полном объеме независимо от места их проживания» [169]. Определение проводит параллель со средой обитания человека [253], однако в определении содержатся важное декларирование элементов цифрового равенства, т.е. доступность для обучающихся цифрового образовательного контента вне зависимости от места проживания и обучения, ИКТ-насыщенность образовательной организации и т.п.

Однако еще в 1990-х годах активно использовались компьютерные практикумы, позволяющие в рамках конкретного предмета получать знания и овладевать навыками, осваивать умения [135]. Поэтому в настоящей работе *цифровая образовательная среда* рассматривается в более узком смысле, как программное обеспечение, приложение, практикум, позволяющий обучаемому получить определенные компетенции в фиксированной предметной области,

ориентированная, как правило, на одну дисциплину или смежные группы дисциплин, включавшие также автоматизированную или внешнюю проверку достижений учащегося. Часто цифровые образовательные платформы пытаются отождествить с цифровыми образовательными средами, что, вообще говоря, неверно. Последние, как правило, обеспечивают поддержку освоения учащимся предмета или даже темы, не предоставляя полноценной коммуникации с педагогом и не привязывая освоение курса к временным рамкам.

Автоматизацию образовательного процесса в вузах проводят с так называемыми LSM системами управления обучением (или виртуальной средой обучения VLE) — программное обеспечение для администрирования, документирования, отслеживания, отчетности, автоматизации и доставки образовательных курсов, программ обучения, материалов или программ обучения.

ЦОП – *цифровая образовательная платформа* – интегрированная программная система, в которой полноценный образовательный процесс осуществляется с использованием цифровых технологий и включает в себя различные образовательные среды, приложения, интерактивные учебные материалы, возможность проведения занятий в режиме видеоконференций, электронные учебники, электронные средства коммуникации между студентами и преподавателями, позволяет создать гибкую, доступную и инновационную образовательную среду (в терминах среды обитания), которая поддерживает разнообразные методы обучения и помогает стимулировать интерес к учебе.

Фактически цифровая образовательная платформа – это программная система, предоставляющая современную мультипредметную цифровую методологию ведения учебного процесса. Эта методика позволяет не только включать обязательные элементы, начиная с лекционных и семинарских занятий и ставшие уже стандартом автоматизированные средства проверки контрольных заданий, но и предоставляет преподавателю и студентам новые формы непрерываемого (в том числе внеурочного) взаимодействия с использованием цифровых методов коммуникации [147; 158].

Цифровая трансформация как процесс в рамках построения цифровой экономики [259; 101] сначала начал развиваться в корпорациях и госструктурах [38], что оказало определенное влияние на цифровую трансформацию в высшем и среднем образовании. Если следовать корпоративной модели цифровой трансформации, то подход состоит в том, что образовательная программа вуза реализуется в цифровом виде в процессе обучения на основании:

• компетентностной модели специальности;

• семантической модели образовательной программы.

К цифровому наполнению программы относятся лекции, практические и лабораторные работы, основные и дополнительные материалы, фонд оценочных средств (формируемые компетенцией на основании паспорта специальности и требований ФГОС) и материалы готовых дистанционных/гибридных курсов.

Для трех участников цифровизации образования, преподавателей, студентов и администрации вуза цифровая трансформация дает в том числе следующее:

- Для преподавателя: расширение учебных ресурсов, возможность индивидуализировать обучение, новые формы взаимодействия, автоматизацию оценивания, мониторинг процесса;
- Для студента: доступ к образовательным ресурсам, интерактивные и индивидуальные формы обучения, обратная связь, развитие навыков цифровой грамотности, возможность самостоятельного обучения;
- Для вуза: гибкие формы обучения, расширение территориальной доступности, повышение качества образования, гибкое управление учебным процессом, развитие научных исследований, создание новых образовательных продуктов и услуг.

Архитектура цифрового наполнения вуза в рамках цифровизации образовательного процесса изображена на рисунке, Рисунок 40.

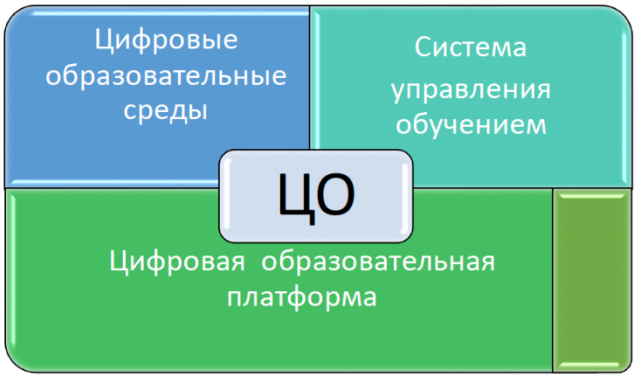


Рисунок 40 – Три инструмента цифровизации образования (ЦО): Цифровые образовательные среды, Система управления обучением, Цифровая образовательная платформа.

Цифровые образовательные ресурсы вузов (платформы, среды, системы [111; 159; 168]) можно разделить на три класса, исходя из их характерных особенностей:

- Системы массового онлайн образования, предоставляющие универсальные цифровые образовательные услуги, которые может использовать учебное учреждение при организации образовательного процесса.

- Основанные на свободном программном обеспечении онлайн-системы, развернутые в том числе и на инфраструктуре образовательной организации.

- Авторские образовательные среды, ориентированные на обучение в какой-либо одной области, например, цифровые среды для обучения программированию.

Примерно с 2010-х годов массовые онлайн-курсы уверенно вошли в структуру системы образования, дополняя ее за счет предоставления готовых курсов по различным направлениям. Яркими представителями подобных систем являются Coursera [314; 13] и Stepic.ru [231]. Данные системы ориентированы на максимальный охват аудитории и возможность проходить курсы без привязки к учебному графику вуза.

Для более глубокой связи с образовательными программами вуза платформы предлагают методику программной интеграции к системе управления учебным

процессом вуза, примером которого может послужить система Coursera for Campus [395].

Разработчики массовых онлайн образовательных курсов размещают свои материалы, снабжая заданиями для промежуточных и финальных испытаний для определения уровней компетенций, освоенных обучаемым. Поскольку предполагается универсальное решение, то тесты чаще всего ограничены заданиями с выбором варианта из списка или кросс-проверкой самих студентов курса (в последнем случае курс должен одновременно проходиться большим количеством студентов). Более сложные проверки, такие как проверки правильности результатов выполнения программного кода [226] или правильности выполнения графических заданий, могут быть реализованы в ограниченном формате и, как правило, не являются частью базовых возможностей платформы.

В массовых онлайн образовательных курсах возможности по адаптации, размещенных на платформе курсов, как правило, не предусматриваются, построение индивидуальных треков обучения возможно только на уровне выбора готовых учебных курсов и порядка их прохождения. Безусловно, наличие на платформе типа Coursera нескольких тысяч готовых к использованию курсов является существенным преимуществом, но для использования конкретных курсов в школьной или вузовской системе образования необходимо, чтобы эти курсы отвечали всем требованиям используемых стандартов организации. Однако это не является исходной целью авторов данных курсов, ориентированных в большей степени на повышение квалификации и переобучение, чем на обеспечение конкретных требований, возникающих в различных вузах.

Зарубежные исследования МООК-платформ показали, что на рубеже 2020-х годов лишь 3% учащихся, оплативших выбранный курс дистанционного образования, заканчивали его [207]. Считается, что качество онлайн-курсов серьезно уступало очному образованию, например, за-за частого отсутствия четкой компетентностной модели курса, сквозного построения курсов, отсутствия единой методики, нехватки методически продуманного контента для наполнения курсов.

Создание цифровых учебников и онлайн-курсов для школьников в России было обусловлено рядом мероприятий, проведенных Министерством просвещения [257], включая федеральный проект «Цифровая школа», стартовавший в 2018 году [233]. В рамках сформулированных требований к цифровой школе должен, в частности, предоставить ученикам возможность получать доступ к материалам в электронном формате, что дает им возможность обучаться в любое время, находясь в любом месте.

Также необходимо учитывать, что при подключении студентов вузов и школ России к массовым онлайн образовательным курсам провайдер услуги должен обеспечить безопасное хранение персональных данных учащихся [167], возможность длительного доступа как к результатам тестирования, так и материалам курсов.

Альтернативой массовым онлайн образовательным курсам является использование одной из систем (платформ) с открытым программным кодом, таких как Moodle LMS [8]. Безусловным преимуществом систем с открытым исходным кодом является возможность их глубокой интеграции в систему управления учебным процессом вуза и школы, но при этом надо учитывать, что существенный объем интеграционных работ потребует привлечения специалистов и наличия развитой IT-службы вуза и школы.

Поскольку системы с открытым программным кодом являются универсальными, наличие встроенных средств проверки знаний студентов также будет ограничено только типовыми проверочными заданиями, для подключения специфичных проверок потребуются дополнительная разработка и интеграция. Адаптация учебных материалов в системах с открытым программным кодом также ограничена возможностями по выбору набора курсов и порядка их прохождения.

Особенности авторских систем можно проследить на примере авторской системы проведения олимпиад по программированию ejudge [226] – системы для организации испытаний в режиме соревнования, в которых необходима автоматическая проверка программ. Многие преподаватели пытались применять

эту систему для поддержки учебных курсов. Существуют определенные требования к функционированию подобных цифровых образовательных сред, которым ejudge не удовлетворяет [226].

Авторские системы автоматической проверки, предназначенные для образования, должны иметь и авторское сопровождение и поддержку. Кроме того, ejudge полноценно не реализует BYOD-Learning или Learning at Any Time, at Any Place via any Device, поддержание которого гарантирует корректность работы на любых, в том числе и мобильных устройствах [384], Рисунок 41.

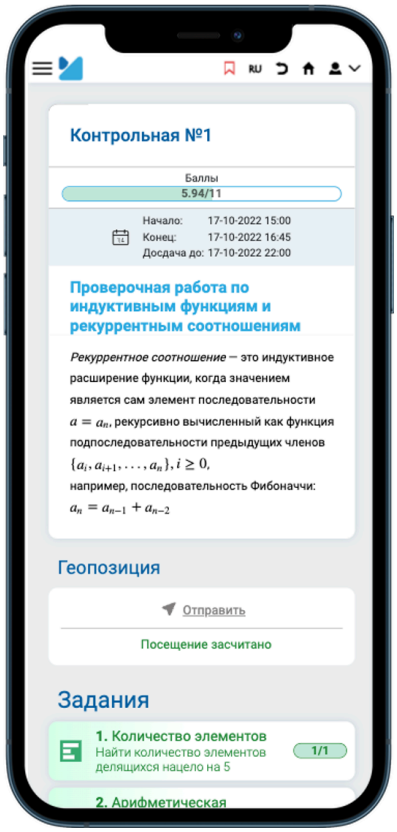


Рисунок 41 – Пример экрана мобильного устройства в авторской ЦОП Мирера, осуществляющий принцип BYOD-Learning: Bring Your Own Device (Принеси свое личное устройство).

Подобные системы должны предоставлять гибкий и дружественный режим работы для студента, что подразумевает в том числе предоставление возможности редактировать отдельные файлы, то есть система должна предоставлять учащимся

возможность писать и исправлять свой код прямо в своей среде. Хотя в ejudge имеется возможность включить отправку текста программы в качестве решений, но такой подход не заменяет полноценную среду программирования.

Для введения элементов геймификации, поддерживая утверждение, что этот подход носит универсальный, доступный любому возрасту, характер, а также традиции олимпиад по программированию, академические аудиторные занятия с использованием цифровых сред именуются *контестами* [262], подчеркивая, что ученики и студенты являются участниками соревнований между собой и сами с собой, непрерывно наблюдая оценку своих достижения в бальной системе.

Для преподавателя также необходимо предоставлять простой удобный механизм, позволяющий интерактивно вмешиваться в идущий образовательный процесс, например, останавливать, редактировать и перезапускать отдельный контест для другой группы обучающихся. При этом такой перезапуск не должен влиять на данные предыдущих участников, и они не должны быть не доступны новым участникам. Тем самым такой механизм не должен заключаться в добавлении новой группы к прошедшему контесту, а должен быстро и надежно создавать новый, что обеспечивает накопление и сохранность цифрового следа.

Повышение эффективности процесса обучения достигается не только за счет технических и дидактических средств, но и с помощью организации обратной связи с обучаемым, которая обеспечивается контролем знаний в интеллектуальных обучающих системах, где контроль знаний, как правило, осуществляется автоматизированными тестирующими компонентами.

4.4 Необходимость создания отечественных цифровых образовательных платформ как средства обучения профильной методической системы обучения и элемент технологического суверенитета

ЦОП Мирера, как средство обучения профильной методической системы обучения – это образовательная платформа для поддержки различных форм

обучения от гибридного формата обучения до дистанционного и аудиторного. Это инструмент для помощи преподавателю в создании курса с сохраняемой иерархией и размещением на платформе всех учебных материалов, настройкой расписания курса независимо по всем учебным группам, проведением занятий со встроенной или внешней системой видео-конференций, сдачей и проверкой домашних заданий с автоматизированной или ручной проверкой, подведением статистики по всем курсам и группам со встроенным журналом посещаемости и с автоматическим формированием отчетности.

В эпоху активного использования дистанционных форм обучения (связанную с пандемией Covid-19) создание отечественных авторских цифровых образовательных платформ становится актуальной научно-практической задачей. С одной стороны, отечественные авторские платформы будут иметь существенно более надежную поддержку разработчиками в России, с другой стороны, чисто дистанционные автоматизированные курсы показали низкую эффективность с точки зрения качества выпускников, так как учебный процесс без преподавателя слабо мотивирует студентов к освоению предмета. Напротив, поддерживаемое цифровыми образовательными платформами смешанное обучение позволяет выстраивать процесс обучения на основе сильных сторон слушателей и находить индивидуальный подход, формируя индивидуальные образовательные треки для обучаемых.

Использование смешанной формы обучения освобождает преподавателя от выполнения рутинных операций по проверке текущей успеваемости слушателей и верификации результатов. С одной стороны, это позволяет преподавателю контролировать учебный процесс и в любой момент времени иметь актуальную информацию об успеваемости. С другой стороны, в этой методической парадигме преподаватель избавлен от необходимости ежедневной проверки заданий, что не только спасает педагога от переписки со слушателями по ошибочным заданиям, но и позволяет существенно увеличить число заданий, выполняемых обучаемыми в рамках изучаемого курса и (или) предмета. Контроль выполнения заданий и

продвижение по предмету слушателя может поддержать, например, система образовательных ботов, которые не только отвечают на содержательные вопросы по материалу, но и сообщают обучаемому о его задолженностях, успехах, расписании занятий и актуальном текущем изучаемом материале.

Использование диалоговых интерфейсов и чат–ботов в коммерции уже является устойчивой тенденцией оптимизации бизнес–процессов [101]. При этом, хотя использование мессенджеров для реализации диалоговых интерфейсов является стандартной практикой, применение чат–ботов в образовательном процессе пока находится в экспериментальной стадии. Чат–боты могут заменить традиционные поисковые интерфейсы, позволяя в форме диалога найти необходимую информацию на сайтах образовательных организаций [33].

Однако основной потенциал применения чат–ботов − в роли виртуальных ассистентов преподавателя. Такие ассистенты могут снять часть рабочей нагрузки с преподавателя в области автоматической проверки знаний, проведения опросов и анкетирования, информирования студентов о ближайших событиях учебного курса [272].

В процессе преподавания педагогу приходится часто отвечать на типовые вопросы студентов, что отнимает много времени и часто нерационально расходует аудиторное время. В снижении нагрузки на учителя поможет виртуальный ассистент, который также будет доступен во внеаудиторные часы и сможет отвечать на основные вопросы учеников. Фактически цифровой интерактивный помощник представляет собой информационно-поисковую систему, позволяющую формулировать запросы к ней на естественном языке.

Сама идея диалогового режима общения с учениками с использованием элементов искусственного интеллекта появилась еще в 1970-х [299], но широкое применение относится к 2010-м годам благодаря росту производительности и мобильности вычислительной техники, а также новых технологий, позволяющих эффективную реализацию методов анализа естественного языка. В список современных требований включаются: поддержка лексики учащихся

(аббревиатуры, сленг и т.д.), устойчивость к орфографическим и синтаксическим ошибкам, сбор статистики по вопросам для ее дальнейшего анализа, возможность переадресации вопроса оператору-человеку, если цифровой ассистент не смог на него ответить.

Несмотря на сложность подобных систем, для преподавателя нужно предоставить возможность решения без специальных знаний, описывая предметную область в виде типовых пар вопросов и ответов со справочными материалами.

Разработчику виртуального ассистента преподавателя (чат-бота) необходимо учитывать особенности образовательного процесса, такие как возможность накопления базы знаний с автоматическим обновлением при небольшом объеме тематических данных для обучения на этапе запуска бота. Примечательно, что также целевая группа, состоящая из студентов и старших школьников, обладает нехарактерным для общей массы людей сленгом и стилем переписки. Поддержка свободного диалогового режима с ботом за пределами учебных целей позволяет студентам чувствовать себя свободнее в процессе диалога с чат-ботом.

В работе ученых Токийского столичного университета (англ. Tokyo Metropolitan University, Japan) предлагались алгоритмы создания новых пар вопрос-ответ или объединения сходных пар [389]. Также проводились исследования по созданию чат-ботов, умеющих поддерживать нетематический диалог [277].

В ЦОП Мирера используются тематический и информационные чат–боты с поддержкой распознавания естественного языка и с закрытым доменом. Последнее позволяет создавать предметно–ориентированную систему со сменяемой тематической базой. Если информационный чат-бот не требует регулярно переучивать свою нейросеть, то для тематического процесс машинного обучения является достаточно частым при расширении базы часто встречающихся пар вопрос-ответ на основании эмпирического опыта. При этом чат–бот позволяет

реализовывать сценарии, которые поддерживают особенности обычной речи с многообразием речевых оборотов.

Авторская цифровая образовательная платформа Мирера [146] проектировалась, как интегрированная с национальной социальной сетью ВКонтакте система для организации доступа к образовательному онлайн-контенту студентов университетов в курсах программирования. Такой подход показывает свою эффективность, хотя нагрузка на преподавателя возрастет, так как цифровая трансформация образовательного процесса, цифровые курсы требуют от педагога находиться на связи со студентами постоянно, круглые сутки без праздников и выходных дней. Такой образовательный процесс называют *непрерываемым.*

Изменение подхода при гибкой и гибридной (смешанной) форме обучения обладает следующими характерными свойствами:

- Самостоятельная работа студента в цифровой образовательной среде и(или) платформе;
- Непрерывная поддержка контакта с преподавателем, в том числе и во внеурочное, внеаудиторное время;
- Образовательный процесс представляет собой непрерываемый поток освоения желаемой компетенции.

Таким образом, происходит двойная конвергенция аудиторного и дистанционного занятия и замена лекционно-семинарской формы на *леминар* (ЛЕкция+сеМИНАР, как правило, во временном соотношении 1:3) по конкретной теме излагаемого предмета, с теоретической частью, с демонстрацией, связанной с темой занятия материалом, с использованием (виртуальной) доски, ответами на возникшие вопросы в реальном времени и практической частью с большим количеством заданий, проверяющихся автоматически в ЦОП Мирера, что позволяет большую часть задач выполнять в аудиторное время.

К сложностям непрерываемого образовательного процесса относится значительная временная нагрузка постоянного (личного) контакта с обучаемыми, в том числе с использованием цифровых методов коммуникации, а именно: обмен

сообщениями, в том числе и во вне цифровой образовательной платформы, социальные сети, мессенджеры, видео-конференц связь и т.п. Однако без этой связанности ученик, потеряв контакт с учителем, как правило, не сможет эффективно пройти обучение, полноценно воспользоваться всей мощью сервисов, предоставляемой цифровой образовательной платформой. В распоряжении учителя и ученика находится весь функционал цифровой платформы, всегда есть доступ к набору инструментов, который отсутствует в классической форме аудиторного образовательного процесса без применения цифровых платформ, Рисунок 42.

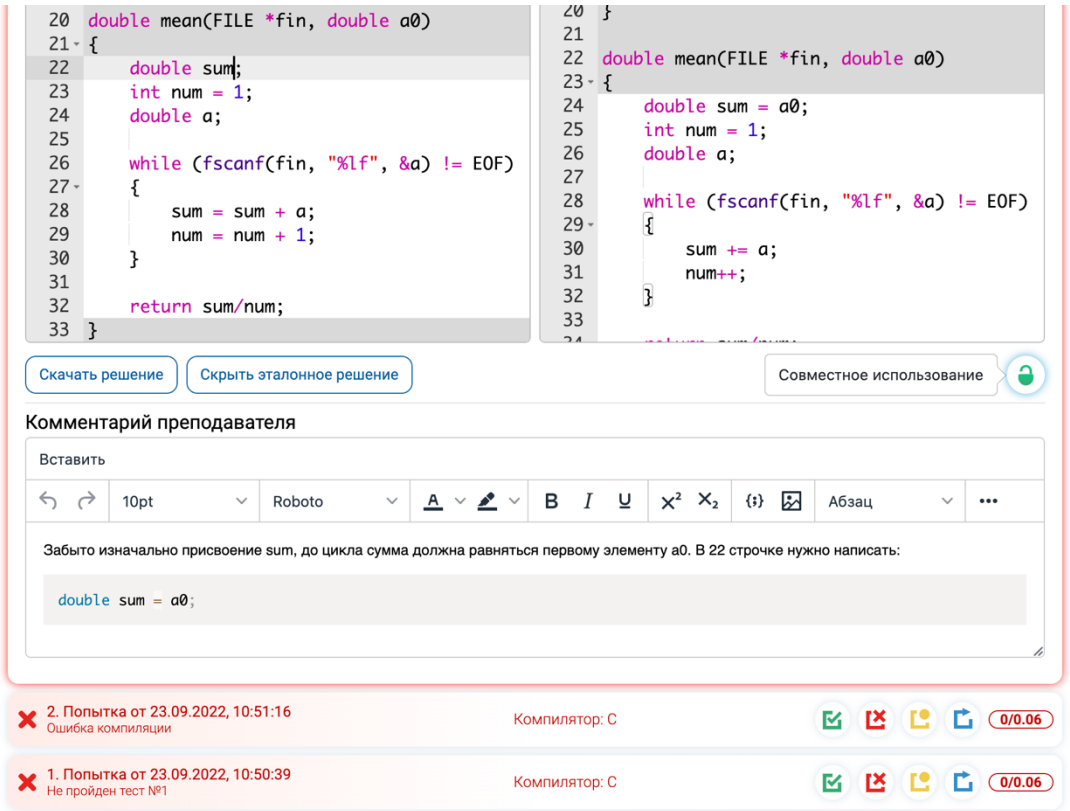


Рисунок 42 – Пример экрана ЦОП Мирера в момент помощи преподавателя студенту в решении задания. Преподаватель редактирует студенческую задачу, и слушатель наблюдает за процессом на личном компьютере (справа окно с эталонным решением задачи)

Для предоставления педагогу возможности конструирования онлайн-курсов с большим объёмом цифрового материала, автоматизированной проверкой заданий

и накоплением базы выполненных заданий студентами, была разработана технология интеграции обособленных компонентов ЦОС ПиктоМир, ЦОС ПиктоМир-К для использования последних элементов трансформируемого образовательного процесса в смешанной форме обучения. Такой единообразный подход модели обучения повышает эффективность учебного процесса, позволит нарастить объем знаний и навыков как у студентов (получить новый уровень компетенции в изучаемом предмете, так как объем выполненных заданий возрастает на порядки), так и у студентов, так и у школьников, Рисунок 43.

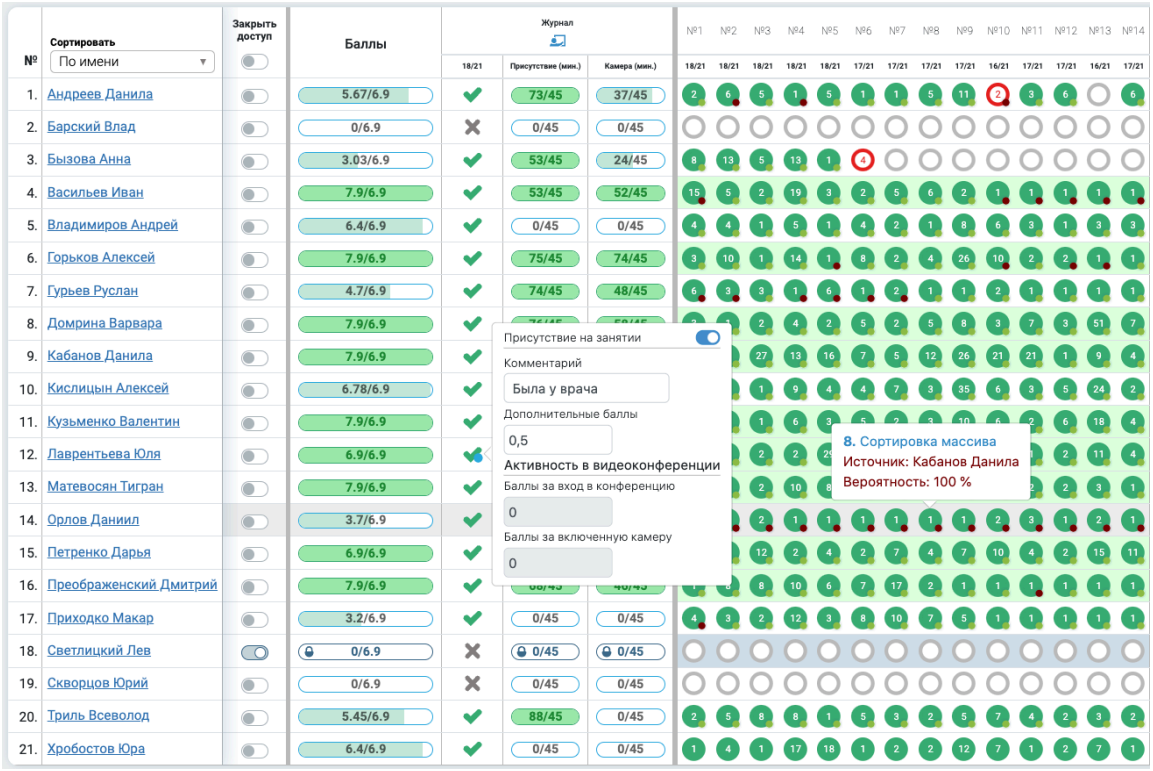


Рисунок 43 – Пример экрана ЦОП Мирера с автоматически обновляемой страницей с результатами контеста-семинара

Мирера – это проект платформы для смешанного (гибридного) обучения, в котором личный контакт педагога с обучаемыми в процессе прохождения курса/предмета является необходимым условием. Кроме того, ЦОП Мирера обладает глубоким уровнем интеграции авторских ЦОС, наличием встроенной системы вебинаров, face-to-face ВКС семинаров, интеграцией с социальной сетью

ВКонтакте, цифровыми помощниками преподавателя (ботами), а также мессенджером Telegram и многим другим, что не доступно в многих других образовательных платформах [378; 274; 198; 209].

ЦОП Мирера поддерживает автоматизированную проверку абсолютно любых задач по программированию. В задаче для любого языка программирования, который можно выбрать из нескольких предложенных, можно установить все настройки и ограничения компилятора, создать все входные и выходные файлы тестирования, а также выставить один из многих типов автоматизированной проверки. Можно создать тесты для задачи, которые возможно мгновенно сгенерировать по эталонному решению преподавателя. Учебная программа может состоять из нескольких файлов, преподаватель может настроить в помощь студенту шаблон, скрыть дополнительные служебные файлы. Благодаря всему этому необходимость установки студентами под каждую изучаемую тему отдельной системы программирования полностью отпадает, студенты могут писать программы непосредственно в ЦОП Мирера.

Помимо этого, ЦОП Мирера поддерживает проверку всех видов тестовых задач, а также ручную проверку и оценку домашних заданий в свободной форме с последующим оставлением комментария и выставлением баллов преподавателем. Задачи могут быть разделены по вариантам, каждая задача имеет настраиваемый вес, студенту за задачу можно выставить любое количество баллов.

В ЦОП Мирера большое внимание уделяется скорости и удобству внесения преподавателями в систему своего курса. В задачах по программированию есть эталонное решение и автоматическая генерация тестов по нему, а также есть множество средств копирования на всех уровнях и развитый функционал, с которым разработать любой цифровой курс в ЦОП Мирера получается быстрее, чем вне ее.

В ЦОП Мирера реализована многоуровневая система прав доступа для курсов и групп, позволяющая настроить определенный доступ преподавателям и студентами к каждой сущности системы.

В ЦОП Мирера реализована поддержка адаптивных образовательных траекторий, благодаря которым можно автоматически выдавать различной сложности задачи успешным или, наоборот, отстающим студентам, а также автоматически выдавать задачи на зачет или экзамен по незакрытым темам, в том числе и открытия контестов для досдачи с понижением баллов, последовательное открытие задач и многое другое [11].

В платформу встроена автоматическая система антиплагиата, примененная к решению студента и эталонному решению преподавателя, Рисунок 44.

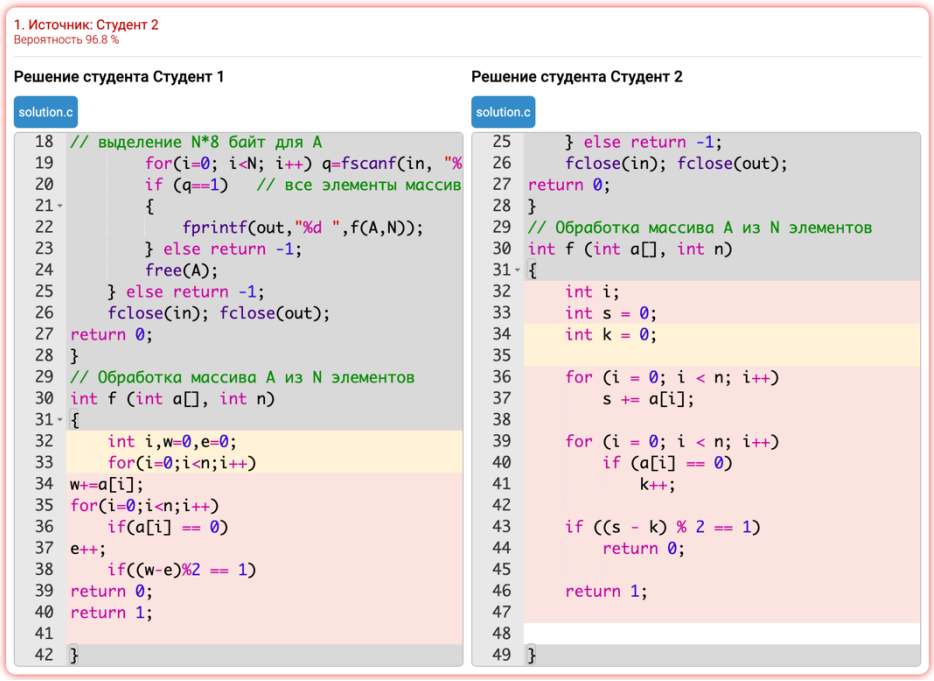


Рисунок 44 – Пример экрана ЦОП Мирера, демонстрирующий работу антиплагиата.

Авторская система антиплагиата с возможностью обнаружения цепочек заимствований между студентами и выделением для преподавателя подсветкой списанных участков каждой пары решений [5]. При этом модуль антиплагиата ЦОП Мирера исключает из проверки шаблон программы, который представляет

некую подготовленную педагогом часть задания текста программы или решения, защищенного от удаления и изменения. Модуль антиплагиата ЦОП Мирера также учитывает созданное преподавателем эталонное решение, игнорируя факт заимствования этих частей эталонного решения, что дидактически рассматривается как успешный факт освоения учебного материала учеником. Высокий процент плагиата на эталонное решение гарантирует, что студент использовал правильный подход построения алгоритма решения задания, при этом стилистические особенности программного кода игнорируются и не влияют на результат тестирования. Такая методика проверки приводит к верифицированию только оригинальных фрагментов студенческого решения, что минимизирует количество ложноположительных результатов проверки.

Визуализация для педагога сравнительного анализа эталонного решения и решения ученика, то есть применение модуля антиплагиата к решению студента и решению преподавателя выявляет совпадения программного кода, что позволяет педагогу в режиме онлайн-консультации помогать учащемуся в поиске логических ошибок в решении, фокусируя внимание преподавателя на различиях кода, упрощая локализацию ошибки. При этом принцип «делай как я» хорошо себя зарекомендовал на начальном этапе освоение программирования, когда ученик больше должен сосредотачиваться на алгоритмических проблемах задания, используя шаблонный подход к кодированию на языке программирования. Это позволяет быстрее овладевать навыками корректного написания программного кода, что позволяет ученику избегать большего количества ошибок в будущем и никак не ухудшает креативность при программировании будущих сложных проектов.

В ЦОП Мирера реализованы различные методы проверки заданий по естественно-научным, например, инженерных задач для вычислительных, графических и табличных заданий, а также возможности по проверке полнотекстовых ответов на открытые вопросы [90].

Тестовые задания, предлагаемые обучающимся, можно разделить на два типа: открытые и закрытые. Первые подразумевают конструируемый ответ на вопрос, а вторые – выбор одного или нескольких ответов из предложенного списка. Основная проблема закрытых вопросов заключается в том, что они проверяют, насколько хорошо обучающийся запомнил и ассоциировал ответ с вопросом, но не требуют от него размышлений. Проверить логику и степень понимания материала таким способом часто не удастся. Открытые вопросы, напротив, заставляют ученика и студента поразмышлять над вопросом, составить конструктивный ответ. Это является наиболее привычной и естественной формой контроля знаний. Проблемой открытых вопросов является огромная рутинная нагрузка по проверке студенческих ответов у преподавателя. Основная задача, которую должна выполнять интеллектуальная система для проверки ответов на открытые вопросы, заключается в выявлении и проверке смысла и мыслей в проверяемом ответе. За счет сравнения этих мыслей с извлекаемыми из эталонного ответа преподавателя определяется правильность студенческого ответа. Преимуществом автоматизации проверки является не только снижение рутинной нагрузки преподавателя, но и прозрачность и справедливость результата, поскольку исключается фактор предвзятости со стороны преподавателя.

Автоматизация проверки знаний – источник обратной связи, поэтому она важна и является ядром современной технологии автоматизации обучения. Расширяя возможности автоматической проверки, расширяются возможности применения наработок в областях обучения, традиционно считавшихся плохо поддающимися автоматизации. Как указано выше, система проверки заданий, в том числе и с открытыми вопросами, должна дополняться анализом заимствований, чтобы сохранять уверенность в качестве получаемых данных.

Педагогический опыт преподавания программирования на мехмате МГУ имени М.В. Ломоносова дает возможность сделать вывод, что ЦОП Мирера значительно снизила рутинную нагрузку на преподавателя. Ранее преподавателю приходилось все время занятия тратить на проверку задач, и студенты успевали за

семестр выполнить в среднем около 10 заданий, а у преподавателя не хватало времени на объяснение новых тем и на индивидуальный подход к студентам. Студентам приходилось много времени тратить на установку программного обеспечения, а также у большинства студентов не получалось сразу разобраться со всеми техническими тонкостями языков программирования.

Благодаря предложенным методическим решениям получилось:

1) Повысить количество решаемых студентами задач с 10 до 300 за семестр.

2) Снять значительную рутинную нагрузку с преподавателя за счет автоматизированной системы проверки с подведением отчетности, а также нейросетевых тематических чат-ботов для ответов на типичные вопросы студентов.

3) Существенно повысить успеваемость студентов за счет шаблонов программ и скрытых технических файлов на первых этапах изучения, а также отсутствия необходимости устанавливать различное программное обеспечение.

Важная часть платформы – интеграция внешних систем и сервисов, например, цифровых образовательных сред. При этом процесс интеграции одной системы в другую необязательно связан с односторонним расширением функциональных возможностей платформы. Интеграция позволяет добавить функционал в обе системы, при этом сохранив целостность и функциональность каждого цифрового образовательного продукта.

Одним из примеров описываемого процесса является интеграция программной системы ЭВМ-практикум в ЦОП Мирера. ЭВМ-практикум – это цифровая образовательная среда, в которой в качестве объектов выступают оперативная память и регистры центрального процессора некой виртуальной ЭВМ [125] (модифицированный и сокращенный вариант Intel x86), Рисунок 45.

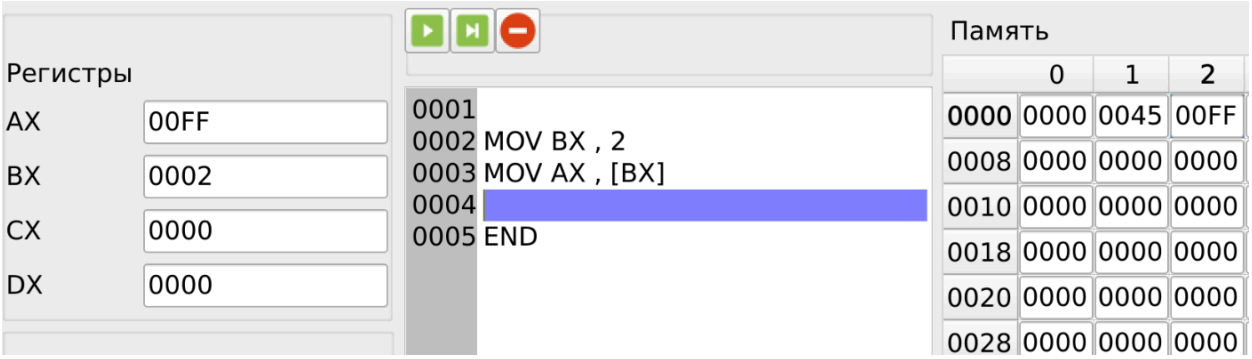


Рисунок 45 – Пример экрана ЦОС ЭВМ-практикума.

С помощью ЭВМ-практикума можно не только непосредственно редактировать эти объекты, но и создавать программу на ассемблере виртуальной ЭВМ, выполнять программу непрерывно или по шагам и наблюдать, как меняются объекты – память, регистры, стек и пр. Таким образом, с точки зрения ЦОП Мирера, ЭВМ-практикум предоставляет интерфейс для написания программного кода, загрузки тестирующего файла и проверки решения на загруженных тестах. Идея интеграции и ее непосредственная реализация заключается в следующем [124:

- ЦОП Мирера поддерживает стандартный формат курса, где создаются задания для ЦОС ЭВМ-практикум;
- Студент, работая в ЦОС ЭВМ-практикум, получает из платформы Мирера свои задания, выполняет их, проводит локальное независимое тестирование задач;
- По желанию студента отлаженная программа как решение задания посылается в ЦОП Мирера, которая и осуществляет финальный контроль (проверку на собственных ресурсах) и фиксирует у себя в базе данных факт решенной студентом задачи, сохраняя полный цифровой след контакта ЦОП Мирера с ЦОС ЭВМ-практикум.
- ЦОС ЭВМ-практикум, в свою очередь, может запрашивать задачи нужного типа из ЦОП Мирера и использовать полученные данные.

Таким образом, под интеграцией сторонних ЦОС подразумевается не только взаимодействие платформы со средой по определенному протоколу с обменом

сообщений, но и непосредственное внедрение тестирующей системы ЦОС в ЦОП Мирера. В качестве дополнительной возможности такая интеграция позволяет студенту не использовать ЦОС, а выполнять задания прямо в ЦОП Мирера. Примером такого подхода может служить ЦОС КуМир. Интеграция с ЦОС КуМир обеспечила функциональную возможность создавать, редактировать и запускать задачи на школьном алгоритмическом языке, прямо находясь на платформе Мирера, проверять решение, используя компилятор, который является частью ЦОС КуМир. В результате ученик может не переключаться между различным программным обеспечением, а осваивать различные предметы в знакомом интерфейсе единой универсальной цифровой образовательной платформы. В ЦОП Мирера имеется встроенный редактор для создания и редактирования учебных решений, в том числе и кода программ. Этот редактор в том числе позволяет создавать и использовать собственные стили текста, подсветку и конфигурации препроцессора для максимального погружения в привычную среду разработки.

Редактор предоставляет средства для гибкого создания правил подсветки и набора управляющих выражений, определяющих синтаксис языка написания, в том числе, естественно, и языка программирования. При интеграции ЦОС КуМир и ЦОП Мирера были созданы поведенческие шаблоны, структуры, описывающие грамматику и механизмы, присущие школьному алгоритмическому языку. Были также определены ключевые слова, типы данных и команды, доступные для взаимодействия с языком программирования. При этом редактор ЦОП Мирера автоматически использует описанные указанные выше параметры синтаксиса и подсветки, когда ученик находится в редакторе программы, написанной на школьном алгоритмическом языке (как в ЦОС КуМир).

ЦОП Мирера аналогично осуществляет взаимодействие с ЦОС ПиктоМир. Преподаватель может создавать, редактировать и удалять объекты в ЦОС ПиктоМир, которые имеют свои объекты-двойники в ЦОП Мирера. Это позволяет объединять функционал двух систем, превращая их в единый цифровой образовательный продукт. Когда ученик решает задания, назначенные ему

учителем в ЦОП Мирера, сам при этом находясь в ЦОП ПиктоМир, то сведения о решенных им задачах и само решение передаются в ЦОП Мирера, где учитель может отслеживать его в режиме реального времени и взаимодействовать с учеником напрямую.

ЦОП Мирера ориентируется на российских студентов 21 века, обладающих современными средствами коммуникации и доступом в Интернет. Платформа обеспечивает студентам возможность работы с использованием любых массовых фаблет-устройств и осуществляет автоматизированную онлайн-помощь в режиме 24/7.

С технологической точки зрения ЦОП Мирера является распределенной многопроцессорной системой, не привязанной жестко к каким-либо аппаратным или программным платформам. Персональные данные студентов и материалы курсов хранятся на серверах, расположенных на территории Российской Федерации. ЦОП Мирера не использует проприетарное ПО, а из свободно распространяемого ПО использует зрелые продукты, получившие широкое распространение.

4.5 Методы формирования индивидуальных образовательных траекторий в ЦОП Мирера

Актуальным и инновационным подходом для автоматизации образовательного процесса является индивидуализация обучения, которая позволяет реализовать часть традиционной работы преподавателя – адаптацию учебного материала под студента с учётом демонстрируемых им результатов. С учетом трудоёмкости на стороне преподавателя такая адаптация без автоматизации возможна, как правило, при работе в небольших группах, поэтому попытки реализации автоматизации этой возможности являются неотъемлемым элементом текущего поколения учебных систем и подходов.

Как было сказано в главе 1, методики индивидуализации образовательного процесса в педагогике используются достаточно давно. Наиболее простой метод

индивидуализации, используемый еще в советской школе, состоит в выборе одного из трех направлений при практическом освоении предмета, в рамках которых педагог индивидуально или по малым группам предлагает учащимся индивидуальные образовательные траектории:

1. дополнительные занятия и задания на повторение при неудовлетворительных результатах промежуточного контроля или перед финальными испытаниями;
2. дополнительный или замещающий материал повышенной сложности за пределами основного содержания предмета программы. В школьных и вузовских учебниках и задачниках такой учебный материал помечен звёздочками [3; 114; 37];
3. ординарный материал в рамках программы предмета из учебников и задачников.

Индивидуализация обучения рассматривается как одна из целей автоматизации образовательного процесса, при которой адаптация учебного материала, методика и дидактические приемы функционально зависят от конкретного обучаемого или группы, базируясь, как правило, на достижениях последних. Ручная индивидуализация обучения практически невозможна, так как является трудоёмкой задачей. Легко подсчитать, что если в курсе «Работа на ЭВМ и программирование» на механико-математическом факультете МГУ имени М.В. Ломоносова, г. Москва, группа студентов состоит, например, из 30 человек, семестр составляет 17 недель при занятиях 1 пара (90 минут) в неделю и при этом на проверку задания преподаватель тратит 15 минут, то за время курса он сможет проверить в среднем $17 \cdot 6/30 = 3.4$ задачи у студента. При этом ни на какую другую педагогическую деятельность в рамках аудиторных занятий у преподавателя не будет времени. Это убедительно показывает, что задача автоматизации учебного процесса является наиболее актуальной в современной практической педагогике.

Вопросы индивидуализации образовательного процесса исследуются в России и за рубежом. Так, в работе И.А. Кречетова, М.Ю. Дорофеевой и

А.В. Дягтерева [96] представлены результаты исследования в области внедрения адаптивного обучения в вузе, подробно описаны шаги создания адаптивного курса, однако авторы не предлагают подходы к масштабированию применения предложенных методик, которые могли бы обеспечить широкое применение адаптивного обучения. В другой работе И.А. Кречетова и В.В. Романенко [97] детально рассматривается алгоритм построения траектории изучения модулей адаптивного курса, прохождение по которой максимизирует уровень знаний студента на момент окончания курса на основании аппарата генетических алгоритмов для построения оптимальной траектории. Примечательно, что при этом остаточные знания студентов оцениваются с использованием кривых забывания. Исследование авторов характеризуется технической сложностью реализуемого алгоритма адаптации, но авторы уделяют меньше внимания оценке результатов эксперимента и ограничиваются проведением эксперимента на ограниченном количестве студентов.

В исследовании Ю.В. Вайнштейн, Р.В. Есина, Г.М. Цибульского [18] рассмотрено построение адаптивного курса обучения с вариацией сложности проверочных заданий и языка представления учебных материалов. Данное исследование характеризуется глубокой научной проработкой теоретико-методологических особенностей применения адаптивного обучения и организацией комплексного педагогического эксперимента, однако в основе технологии адаптации находятся, по сути, эвристические методы.

Работы Н.В. Комлева, Д.А. Вилявина [92] и Т.М Шамсутдиновой [263] объединяют использованный подход к адаптации - изменение порядка изучения учебных материалов. К отличиям можно отнести то, что в первом исследовании основное внимание уделено технической стороне реализации системы адаптивного обучения, разработке курсов и представлению учебных материалов, тогда как во втором исследовании больше внимания уделяется методологической проработке исследования. Однако оба исследования характеризуются ограниченным по объему экспериментом по оценке эффективности предлагаемых решений.

В обзоре состояния области исследования и применения адаптивного обучения К.А. Вилковой, Д.В. Лебедева [21] авторы отмечают, что имеющиеся в научных источниках результаты исследований не дают однозначного ответа об эффективности адаптивного обучения в общем случае, при этом замечено, что всего лишь одна система адаптивного обучения Plario является отечественной. Фактически адаптивные курсы оправданы в качестве выравнивающих курсов, когда хорошо подготовленный студент быстро проходит адаптивный выравнивающий курс, поскольку для него формируется минимальная по длине траектория обучения, в свою очередь, студент с пробелами в подготовке сможет пройти расширенную версию выравнивающего курса, повторив материал, вызывающий наибольшие сложности. Но в [96] указывается, что вариативность адаптивных курсов является ограниченной и предполагает наличие минимальных начальных знаний.

Несмотря на то, что в описанных выше исследованиях применялись сложные методы адаптации, такие как генетические алгоритмы, передовым подходом к решению задачи построения адаптивной траектории можно считать использование искусственных нейронных сетей. В исследовании C. Piech и др. [319] предложена реализация технологии трассировки знаний с использованием глубокого обучения, особенностью которой является необходимость использования для разработки решения больших объемов предварительно накопленных экспериментальных данных. Данное направление активно развивается, так, в обзоре G. Abdelrahman и др. [273] приводится описание решения на основе различных аспектов модели обучаемого, такие как модели забывания и модели обучения студента. Важно отметить, что авторы приводимых в России исследований, например, И.А. Кречетов и В.В. Романенко [97] отмечают, что для использования подобных методов в вузе необходимо накопить большой объем экспериментальных данных, что на практике возможно только для курсов в системах массового онлайн-обучения.

В исследовании S. Cheng и др. [275] предлагается решение по адаптации к новым курсам моделей, обученных на данных другого курса, при этом авторы ставят перед собой задачу осуществить перенос модели - адаптацию - с использованием меньшего объема данных, чем требуется для создания изначальной модели.

Цель индивидуализации обучения состоит в построении и(или) выборе персонализированных образовательных технологий, применение которых позволит студенту улучшить его учебные результаты относительно ожидаемых. В идеальном случае со стороны обучаемого учебный процесс наблюдается как полноценная индивидуальная форма обучения, при которой педагог постоянно доступен для контакта, то есть образовательный процесс происходит в непрерываемой образовательной парадигме. Со стороны педагога, решающего задачу масштабирования учебного процесса с единичных слушателей и малых групп с сохранением индивидуализации обучения, задача лимитируется требованиями к качеству образования, чтобы увеличение числа обучаемых не приводило одновременно к деградации качества обучения в целом [337]. При индивидуализации, подборе подходящей образовательной технологии преподаватель тестирует различные методы обучения, применяемые для преподавания дисциплины, использует свой накопленный опыт и метод «проб и ошибок» для отбора образовательных технологий, применение которых даёт ожидаемый результат в обучении. Наблюдаемое поведение и методы педагога дают уверенность в том, что этот подход может быть успешно автоматизирован для реализации поиска индивидуальной образовательной технологии в условиях масштабирования.

Большинство образовательных технологий появились в результате формирования устойчивых традиций, наблюдений и экспериментов. Психолого-педагогические исследования предполагают подтверждение результатов применения новых технологии на основе экспериментов, в рамках которых выдвигаются гипотезы, которые статистически проверяются [271]. Для получения

статистически-значимых результатов проверки гипотезы необходима длительная проверка на большом количестве студентов и школьников. Такой процесс подходит для исследования эффективности образовательных технологий, но его невозможно применить для индивидуализации обучения конкретного студента. В этом случае селекция персонализированных образовательных технологий должна проводиться по другим алгоритмам. Процесс подбора технологий может быть оптимизирован в случае, если определены критерии применимости образовательной технологии для конкретного студента. На практике преподаватель интуитивно обычно так и действует, относя конкретного студента к тому или иному сформированному классу обучающихся. В случае, если такой метод не дает существенных результатов в краткосрочном периоде наблюдения, то далее педагог может применять метод «проб и ошибок», который также возможно автоматизировать.

В случае гибридной или дистанционной формы обучения с участием на занятиях преподавателя, находясь вне аудитории и не наблюдая реакцию слушателей, даже опытному преподавателю сложно оценить уровень вовлеченности студентов и верифицировать достаточный уровень восприимчивости в удаленной аудитории. Не наблюдая за результатами практической работы студентов, практически невозможно определить, кому из обучающихся требуется помощь педагога, а кто сможет без дополнительной поддержки самостоятельно пройти предложенный контест. В случае, когда педагог использует ЦОП Мирера, ему представляется возможность в интерактивном режиме наблюдать за практической работой студентов. При этом, как было указано выше, преподаватель может индивидуально подключиться к выполнению заданий любого слушателя и совместно с ним исправить ошибку в процессе работы или дать рекомендации по решению.

Фактический уровень остаточных знаний в курсе фиксируется, как правило, на этапе прохождения итогового испытания или осуществляются попытки отследить отставание на промежуточных контрольных. В условиях дистанционных

и смешанных форм обучения также необходимы методы раннего определения отстающих студентов для предотвращения их неуспешного завершения курса. Проблема предсказания провала студентов на итоговых испытаниях известна и исследовалась в контексте массовых открытых онлайн- курсов и учебных платформ [294; 350]. Несмотря на явную зависимость прогнозирования ожидаемого результата студента по освоению курса и вероятности отчисления, важно получать априорную оценку на основе результатов промежуточных результатов испытаний, в том числе с учетом или без индивидуальных особенностей студентов [355].

В исследовании были использованы данные цифрового следа ЦОП Мирера о выполнении студентами домашних заданий, классных работ и промежуточных проверок, а априорные оценки успеваемости студентов были построены с использованием искусственных нейронных сетей [127].

Процесс формирования индивидуализированной образовательной технологии начинается с применения ранее выявленной композиции образовательных технологий, использованной для обучения студента другой, возможно, близкой дисциплине, или набора образовательных технологий, традиционно применяемых при изучении конкретной дисциплины. Образовательные технологии описываются в форме педагогических экспериментов – традиционной формы первичного описания новых технологий. Для получения данных для верификации гипотезы в автоматизированных педагогических экспериментах используется цифровой след обучаемого в ЦОП Мирера. Сформированное множество образовательных технологий разбивается на три класса: *доступные* образовательные технологии, которые еще не прошли верификацию, и *проверяемые* образовательные технологии и *применяемые,* которые прошли проверку. Образовательные технологии, не прошедшие проверку, возвращаются в доступные, снабженные весовым атрибутом, ставящим их в конец очереди для следующей выборки. Применяемые образовательные технологии получают больший вес, что дает возможность их повторно использовать сразу в

следующих курсах при следующем отборе. Схема формирования классов педагогических экспериментов в процессе поиска индивидуализированной образовательной технологии приведена на рисунке, Рисунок 46.

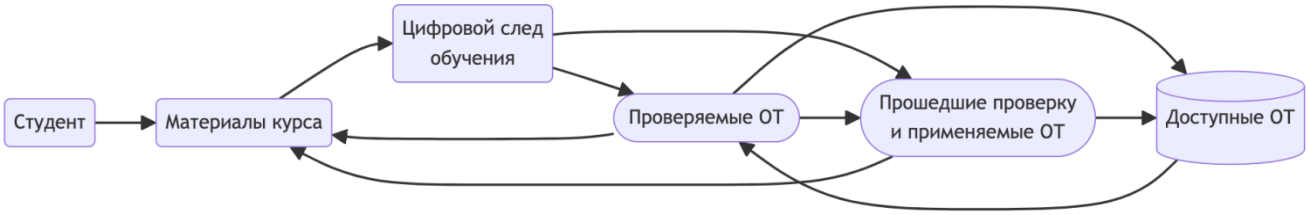


Рисунок 46– Индивидуализированная образовательная технология (ОТ) как композиция педагогических экспериментов

Выше введено определение индивидуализированной образовательной технологии как композиции, которая формально может рассматриваться как подмножество множества педагогических экспериментов. При этом педагогический эксперимент рассматривается в двух аспектах:

1. продолжительный педагогический эксперимент, направленный на проверку гипотезы, как правило, на длительном интервале времени и на большой выборке студентов.
2. краткосрочный педагогический эксперимент на конкретном студенте для проверки применимости для него выбранной образовательной технологии.

В общем случае для педагогического эксперимента описываются оба варианта применения: на этапе масштабной проверки – «гипотеза об эффекте» и при оценке индивидуальной применимости – «гипотеза о применимости».

К преимуществам представления образовательной технологии как композиции педагогических экспериментов относится лучшая интерпретируемость результатов по сравнению с технологиями «черного ящика» или решениями на основе машинного обучения, а также возможность связывать наблюдаемые и полученные в рамках экспериментов результаты, что дает

возможность формировать цифровой след о практике применения и результатах от применения образовательной технологии. Возможная интерференция экспериментов не мешает преподавателю проводить анализ текущего состояния индивидуализированной образовательной технологии, а также корректировать ее.

Формализованное описание автоматизированных экспериментов, в которых участвует студент в рамках модифицированного учебного процесса, позволяет накапливать не только цифровой след в изменяемой области образовательного процесса и результаты, но и алгоритмы проведения таких экспериментов, что дает возможность эффективно проводить аналогичные эксперименты на других курсах. Регистрируемые в процессе экспериментов результаты, соотнесенные с временной шкалой, позволяют прогнозировать и(или) оценивать нагрузку на студента от применения образовательной технологии в широком диапазоне как моментальный снимок достижений, так и как результат длительного применения исследуемой образовательной технологии. Последнее позволяет отбирать технологии, которые в принципе применимы на данном курсе с учетом ограничения по времени прохождения курса и к планируемым зачетным единицам учебной программы.

В целях ресурсной оптимизации процесс исследования применимости новой образовательной технологии состоит из двух этапов, где сначала проверяется эффект от применения технологии в рамках масштабного эксперимента. В случае подтверждения гипотезы эксперимент попадает в пул для использования при формировании индивидуализированной образовательной технологии, что и проверятся на втором этапе, когда прошедшие первый этап эксперимента непосредственно применяются для формирования индивидуализированной образовательной технологии.

На первом этапе при проверке гипотезы об эффекте на практике прибегают к сравнению средних значений успеваемости (распределение успеваемости по группе) у экспериментальных и контрольных групп, при получении значимой разницы считают гипотезу (и реализующую ее технологию) верифицированной, реже оценивают влияние на другие аспекты обучения. На втором этапе

рассматривается индивидуальный результат в определенном блоке материала или задании.

Современные образовательные технологии основаны на анализе данных, получаемых в результате взаимодействия студента с учебной системой – цифрового следа обучения [367]. В результате образовательная технология, разработанная с использованием собранных при обучении на одном курсе-доноре данных, как правило, не может быть перенесена без изменений для автоматизации обучения на другом курсе − реципиенте. Здесь под переносом или адаптацией технологии понимается изменение дисциплины, учебного заведения и даже потока курса в одном учебном заведении. Причиной сложности переноса является то, что образовательная технология оказывается связана с учебным курсом-донором, структура и содержание которого отображается в цифровой след обучения, и по этой причине технология не может быть применена на курс-реципиент без адаптации. Без адаптации технологии применение её в других условиях не позволяет получить сопоставимые с полученными на исходном курсе-доноре результаты. Существующие методы допускают перенос технологии с одного курса-донора на другой курс-реципиент путем адаптации решения с использованием новых данных, накопленных по результатам обучения на курсе-реципиенте [273]. Проблема переноса решения характерна не только для технологий на основе глубокого обучения, но и также проявляется на технологиях, разработанных на основе анализа данных.

Другой проблемой является недостаточный объём данных, который можно накопить при преподавании на старших курсах вузов или на курсах повышения квалификации, поскольку в силу специализации студентов и малой численности студентов в группах охват на таких курсах незначительный, а следовательно, накопление больших объемов данных для анализа требует длительного периода времени, который будет превышать время устаревания данных.

Надо отметить еще одну особенность педагогических экспериментов – интерференцию экспериментальных воздействий. Участники педагогических

экспериментов обладают памятью и обучаемостью, по этой причине дважды повторить один эксперимент с одинаковыми параметрами в рамках обучения на курсе невозможно. В общем случае: каждый раз в эксперименте будут участвовать разные студенты. По этой причине в эксперименте участвует большое количество студентов, разделенных на несколько экспериментальных и контрольных групп, разница в результатах которых используется для верификации гипотезы. Соответственно, на индивидуальном уровне повторить какой-то эксперимент дважды невозможно, однако если сделать предположение, что образовательная технология отделена от курса и не связана непосредственно с его структурой и содержанием, то образовательная технология сводится к приёму, использование которого может быть перенесено с одной дисциплины на другую без потери эффективности от применения [261]. То есть это специфичная для студента композиция приёмов обучения, которая может применяться для обучения различным дисциплинам, а не только для обучения на одном конкретном курсе. Общая схема повторного использования индивидуализированных образовательных технологий приведена на рисунке, Рисунок 47, где изображена взаимосвязь учебных курсов, цифрового следа обучения посредством автоматизированной системы обучения (АСО) и индивидуализированной образовательной технологии, которая может одновременно применяться к нескольким курсам в разных видах с учетом специфики курса. То есть не является общей, но при этом применимой для конкретного студента. В этом случае результаты обучения на одном курсе влияют на технологии обучения на другом курсе.

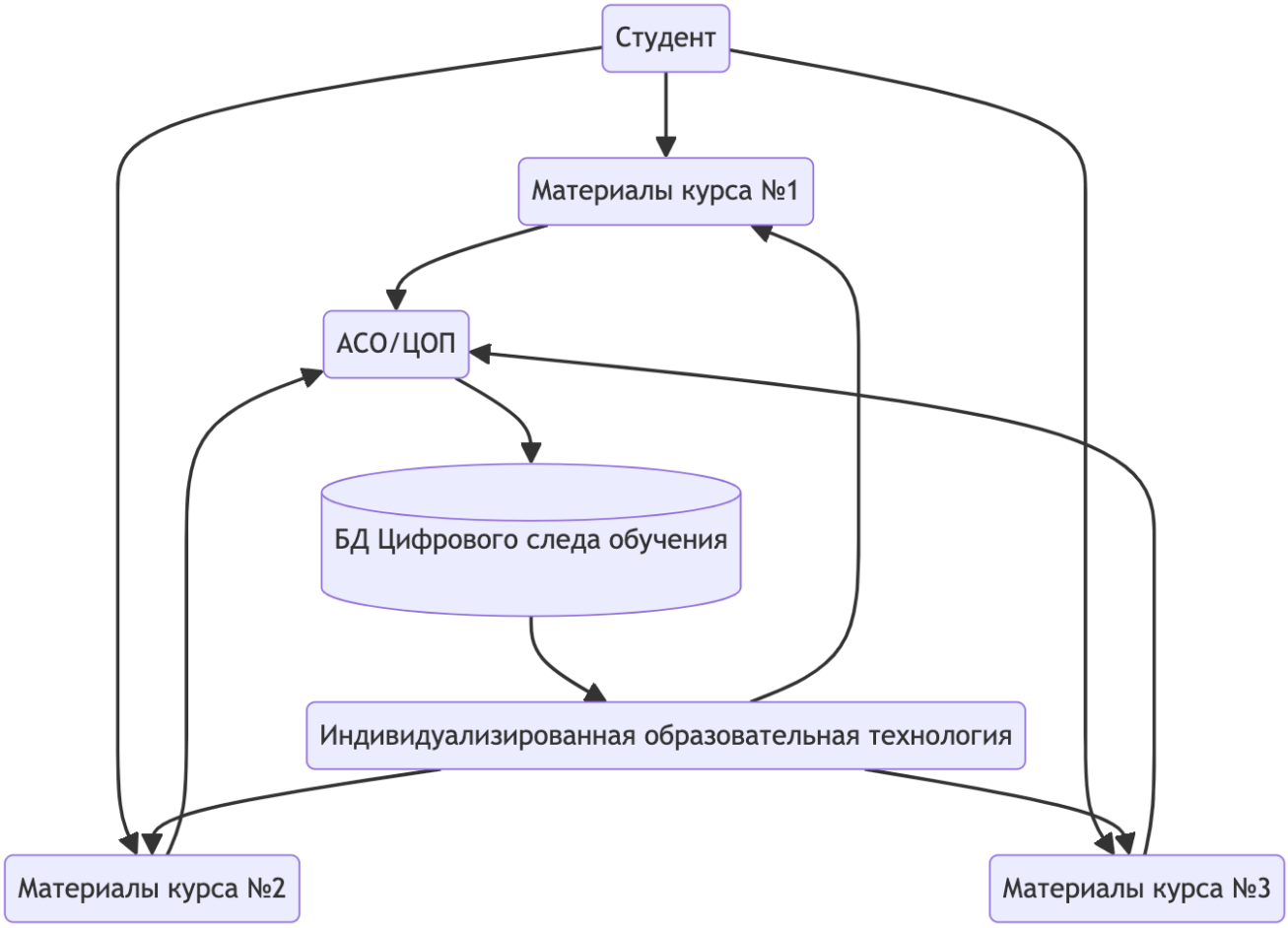


Рисунок 47 – Повторное использование и формирование индивидуализированной образовательной технологии через цифровой след обучения при обучении на нескольких курсах

Сейчас принято считать автоматизацию обучения средством снятия нагрузки с преподавателя, в новом качестве это является автоматизированным инструментом по поиску индивидуальной образовательной технологии обучения конкретного студента и, как следствие, освобождает от излишней нагрузки преподавателя. Из предположения, что образовательная технология отделима от конкретного курса и может применяться на других курсах, следует, что в процессе обучения студента в вузе на младших курсах выполняется поиск оптимальной для студента образовательной технологии, после чего студент обучается в рамках корректируемой индивидуализированной образовательной технологии. На

старших курсах образовательные технологии являются общеприменимыми вне зависимости от студента, поскольку объёма данных для исследования недостаточно из-за специализации студентов и обучения в малых группах.

Определяющим фактором при разработке современных технологий на основе данных является возможность быстро собирать данные для разработки и последующей проверки эффективности решения, после разработки возникает проблема распространения полученных решений с учетом проблем с переносом решений. Для реализации подхода автоматизации персонализированных образовательных технологий необходимо использовать интеграционную платформу или функцию интеграционной платформы в составе цифровой образовательной платформы, к функции которой относится координация экспериментов на этапе накопления данных и проверки эффективности.

Функции интеграционной платформы не ограничиваются координацией проводимых экспериментов, она также отвечает за поддержку преподавателей в процессе участия в эксперименте и при внедрении новых образовательных технологий, поскольку автоматически обеспечивает контроль результатов и обнаруживает аномалии в поведении учебных групп и отдельных студентов.

Помимо увеличения охвата, интеграционная платформа также является единой точкой входа для исследователей для ознакомления с текущими результатами исследований по выбранным технологиям, их повторным использованием и развитием, а наличие формального описания эксперимента позволяет независимо проверять эксперименты, где среда распространения новых технологий ускоряет их появление в учебных заведениях.

Представление индивидуализированной образовательной технологии как композиции отдельных образовательных технологий является развитием идеи адаптивного обучения, в рамках которой исследователи стремятся найти универсальную параметрическую модель обучения, применение которой заключается в поиске оптимальных параметров модели для конкретного студента. В исследовании констатируется, что применение параметрических моделей

позволяет улучшить результаты только части студентов, а не всех участников эксперимента. Предлагается изначально поставить максимальную цель – охватить всех студентов, но при этом допускается применение композиции из нескольких образовательных технологий, что позволит охватывать различными образовательными технологиями всех студентов, а не фокусироваться лишь на подгруппе студентов, которым даёт преимущество применение конкретной образовательной технологии. Предлагаемая интерпретация ближе к подходам исследователей и разработчику системы обучения GermanTutor [330], которые в своей системе используют сразу несколько различных образовательных технологий для достижения максимального охвата аудитории и повышения результативности обучения на платформе.

Высказанная идея не противоречит текущим подходам к адаптивному обучению, основанным на данных, таким как DKT [319] и его разновидностей, которые могут быть использованы как отдельные образовательные технологии. Однако при этом важно учитывать ограниченную переносимость таких решений (AdaptDKT) [275], которая, следовательно, приведет к ограниченной применимости полученной индивидуализированной образовательной технологии для использования при обучении новым дисциплинам.

Предложение не противоречит сложившейся модели построения учебных систем, состоящей из модели предметной области, модели обучения и модели студента или школьника [21]. В этом случае предлагаемый подход дает возможность сформировать индивидуализированную модель обучения как композицию образовательных технологий, подходящих для конкретного студента. Предложенный подход возвращает к первоначальному пониманию модели обучения как универсального решения, которое со временем оказалось зависимым от модели предметной области (структуры и содержания учебного курса).

Необходимость поддержки преподавателя в процессе внедрения новых технологий, обозначенная авторами как функция интеграционной платформы, также подтверждается в исследовании С.В. Горбатова и Е.А. Красновой [44],

согласно которому педагогам надо дать возможность сосредоточиться на педагогической поддержке, убрав из работы рутинные проверки, которые могут быть выполнены учебной системой. Поскольку исследования сложно отнести к рутинной работе, предлагается часть контроля проведения эксперимента также переложить на систему, дав возможность получать доступ к данным организаторам эксперимента, что позволяет проводить распределённые масштабные эксперименты, которые находятся под контролем учебной системы, а не только преподавателей-участников эксперимента.

Полученные результаты частично реализованы в ЦОП Мирера, например, функциональность «адаптивные траектории», которая использует только результаты студента и не связана со спецификой преподаваемой дисциплины. Однако остается проблемой автоматизация проверки знаний, отсутствие учебных материалов высокой гранулярности и вариативности, которые нужны для создания адаптивных курсов, а также общая методология создания этих материалов, что является препятствием для широкого внедрения методов индивидуализации [136].

Выводы главы 4

Методология формирования алгоритмического мышления включает в себя получение знаний, освоение определенных навыков, формирование умений при прохождении уровней образования: уровень пропедевтики – знакомство дошкольников и младших школьников с основами алгоритмизации, уровень генерализации знаний — обучение школьников программированию, и уровень профессионализации (профильное обучение) – высшее образование, когда студенты изучают информатику и программирование в разрезе специальностей и различных предметов. Все три ступени объединены ориентированными на возраст и уровень компетенции методическими системами обучения, позволяющих выстроить бесшовный непрерывный образовательный процесс и использующих вариативные, ориентированные на возраст, методики и формы обучения, базирующиеся на средствах обучения, включающих практику освоения

программирования на пиктографическом учебном языке для пропедевтики (встроенного в цифровую образовательную среду ЦОС ПиктоМир). Для уровней генерализации и профессионализации (профильный уровень) переход к текстовым, учебным и производственным языкам программирования с использованием ступенчатой методики на базе средств обучения – тройки цифровых образовательных сред, включающих пиктографическую, блочную и текстовую среду программирования (ЦОС ПиктоМир, ЦОС ПиктоМир-К, ЦОС КуМир). Процесс перехода к обучению производственным программным языкам в рамках методической системы обучения не ставит целью приобрести конкретные навыки владения той или иной системой.

Содержание обучения составляет универсальный, доступный любому возрасту обширный набор заданий для изучения основ информатики и программирования, который организует практическую подготовку дошкольников, школьников и студентов с использованием, как средств обучения, ИКТ-насыщенных цифровых образовательных сред с автоматизированной верификацией уровня освоения требуемых компетенций. Для интенсификации образовательного процесса и внедрения методики обучения, использующей элементы индивидуализации, которые основываются на элементах искусственного интеллекта, необходимо в качестве средства обучения использовать интеграционную цифровую образовательную платформу (ЦОП Мирера), где выполнение заданий обучаемыми оценивается не только по результатам испытаний, но и по процессу выполнения задач, контролирующему из омниканальной цифровой образовательной платформы, использующей искусственные нейронные сети.

Использование, как средств обучения, профильной методической системы обучения ЦОС ПиктоМир совместно с цифровой образовательной платформой Мирера на начальном курсе алгоритмики и программирования позволило увеличить объем практики студентов педагогических вузов с единиц до нескольких сотен за семестр. Освоенная практика будущих преподавателей курса

информатики и ИКТ позволила интенсифицировать и индивидуализировать подготовку молодых специалистов для современной цифровой реальности.

На основании проведенных исследований сформирован подход построения индивидуализированной образовательной технологии обучаемого, как композиции педагогических экспериментов, которые оказывают положительный эффект на результаты конкретного слушателя. Предложенный метод обладает лучшей интерпретируемостью, так как в его основе лежат интерпретируемые педагогические эксперименты. Автоматизация процедур отбора образовательных технологий для каждого студента проводится за счет использования метода «проб и ошибок» и его оптимизированных версий.

Для координации проведения экспериментов в области автоматизации обучения, накопления данных, для разработки и оценки эффективности новых технологий создается расширение функциональности средства обучения ЦОП Мирера. Важно отметить, что предложенные методы автоматического построения индивидуализации образования являются мультипредметными.

ГЛАВА 5. ОРГАНИЗАЦИЯ И РЕЗУЛЬТАТЫ ПЕДАГОГИЧЕСКОГО ЭКСПЕРИМЕНТА

5.1 Достижения дошкольников и младших школьников при формировании основ алгоритмического мышления в рамках масштабного эксперимента в образовательных организациях

На основе методик понижения возраста первичного знакомства с основами программирования, представленных в главе 3, используя разработанную авторскую ИКТ-насыщенную предметно-цифровую образовательную среду ПиктоМир, был проведен ряд педагогических экспериментов, как практическое обоснование сформулированных в работе гипотез.

В федеральном государственном учреждении «Федеральный научный центр Научно-исследовательский институт системных исследований Российской академии наук» (ФГУ ФНЦ НИИСИ РАН) в рамках реализации Национального проекта «Образование» с 1 августа 2020 года развернута площадка по теме «Апробация и внедрение основ алгоритмизации и программирования для дошкольников и младших школьников в цифровой образовательной среде ПиктоМир» по направлению дошкольного, начального общего и дополнительного образования. Данная сетевая площадка развернута в 2020 году Академией Наук России совместно с Самарским Институтом Образовательных Технологий и по состоянию на середину 2024 года охватывает около 800 детских садов и начальных школ России из 69 субъектов России. Список образовательных организаций, присоединившихся к площадке «ПиктоМир», приведен в Приложении 1.

Для оценки эффективности внедряемых методик формирования основ алгоритмического мышления у детей, начиная с четвертого года жизни, в рамках проводимых педагогических экспериментов используются результаты достижений детей, рассматриваемые при участии в парциальных программах и программах дополнительного образования протяженностью от одного года до трех. Как указано

в ФОП ДОО [333], освоение основной образовательной программы и вариативной части программ не сопровождается проведением промежуточных аттестаций и итоговой аттестации обучающихся, а основным методом педагогической диагностики является наблюдение:

«16.8. Анализ продуктов детской деятельности может осуществляться на основе изучения материалов портфолио ребенка (рисунков, работ по аппликации, фотографий работ по лепке, построек, поделок и другого). Полученные в процессе анализа качественные характеристики существенно дополнят результаты наблюдения за продуктивной деятельностью детей (изобразительной, конструктивной, музыкальной и другой деятельностью).

16.9. Педагогическая диагностика завершается анализом полученных данных, на основе которых педагог выстраивает взаимодействие с детьми, организует РППС, мотивирующую активную творческую деятельность обучающихся, составляет индивидуальные образовательные маршруты освоения образовательной программы, осознанно и целенаправленно проектирует образовательный процесс» [333].

Вся деятельность педагога-воспитателя, в том числе и в рамках вариативной части образовательной программы, соответствует нормативной базе [163; 243; 164]. Площадка «Апробация и внедрение основ алгоритмизации и программирования для дошкольников и младших школьников в цифровой образовательной среде ПиктоМир» реализуется в образовательных организациях в рамках вариативной части программы с учетом специфики этнонациональных, социокультурных, региональных условий и сложившихся форм организации работы с детьми в субъектах федерации и конкретных образовательных организациях [206; 105; 139; 172]. При этом воспитатель использует различные формы и вариативность подачи материала на площадке, вплоть до различного именования занятий и включения в программу элементов смежных предметов.

В качестве примера исследований приводятся выборочные результаты 2022-2023 учебного года на основании отчетов о проделанной работе по внедрению и

апробации курса «Алгоритмика для дошкольников» в 265 организациях дошкольного образования. В указанном учебном году согласно отчетам программой было охвачено 12 тысяч 319 детей в 718 группах, включая 2031 ребенка с ОВЗ.

Занятия проходили в каждой организации по внутреннему расписанию: 1 или 2 раза в неделю по 25 минут. Из них 50% занятий было проведено с использованием Робототехнического набора (предметной среды ЦОС ПиктоМир), что предусмотрено программами и календарно-тематическим планом, так называемый допланшетный период в средней, старшей и подготовительной группе. Бескомпьютерная часть занятия по санитарным правилам проходит без использования электронных средств обучения [216].

Число детей, погруженных в изучение азов алгоритмизации, можно поделить по возрастам:

- средняя группа (4-5 лет) – 3654 ребенка (213 подгрупп);
- старшая группа (5-6 лет) – 4872 ребенка (284 подгруппы);
- подготовительная группа (6-7 лет) – 3793 ребенка (221 подгруппа).

В образовательных организациях было занято 840 педагогов. Все воспитатели прошли дополнительное профессиональное обучение, из них продвинутый уровень, освоившие парциальную программу «По алгоритмическим дорожкам» в рамках второго курса дополнительного профессионального обучения, получили 334 педагога.

В 2022-2023 году по дополнительной программе «Алгоритмика. Старт 4+» работало 134 образовательных организаций, по основной программе «По алгоритмическим дорожкам» работало 102 ДО. В 29 образовательных организациях работали в разных подгруппах по двум программам. Педагоги, предоставившие отчеты, принимают активное участие в разработке методических материалов: настольных игр, пособий, лэпбуков и ПиктоБоксов. Эти материалы помогают в игровой форме познакомить детей с основными понятиями алгоритмики.

Из основных результатов педагоги выделили:

- личностные результаты:
  - осмысление мотивов своих действий при выполнении заданий;
  - развитие любознательности, сообразительности при выполнении разнообразных заданий проблемного и эвристического характера;
  - развитие внимательности, настойчивости, целеустремленности, умения преодолевать трудности;
  - развитие самостоятельности суждений, независимости и нестандартности мышления;
  - воспитание чувства справедливости, ответственности;
- Метапредметные результаты [6]:
  - регулятивные универсальные учебные действия:
    - планировать последовательность шагов алгоритма для достижения цели;
    - формировать умения ставить цель;
    - создание творческой работы, планировать достижение этой цели;
    - осуществлять итоговый и пошаговый контроль по результату;
    - различать способ и результат действия;
    - вносить коррективы в действия в случае расхождения результата решения задачи на основе ее оценки и учета характера сделанных ошибок;
    - оценивать получающийся творческий продукт и соотносить его с изначальным замыслом, выполнять по необходимости коррекции этого продукта, либо замысла;
  - познавательные универсальные учебные действия:
    - ориентироваться на разнообразие способов решения задач;
    - проводить сравнение по заданным критериям;

        - строить логические рассуждения в форме связи простых суждений об объекте;
        - устанавливать аналогии, причинно-следственные связи;
    - коммуникативные универсальные учебные действия:
        - аргументировать свою точку зрения на выбор оснований и критериев при выделении признаков, сравнении и классификации объектов;
        - выслушивать собеседника и вести диалог.

Таким образом, по результатам наблюдения можно утверждать, что у детей сформированы основы алгоритмического мышления, которые дают содержательные личностные результаты. По завершении курса у детей сформированы начальные представления о программировании и алгоритмизации; дети познакомились с основными алгоритмическими понятиями и определениями; дети научились "прочитывать" схемы, по схемам и инструкциям строить модели. Результаты мониторинга показывают, что у детей развились творческие способности, умение анализировать, сравнивать, сопоставлять, логически мыслить, решать логические и алгоритмические задачи. Дети овладели основами алгоритмики, умеют запускать программы на планшете для роботов-исполнителей, познакомились с основными понятиями, командами.

Среди результатов занятий «Алгоритмика для дошкольников» с использованием ЦОС ПиктоМир можно выделить следующее:

- высокий познавательный интерес детей (78% воспитанников посещают занятия с удовольствием, активны во время работы);
- у детей улучшились навыки ориентировки в пространстве (60% воспитанников имеют высокий уровень);
- 95% детей старшей группы освоили основной набор команд: обозначения, именования, действия команд роботов, используемых в ЦОС ПиктоМир, могут их применять в процессе игры, а также переносить игры и результаты в другие виды деятельности.

Дети, занимающиеся в планшетном периоде, самостоятельно могут запускать игры в приложении среды ПиктоМир на планшете, компьютере, могут играть и самообучаться. Дети познакомились с несколькими вариантами записи решения одной программы, например, с короткой и длинной программой.

Один из особенных результатов – это 100% удовлетворенность родителей детей. Родители оценивают работу педагогов с детьми и родителями, доступность дополнительной деятельности, коллаборации родителей с детьми в домашней обстановке.

При этом воспитатели получают полноценную методическую поддержку от авторов курса, отмечают ее важность: чат, в котором можно задать вопрос авторам программы, вебинары по темам, курсы повышения квалификации, методические марафоны, выездные школы, на которых педагоги делятся опытом.

Ниже приведен мониторинг некоторых выборочных результатов освоения программы «Алгоритмика для дошкольников» в разных курсах.

В годовом курсе для детей 6+ «Алгоритмика (1-30)» в дошкольной образовательной организации проводят занятия для детей подготовительной группы, в котором предусмотрен мониторинг полученных навыков.

Единица уровня ЦОС ПиктоМир, называемая мир «Алгоритмика», - это набор из 30 компьютерных игр для курса «Алгоритмика для дошколят», по одной игре для каждого из 30 компьютерных занятий, поддерживаемый подробными методическим пособием [107]. Пособие описывает 30 занятий, каждое из которых состоит из двух частей с перерывом между ними: бескомпьютерной и компьютерной [216]. Часто в детском саду проводят каждую из описанных в методичке частей занятия отдельно, тем самым 30-недельный курс включает по два занятия на каждой неделе: бескомпьютерное и компьютерное, суммарно 60 занятий. Согласно программе курса, в декабре каждого текущего учебного года проводится промежуточный этап мониторинга результатов занятий. Итоговый мониторинг освоения программы проводится в конце учебного года. Основной целью мониторинговых занятий является анализ навыка придумывания линейных

алгоритмов и составления по алгоритмам программ в ЦОС ПиктоМир. Ниже приведены примеры нескольких мониторингов:

- МБДОУ №20 «Югорка» г. Сургута;
- Частный детский сад «Феникс», г. Москва;
- МБДОУ детский сад №49 г. Красногорск Московской области.

В промежуточном мониторинге результатов освоения курса «Алгоритмика для дошколят» в МБДОУ №20 «Югорка» г. Сургута приняли участие две группы воспитанников старшего дошкольного возраста (6-7 лет) в количестве 50 человек.

Для диагностики результатов обучения детям было предложено выполнить задания, предусмотренные в игре 9 «Соревнование» ЦОС «ПиктоМир» [107]. В отличие от регулярных занятий, где работа на планшетах продолжается не более 15 минут, на диагностическом занятии на работу с планшетами было отведено 30 минут с 5-минутным физкультурным перерывом в соответствии с медицинскими рекомендациями [216]. Автоматический сбор результатов выполнения заданий группой дошкольников не использовался, воспитатель наблюдал на экране планшетов персональную деятельность детей и результаты игры в ЦОС ПиктоМир, как кумулятивный результат работы ребенка над заданиями.

Девять заданий игры «Соревнование» посвящены составлению содержательных линейных алгоритмов (от 9 до 17 команд каждый) с использованием двух разных исполнителей-роботов Вертун и Двигун ЦОС ПиктоМир. В заданиях 1, 2, 5-9 ребенку открыт доступ к «копилке команд», что позволяет при составлении алгоритма сразу отслеживать, насколько правильно подобраны команды в режиме непосредственного управления, а в заданиях 3 и 4 доступ к «копилке» отсутствует, что предполагает предварительное «умозрительное» составление ребенком всего алгоритма или его фрагмента и только после этого проверку и отладку алгоритма.

Результаты мониторинга показали, что 100% детей в целом справились со всеми предложенными заданиями. При этом 24 ребенка (это составляет 48%) справились с заданиями без какой-либо посторонней помощи. Остальным ребятам

потребовалась небольшая подсказка со стороны педагога для выполнения одного-двух заданий. Наибольшие затруднения были вызваны заданиями 2 и 3, в которых не была доступна «копилка». Отмечено, что после успешного выполнения заданий 2 и 3 (в которых «копилка» недоступна) из 7 ребят (14% воспитанников) не воспользовались копилкой и при выполнении следующих заданий 4-9, хотя при выполнении этих заданий «копилка» была доступна.

Наибольшие затруднения возникли у детей, которые по тем или иным причинам пропустили часть занятий. Из 50 детей по тем или иным причинам 4 ребенка посещали занятия нерегулярно. В общей сложности ими было пропущено от 5 до 10 занятий. Большая часть пропусков пришлась на цикл занятий, посвященных работе с реальным роботом Ползуном. Тем не менее ребята освоили линейные алгоритмы. Этот вывод подтверждается тем, что эти дети, хотя и с некоторой помощью педагога, смогли справиться с предложенными им заданиями игры «Соревнование».

Как и предполагалось, участие в данном соревновании-занятии оказалось психологически комфортным для всех без исключения детей и все дети остались довольны результатами своего участия в «Соревновании». По итогам промежуточного этапа мониторинга можно утверждать, что за промежуток 8 недельных занятий, как и предусматривается утвержденной программой курса, линейные алгоритмы были освоены всеми детьми на 100%.

Далее приведены результаты промежуточного мониторинга освоения курса «Алгоритмика для дошколят» в детском саду «Феникс».

Группа образовательной организации состояла из 10 человек. Занятия посещались детьми нерегулярно. В группе 4 человека занимались более одного года. Работа такими детьми строилась по схеме: «выполнил своё задание, проверь у соседа, ответь на вопрос и помоги».

В группе два ребенка не являются носителями русского языка, но занятия позволили лучше освоить язык страны пребывания, включая необходимые основные понятия программирования. За 3 месяца занятий дети выучили слова и

овладели терминами: программа, робот, команды робота, исполнитель, непрерывное выполнение, пошаговое, пуск, включая бытовые слова, например: кнопка, стрелка и т.п.

Среди группы один ребенок, посетивший всего 8 занятий, сумел научиться пользоваться ЦОС ПиктоМир на планшете и самостоятельно составлять алгоритмы до 5 команд без использования копилки.

На заключительном занятии дети играли в игре «Соревнование» (9 заданий) в режиме ограниченного времени, то есть им необходимо было самостоятельно решить максимальное число задач за 15 минут, включая сложные задания, в которых отключена копилка. В соревновании участвовало 6 человек.

Дети (имена в исследовании сокращены до 1 буквы), занимающиеся второй год выполнили Н(6 лет 8 месяцев) – 6 заданий, М(6 лет 9 месяцев) – 8, С(7 дет 1 месяц) -7. Двое детей возраста 5 лет и 6 месяцев выполнили самостоятельно 4 задания, включая сложные задания (одна задача - 12 команд, другая 17 команд) с отключенной копилкой.

Ребёнок (6 лет 5 месяцев), для которого русский язык не являлся родным, выполнил 2 задания. Для решения первых двух заданий можно было использовать копилку, но ребёнок отказался и шаг за шагом, проверяя составленную программу по частям, отладил программу.

Диагностику-мониторинг в МБДОУ детский сад №49 г. Красногорск Московской области развития алгоритмических умений проводили с двумя подготовительными группами: экспериментальной (посещали занятия «Алгоритмика») и контрольной (не посещали). В эти группы вошли 60 детей возраста 6-7 лет.

Занятия в экспериментальной группе проходили два раза в неделю. За годовой курс дети освоили такие важные структуры программирования как: линейный алгоритм, циклы с повторителями, программа с подпрограммой. Пробовали решение задач на кооперативное программирование.

Диагностика проводилась с помощью диагностической методики Кубики Кооса (англ. Kohs Block Design Test) – специальной методики для диагностирования наглядно-действенного интеллекта или невербального теста интеллекта [249].

Тест «Кубики Кооса» благодаря своей сложной и многогранной природе заданий оказался наиболее информативным инструментом для изучения невербального интеллекта и стал рассматриваться как самостоятельный метод его измерения. Невербальный интеллект тесно связан с особенностями восприятия, моторики, зрительно-моторной координации, а также с пространственными представлениями и конструктивными и эвристическими способностями. Тест «Кубики Кооса», требующий проявления всех этих качеств, показал высокую чувствительность к выявлению невербальной одаренности.

Он широко применяется в профориентации и профотборе для оценки способностей к практической деятельности и интеллектуальных предпосылок технических навыков в образовании, а также в медицине для диагностики различных нейропсихологических и психопатологических симптомов. В тесте Кооса ребенку выдаются фигурки и карточка с заданием. Малыш должен сложить кубики так, чтобы рисунок на верхней части фигур полностью совпадал с рисунком (узором) на карточке. Необходимый принцип тестирования: задания должны быть упорядочены в порядке возрастающей сложности.

Для этого очень важно следовать правилам:

- сначала ребенок составляет рисунок только из одноцветных граней кубиков,
- затем добавляют пару-тройку кубиков с двухцветными гранями,
- после чего убирают простые кубики и оставляют только двухцветные,
- с каждым следующим упражнением должно быть задействовано все больше фигур.

Тестирование проходит либо в течение 15-20 минут, либо завершается после того, как малыш неудачно выполнил уже пятое по счету задание. При расшифровке

результатов учитывается время, за которое тестируемый смог правильно выложить узор, Таблица 7.

Из рисунка, Рисунок 48, видно, что диагностические показатели высокого уровня невербального интеллекта экспериментальной группы - детей, посещающих кружок по алгоритмике, превышают показатели контрольной группы. Значит, используемая программа «Алгоритмика для дошкольников» с использованием ЦОС ПиктоМир способствовала более полному раскрытию интеллектуальных способностей детей.

Таблица 7 – Результаты диагностики в МБДОУ детский сад №49.

Группа	Начало года			Конец года		
	Уровень	Человек	Процент	Уровень	Человек	Процент
Экспериментальная	Высокий	2	5	Высокий	13	42
	Средний	25	87	Средний	16	55
	Низкий	3	8	Низкий	1	3
	Высокий	2	5	Высокий	5	17
	Средний	25	87	Средний	23	78

Контрольная	Низкий	3	8	Низкий	2	5

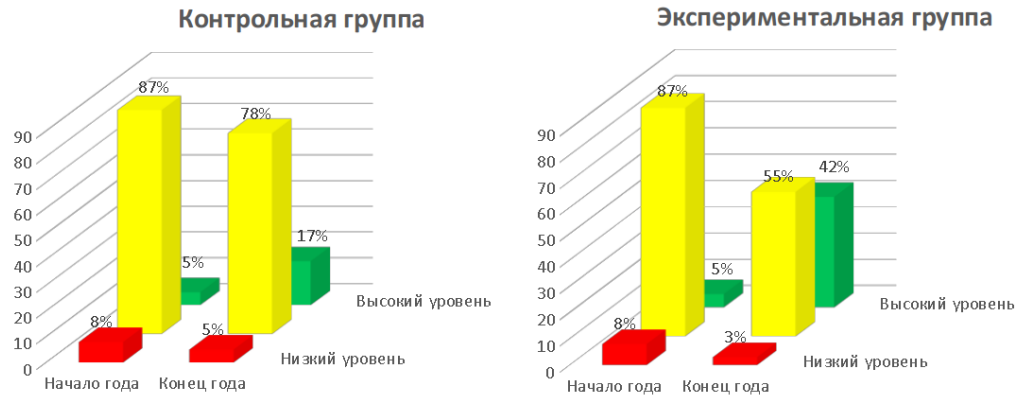


Рисунок 48 – Диаграмма результатов диагностики в МБДОУ детский сад №49

В качестве выборочного примера мониторинга освоения основ алгоритмизации в рамках 3-х летний курс дополнительной образовательной программы «Алгоритмика. Старт. 4+» приведены результаты испытаний в БДОУ Вытегорского муниципального района, детский сад «Гармония».

Диагностика проводилась с детьми индивидуально в форме игры. Методы диагностики относятся к группе эмпирических. Педагог вел наблюдение за процессом выполнения и фиксировал информацию в протоколе наблюдения.

Каждое предложенное ребёнку задание (из 3-х) имеет свою оценочную шкалу. После выполнения ребёнком всех заданий выводится среднее значение. Задания рассчитаны на обучающихся 4 – 7 лет. Педагог приглашает ребёнка поиграть и по очереди предъявляет ему индивидуально стимульный материал. Анализируемые показатели:

- характер деятельности (целенаправленность, хаотичность и т. п.);
- доступность выполнения задания;
- характер ошибок при выделении признаков;
- характер рассуждений ребенка и уровень обобщающих операций;
- объем и характер необходимой помощи со стороны взрослого.

Тип заданий:

1. Методика «Что лишнее?»;
2. Методика «Последовательность событий» (предложена Н.А. Бернштейном) [217];
3. Методика «собери по образцу».

В начале года диагностика проводилась с использованием карточек, знакомых детям. Например, Рисунок 49.

Рисунок 49 – Задание 1.

Ребенку предъявляются изображения четырех предметов, три из которых можно объединить обобщающим словом, а четвертый предмет по отношению к ним окажется «лишним». «Посмотри, здесь нарисованы предметы (объекты). В каждом ряду есть «лишний», он не подходит к трем другим. Какой предмет (объект) «лишний» и почему?».

В задании 2 ребенку показывают беспорядочно разложенные картинки, которые он должен разложить по порядку, Рисунок 50.

Рисунок 50 – Пример задания 50

В задании 3 предлагалось собрать по схеме из конструктора фигуру.

Результаты оценивались следующим образом:

- Высокий уровень – ребенок правильно выполнил всё задание;
- Средний уровень – ребенок правильно справляется с частью задания;
- Низкий уровень – ребенок не выполнил предложенное задание.

Диагностика проводилась на одной подгруппе, состоящей из 8 детей старшей группы. Для сравнения приводится результаты в начале года и в конце, Таблица 8, Таблица 9.

Таблица 8 – Результаты диагностики в начале года до прохождения дополнительной образовательной программы «Алгоритмика»

№ п/п ребенка	Задание 1	Задание 2	Задание 3	Общий итог
1	в	в	с	в
2	в	в	в	в
3	с	с	с	с
4	с	в	с	с
5	с	в	в	в
6	с	с	с	с
7	в	с	с	с
8	с	с	с	с
Результаты высокого уровня	3 (37,5%)	4(50%)	2(25%)	3(37,5%)

Таблица 9 – В конце года обучения провели диагностику на тех же заданиях

№ п/п ребенка	Задание 1	Задание 2	Задание 3	Общий итог
1	в	в	в	в
2	в	в	в	в
3	в	в	с	в
4	в	в	с	в
5	в	в	в	в
6	в	с	в	в
7	в	в	в	в
8	в	в	с	в
Результаты высокого уровень	8 (100 %)	7 (87,5%)	5 (62,5%)	8 (100%)

Следуя этим трем методикам, были подготовлены карточки с заданиями по программе «Алгоритмика для дошкольников» с использованием ЦОС ПиктоМир. Например, найди лишнюю команду, подбери правильную последовательность картинок с результатом выполнения программы Вертуна, собери по образцу коврик-карту для робота Ползуна. Например, к заданию 2 «Разложи по порядку» разработаны картинки, на которых нужно установить порядок рисования роботов. Это позволяет также проверить, насколько ребенок знаком с образами исполнителей ПиктоМир, Рисунок 51.

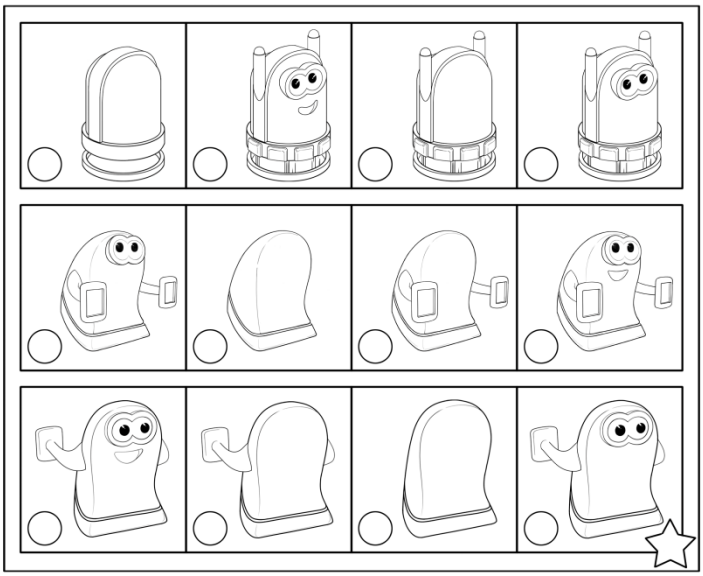


Рисунок 51 – Пример задания 2 с роботами ЦОС ПиктоМир.

В конце года была проведена еще одна диагностика, с использованием новых карточек и заданий. Результаты отражены в таблице, Таблица 10.

Таблица 10 – В конце года обучения провели диагностику на новых заданиях

№ п/п	Задание 1	Задание 2	Задание 3	Общий итог
1	в	в	в	в
2	в	в	в	в
3	в	с	в	в
4	в	в	с	в
5	в	в	в	в
6	в	с	в	в
7	в	в	в	в
8	в	в	с	в
Результаты высокого уровень	8 (100%)	6 (75%)	6 (75%)	8 (100%)

Заметно, что те дети, у которых в начале года результаты были средние, смогли их улучшить после занятий «Алгоритмикой». Все дети показали высокие результаты Таблица 11.

Таблица 11 – Результаты показывают, насколько занятия «Алгоритмикой» повысили уровень детей

Задания	Уровень	Начало года		Конец года			
		Кол-во детей	Процент	Кол-во детей	Процент	Кол-во детей	Процент
Задание 1	Высокий	3	37,5	8	100	8	100
	Средний	5	62,5				
Задание 2	Высокий	4	50	7	87,5	6	75
	Средний	4	50	1	12,5	2	25
Задание 3	Высокий	2	25	5	62,5	6	75
	Средний	6	75	3	37,5	2	25
Общий результат теста	Высокий	3	37,5	8	100	8	100
	Средний	5	62,5				

2-х-летний курс основной программы «По алгоритмическим дорожкам» в ГАДОУ детский сад №15 Колпинского района Санкт-Петербурга реализовывался в 2023-2024 учебном году с воспитанниками 2-го года обучения, подготовительной

к школе группы (возраст 6-7 лет) во второй половине дня в объёме 72 часа (режим занятий 2 часа в неделю, продолжительность занятия согласно [216] – 30 минут).

Исследования проводились с использованием диагностического инструментария «ПиктоМир: программируем будущее», 2023г (авторы: Ю.А. Помещенко, заместитель заведующего по УВР, А.В. Дроздецкая, воспитатель).

К основным задачам диагностики относятся:

1. Диагностика алгоритмических умений воспитанников, умений ориентироваться и выполнять учебные задачи в цифровой образовательной среде ПиктоМир;
2. Диагностика сформированности «гибких навыков» по критериям: выделение основной мысли, выявление противоречий, выявление нехватки информации, формулировка собственного вывода.

В качестве основного метода педагогической диагностики был использован метод получения ответов детей (в процессе беседы по заранее подготовленным вопросам и настольных игр). Направлен данный метод на получение конкретных ответов на заданные воспитателем вопросы.

Для фиксации уровня сформированности алгоритмических навыков и «гибких навыков», в том числе предпосылок формирования критического мышления, у старших дошкольников использовались следующие критерии:

1. Выявление основной мысли;
2. Выявление противоречий;
3. Выявление нехватки информации;
4. Формулировка собственного вывода.

Итоговые показатели измеряются тремя уровнями:

- высокий уровень (В) усвоения – это получение правильных ответов ребенка без использования наводящих вопросов;
- средний уровень (С)– это получение правильных ответов ребенка после использования наводящих вопросов;

- низкий уровень (Н) – это отсутствие правильных ответов ребенка.

По результатам диагностики выявлены уровни сформированности навыков критического (алгоритмического) мышления воспитанников:

- низкий уровень – 1 воспитанник (4%),
- средний и высокий уровни – 24 воспитанника (96%).

Подробные результаты представлены в таблице, Таблица 12 и на диаграмме, Рисунок 52.

Таблица 12 – Таблица фиксации результатов диагностики

Число детей	Выявление основной мысли			Выявление противоречий			Выявление нехватки информации			Формулировка собственного вывода		
	1			2			3			4		
	Н	С	В	Н	С	В	Н	С	В	Н	С	В
ИТОГО	**1**	**13**	**11**	**1**	**13**	**11**	**1**	**13**	**11**	**1**	**7**	**17**

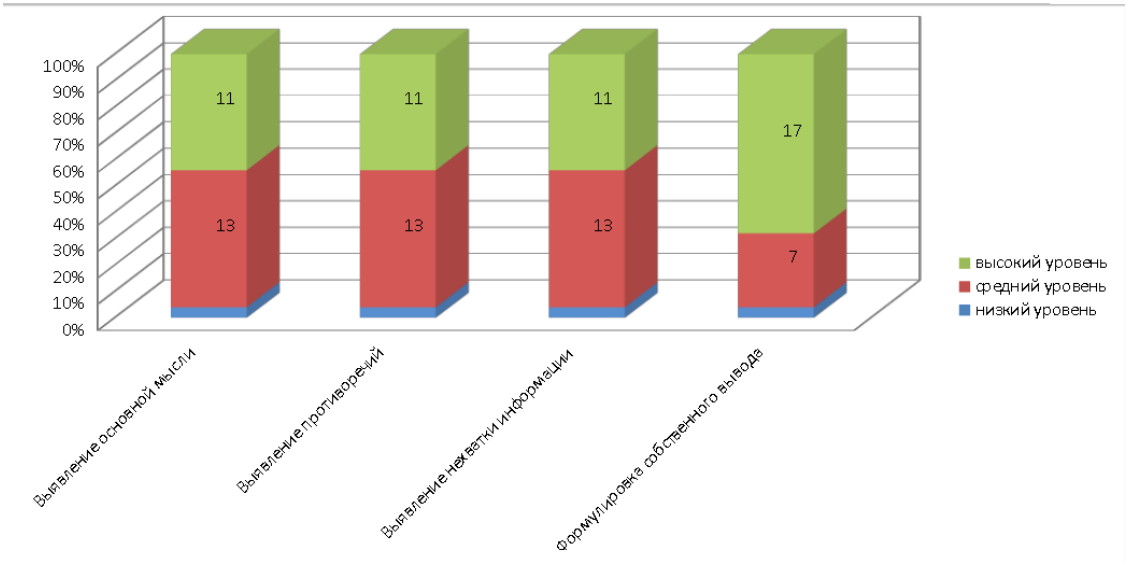


Рисунок 52 – Диаграмма уровней сформированности гибких навыков у воспитанников по результатам программы в 2023 -2024 учебном году

Надо отметить, что также диагностика позволила определить степень усвоения воспитанниками знаний, полученных в результате прохождения программы дополнительного образования «ПиктоМир», таких как: понимание понятийного аппарата ЦОС ПиктоМир, знание легенд виртуальных роботов и элементов игровых полей, а также умение составлять простые программы и программы с повторителями. 24 воспитанника (96%) знают и называют реальных и виртуальных роботов ЦОС ПиктоМир, знают легенды роботов, ориентируются в схемах игровых полей, могут выделить лишние или недостающие элементы, умеют составлять простые программы и программы с повторителями.

По результатам можно сделать вывод: на конец учебного года у воспитанников наблюдаются высокие и средние показатели сформированности алгоритмических навыков и «гибких навыков», в том числе предпосылок формирования критического мышления, что говорит о качественном построении образовательного процесса и доказывает высказанные во введении гипотезы.

Приведены всего несколько примеров результатов наблюдений педагогов-воспитателей за процессом и результатом освоения основ алгоритмизации и формирования алгоритмического мышления у малышей, начиная с четвертого года жизни в дошкольных образовательных организациях и начальной школе в рамках программ:

- дополнительная общеобразовательная программа технической направленности «Алгоритмика. Старт» для детей 4-7 лет.
- парциальная образовательная программа по формированию алгоритмической грамотности «По алгоритмическим дорожкам» (для детей 5-7 лет). Эта программа основана на классических художественных произведениях для детей на русском языке. Важная задача ближайшего будущего−включить в нее элементы произведений на национальных языках.
- «Алгоритмика» для начальной школы.

В результате указанных двухлетних и трехлетних программ «Алгоритмика», в которых дошкольники осваивают на практике азы программирования, составляя

самостоятельно 200-300 программ в ЦОС ПиктоМир, дети стали способны решать в задачи по программированию уровня, предусмотренного действующим ФГОС ООО [170] для выпускников 9 класса [171], Рисунок 53.

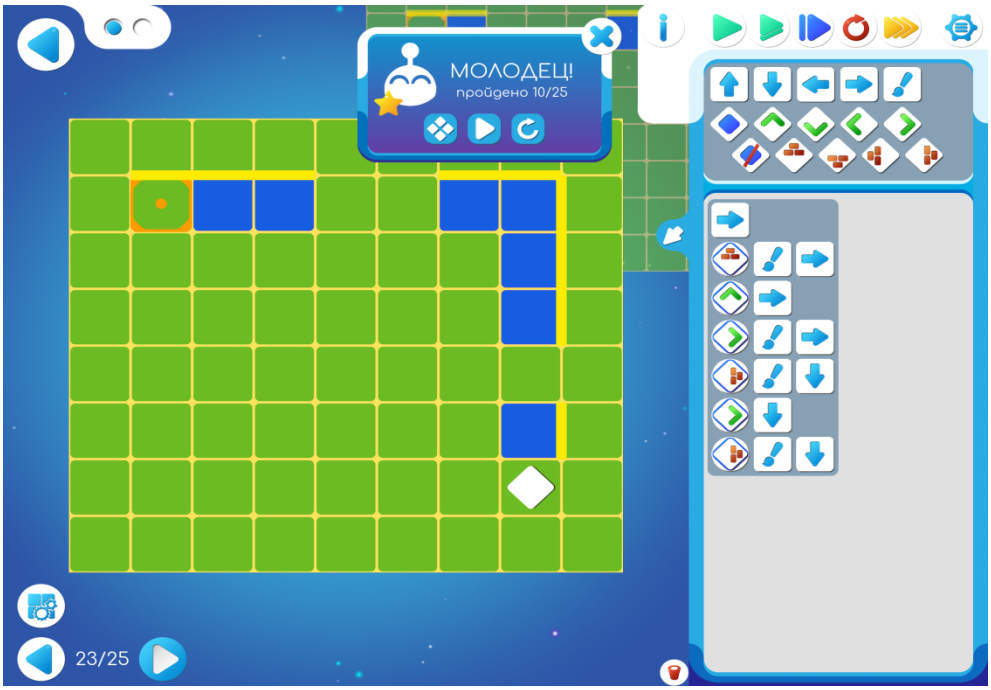


Рисунок 53 – Решение в ЦОС ПиктоМир высоко оцениваемой (2 балла) задачи по программированию в Демоверсии ОГЭ по информатике 2023-2024 гг.

Дошкольники и младшие школьники, участвующие в площадке федерального уровня имеют следующие результаты:

- практическое освоение всех конструкций структурного программирования в бестекстовой ЦОС ПиктоМир, а младшие школьники в блочной ЦОС ПиктоМир-К;
- практическое освоение простейших приемов составления и отладки программ в императивных однопоточных языках программирования;
- освоение системы научных понятий программирования на уровне, доступном 7-летнему ребенку;
- приобретение опыта кооперативного программирования, требующего речевого взаимодействия с партнером в процессе разделения общей

работы между участниками и отладки параллельно работающих алгоритмов.

В докладе Симора Пейперта [362] был представлен «сциентистский» подход к педагогическим исследованиям. Классическое педагогическое исследование состоит в обосновании педагогических новаций методами математической статистики, то есть анализ результатов выборочного сравнительного контроля в некоторый промежуток времени. При радикальном изменении в образовательном процессе детей такой метод, как правило, ничего не может гарантировать, так как для получения реальных результатов необходимо проводить педагогические измерения на большом количестве учеников в течение продолжительного времени. К таким изменениям можно отнести и кардинальное понижение возраста первичного знакомства детей с программированием. Лого С. Пейперта, привнесенное в школу, позволило детям использовать систему не только как учебный инструмент, но и как язык общения ребенка с компьютером, чтобы изменить сам образовательный процесс [223]. Аналогично и сейчас, десятки тысяч детей систематически участвовали в развивающей деятельности с помощью предметно-цифровой образовательной среды ПиктоМир на основе новой методологии в дошкольном и младшем школьном обучении, показали массовый наблюдаемый результат успешного освоения основ алгоритмизации и формирования фундамента алгоритмического мышления.

Таким образом, доказана возможность формирования алгоритмического мышления в раннем возрасте у дошкольников и младших школьников при использовании методологического подхода, использующего ИКТ-насыщенные предметно-цифровые образовательные среды, наполненные необходимым педагогическим инструментом и учебным содержанием, позволяющие не только успешно усвоить системы научных понятий, но и приобрести позитивный опыт практического освоения естественно-научной тематики, что гарантирует дальнейшие успехи в STEM-дисциплинах.

5.2 Результаты внедрения методологии ступенчатого подхода при преподавании информатики и программирования для студентов педагогических университетов.

Эксперимент проходил в течение 2020-х годов в Институте детства МПГУ в рамках дисциплин, содержащих практическое программирование для студентов в своей массе, не освоивших основы алгоритмизации ранее или имеющих поверхностные знания в этой области. Для проведения корректировки группового расписания и тонкой настройки параметров индивидуальных образовательных траекторий студентов, проводилось предкурсовое тестирование, в котором студентам предлагалось выполнить простейшие задания по программированию и оценить свои успехи в области STEM-предметов. Анкетирование проходило в ЦОП Мирера, в приведенном примере опроса участвовали 25 студентов 4 курса МПГУ сдвоенной специальности будущих учителей информатики и начальных классов (гендерная картина: 24 девочки и 1 мальчик). В анкете было предложено 13 вопросов. Ниже приведены ответы на некоторые вопросы, Таблица 13, Таблица 14, Таблица 15, Таблица 16.

Таблица 13 – Статистика ответов на вопрос 5 анкеты: «Баллы ЕГЭ по математике»

Количество баллов за ЕГЭ по математике	Количество студентов
100	0
95-99	0
90-94	0
80-89	2
70-79	11
меньше 70	12

Таблица 14 – Статистика ответов на вопрос 6: «Баллы ЕГЭ по информатике»

Количество баллов за ЕГЭ по информатике	Количество студентов
100	0
95-99	0
90-94	0
80-89	0
70-79	0
меньше 70	2
не сдавал	23

Таблица 15 – Статистика ответов на вопрос 7: «Оцените свои знания по математики»

Оценка своих знаний по математике	Количество студентов
блестящие	1
отличные	2
хорошие	14
нормальные	8

Таблица 16 – Статистика ответов на вопрос 8: «Учили ли Вас в школе на уроках информатики программировать? Если да - то на каких языках?»

Языки программирования	Количество студентов
не изучали программирование в школе	17
Паскаль	6
C / C++	1
Питон	2
КуМир (алгоритмический язык)	5

Как видно, из группы в 25 человек 17 студентов (почти 2/3 группы) не изучали программирование в школе. Такой показатель является не только статистически медианным в различных годах, но и статически имеет малое стандартное отклонение.

Программа педагогического университета относит сложные предметы, связанные с программированием к старшим курсам, поэтому, как правило, меньше 20% студентов изучают информатику и программирование в рамках курсов по выбору в предыдущие годы обучения. В рамках анкетирования студентам предлагается ответить на математический вопрос 11, не приводя аргументов и доказательств, например, «угадать» число решений уравнения *10*sin(x) = x*. Последний вопрос необходим для формирования уровня сложности и повторений в математическом содержании заданий курса.

В последних заданиях тестирования предлагалось описать на любом языке программирования, входящем в ФОП ООО [246], Python, C++, Паскаль, Java, C#, Школьный Алгоритмический Язык или описать алгоритм любым доступным способом. Решение на вышеперечисленных языках программирования студент мог сразу проверить, поскольку анкетирование проводилось в ЦОП Мирера, где в контест допускается включать задания различного типа, Рисунок 54, Рисунок 55.

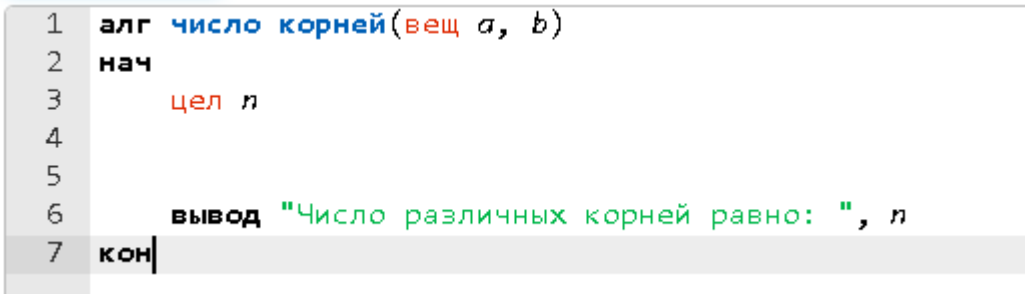


Рисунок 54 – Пример задания 542 на школьном алгоритмическом языке: «Допишите программу вычисления числа различных корней уравнения *x(x-a)(x-b)=0*»

```
1  алг число корней(вещ a, b)
2  нач
3      цел n
4
5
6      вывод "Число различных корней равно: ", n
7  кон
```

Рисунок 55 – Пример задания 13 на школьном алгоритмическом языке: «Допишите программу вычисления числа различных корней уравнения $(x2- a)(x- b)=0$»

В приведенном примере анкетирования группы на 12 вопрос 9 студентов не пробовали отвечать, из 16 студентов, попытавшихся доделать программу решения задачи, справились только 3 студента. К заданию вопроса 13 приступили 12 студентов из них было 3 успешных ответа. Остальные 13 человек не приступали к решению задачи.

Анкетирование показало, что 68% студентов не изучали программирование до этого курса и 32% знакомились с программированием в школе, на дополнительных курсах и т.п. По окончании годового курса по программированию, студентам было предложено повторно решить математические задачи на программирование, аналогичные заданиям в анкете, Таблица 17.

Таблица 17 – Статистика ответов на вопрос 5 анкеты: Баллы ЕГЭ по математике

	В начале года правильно ответили	В конце года правильно ответили
Вопрос 11 (Математическая задача)	4 студента (16 %)	25 (100%)
Вопрос 12 (Программа)	3 студента (12 %)	25 (100%)
Вопрос 13 (Программа)	3 студента (12 %)	25 (100%)

Результаты таблицы позволяют испытывать осторожный оптимизм к достижениям студентов в области освоения азов программирования в рамках основной школьной программы.

В течение ряда лет, начиная с 2015 года, в МПГУ на различных факультетах велся практический курс освоения азов программирования [104]. Основной целью преподаваемой дисциплины являлось формирование алгоритмического мышления у будущих учителей информатики, математики, физики и умение решать задания по программированию в объеме ФГОС [170; 232].

Как изложено в главе 4, используя внедрение элементов геймификации в образовательный процесс в качестве базы для цифровых курсов, например, курса «Алгоритмика в начальной школе», для студентов 3 или 4 курса факультета начального образования ФНО «Институт детства» в МПГУ использовались цифровые образовательные среды ПиктоМир, ПиктоМир-К и КуМир, интегрированные в цифровую образовательную платформу Мирера. На платформе Мирера студенты выполняли задания по программированию на школьном алгоритмическом языке и языке Python (с 2022 года). Следуя методологии ступенчатого вневозрастного подхода обучения, освоение основных понятий программирования и составление алгоритмов происходит поэтапно в цифровых образовательных средах. При этом до 2019 года в цифровом курсе использовались две ЦОС ПиктоМир и КуМир и структура курса выглядела так, Таблица 18.

Таблица 18 – Темы и структура курса до 2019 года

Номер темы	Содержание темы	Число практикумов (контестов) /трудоемкость	Число заданий (суммарно)
1	Методика пиктограммного программирования». Использование ЦОС ПиктоМир и исполнителей Вертун, Двигун, Кувшин (счетчик)	3/60	150
2	Программное управление исполнителями. Кооперативное программирование. Многообразие роботов-Исполнителей.	2/20	30
3	Исполнитель Робот в системе КуМир. Методика изучения основных понятий программирования путем составления алгоритмов управления виртуальным роботом. Переход к текстовому программированию	2/40	30
4	Решение стандартных задач по программированию (в объеме кодификатора ЕГЭ 2011-2015гг.) в системе КуМир.	5/24	60

Задачи по темам 3 и 4 решались в рамках практикумов ЦОС КуМир. Для данного курса было подготовлено 7 практикумов с автоматизированной проверкой заданий. Каждый практикум подготовлен в трех вариантах; демо - вариант и 2 варианта для аудиторной и самостоятельной работы. При выполнении работы студенту предоставлялась возможность самостоятельно (на своем компьютере) проверять правильность решения каждой задачи и предъявлять преподавателю итоговую таблицу решенных им задач.

В этом курсе студенты решали с исполнителями около 30 заданий. Итоги выборочного курса 2019 года показаны в таблице, Таблица 19.

Таблица 19 – Итоги студентов в балльной системе ЦОП Мирера 2019 года

Номер студента	Работа в течение года	Тесты (проверка знаний)				Контрольные работы (практика программирования)			Итого
		1	2	3	4	1	2	3	
1	20	2	2	4	4	8	12	20	72
2	26	2	2	4	4	8	12	20	78
3	28	2	2	4	4	8	12	20	80
4	24	2	2	4	4	8	12	18	74
5	28	2	2	4	3	7	10	17	74
6	28	2	2	4	4	8	12	20	80
7	28	2	2	4	4	8	12	20	80
8	26	2	2	3	4	8	12	17	74
9	28	2	2	4	4	8	12	20	80
10	22	2	2	4	4	8	12	20	74
11	26	2	2	3	4	8	12	17	74
12	28	2	2	4	3	7	10	17	74
13	28	2	2	4	4	8	12	20	80
14	28	2	2	4	3	7	10	17	74
15	26	2	2	3	4	8	12	17	74
16	14	1	1	2	2	8	12	20	60
17	24	2	2	4	4	8	12	18	74
18	28	2	2	4	3	7	10	17	74
19	26	2	2	4	4	8	12	20	80
20	22	2	2	4	4	8	12	20	74
21	26	2	2	4	4	8	12	20	78
22	28	2	2	4	4	8	12	20	80
23	26	2	2	3	4	8	12	17	74
24	14	2	2	4	4	8	6	10	50
25	14	1	1	2	2	8	12	20	60

Баллы в графе «Работа» студент получал за посещение практических занятий и выполнение практикумов в аудитории. Графа «Тесты» содержит результаты знаниевых тестов, испытаний по темам:

1. Основные понятия алгоритмики.
2. Управляющие конструкции на примере задач с исполнителем Робот в КуМире.
3. Табличные величины.
4. Символьные величины и строки.

Контрольные работы включали оценки за выполнение заданий по темам:

1. ПиктоМир.
2. Робот в КуМире.
3. Массивы (ЦОС КуМир).

Все задания первых практикумов (принцип программного управления, повторители, подпрограммы, исполнители Вертун и Двигун в пиктограммной ЦОС ПиктоМир) ранее были апробированы в курсах для дошкольников возраста 6+. Использование вневозрастного подхода в обучении программированию в ЦОС ПиктоМир позволило сократить время на освоение основных понятий программирования и, используя выбранный набор задач, необходимый для формирования основ алгоритмического мышления, закрепило на практике основные концепции структурного программирования. Однако педагогический опыт показал, что в рамках ступенчатой методологии преподавания программирования сложным оставался переход от пиктографического программирования к текстовому. В 2020 года с появлением авторской гибридной ЦОС ПиктоМир-К возник дополнительный уровень в ступенчатой методике курса по программированию, позволяющий решать задачи с исполнителями, которые были доступны только в ЦОС КуМир. Соответственно поменялась и тематика курса, Таблица 20.

Таблица 20 – Темы, структура и примерное содержание ступенчатого курса с 2020

Номер темы	Содержание темы	Число практикумов (контестов) /трудоемкость	Число заданий (суммарно)
1	Методика пиктограммного программирования. Использовалась ЦОС ПиктоМир. Принцип программного управления, повторители, подпрограммы, исполнители Вертун и Двигун. Команды с обратной связью, цикл пока, условные конструкции. Использование нескольких конструкций в одном подпрограмме (понятие «блока»). Понятие «счетчик». Разработка универсальных программ, способных работать в разных однотипных обстановках. Подготовка к переходу к традиционным языкам программирования и использованию числовых переменных. Решение математических задач с использованием «кувшина с памятью». Знакомство с новыми исполнителями. Знакомство и решение задач на параллельное выполнение программы для знакомых исполнителей Практикум Ползун. Управление реальным роботом с использованием ЦОС ПиктоМир. Пультовое управление роботом.	5/40	250
2	Переход к текстовому программированию. Работа в ЦОС ПиктоМир-К Задания на освоение текстового представления основных конструкций программирования. Учимся превращать «волшебный кувшин» в «счетчик» с переменными. Осваиваем использование понятий переменных при решении задач с исполнителем Робот	2/30	200

	Понятия «температура» и «радиация». Решение задач на поиск средних значений. Решаем математические задачи: считаем периметр; вычисляем площадь. Понятие параметры. Решаем задачи с исполнителем Черепашка. Система координат, Векторы. Решаем задачи с исполнителем Чертежник. Решаем задачи с использованием рекурсии. Решение задач с Роботом из списка задач ОГЭ. Задача 15.		
3	Решение задач на алгоритмическом языке. ЦОС КуМир. Исполнитель Робот в ЦОС КуМир. Программное управление исполнителем Робот. Задачи на закрашивание всех клеток прямоугольника или граничных клеток прямоугольника. Цикл N раз. Вложенные циклы. Вспомогательные алгоритмы. Цикл пока. Целочисленные аргументы. Подготовка к работе с массивами; поиск максимальной или средней радиации, температуры на двумерном поле Робота. Используемые конструкции и понятия; Команда присваивания. Виды величин Алгоритмы с величинами. Простейшие алгоритмы с результатами.	2/30	20
4	Решение стандартных задач по программированию. Знакомство со встроенными в ЦОС КуМир функциями (min, imin, max, imax и т.д.). Решение математических задач: нахождение минимума или максимума 3-4 чисел, число максимумов, нахождение количества различных чисел, определение вида треугольника по трем длинам сторон, расстояние от точки плоскости до отрезка или прямоугольника со сторонами параллельными осям координат, максимальное значение квадратного трехчлена на отрезке. Цикл для. Знакомство с табличными величинами (массивами). Однопроходные алгоритмы работы с массивами:	5/24(до 2022 года) 7/44(с 2022 года)	100

	заполнение массива, сумма, максимум, минимум, номер первого или последнего максимума, значение многочлена. Задачи на сдвиг элементов. Однопроходные алгоритмы работы со строками. Знакомство с символьными величинами. Перевод натурального числа в десятичную и другие системы счисления.		
5	Решение задач темы 4 на языке программирования Python в ЦОП Мирера.	5/20 (с 2022 года)	100

В рамках курса проводились промежуточные испытания, контрольные и знаниевое и навыковое тестирование. Так как курс читался для студентов будущих учителей информатики в основной школе, то необходимо было сформировать умения и навыки решения задач из списка ОГЭ [171]. Ниже приведен пример одного задания курса с исполнителем Робот из списка задач ОГЭ (Задача 15).

«На бесконечном поле есть горизонтальная и вертикальная стены. Правый конец горизонтальной стены соединён с нижним концом вертикальной стены. Длины стен неизвестны. В каждой стене есть ровно один проход, точное место прохода и его ширина неизвестны. Робот находится в клетке, расположенной непосредственно слева от верхнего конца вертикальной стены. На рисунке указан один из возможных способов расположения стен и Робота. Напишите для Робота алгоритм, закрашивающий все клетки, расположенные непосредственно выше горизонтальной стены и левее вертикальной стены. Проходы должны остаться не закрашенными. Робот должен закрасить только клетки, удовлетворяющие данному условию. При исполнении алгоритма Робот не должен разрушиться, выполнение алгоритма должно завершиться. Конечное расположение Робота может быть произвольным. Алгоритм должен решать задачу для любого допустимого расположения стен и любого расположения и размера проходов внутри стен» [171], Рисунок 56.

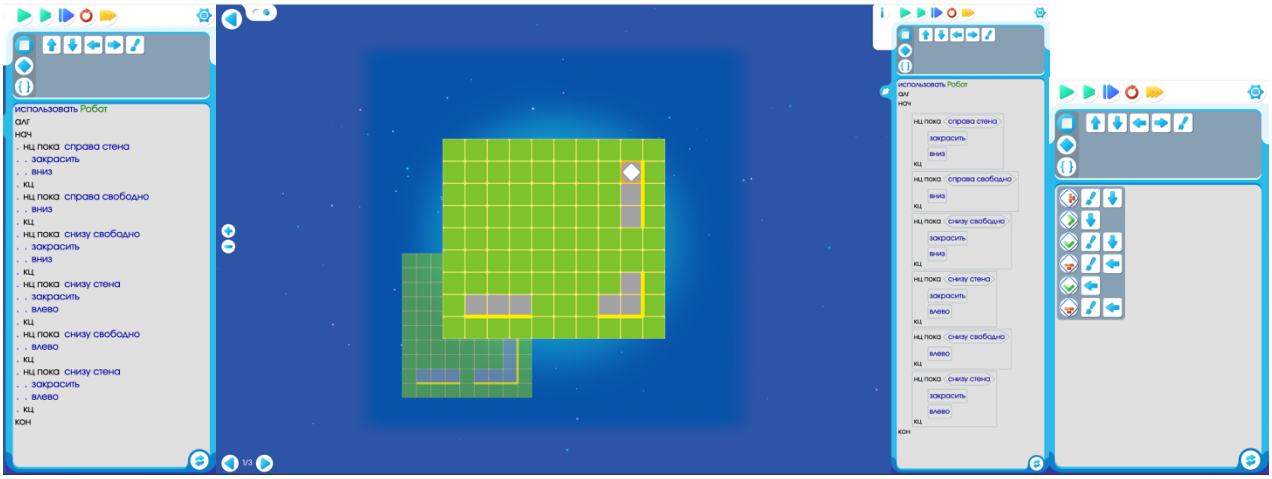


Рисунок 56 – Пример решения задачи 15 ОГЭ по информатике в ЦОС ПиктоМир-К на языке Пикто и школьном алгоритмическом языке.

Ниже приведены результаты выполнения контестов Робот 1 и Робот 2 — практикумов с автоматической проверкой задач (по 10 заданий разного уровня сложности) с исполнителем Робот в среде ЦОС КуМир в 2019 году, когда в программу курса были включены две темы: пиктограммное программирование и текстовое, Таблица 21.

Таблица 21 – Выборка 2019 года решения студентами практикумов Робот 1 и Робот 2

Номер студента	Робот 1(15 заданий)	Робот 2(15 заданий)
1	15	15
2	15	15
3	15	15
4	15	0
5	2	3
6	15	14
7	14	14
8	15	15
9	15	0
10	15	13
11	13	2
12	15	3
13	15	2
14	15	15
15	15	0
16	3	3
17	15	0
18	15	0
19	14	14
20	15	15
21	15	0
22	15	15
23		

24		
25		
Прошли порог	21 (84%)	11(44 %)
не прошли порог	4 (16%)	14(56 %)

Ниже приведены результаты выполнения заданий в ЦОС КуМир в 2020 году, когда в программу курса была добавлена работа в гибридной ЦОС ПиктоМир-К. Внедрение гибридной ЦОС ПиктоМир-К позволило увеличить число задач с исполнителем Робот до 100 в этой среде. Это значительно улучшило результаты выполнения практикумов с роботом в КуМире в группе, Таблица 22.

Таблица 22 – Выборка 2020 года решения студентами практикумов Робот 1 и Робот 2

Номер студента	Робот 1(15 заданий)	Робот 2 (15 заданий)
1	15	15
2	15	15
3	15	15
4	15	15
5	14	15
6	15	15
7	15	15
8	15	15
9	15	15
10	15	15
11	14	15
12	15	15
13	15	15

14	15	15
15	15	15
16	15	15
17	15	15
18	15	15
Прошли порог	18 (100%)	18(100 %)
Не прошли порог	0	0

Можно сделать вывод, что практика решения большого количества задач в комфортной переходной ЦОС ПиктоМир-К помогает качественнее решать задачи в ЦОС КуМир на текстовом языке. Систематическое погружение в ЦОП Мирера, предоставляющую омниканальный доступ к материалам и автоматическую проверку студенческих заданий, позволяет отстающим успешно освоить основы программирования (включая базовые фундаментальные понятия). Для этого используются, как в школе и ДОО, ЦОС ПиктоМир и ПиктоМир-К, интегрированные в ЦОП Мирера. Наблюдаемая геймификационная направленность ЦОС ПиктоМир, ЦОС ПиктоМир-К помогает снимать дидактические затруднения даже в таком серьезном материале, например, как алгоритмы взаимного исключения [20].

Освоение основ программирования в выравнивающем курсе или начальном курсе в вузах проводится с использованием тех же парадигм с переходом к гибридной ЦОС ПиктоМир-К аналогично школе и ДОО на основе тех же заданий, но с более высокой интенсивностью. Чтобы избежать проблемы «цифрового равенства» при обучении, когда группа отстающих студентов является тормозом общегруппового образовательного процесса, в ЦОП Мирера используется аппарат динамических траекторий, который позволяет лидерам освоить дополнительный материал.

Проведенные исследования и педагогические эксперименты с использованием ЦОС ПиктоМир, ЦОС ПиктоМир-К, ЦОП Мирера и др. показали

высокую эффективность интеграционного вневозрастного подхода при преподавании основ программирования и алгоритмизации в ДОО, школах, вузах. Использование единого подхода в пропедевтических курсах позволяет организовать множественные точки входа в систематический образовательный процесс, который обеспечен полным входным и выходным контролем освоенных компетенций.

5.3 Опыт построения априорных оценок успеваемости студентов в ЦОП Мирера

Специфика ЦОП Мирера состоит в глубокой интеграции в платформу цифровых курсов с заданиями, тестирующих системам и цифровых образовательных сред, что позволяет студентам выполнять все практические работы внутри ЦОП Мирера. При этом у обучаемых сохраняется их подробный цифровой след о работе с платформой, что позволяет автору курса проводить анализ различного уровня глубины для последующего использования результатов исследования в коррекции курса, набора заданий, а также для формирования индивидуальных траекторий учащихся [127; 85; 11]. В ЦОП Мирера цифровой след содержит обширные данные о выполнении студентами домашних заданий, классных работ и промежуточных проверок и многое другое.

В эксперименте, который проходил в нескольких группах механико-математического факультета МГУ имени М.В. Ломоносова в 2020-2021 учебном году, стояла задача в получении априорных финальных оценок студентов, используя информацию о текущих достижениях. Для этого учащиеся были разделены на три класса, чтобы можно было автоматически вычислить вероятность попадания студента в один из следующих трех классов на финальных испытаниях:

- Отстающие учащиеся, то есть те, кто получит на зачете или экзамене неудовлетворительную оценку. Обнаружение такого ученика заблаговременно позволяет педагогу предоставить серьезный объем

помощи, вплоть до индивидуальных занятий, предотвратив провал в обучении конкретных студентов.

- Успевающие учащиеся, которые в изучаем курсе имеют средние результаты. При 5-ти бальной системе оценивания на финальных испытаниях такие студенты получат оценку удовлетворительно или хорошо. Этой группе требуется умеренная поддержка со стороны преподавателя, в основном для перехода в следующий, более высокий класс учащихся с высокими достижениями.
- Учащиеся высоких достижений. Студенты этого класса превосходят результатами среднестатистического студента группы и, как правило, требуют дополнительных сложных заданий и материала сверх осваиваемой программы.

В ЦОП Мирера курс представляет собой иерархическую структуру неограниченной вложенности. То есть курс может состоять из более мелких курсов, называемых промежуточными. В свою очередь, последние могут вновь состоять из промежуточных курсов и тем, которые уже делятся на контесты (практические семинарские и домашние работы, контрольные и т.п.), ассоциированные с реальными аудиторными занятиями, которые, в свою очередь, состоят из задач, проверяющихся ЦОП Мирера автоматически, Рисунок 57. В проведенном исследовательском эксперименте были отобраны простейшие одноуровневые курсы и темы, то есть финальные оценки не всегда являлись отчетными отметками в зачетных/экзаменационных ведомостях по истечении учебного семестра или года. Протяженность таких курсов варьировалось от нескольких недель до семестра.

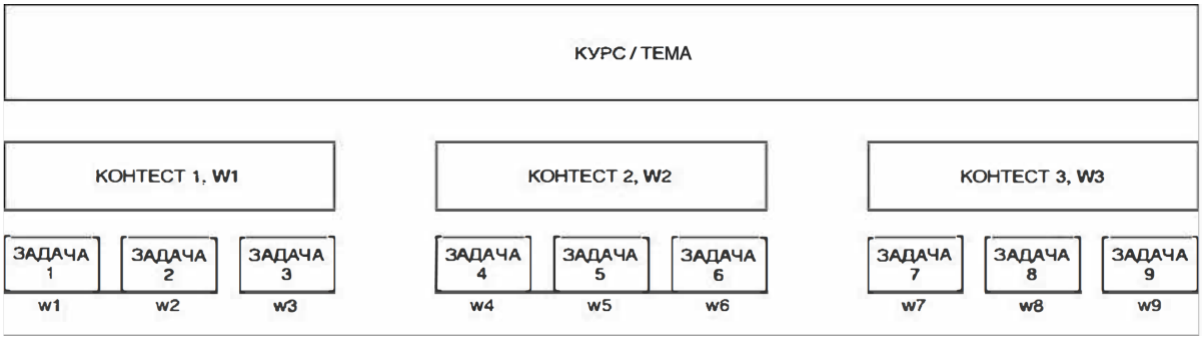


Рисунок 57 – Иерархическая структура оценок курса/темы.

Для учета вклада контестов в курс на основании эмпирической оценки педагога присваиваются веса $W_1, W_2, W_3, \ldots$ Задачи контеста оцениваются весами $w_1, w_2, w_3, \ldots$, отражающими сложность и уровень освоенной компетенции в контесте. При этом общий вклад (важность, уровень) задачи i контесте j оценивается нормированным весом, который считается по Формуле 1:

$$w_i' = \frac{w_i}{w_1 + w_2 + w_3 + \cdots} \times \frac{W_j}{W_1 + W_2 + W_3 + \cdots}$$

Предполагается, что в процессе обучения студент последовательно совершает попытки выполнить задачу i в рамках некоторого контеста j и может получить за ее выполнение максимальную оценку w_i'. По окончании курса все нормированные веса всех сданных задач учащегося суммируются, и преподаватель выставляет окончательную оценку студенту, опираясь на полученный числовой результат. Понятно, что если студент успешно выполнил все задания всех контестов, то его достижения оцениваются равными 1, напротив, абсолютно неуспевающий студент получит оценку 0. Для классификации студента по финальному баллу используются эмпирически подобранные пороги, разделяющие классы.

Кроме веса задачи, в эксперименте для предсказания финального результата с использованием искусственных нейронных сетей используются данные о *попытках сдачи задачи* вне зависимости от принадлежности к контесту. Как величина попытка сдачи задачи использует следующие признаки:

1) Временная отметка попытки сдачи.

2) Статус попытки (пройдено, не пройдено, ошибка, плагиат и т. д.).
3) Тип задачи (обычная, легкая, сложная и т. д.).

Если выбрать случайный временной отрезок от начала курса *t*, то на основании собранной информации о попытках сдачи задач конкретного студента требуется с большой вероятностью совпадения предсказать финальный балл студента на основании текущей информации о прохождении курса. При этом учитываются не только результаты прохождения контестов, но и суммарный балл, набранный за время *t*, Рисунок 58.

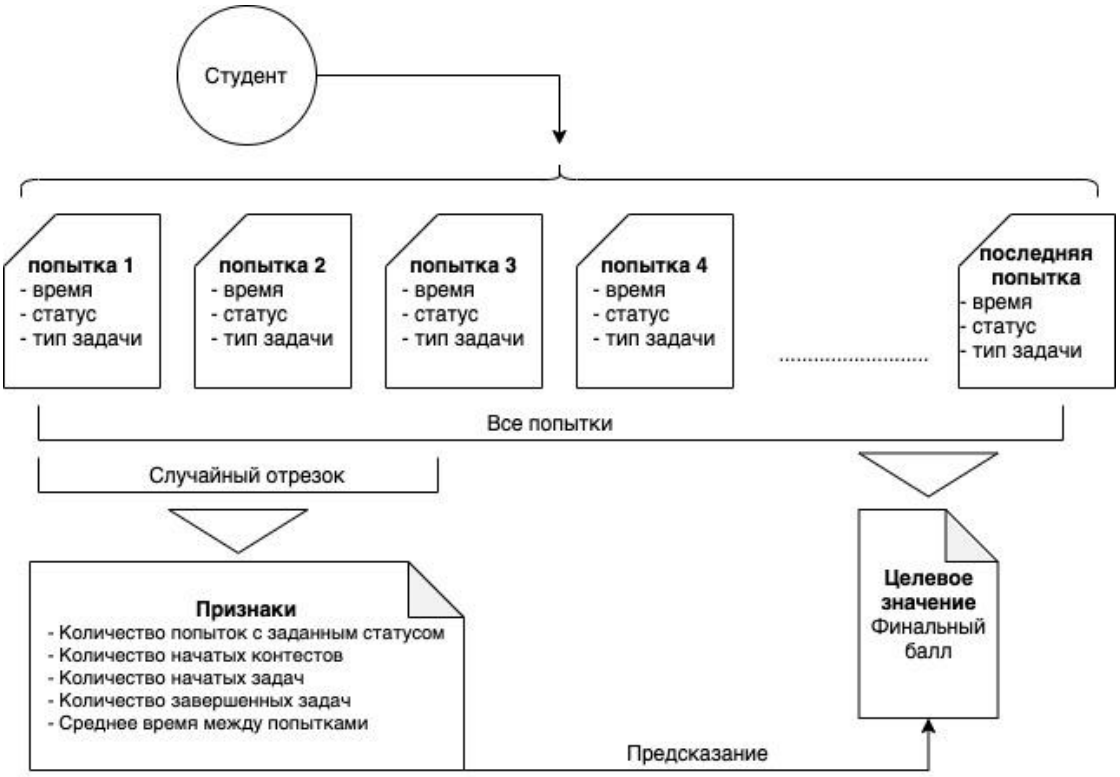


Рисунок 58 – Структура данных о попытке сдачи задач

Полученные предсказания позволяют педагогу определить студента или группу студентов, которым потребуется от него непосредственная помощь для перехода в более высокий класс. Для получения высококачественного результата предсказания необходим большой объем данных, используемых для создания модели методом глубокого обучения. Эксперимент проводился на данных курсов предыдущих 3-х лет, накопленных в цифровом следе ЦОП Мирера из 3206 пар

студент-курс. Каждая такая пара была представлена последовательностью попыток сдачи задач, данным студентом в рамках данного курса. Гистограмма числа студентов, распределенных по группам логарифма (натурального) числа попыток представлена на рисунке, Рисунок 59.

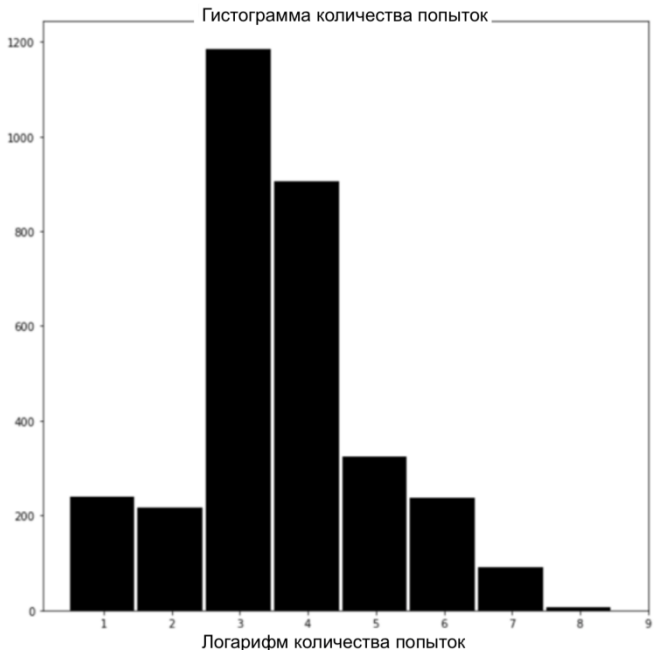


Рисунок 59 – Гистограмма логарифма длин последовательностей попыток сдачи.

По окончании курса большинство студентов предприняло 20-150 попыток сдачи. При этом значительной доле студентов, 7.8%, потребовалось от 200 до 2200 попыток. С целью повысить устойчивость решения в рамках эксперимента использовались только признаки, не зависящие от общего числа попыток, предпринятых студентом.

Распределение финального балла студента и пороговые значения финального балла, показанные на рисунке выше, были предоставлены тремя опытными практикующими преподавателям, Рисунок 60.

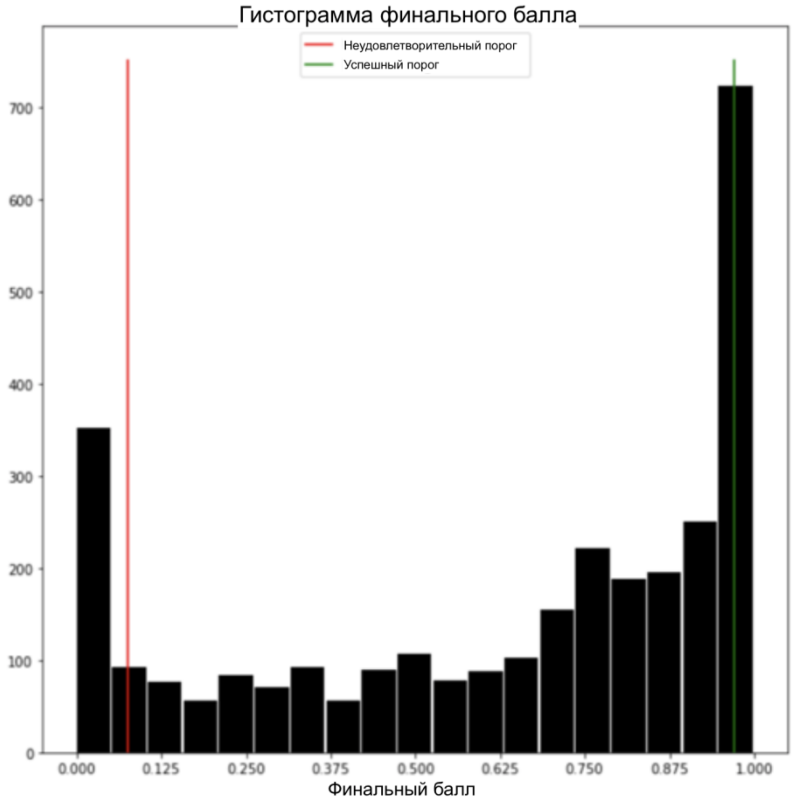


Рисунок 60 – Распределение финального балла студента

Пропорции классов студентов в соответствии с данными порогами представлены на рисунке, Рисунок 61.

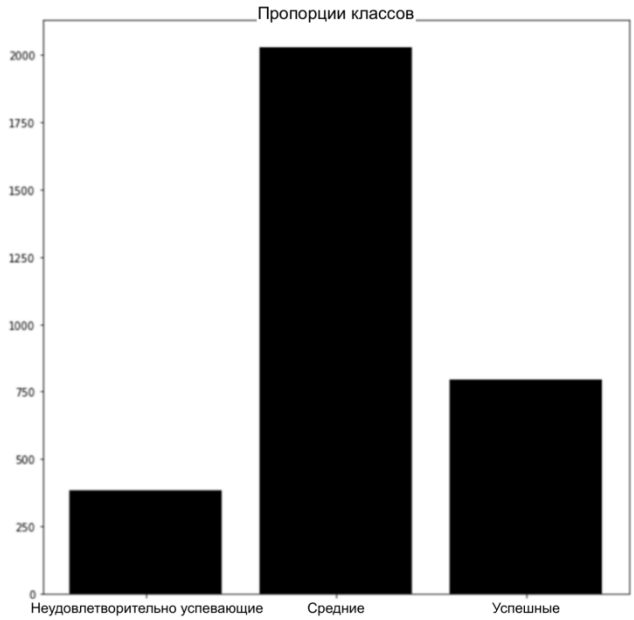


Рисунок 61 – Пропорции классов студентов.

По итогам для каждого класса определяется характер и целесообразность автоматизации корректирующих действий для «подтягивания» отстающих. Оценка строится с использованием искусственных нейронных сетей. Полученная априорная оценка может быть использована для ранней диагностики недостижимости уровней компетенций, а также для построения адаптивных треков обучения для хорошо успевающих.

Результаты экспериментов показали, что при прохождении около 20% курса можно предсказать будущую успешность студента на основании текущих результатов прохождения курса. Априори, зная класс, к которому отнесен тот или иной студент, можно определить не только методы педагогического воздействия, но и целесообразность автоматизации предпринимаемых преподавателем действий.

Так, автоматизация действия преподавателя для помощи студентам из первого класса состоит в том числе из набора задач на повторение с целью выровнять компетенции студента, необходимые для обучения в данном курсе. Отметим, что помощь студентам с высокими достижениями может быть оказана предложением повысить простой эвристикой сложность заданий. При этом автоматизация в области персонализации обучения наиболее перспективна для успевающего класса студентов, так как результат применяемых педагогических технологий будет наиболее массовым. Так, например, в ЦОП Мирера автоматизация предполагает использование в учебном процессе «виртуального ассистента преподавателя». Он учитывает особенности каждого студента и в зависимости от характера возникающих у него проблем выполняет корректирующие действия, например, предлагает повтор материала, который не был полностью усвоен или требует повторения для закрепления [296; 279; 203].

5.4 Исследование данных цифрового следа ЦОП Мирера

Вся активность студентов во время обучения при использовании ЦОП Мирера приводит, в том числе, к сохранению цифрового следа, данные которого впоследствии можно использовать, например, для построения индивидуальной адаптивной траектории студента. Для преподавателя аналитика, построенная на цифровом следе студента, очень важна для проведения педагогического управляющего воздействия на образовательный процесс, например, путём внесения изменений в контент курса для конкретной группы или отдельных студентов. При большом количестве студентов педагогу трудно выделить неуспевающих студентов или отличников только на основании кумулятивной статистики. Даже пополнение вектора успеваемости студента данными о посещаемости аудиторных занятий и заимствовании решений не дает возможности построить цифровой портрет обучаемого, а позволяет лишь выделить класс успеваемости студентов, дифференцируя последнего по трем классам: «отличники», «хорошисты» и «двоечники».

Как и было указано в предыдущем параграфе отличники – это студенты высоких достижений, которые способны самостоятельно осваивать материал и которым, как правило, не нужна помощь преподавателя. Студены этого класса в основном посещают все занятия, чтобы не пропустить важного материала, при этом всегда выполняют задания первыми. Таким слушателям необходимо предоставлять большее количество сложных, дополнительных заданий, что позволяет повысить уровень их компетенции в предмете. Дополнительные набранные баллы мотивируют студентов высоких достижений на самостоятельное планирование своего учебного времени и участие в управлении персональным образованием, что позволят им, учитывая накопленные баллы, пропустить какие-то базовые задания. Статистически отличники имеют низкий уровень заимствования.

Успевающие студенты, хорошисты – это слушатели, которые удовлетворительно и хорошо усваивают материал, имеют достаточно высокий

уровень посещаемости. Несмотря на то, что, в основном, студенты этого класса стараются избегать заимствований, они иногда просят помощи у отличников, что повышает уровень плагиата. Это значительный по количеству студентов класс в учебной группе. Так как всегда есть вероятность перехода учащегося из этого класса в класс высоких достижений или снижение своих учебных показателей и переход в низший класс, то статистически и педагогически эти студенты представляют наибольший интерес.

Двоечники, или класс неуспевающих – это студенты, которые не способны самостоятельно освоить учебный материал. Учащиеся этого класса, как правило, не посещают аудиторные занятия по психологическим или социальным причинам. Такое поведение приводит к ухудшению общего показателя успеваемости группы и персональных показателей неуспевающих студентов. На контрольных мероприятиях такие студенты пытаются активно заимствовать решения других студентов. Преподаватель может наблюдать интерактивно картину поведения двоечников на контрольных, когда задания выполняются ближе к временному окончанию испытания и с высоким уровнем заимствования результата, что видно в ЦОП Мирера, а также когда такие студенты выполняют задания с первой попытки.

Локальной целью преподавателя можно декларировать позитивную устойчивость успевающих студентов, когда студент может перейти в высший класс, но не понизить свои успехи, не попасть в класс неуспевающих студентов. Таким образом, цель модели в ЦОП Мирера уметь предсказать момент возможного ухудшения успеваемости хорошиста. Это позволит педагогу исправить положение конкретного студента, а в случае массового ухудшения показателей успеваемости студентов произвести соответствующие изменения в курсе, например, добавить поясняющие и поддерживающие задания в контест или декомпозировать сложную для прохождения тему.

Базовым критерием, демонстрирующим ухудшение успеваемости студента, будет факт сохраняющегося или нарастающего падения числа баллов, набранного

студентом за контест или тему. Однако нельзя базироваться только на этом показателе. Важными дополнительными критериями будут падение посещаемости и увеличение процента плагиата. Также важен критерий оценки количества потраченных на задачу попыток. Можно сформировать критерий отнесения студента к неуспевающей группе следующим образом: малое количество баллов за продолжительный период, или, наоборот, достаточное количество баллов, но малое количество попыток и высокий уровень заимствований. Используя цифровой след в ЦОП Мирера, можно рассмотреть временной ряд из попыток, при этом отмечая, что неуспевающие студенты сдают задания ближе к концу контеста.

Представление попыток решений студентов, а также результатов за контесты в виде временного ряда предоставляет широкий спектр возможностей по построению поведенческой аналитики всех классов студентов. При этом чистые метрики, без привязки друг к другу и ко времени показывают ограниченную картину с большим количеством ложных результатов. Обучившись на поведенческой аналитике и цифровом почерке студентов, можно детектировать изменение этого почерка, что является основным сигналом к тому, что студент в скором времени может сменить свой класс.

Использование аппарата временных рядов позволяет не просто предсказать финальный класс студента по небольшому количеству начальных данных, но и предсказать дальнейшую траекторию успешности обучения студента, то есть предсказать, какие баллы будет набирать слушатель в будущем, с каким плагиатом, с какой посещаемостью и так далее. Предсказываются не какие-то конкретные итоговые оценки, а весь ряд обучения студента в целом, по которому потом можно построить любые метрики и критерии, например, класс студента в будущем. Благодаря этому можно заранее узнать, когда студент будет ухудшать свои показатели.

Как сказано выше, основные данные, которые подлежат рассмотрению: процент набранных баллов за контест, посещаемость, плагиат и количество попыток. Дальше будут обсуждаться результаты аналитических исследований

педагогического эксперимента на примере двух групп 2021 и 2022 года двухлетнего курса механико-математического факультета МГУ имени М.В. Ломоносова.

Практически на всех графиках будет рассматриваться скользящее среднее, а не чистые метрики студентов. Это объясняется тем, чтобы визуально сделать графики более сглаженными и понятными для восприятия. Однако, и для работы модели в целом это имеет большое значение. Используя скользящее среднее, дается возможность учитывать часть предыдущих результатов студента, наблюдать его результаты в динамике. Результаты за весь период обучения могут привести к необъективности оценок, поскольку плохая успеваемость студента при изучении какой-то конкретной темы не влечет за собой плохую успеваемость в целом.

Ниже приведены графики с временными рядами всех студентов за целый курс, начатый в 2021 году. На первом графике приведено количество попыток студентов по времени, на этом графике каждая точка – это среднее количество попыток студента за конкретный контест. Таким образом, по оси X отложены отсортированные по времени контесты, по оси Y – усредненное по всем задачам контеста количество попыток студента, потраченного на задачу, после чего от этого берется скользящее среднее, Рисунок 62. На графике четко видно 4 больших непрерывных промежутка, а также один короткий в январе 2023 года. Это объясняется тем, что в этом двухлетнем курсе 4 семестра, а в январе 2023 проводится экзамен за весь курс. На графике, например, первого семестра, четко виден небольшой всплеск в середине семестра – это объясняется проведением промежуточных контрольных испытаний, а также большой всплеск к концу семестра, когда студенты, в основном неуспевающие, пытаются с большим количеством попыток закрыть зачетные мероприятия. Такая же картина видна во втором семестре, в третьем семестре она более сглажена, поскольку студенты уже приобрели определенный опыт и компетенции, когда зачетные задачи стали сложнее. В четвертом семестре явно видно падение среднего количества попыток,

поскольку изучаются такие темы, как Python, OpenGL, где для решения задач не требуется большого количества попыток.

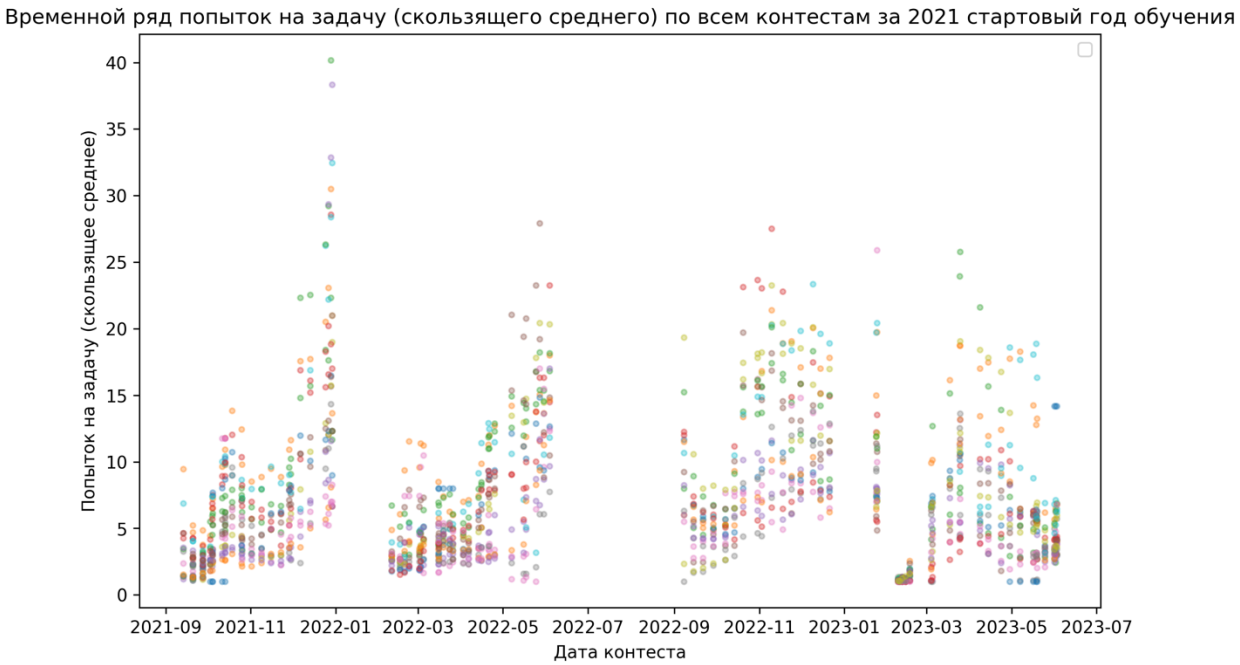


Рисунок 62 – Среднее количество потраченных на задачу попыток студентов, начавших обучение в 2021 году. Каждая точка на графике соответствует конкретному студенту в группе.

Если посмотреть на аналогичный график плагиата, то можно увидеть, что к концу семестра плагиат, вопреки ожиданиям, падает, Рисунок 63. На этом графике также каждая точка – это средний плагиат по всем задачам за контест у конкретного студента, по оси X контесты, отсортированные по времени, по оси Y – средний плагиат по всем задачам за этот контест. Падение плагиата к концу семестра объясняется опять же спецификой преподавания курса, когда в конце семестра отличники получают зачеты без сдачи экзаменов и других контрольных мероприятий, а остальным студентам становится не у кого заимствовать решения. Помимо этого, на контрольных мероприятиях выдаются индивидуальные варианты каждому студенту, чем объясняется также, например, падение плагиата в середине первого семестра.

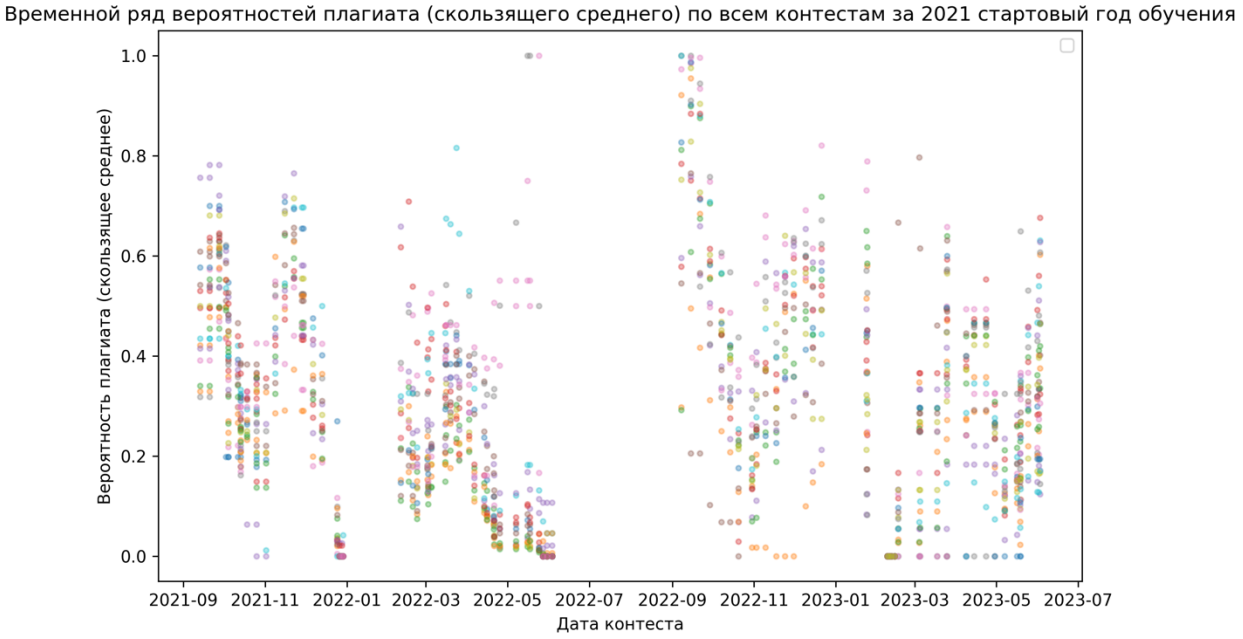


Рисунок 63 – Средний плагиат по всем задачам за контест у студентов, начавших учебу в 2021 году. Каждая точка на графике соответствует конкретному студенту в группе.

Анализа графика посещаемости, как бинарной величины (0, студент не посетил контест, или 1, студент посетил контест) представляет сложность наблюдения. Для отслеживания динамики посещаемости считается последовательная сумма посещаемости для каждого студента, Рисунок 64. На данном графике любая ступенька означает, что студент пропустил данный контест. Если ступенька является продолжительной и их становится много, то есть, когда падает общий наклон графика, то можем сделать вывод, что у студента есть ненулевая вероятность покинуть свой класс. На рисунке наблюдаемы 3 выраженные ступеньки – это каникулы между семестрами. Отчетливо видны студенты, посещающие большинство контестов – на таком графике они всегда будут верхними линиями, а также видны студенты, мало посещающие занятия – они всегда будут внизу графика. Также видны студенты, подключившиеся к курсу позже, например, которые пропустили начало или часть курса по уважительным или неуважительным причинам. Помимо этого, видны студенты, которые покинули курс досрочно, что выражено продолжительной горизонтальной линией.

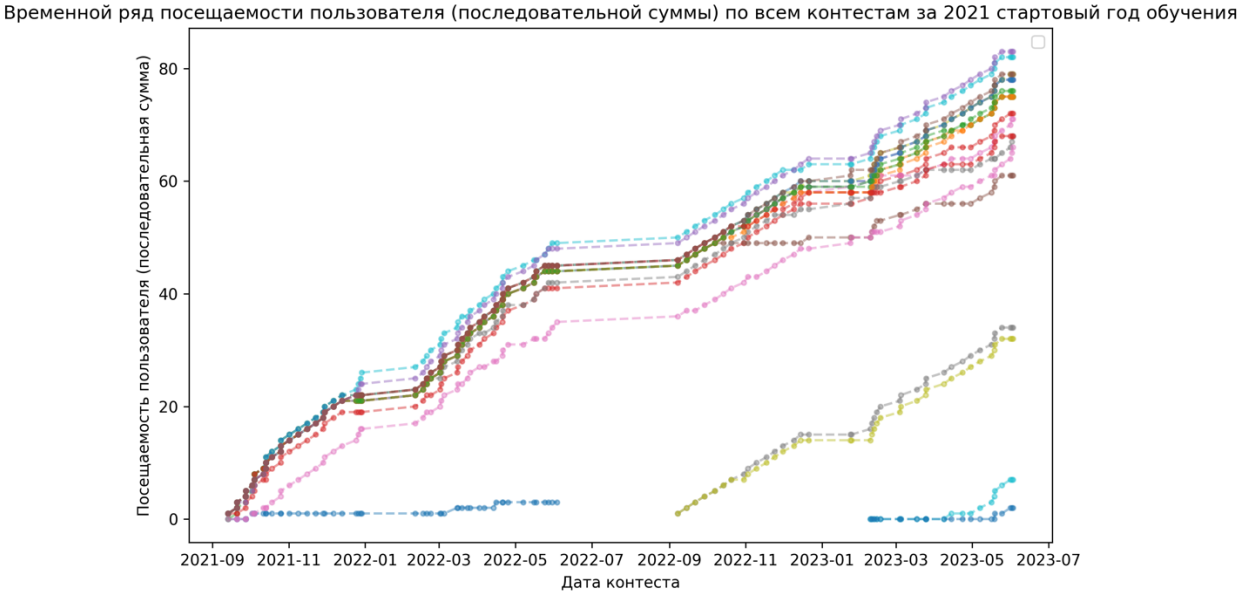


Рисунок 64 – Последовательная сумма посещаемости по контестам у студентов, начавших учиться в 2021 году. Каждая точка на графике соответствует конкретному студенту в группе.

Ниже приведено сравнение результатов студентов, начавших курс в 2021 и в 2022 годах. Основной график – сравнение процента набранных баллов студента в 2021 и 2022 году, Рисунок 65. По оси X на данном графике также отсортированные по времени контесты, по оси Y сплошной линией нарисован средний процент набранных баллов по контесту студентами группы, а область вокруг – это 20% и 80% квантили набранных баллов. Таким образом, можно сказать, что сравниваются диапазоны условных хорошистов курсов, их количества и среднего уровня. Уже на этом графике видно, особенно, на примере 3 семестра со сложными заданиями, что в 2022 году квантили стали значительно ближе к среднему, то есть по итогу обучения сформировался большой костяк уверенных хорошистов (получающих оценку отлично за курс, конечно), набирающих без разброса большее количество баллов. Можно сделать вывод, что чем меньше разброс квантилей хорошистов – чем более качественное, равномерное обучение по курсу и тем успешнее результаты педагогического эксперимента.

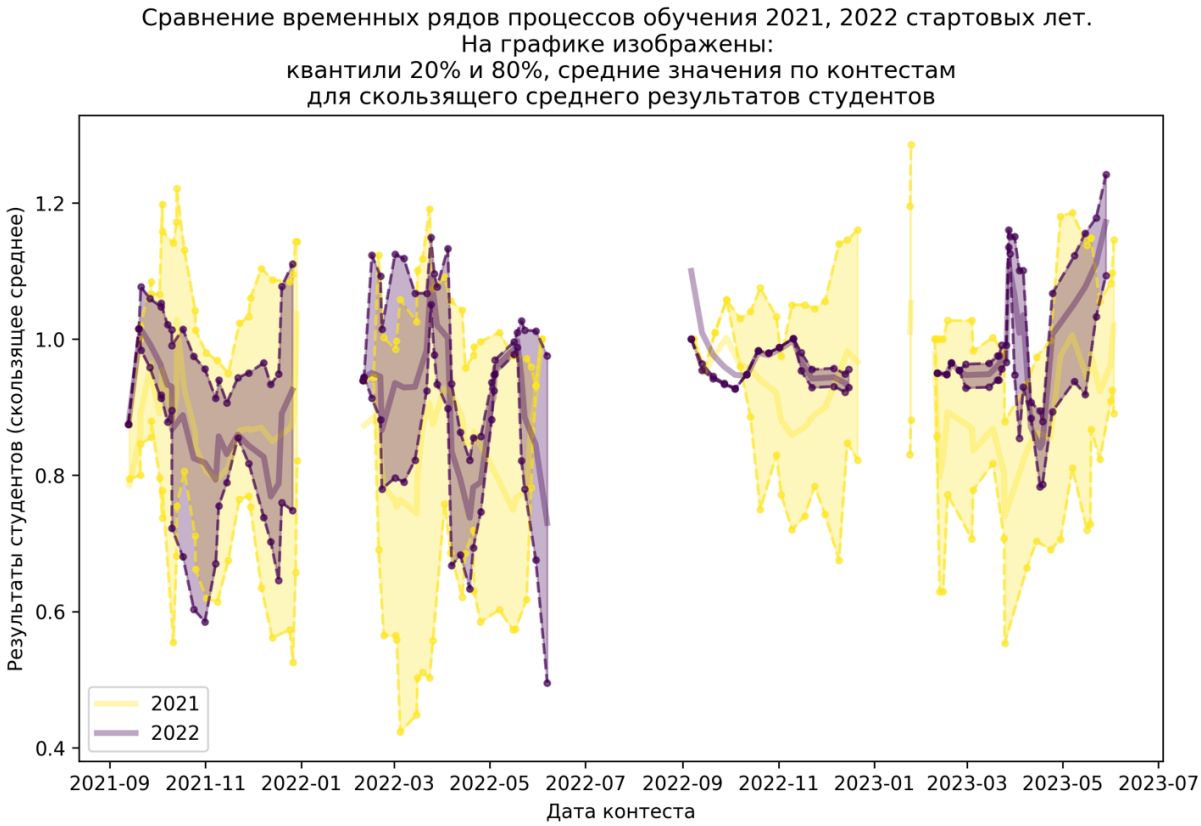


Рисунок 65 – Процент набранных баллов по контестам в среднем у студентов группы, среднее, а также квантили 20% и 80% при сравнении результатов 2021 и 2022 гг. Мехмат МГУ.

Для проведения итогового сравнения представленные выше графики усредняются по всем контестам. Для наглядности сравнение проводится с помощью графика boxplot, на котором линия в середине прямоугольника – это среднее, верхняя и нижняя границы прямоугольника – это 20% и 80% квантили, «усики» – это минимум и максимум. Графики посещаемости, Рисунок 66, плагиата, Рисунок 67, и процента набранных баллов по контестам, Рисунок 68. На графике посещаемости в связи с тем, что данные бинарны, квантили и минимум и максимум совпадают со значениями 0 и 1, однако можно увидеть, что посещаемость в среднем в 2022 по сравнению с 2021 году увеличилась. Средний плагиат в группе практические не изменился, однако квантили стали ближе к низкому плагиату, то есть можно сделать вывод, основная масса студентов стала меньше списывать. Средний процент набранных баллов увеличился, а также квантили стали намного

ближе к среднему, что говорит о более успешном освоении курса группой в целом, а не ее отдельными студентами, то есть о более однородном и качественном обучении.

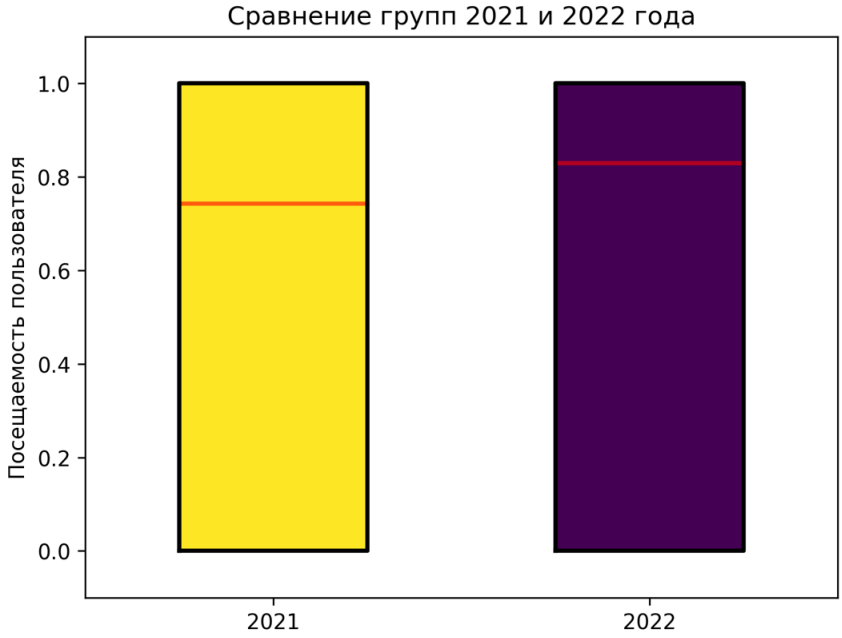


Рисунок 66 – Средняя посещаемость по всем контестам и всем студентам в группах, среднее, 20% и 80% квантили, минимум и максимум, при сравнении результатов 2021 и 2022 гг. Мехмат МГУ.

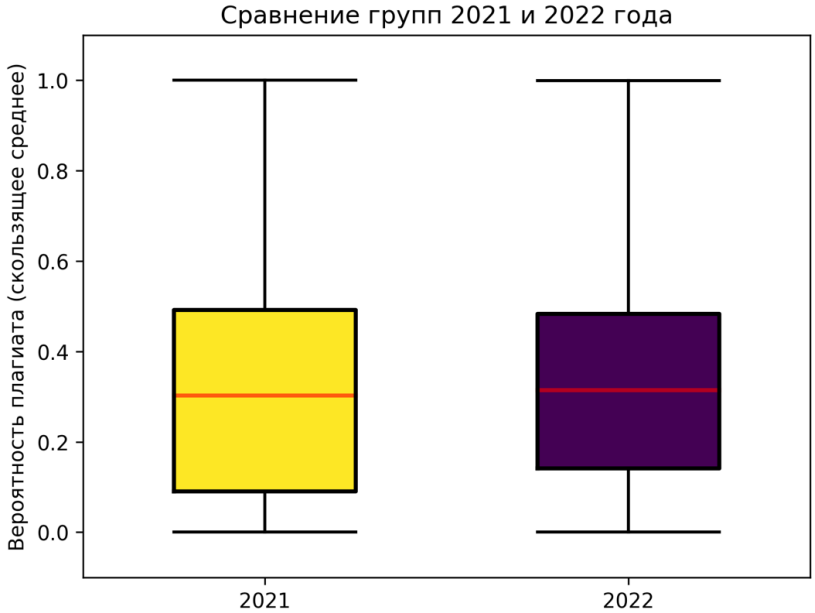


Рисунок 67 – Средний плагиат по всем контестам и всем студентам в группах, среднее, 20% и 80% квантили, минимум и максимум, при сравнении результатов 2021 и 2022 гг. Мехмат МГУ.

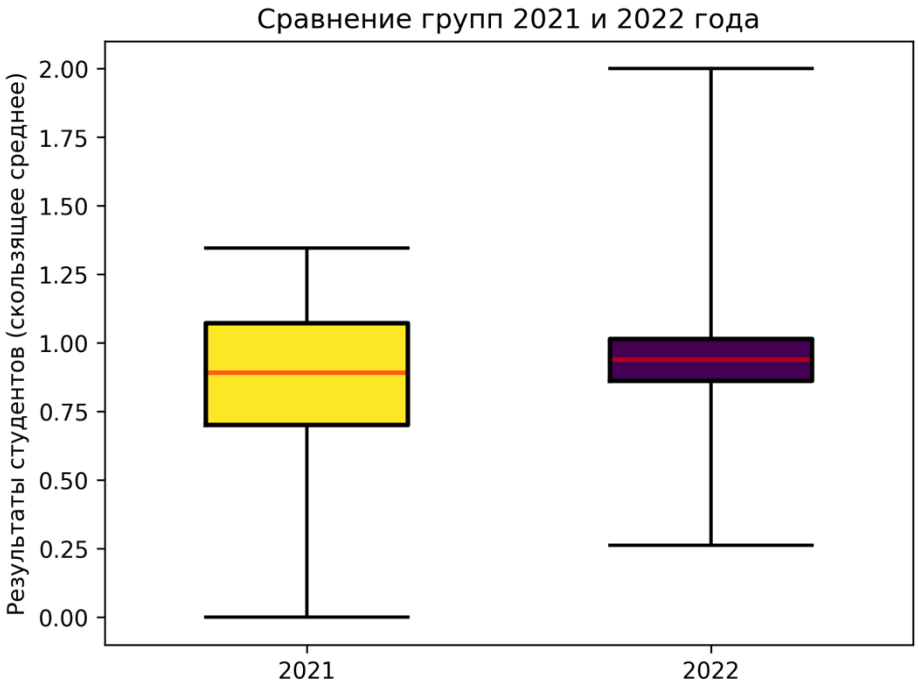


Рисунок 68 – Средний набранный процент баллов по всем контестам и всем студентам в группах, среднее, 20% и 80% квантили, минимум и максимум, сравнение 2021 и 2022 года, Мехмат МГУ.

Основываясь на реальных критериях «отличников», «хорошистов» и «двоечников», используемых в реальном образовательном процессе, из этих данных можно построить график динамики перехода студентов между этими классами. В данном критерии отличник – это студент, который присутствует практически на всех занятиях и самостоятельно выполняет все задания, двоечник – студент с маленькой посещаемостью, который или не выполняет задания, или выполняет их с большим процентом плагиата, хорошисты – остальные студенты, Рисунок 69, Рисунок 70. На следующих графиках по оси X – отсортированные по времени контесты, по оси Y – распределение студентов по категориям. На графиках можно увидеть ступени – это связано с вошедшими в середину курса студентами, например, вернувшимися из академического отпуска. Также можно увидеть 5 больших всплесков двоечников – это связано с тем, что к сессии и другим контрольным мероприятиям отличники и хорошисты в представленной модели преподавания уже получают зачет или положительную высокую экзаменационную

оценку , в связи с чем им нет смысла посещать зачетные мероприятия и модель в этот момент ошибочно предполагает, что данный студент является конкретно в этом контесте двоечником. Проанализировав графики можно увидеть, что в 2022 система оценивания каждого контеста была выбрана более консистентно и справедливо, а образовательная программа была построена так, что студенты получили более качественное, равномерное обучение по курсу, таким образом, основное количество студентов стало хорошистами, что подтверждает успешность педагогического эксперимента.

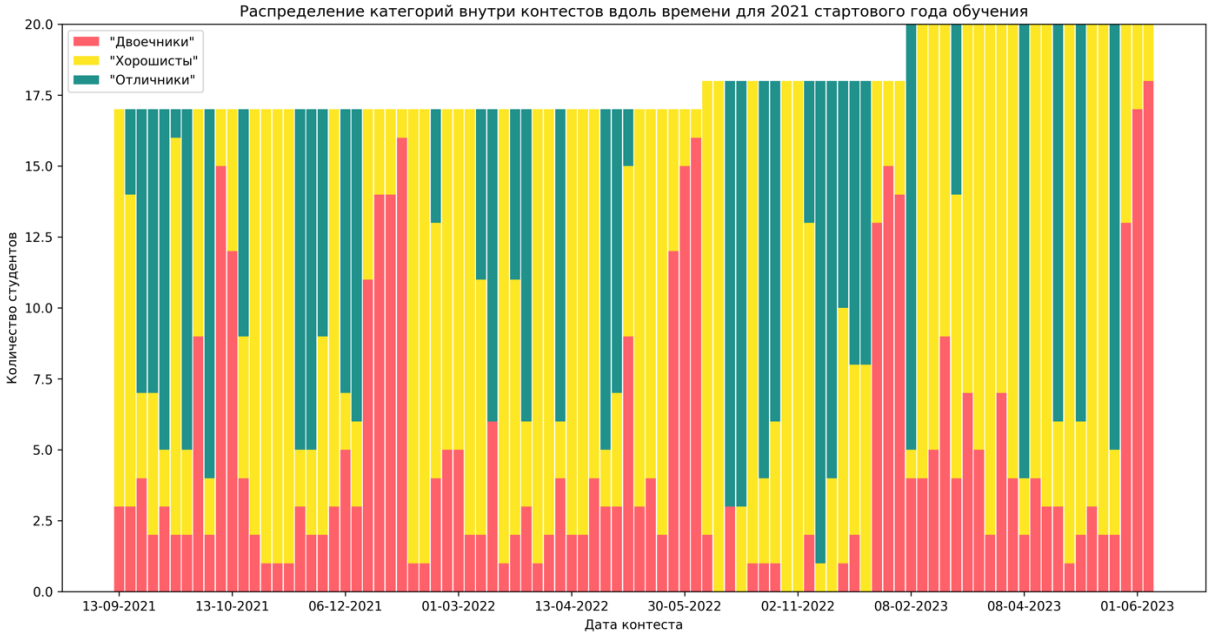


Рисунок 69 – Динамика распределения категорий студентов между «отличниками», «хорошистами» и «двоечниками», 2021 год, Мехмат МГУ.

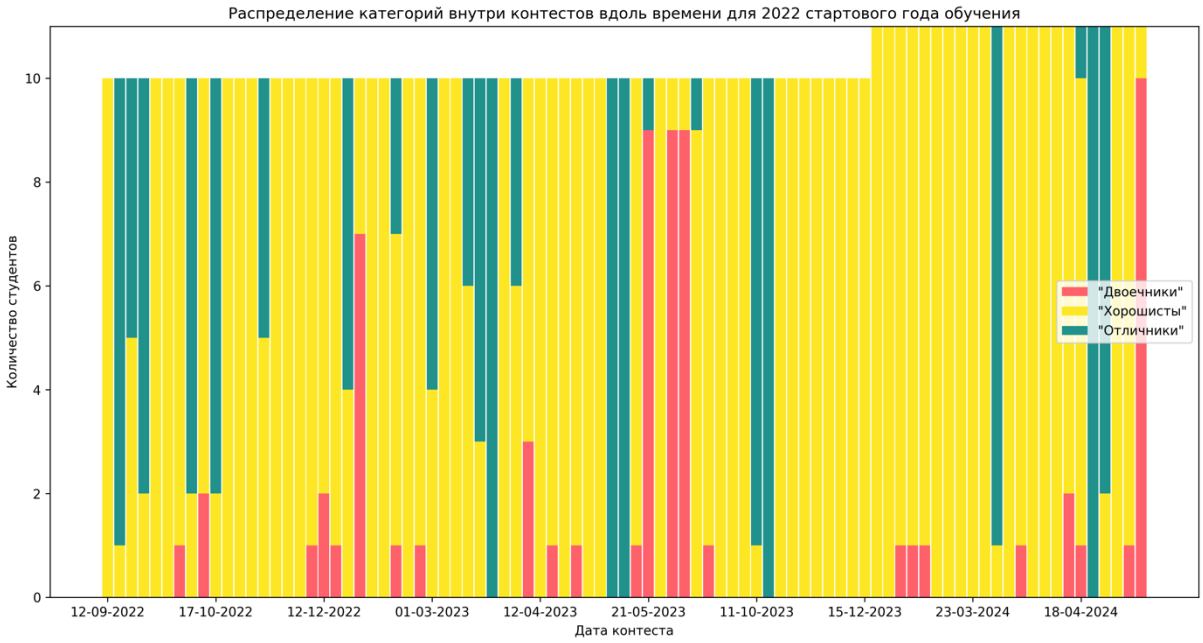


Рисунок 70 – Динамика распределения категорий студентов между «отличниками», «хорошистами» и «двоечниками», 2022 год, Мехмат МГУ.

Данный двухлетний курс проводился в ЦОП Мирера и другим, контрольным преподавателем, который только в 2022 году подключился к эксперименту. Следующие аналогичные графики посещаемости, Рисунок 71, плагиата, Рисунок 72, и процента набранных баллов, Рисунок 73, показывают, что одной платформы недостаточно для успешного обучения студентов, нужен набрать определенный опыт и сформировать у педагога необходимые компетенции ведения цифрового курса, чтобы, основываясь на данных, быстро анализировать и корректировать курс под конкретную группу. На графиках видно, что средняя посещаемость у второй группы хуже, средний плагиат выше и средняя успеваемость также хуже.

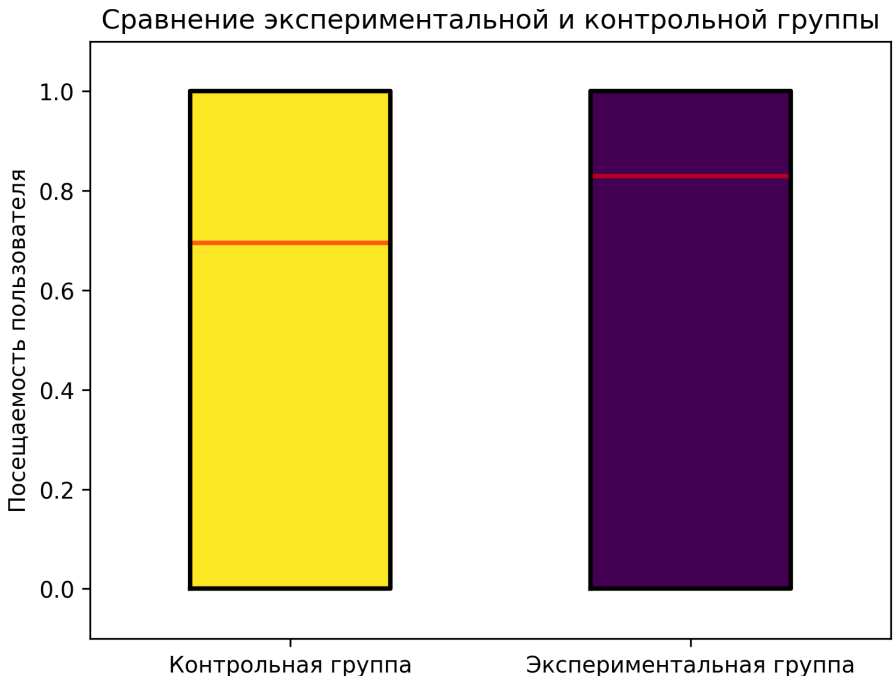


Рисунок 71 – Средняя посещаемость по всем контестам и всем студентам в группах, среднее, 20% и 80% квантили, минимум и максимум, сравнение экспериментальной группы и группы контрольного преподавателя, Мехмат МГУ, 2022 год.

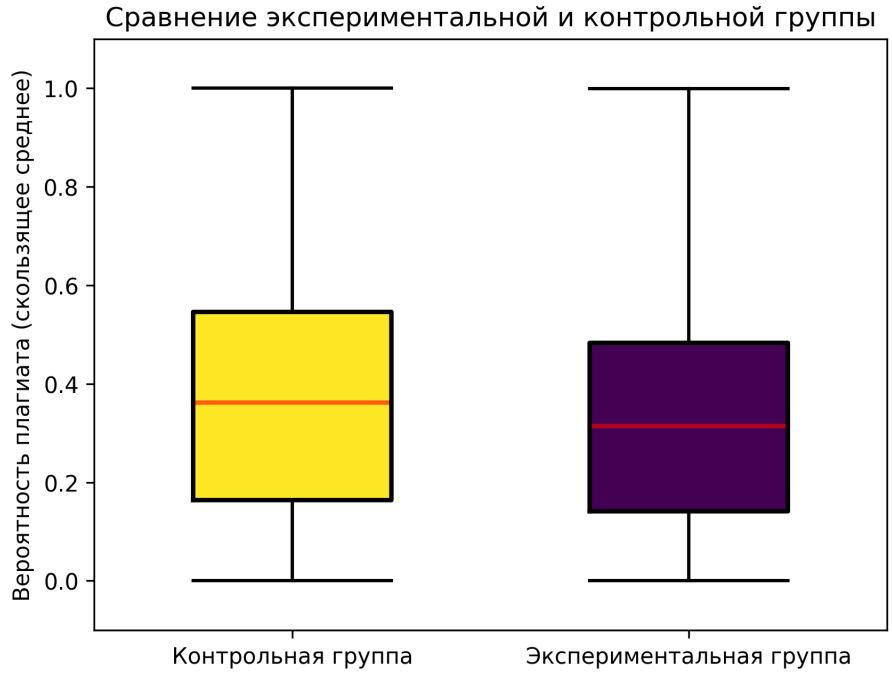


Рисунок 72 – Средний плагиат по всем контестам и всем студентам в группах, среднее, 20% и 80% квантили, минимум и максимум, сравнение экспериментальной группы и группы контрольного преподавателя, Мехмат МГУ, 2022 год.

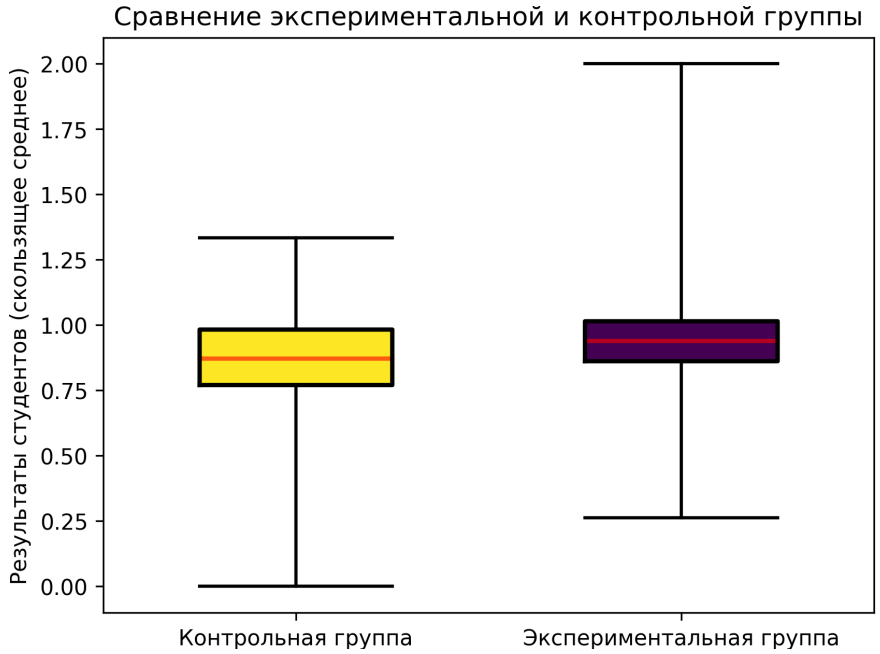


Рисунок 73 – Средний набранный процент баллов по всем контестам и всем студентам в группах, среднее, 20% и 80% квантили, минимум и максимум, сравнение экспериментальной группы и группы контрольного преподавателя, Мехмат МГУ, 2022 год.

5.5 Сигнатурные методы и генеративные модели в ЦОП Мирера

Пусть имеется результат наблюдения за одной или несколькими траекториями некоторого случайного процесса, который описывает изменение какой-либо величины, тогда возникает вопрос, как численно произвести достаточно большое число новых траекторий, которые были бы в некотором смысле “похожи” на наблюдаемый процесс? Для обучения моделей машинного обучения требуется выделить набор признаков, которые бы «хорошим» образом характеризовали эту траекторию, то есть наиболее полно описывали бы ее и отличали от других траекторий. Можно выделить приращения траектории по каждому аргументу, среднее, волатильность, и многое другое, однако на данный момент известно не так много подобных характеристик. В то же время, для обучения сложных моделей глубокого машинного обучения требуются тысячи таких характеристик.

Центральную роль в предложенном методе играет преобразование траектории случайного процесса, состоящее в вычислении его сигнатуры [302; 303]. Сигнатурой пути в *n*-мерном пространстве (то есть некоторой функции, аргументом которой является время – например, траектории случайного процесса или обучения студента), является семейство повторных интегралов от компонент этого пути по другим компонентам. Множество различных типов данных можно считать последовательно поступающими, или упорядоченными по времени, например финансовые временные ряды, тексты на естественных языках, траектория обучения студента.

На практике, поскольку повторные интегралы являются определенными, это является набором вещественных чисел, хорошо обобщающих и полноценно описывающих путь, так как сигнатуры содержат в себе полную информацию об его аналитических и геометрических свойствах [276]. Сигнатуры не зависят от сдвига или начальной точки, а также от репараметризации времени, сигнатура зависит только от геометрических свойств пути. Например, первый уровень сигнатуры является приращением пути по каждому аргументу, а второй уровень сигнатуры связан с так называемой площадью Леви. Таким образом, сигнатурный подход представляет собой непараметрический робастный способ извлечения характерных признаков из данных, которые впоследствии можно использовать для моделей машинного обучения.

В общем случае путь не определяется однозначно его сигнатурой. Например, можно видеть, что по сигнатуре нельзя восстановить точную скорость, с которой этот путь проходится (из-за инвариантности сигнатуры при репараметризации времени). Однако для непересекающихся путей по сигнатуре можно полностью определить все точки, через которые пройдёт путь, и порядок их обхода. При этом в различных приложениях оказывается, что для достаточно хорошей степени восстановления траектории по сигнатуре достаточно использовать лишь несколько первых повторных интегралов из всего этого бесконечного набора, что позволяет описывать траектории весьма экономным образом. Иными словами, сигнатуру

можно рассматривать как способ сжатия информации, содержащейся в траектории. Сигнатуры также позволяют получить еще больше элементов для обучения моделей машинного обучения, поскольку также легко вычисляются и для многомерных путей. Таким образом, можно увеличивать размерность изначального пути, изучая его зависимость от других путей.

Важное свойство сигнатуры состоит в том, что произведение двух её элементов может быть всегда представлено в виде суммы других элементов. Это свойство показывает, что члены сигнатуры не являются алгебраически независимыми. Однако для многих моделей машинного обучения для получения лучшего качества требуются независимые входные данные. Этой проблемы можно избежать, если вместо сигнатур рассматривать логарифмические сигнатуры, которые представляют из себя уже независимый набор признаков. Таким образом, используя логарифмические сигнатуры можно получить способ получения любого необходимого количества независимых признаков, полноценно описывающих геометрические свойства в том числе многомерного пути.

Для эффективного вычисления сигнатур можно использовать несколько подходов. Если известна аналитическая параметризация пути, и интегралы по нему считаются явно, то можно непосредственно аналитически их вычислить. Однако для множества реальных данных, которые являются результатом последовательных наблюдений за некоторыми явлениями, пути являются кусочно-линейными функциями и обладают неудобной для вычисления параметризацией. Эту проблему можно решить, воспользовавшись тождеством Чена. Это тождество позволяет, зная значения сигнатур на двух отрезках, легко вычислить сигнатуру на отрезке, являющимся их объединением. Таким образом, можно легко вычислить все сигнатуры на каждом линейном отрезке кусочно-линейной функции, которые обладают очень удобной параметризацией, а потом вычислить сигнатуру всего пути, по очереди присоединяя каждый следующий отрезок с помощью этого тождества.

Генерация новых данных, похожих по распределению на исходные, может быть необходима в сферах, где, например, требуется анонимизации данных, или когда количество данных заведомо мало из-за ограничения на число экспериментов. Также генеративные модели могут использоваться для тестирования стратегий, которое нельзя проводить на исторических данных во избежание переобучения.

Когда большой объем данных недоступен, генеративная модель, основанная на сигнатурах и использующая малое количество данных, может сгенерировать большое количество новых синтетических данных, статистически неотличимых от изначальных, которые впоследствии можно передать другим моделям, которым в свою очередь необходимо большее количество данных. Используя генеративные модели, основанных на машинном обучении, можно сгенерировать новые данные, похожие по распределению на исходные. Тогда не требуется знать распределение исходных данных, просто генерируются новые и сравниваются по распределению с исходными, что предлагается также делать с помощью сигнатурных методов.

Поскольку модель будет использовать сигнатуры в качестве входных данных, генерировать она будет тоже их. Если необходимо получить конкретные пути, то можно применить эволюционный алгоритм восстановления пути по его сигнатуре, который будет подбирать такой путь, чтобы сигнатура была наиболее похожа на изначальную. После того, как получатся сгенерированные пути, необходимо проверить, что они из того же распределения, что исходный набор. Для этого можно использовать различные тесты, например, с метрикой максимального среднего расхождения, который также будет использовать сигнатуры путей.

В последнее десятилетие сигнатурные методы находят применение в машинном обучении. Например, в сочетании со сверточными нейронными сетями завоевали первый приз в онлайн конкурсе ICDAR 2013 по распознаванию изолированных китайских символов [329]. В сочетании с моделью регрессии на градиентном бустинге выиграли первый приз в конкурсе PhysioNet 2019 по вычислительной технике в кардиологии [393]. Также сигнатурные методы нашли

применение в задачах финансовой математики, связанных с хеджированием производных инструментов [304].

5.6 Сигнатурные методы в применении к траекториям обучения студентов

Траектория обучения студента состоит из множества различных размерностей. Если смотреть это с точки зрения контестов в целом, то это его средний плагиат, количество попыток, набранные баллы, посещаемость. Эти данные распределены по времени контестов, что делает их временным рядом. Кроме этого, внутри каждого ряда есть временной ряд попыток, который также имеет данные о времени, которое студент потратил на эту попытку, ее статус, набранные баллы за нее, плагиат, процент совпадения с эталонным решением, и так далее. Получается, что внутри временного ряда контестов появляется временной ряд попыток, которые также являются важными характеристиками поведения студента в рамках этого контеста. Для того, чтобы закодировать эти данные высокой размерности для использования в моделях машинного обучения, можно использовать сигнатуры.

При экспериментировании с любой образовательной стратегией существует проблема, что на проведение исследований нужны годы, а студентов в группах недостаточное количество, чтобы накопить уверенный объем данных для машинного обучения. В этом случае могут помочь генеративные модели, основанные на сигнатурах, которые можно применить для генерации дополнительных образовательных траекторий студентов того же распределения, которые впоследствии можно использовать для нейронных сетей и других глубоких моделей машинного обучения.

Итоговой целью, по-прежнему, является предсказание будущей образовательной траектории студента, таким образом, можно проверять любые критерии о слушателе, например, станет ли он через месяц двоечником и нужна ли ему сейчас срочная помощь, а также автоматически создавать для студентов адаптивные траектории обучения.

Для этого надо зафиксировать окно от рассматриваемого момента времени, сколько последних контестов нужно учитывать для предсказания будущего. За месяц в среднем проводится 4 аудиторных занятия и одна большая домашняя работа, поэтому окно будет шириной в 5 контестов. Обучающие данные собираются следующим образом: берутся 5 контестов, строится временной ряд образовательной траектории студента, по нему считается сигнатура этого временного ряда. Следующее значение временного ряда считается ответом, и обучается регрессионная модель, предсказывающая по полученной сигнатуре окна временного ряда следующее значение. Важно, что весь временной ряд, в том числе ответ, является многомерным, и содержит в себе информацию о баллах, о плагиате, о посещаемости и так далее. Отличительной особенностью сигнатур является то, что они способны улавливать взаимные зависимости размерностей ряда, что особенно помогает в данном случае: например, уловить зависимость, что при падении посещаемости будет расти плагиат и падать баллы, или уловить зависимость, что при уменьшении количества попыток растет плагиат, и так далее.

На следующих графиках, Рисунок 74, Рисунок 75, Рисунок 76, представлены примеры работы данной модели, первые 5 контестов используются, как основа для дальнейших предсказаний. После этого рисуется следующая предсказанная точка, после чего окно сдвигается. В рамках данных графиков следующие точки при сдвиге окна берутся из реальных данных, что соответствует реальной действительности использования этой модели: это подсказка для преподавателя всегда в режиме онлайн во время проведения курса, поэтому у модели есть возможность использовать всегда актуальные данные. Если такой возможности нет – то можно использовать предсказанные данные как основу для следующих предсказаний, однако, конечно, при каждом последующем сдвиге окна будет накапливаться ошибка.

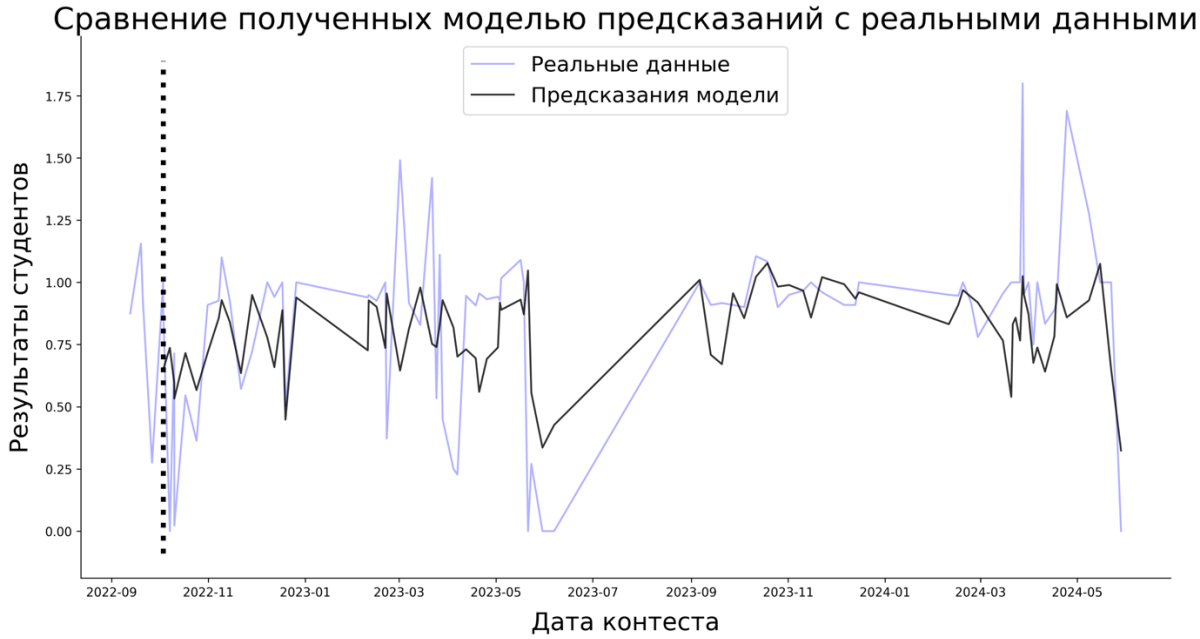


Рисунок 74 – Пример работы модели предсказания обучающей траектории студента с помощью сигнатур, вывод размерности временного ряда с баллами студента

Рисунок 75 – Пример работы модели предсказания обучающей траектории студента с помощью сигнатур, вывод размерности временного ряда с баллами студента

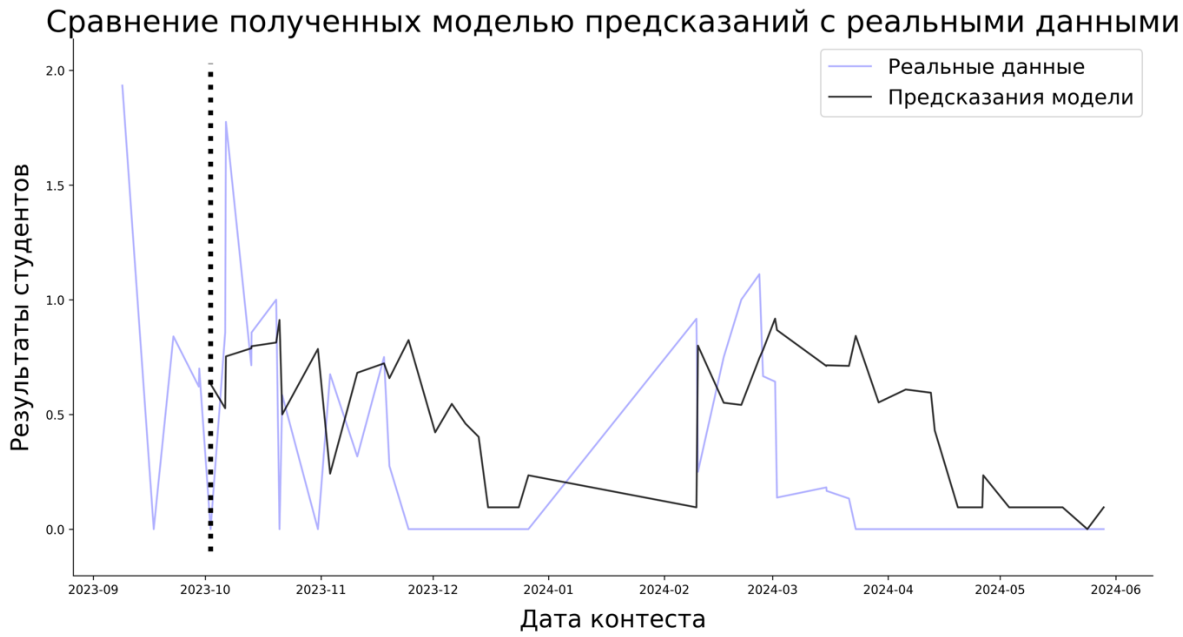


Рисунок 76 – Пример работы модели предсказания обучающей траектории студента с помощью сигнатур, вывод размерности временного ряда с баллами студента

Итоговыми метриками качества данной модели могут считаться не только стандартные метрики регрессионных моделей, но и насколько ее предсказания реально предвосхищают изменение класса студента. Однако, поскольку после предсказания модели преподаватель может изменить курс, тему, контест или изменить набор образовательных технологий, пытаясь избежать негативного результата обучения в будущем, то есть того, что предсказала модель, данная метрика качества некорректна для подсчета при валидации, то есть уже при реальном использовании модели. Тем не менее, на тестовой выборке модель показала более чем 80% качество предсказания будущей категории студентов, что позволило преподавателю в случае предсказанного изменения класса с хорошиста на худший класс неуспевающих студентов вовремя заметить этот момент и точечно помочь отстающим студентам.

Выводы главы 5

Для оценки эффективности методических систем обучения в формировании основ алгоритмического мышления у детей, начиная с четвертого года жизни, в рамках проводимых педагогических экспериментов использовались результаты достижений детей, рассматриваемые при участии в парциальных программах и программах дополнительного образования протяженностью от одного года до трех.

Доказана возможность формирования алгоритмического мышления в раннем возрасте у дошкольников и младших школьников при использовании пропедевтической методической системы обучения, базирующейся на ИКТ-насыщенных предметно-цифровых образовательных средах, как средствах обучения, наполненных необходимым содержанием и поддержанных соответствующими методиками, которые позволяют успешно усвоить систему научных понятий.

Проведенные исследования и педагогические эксперименты с использованием средств обучения – ЦОС ПиктоМир, ЦОС ПиктоМир-К, ЦОП Мирера и др. показали высокую эффективность интеграционного подхода формирования алгоритмического мышления при обучении информатике и программированию в ДОО, школах, вузах.

Разработаны и использованы в реальном образовательном процессе в интеграционном средстве обучения ЦОП Мирера методы предсказания результирующих достижений студентов по классам «отличник», «хорошист» и «двоечник». Результаты экспериментов показали, что при прохождении около 20% курса можно оценить будущую успешность студента, что позволяет определить приемы точечного педагогического воздействия, а также целесообразность автоматизации предпринимаемых преподавателем действий.

Разработаны и использованы в реальном образовательном процессе в интеграционном средстве обучения ЦОП Мирера генеративные модели, основанные на сигнатурах, используемые для генерации дополнительных образовательных траекторий студентов с целью локального предсказания будущей

образовательной траектории студента, что позволяет педагогу независимо управлять образовательным процессом выделенного студента, группой студентов, отнесенных к одному классу, студентами на курсе.

ЗАКЛЮЧЕНИЕ

В ходе исследования была подтверждены выдвинутые гипотезы, решены поставленные задачи педагогического исследования, обоснованы положения, выносимые на защиту, достигнута поставленная цель, получены следующие выводы и результаты.

1. Выделены основные составные части алгоритмического мышления, как исторического наследника операционного стиля мышления. Указано соотношение вычислительного и алгоритмического мышления. Предложена методическая система обучения формирования алгоритмического мышления, как практики освоения содержания обучения – определенного набора алгоритмических задач.
2. Сформулированы основные понятия программирования, которые являются фундаментом формирования алгоритмического мышления у детей дошкольного возраста и в пропедевтических методических системах обучения информатике и программированию.
3. Разработана и обоснована интеграционная методология поэтапного формирования алгоритмического мышления – методическая система обучения, с вариативным содержанием, ориентированным на различный возрастной контингент учащихся, включающая практические методы с большим объемом самостоятельной работы, использующая цифровые и предметно-цифровые ИКТ-насыщенные средства обучения и различные формы, ориентированные на возраст и начальный уровень компетенции учеников, существенно повышающая эффективность систематического освоения информатики и программирования.
4. Исследована методика и средства преподавания информатики и программирования в историческом контексте, что показало, что использование текстовых учебных языков программирования с национальной лексикой положительно влияет на эффективность образовательного процесса, на формирование алгоритмического мышления в рамках национальных культур и ценностей народов, что обеспечивается в том числе близостью учебного языка к естественному языку.

5. Как компонент методической системы обучения, разработана методология внедрения раннего пропедевтического обучения информатике и программированию, пропедевтика парадигм программирования, состоящая в использовании уникального дидактического материала, включая не только последовательный набор задач и методические подходы, но и, как стержневой элемент, средство обучения – спроектированную и разработанную предметно-цифровую образовательную среда ПиктоМир, в которую скомпонован указанный набор задач на основании сформулированных в настоящей работе критериев для поэтапного формирования основ алгоритмического мышления.

6. Методология формирования алгоритмического мышления включает в себя получение знаний, освоение определенных навыков, формирование умений при прохождении уровней образования: уровень пропедевтики – знакомство дошкольников и младших школьников с основами алгоритмизации, уровень генерализации знаний – обучение школьников программированию, и уровень профессионализации (профильное обучение) – высшее образование, когда студенты изучают информатику и программирование в разрезе специальностей и различных предметов. Все три ступени объединены ориентированными на возраст и уровень компетенции методическими системами обучения, которые позволяют выстроить бесшовный непрерывный образовательный процесс и используют вариативные, ориентированные на возраст, методики и формы обучения, базирующиеся на средствах обучения включающих практику освоения программирования на пиктографическом учебном языке для пропедевтики (встроенного в цифровую образовательную среду ЦОС ПиктоМир). Для уровней генерализации и профессионализации (профильный уровень) переход к текстовым, учебным и производственным языкам программирования с использованием ступенчатой методики на базе средств обучения – тройки цифровых образовательных сред, включающих пиктографическую, блочную и текстовую среду программирования (ЦОС ПиктоМир, ЦОС ПиктоМир-К, ЦОС КуМир).

7. Содержание обучения составляет универсальный, доступный любому возрасту обширный набор заданий для изучения основ информатики и программирования, который организует практическую подготовку дошкольников, школьников и студентов с использованием, как средств обучения, ИКТ-насыщенных цифровых образовательных сред с автоматизированной верификацией уровня освоения требуемых компетенций. Для интенсификации образовательного процесса и внедрения методики обучения, использующей элементы индивидуализации, которые основываются на элементах искусственного интеллекта, необходимо в качестве средства обучения использовать интеграционную цифровую образовательную платформу (ЦОП Мирера), где выполнение заданий обучаемыми оценивается не только по результатам испытаний, но и по процессу выполнения задач, контролирующему из омниканальной цифровой образовательной платформы, использующей искусственные нейронные сети.

8. Использование, как средств обучения, профильной методической системы обучения ЦОС ПиктоМир совместно с цифровой образовательной платформой Мирера на начальном курсе алгоритмики и программирования позволило увеличить объем практики студентов педагогических вузов с единиц до нескольких сотен за семестр. Освоенная практика будущих преподавателей курса информатики и ИКТ позволила интенсифицировать и индивидуализировать подготовку молодых специалистов для современной цифровой реальности.

9. Разработаны и имплементированы в средство обучения ЦОП Мирера интерактивные методы анализа и предсказания на основе генеративных моделей и сигнатур, используемых для формирования дополнительных данных, с целью управления образовательными траекториями студентов, а также позволяющие педагогу независимо управлять образовательным процессом выделенного студента, группой студентов, отнесенных к одному классу, а также всеми студентами на курсе.

10. Экспериментально подтверждена результативность педагогического проектирования. Полученные результаты и научно-методический опыт могут быть тиражированы в системе национального образования всех уровней.

Проведенное исследование позволило получить ответы на поставленные вопросы теоретического обоснования. Исследование подтвердило гипотезу о том, что, как содержание методической системы обучения, включающей в качестве средств обучения ИКТ-насыщенные цифровые образовательные среды и платформы, существует набор заданий и понятий, успешное поэтапное освоение которых приводит к формированию алгоритмического мышления у обучаемых. Такой набор заданий и понятий является универсальным для успешного формирования основ алгоритмического мышления на всех ступенях обучения (у студентов вузов, включая будущих учителей информатики педуниверситетов, школьников и у дошкольников). Возможно понижение границ сензитивного периода первичного знакомства с основами программирования для детей 4-ого года жизни при использовании методической системы обучения с предметно-цифровыми образовательными средами и платформами для обучения информатике и программированию в качестве средств и специальных форм обучения.

Применение методической системы обучения для поэтапного формирования алгоритмического мышления в приложении к созданию содержания, методики и форм обучения в пропедевтических разделах курса информатики по основам алгоритмизации и программирования, способствует повышению эффективности образовательного процесса и достижению учебно-воспитательных целей.

Вместе с тем проведенное исследование обозначило ряд новых вопросов и проблем, которые заслуживают специального рассмотрения. Необходимо продолжать исследование соотношения методологии формирования алгоритмического мышления индивидуума и использования генеративных искусственных нейронных сетей в образовательном процессе.

ЛИТЕРАТУРА

1. Авдулова Т.П. Психология игры: Учебник для академического бакалавриата / Т.П. Авдулова. – 2-е изд., испр. и доп. – М.: Юрайт, 2018. – 232 с.

2. Автоматизация проверки семантической составляющей текстовых ответов обучающихся в цифровой образовательной платформе / А.Г. Леонов, Н.С. Мартынов, К.А. Мащенко, А.А. Холькина, А.В. Шляхов // Программные продукты и системы. – 2024. – Т. 37, № 3. – С. 197-206. – DOI: 10.15827/0236-235X.142.197-206

3. Алгебра: Учебное пособие для средних школ с математической специализацией / Н.Я. Виленкин, Р.С. Гутер, С.И. Шварцбурд, Б.В. Овчинский, В.Г. Ашкинузе. – М.: Просвещение, 1968. – 338 с.

4. Алтухова С.О. Формирование вычислительного мышления на основе составления алгоритмов решения задач / С.О. Алтухова, З.А. Кононова // Мир науки, культуры, образования. – 2021. – № 5(90). – С. 60-62. – DOI 10.24412/1991-5497-2021-590-60-62.

5. Анализ методов проверки кода программ на плагиат в цифровой образовательной платформе Мирера / М.С. Дьяченко, В.А. Домрина, А.Г. Леонов, К.А. Мащенко, И.Г. Райко, А.А. Холькина // Труды НИИСИ РАН. – 2022. – Т. 12, № 3. – С. 26-33.

6. Асмолов А.Г. Персонализация образования и антропология будущего / А.Г. Асмолов // Народное образование. – 2021. – №3(1486). – С. 75-82.

7. Бахвалов Н.С. Практикум по программированию / Бахвалов Н.С., Бетелин А.Б., Бетелин В.Б.; Под ред. Н.С. Бахвалова, А.В. Михалёва. – М.: МГУ, 1986. – 208 с.

8. Белозёрова С.И. Опыт применения LMS Moodle для создания и сопровождения учебных курсов / С.И. Белозёрова, О.И. Чуйко // Современные проблемы науки и образования. – 2019. – № 1. – С.78.

9. Бененсон Е.П. Информатика и ИКТ. 3 кл.: Учебник-тетрадь в 2-х частях / Е.П. Бененсон, А.Г. Паутова. – М.: Академкнига, 2007. – 75 с.

10. Берг А.И. О возможностях автоматизации управления народным хозяйством / А.И. Берг, А.И. Китов, А.А. Ляпунов // Проблемы кибернетики: Сборник статей. Выпуск 6. – М.: Физматгиз, 1961. – С. 83-100.

11. Бесшапошников Н.О. Цифровизация образования – Новые возможности управления образовательными треками / Н.О. Бесшапошников, А.Г. Леонов, А.А. Прилипко // Вестник Кибернетики. – 2018. – № 2. – С. 157-163.

12. Бетелин В.Б. Основные понятия программирования в изложении для дошкольников / В.Б. Бетелин, А.Г. Кушниренко, А.Г. Леонов // Информатика и ее приложения. – 2020. – Т. 14, № 3. – С. 56-62. – DOI: 10.14357/19922264200308

13. Боброва Т. Сервис онлайн-курсов Coursera разрешил любым университетам использовать курсы с платформы в своих программах / Т. Боброва // Vc.ru: Платформа для предпринимателей и высококвалифицированных специалистов малых, средних и крупных компаний: [Электронный ресурс]. – URL: https://vc.ru/services/86441-servis-onlayn-kursov-coursera-razreshil-lyubym-universitetam-ispolzovat-kursy-s-platformy-v-svoih-programmah (Дата обращения 17.08.2024)

14. Босова Л.Л. Программирование как инструмент формирования вычислительного мышления обучающихся. / Л.Л. Босова // Информатика в школе. – 2020. – №10. – С. 4-10. – DOI: 10.32517/2221-1993-2020-19-10-4-10

15. Булденко С.В. Изучение изменения выраженности эгоцентризма в детском возрасте в рамках теории Ж. Пиаже / С.В. Булденко, Т.Г. Волкова // Психология и педагогика: Методика и проблемы практического применения. – 2010. – №16-1. – С.4-7.

16. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++ = Object-Oriented Analysis and Design with Applications / Г. Буч; Пер. с англ. под ред. И. Романовского, Ф. Андреева. – 2-е изд. – М..: Бином, 1998. – 558 с.

17. В.П. Зинченко. Сознание и творческий акт, 2010. Автор: Языки славянских культур, 2010 588 с.

18. Вайнштейн Ю.В. Модель образовательного контента: от структурирования понятий к адаптивному обучению / Ю.В. Вайнштейн, Р.В. Есин, Г.М. Цибульский // Открытое образование. – 2021. – Т. 25, № 1. – С. 28–39. – DOI: 10.21686/1818-4243-2021-1-4-28-39

19. Варсанофьев Д.В. Е-практикум – программное обеспечение школьного курса информатики и вычислительной техники / Д.В. Варсанофьев, А.Г. Кушниренко, Г.В. Лебедев // Микропроцессорные средства и системы. – 1985. – № 3. – С. 27-33.

20. Визуализация алгоритмов реализации взаимоисключений средствами пиктограммной среды программирования / И.Н. Грибанова, А.С. Караваева, А.А. Леонов [и др.] // Труды НИИСИ РАН. Математическое и компьютерное моделирование сложных систем: теоретические и прикладные аспекты. – 2022. – Т. 12, № 3. – С. 39-62.

21. Вилкова К. А. Адаптивное обучение в высшем образовании: За и против / К.А. Вилкова, Д.В. Лебедев. – М.: НИУ ВШЭ, 2020. – 36 с.

22. Вирт Н. Алгоритмы + структуры данных = программы / Н. Вирт. – М.: Мир, 1985. – 406 с.

23. Вирт Н. Паскаль. Руководство для пользователя и описание языка / Н. Вирт, К. Йенсен. – М.: Финансы и статистика, 1982. – 151 с.

24. Вирт Н. Программирование на языке Модула-2 / Н. Вирт. – М.: Мир, 1987. – 202 с.

25. Волошина О.А. Создание алфавитов для бесписьменных народов / О.А. Волошина // Первое сентября. Русский язык. – 2009. – №14 (590). – С.10-17.

26. Вопросы психологии ребенка дошкольного возраста: Сборник статей / Под ред. А.Н. Леонтьева, А.В. Запорожца. – М.: Международный образовательный и психологический колледж, 1995. – 44 с.

27. Выготский Л.С. Педагогическая психология / Л.С. Выготский. – М.: Педагогика-Пресс, 2012. – 536 с.

28. Выготский Л.С. Игра и ее роль в психическом развитии ребенка: Стенограмма лекции, прочитанной в 1933 г. в ЛГПИ им. А.И. Герцена // Психология развития ребенка / Л.С. Выготский. — М: Смысл; Эксмо, 2004 – с. 200-233. – (Библиотека всемирной психологии).

29. Выготский Л. С. Вопросы детской (возрастной) психологии // Собрание сочинений: В 6 т. Т.4. Детская психология / Под ред. Д.Б. Эльконина. – М.: Педагогика, 1983. – с. 243-385.

30. Выготский Л.С. Мышление и речь / Л.С. Выготский. – Изд. 5, испр. – М.: Лабиринт, 1999. – 352 с.

31. Выготский Л.С. Психология развития человека / Л.С. Выготский. — М.: Смысл; Эксмо, 2005. – 1136 с.

32. Гальперин П.Я. Психология мышления и поэтапного формирования умственных действий / П. Гальперин // Исследования мышления в советской психологии. – М.: Просвещение, 1966. – 179 с.

33. Гатулин Р.Р. Использование мессенджера Telegram для реализации технологии электронного обучения в вузе / Р.Р. Гатулин, Д.А. Колупаева // Санкт-Петербургский образовательный вестник. – 2017. – №11-12. – С.15-16.

34. Гейн А.Г. Информатика и информационные технологии: учеб. для 8 кл. общеобразоват. учреждений / А.Г. Гейн, А.И. Сенокосов, Н.А. Юнерман. – М: Просвещение, 2009. – 175 с.

35. Гейн А.Г. Методика изучения алгоритмизации с помощью учебных исполнителей: Книга для учителя: [Электронный ресурс] / А.Г. Гейн. – Екатеринбург, 2011. – URL: http://kadm.kmath.ru/files/metodica_algo.docx

36. Гейн А.Г. Основы информатики и вычислительной техники / А.Г. Гейн, В.Г. Житомирский. – Свердловск: Издательство Урал. ун-та, 1989. – 272 с.

37. Геометрия: Учебное пособие для 6-8 классов средней школы / Под ред. А.Н. Колмогорова. – М.: Просвещение, 1979. – 383 с.

38. Гимранов Р.Д. Изобретая информационные системы будущего. Теория и практика / Р.Д. Гимранов, И.Н. Холкин. – Сургут: Аэроплан СОФТ, 2017. – 190 с.

39. Глушков В.М. Макроэкономические модели и принципы построения ОГАС / В.М. Глушков. – М.: Статистика, 1975. – 160 с. – (Методы оптимальных решений).

40. Говоров А.В. Проект базового закона об образовании в контексте проблемы пересмотра конституции Японии / А.В. Говоров // Ежегодник Япония. – 2007. – Т. 36. – С. 27-47.

41. Говорова Н.В. Бедность и неравенство в Европейском Союзе // Современная Европа. – 2016. – №3. – С. 104-113.

42. Гоноскова О.Б. Возрастная периодизация и педагогическая система Марии Монтессори // Психолого-педагогический журнал. – 2013. – № 4(28). – С. 183-191.

43. Гоноскова О.Б. Реализация идей М. Монтессори в высшей школе // Человеческий капитал. – 2016. – № 9 (93). – С. 88- 92.

44. Горбатов С.В. Цифровой след как механизм индивидуализации образовательной траектории студента: (На примере курса «Цифровые технологии самообразования») / С.В. Горбатов, Е.А. Краснова // Перспективы науки и образования. – 2022. – №4(58). – С.193-208. – DOI: 10.32744/pse.2022.4.12

45. Горелик А.М. Эволюция языка программирования Фортран (1957-2007) и перспективы его развития / А.М. Горелик // Вычислительные методы и программирование. – 2008. – № 9(2). – С. 53–71.

46. Григорьев С.Г. Информатизация образования. Фундаментальные основы и практические приложения: Учебник для студентов педагогических вузов и слушателей системы повышения квалификации педагогов / С.Г. Григорьев, В.В. Гриншкун. – Воронеж: Научная книга, 2014. – 232 с.

47. Григорьев С.Г. Конфигурация для Пролога Д / С.Г. Григорьев // Информатика и образование. – 1992. – №5-6. – С. 53-54.

48. Григорьев С.Г. Математическая модель учебного транслятора языка Пролог-Д / С.Г. Григорьев // Компьютерное моделирование в системе образования: Материалы международного коллоквиума по проблемам компьютерного моделирования в системе образования. – М.: МИАН,1991. – С.79-88.

49. Григорьев С.Г. Программирование на Пролог-Д / С.Г. Григорьев // Информатика и образование. – 1990. – №5. – С.50-57.

50. Григорьев С.Г. Работа системы Пролог-Д / С.Г. Григорьев // Информатика и образование. – 1990. – №4. – С.50-57.

51. Гриншкун В.В. Школьная информатика в контексте фундаментализации образования / В.В. Гриншкун, И.В. Левченко // Вестник РУДН. – 2009. – №1. – С.55-63.

52. Дмитриева В.Г. Методика раннего развития Марии Монтессори: От 6 месяцев до 6 лет / В.Г. Дмитриева. – М.: Эксмо, 2011. – 224 с. – (Психология. Искусство быть родителем. Советуют профессионалы).

53. Дошкольное образование: Федеральный государственный образовательный стандарт: Утверждено Приказом Минобрнауки России от 17.10.2013 № 1155: (Ред. от 21.01.2019) // ФГОС / Национальная ассоциация развития образования и науки. – URL: https://fgos.ru/fgos/fgos-do/ (Accessed: 31.07.2024)

54. Дуванов А.А. Азбука Роботлании: Курс информатики для малышей в алгоритмическом изложении / А.А. Дуванов, Н.Д. Шумилина // Вестник Ярославского педагогического университета им. К.Д. Ушинского. – 2014. – Т.8, № 1. – С. 49-51.

55. Дуванов А.А. Методические и программные средства дистанционного обучения школьников и учителей / А.А. Дуванов, Ю.А. Первин // Информационные технологии в образовании: VIII международная конференция-выставка: 3-6 ноября 1998 г.: (ИТО'98/99): Материалы. – М.: МИФИ, 1998. – С.11-30.

56. Дуванов Ю.А. Система программных исполнителей Роботландия в раннем обучении информатике / Ю.А. Дуванов, Ю.А. Первин // Европейский конгресс по применению компьютеров в обучении: Материалы. – Лейпциг, ГДР: Академиздатцентр "Наука", 1987. – С. 15-31.

57. Дюк Гониназ М. Славянское влияние в эсперанто / М. Дюк Гониназ // Проблемы международного вспомогательного языка. – М.: Наука, 1991. – С. 113-116.

58. Единый государственный экзамен по информатике: (Демонстрационный вариант, Кодификатор, Спецификация): [Электронный ресурс] // ФГБНУ Федеральный институт педагогических измерений: [Офиц. сайт]. – URL: https://doc.fipi.ru/ege/demoversii-specifikacii-kodifikatory/2024/inf_11_2024.zip (Дата обращения: 20.08.2024)

59. Ежегодная пресс-конференция Владимира Путина: 17 декабря 2020 года // Президент России: [Офиц. сайт]. – URL: http://www.kremlin.ru/events/president/news/64671 (Дата обращения: 20.08.2024)

60. Ершов А.П. Алгоритмический язык в школьном курсе основ информатики и вычислительной техники / А.П. Ершов // Микропроцессорные средства и системы. – 1985. – №2. – С.48-50.

61. Ершов А.П. Вычислительное дело в США : По материалам поездки в США на III Конгресс IFIP 25-29/V-65 г.: Отчет / А.П. Ершов. – М., 1966. – 338 с.

62. Ершов А.П. Об объектно-ориентированном взаимодействии с ЭВМ / А.П. Ершов // Микропроцессорные средства и системы. – 1985. – №3. – С. 2-5.

63. Ершов А.П. О предмете информатики / А.П. Ершов // Вестник АН СССР. – 1984. – №2. – С. 112-114.

64. Ершов А.П. Основы информатики и вычислительной техники: Пробное учебное пособие для средних учебных заведений: В 2 ч. Ч.1 / А.П. Ершов, В.М. Монахов. – М.: Просвещение, 1985. – 96 с.

65. Ершов А.П. О формализации понятия программы / А.П. Ершов, А.А.nЛяпунов // Кибернетика. – 1967. – № 5. – С. 40–57.

66. Ершов А.П. О человеческом и эстетическом факторах в программировании / А.П. Ершов // Кибернетика. – 1972. – №5. – С. 94-99.

67. Ершов А.П. Программирование – вторая грамотность / А.П. Ершов. – Новосибирск: ВЦ СО АН СССР, 1981. – 18 с.

68. Ершов А.П . Урок 1: правила записи предписаний на языке Робик. Урок 2: гибкие системы предписаний, синтаксические диаграммы и переменные поля / А.П. Ершов, Г.А. Звенигородский // Квант. – 1979. – №9. – С. 21-26.

69. Ершов А.П. Учитель / А.П. Ершов // Очерки по истории информатики в России / Ред.-сост. Д.А. Поспелов, Я.И. Фет. – Новосибирск: Науч.-изд. центр ОИГГМ СО РАН, 1998. – С. 193-197.

70. Житин Д.В. Этнотерриториальные особенности социального неравенства в США / Д.В. Житин, А.Д. Прокофьев // Вестник Санкт-Петербургского университета. Науки о Земле. – 2022. – №67 (2). – С. 333-359.

71. Запорожец А.В. Избранные психологические труды: В 2 т. Т. I. Психическое развитие ребенка / А.В. Запорожец. – М.: Педагогика, 1986. – 315 с.

72. Запорожец А.В. Проблемы дошкольной игры и руководства ею в воспитательных целях / А.В. Запорожец // Игра и ее роль в развитии ребенка дошкольного возраста: Сборник научных трудов / Науч. ред. А.В. Запорожец, Т.А. Маркова. – М.: НИИ общей педагогики, 1978. – 155 с.

73. Звенигородский Г.А. Основные операторы учебно-производственного языка Рапира / Г.А. Звенигородский // Квант. – 1985. – Т. 10. – С. 52-55.

74. Звенигородский Г.А. Первые уроки программирования / Г.А. Звенигородский. – М.: Наука, Библиотечка «Кванта», 1985. – 208 с.

75. Звенигородский Г.А. Школьная информатика: (Концепции, состояние, перспективы) / Г.А. Звенигородский, Ю.А. Первин, А.П. Ершов. – Новосибирск: ВЦ СО АН СССР, 1979. – 51 с.

76. Звонкин А.К. Информатика: Алгоритмика: Учебник для 6 класса / А.К. Звонкин, С.К. Ландо, А.Л. Семенов. – М.: Просвещение, 2006. – 237 с.

77. Йенсен К. Паскаль: Руководство для пользователя и описание языка / К. Йенсен, Н. Вирт; Пер. с англ. Д.Б. Подшивалова. – М.: Финансы и статистика, 1982. – 151 с.

78. Иванова Н.Я. О строении сюжета детских игр / Н.Я. Иванова // Дошкольное воспитание. – 1965. – № 5. – С.18-23.

79. Избранные педагогические сочинения: В 2 т. / Под ред В.Я. Струминского. – М.: Учпедгиз, 1953-1954. – (Бибилиотека учителя).

80. Изучение основ информатики и вычислительной техники: методическое пособие: В 2-х ч. Ч.1. 9 класс. Ч.2. 10 класс / А. П. Ершов, [и др.]; Под ред. А.П. Ершова, В.М. Монахова. – М.: Просвещение, 1986. – 192 с, 176 с.

81. Иноземцев В.Л. Социология Д. Белла и контуры современной постиндустриальной цивилизации / В.Л. Иноземцев // Вопросы философии. –2002. – № 5. – С.138-144.

82. Информатика: 7-9 класс / А. Г. Кушниренко, Я. Н. Зайдельман, В. В. Тарасова, А. Г. Леонов // Свободное программное обеспечение в высшей школе: Двенадцатая конференция: Тезисы конференции. – М.: BaseAlt, 2017. – С. 41-47.

83. Информатика и ИКТ: учебник для 3 класса / Н.В. Матвеева, Е.Н. Челак, Н.К. Конопатова, Л.П. Панкратова. – М.: БИНОМ Лаборатория знаний, 2009. – 235 с.

84. Информационная культура. Кодирование информации. Информационные модели. 9-10 классы: Учебник для общеобразовательных учреждений / А.Г. Леонов, М.Г. Эпиктетов, В.В. Борисенко, М.А. Кузьменко, Б.А. Назаров, С.Б. Ханжин, А.Г. Кушниренко. – М.: Дрофа, 1996. – 205 с.

85. Использование элементов искусственного интеллекта в виртуальных помощниках преподавателя / Н.О. Бесшапошников, М.С. Дьяченко, А.Г. Леонов, К.А. Мащенко // Успехи кибернетики. – 2020. – Т. 1, № 3. – С. 15-22. – DOI: 10.51790/2712-9942-2020-1-3-2

86. Карьера в информационных технологиях / Отв. ред. М. Шинкарук. – М.: Аванта+, 2003. – 365 с.

87. Каталог профессий // Атлас новых профессий [Электронный ресурс] / Creative Commons Attribution 4.0 International. – URL:https://atlas100.ru/catalog/ (Дата обращения: 24.08.2024)

88. Кириенко Д.П. Изучение алгоритмизации с использованием исполнителей и автоматического тестирования / Д.П. Кириенко // Всероссийский съезд учителей информатики. Москва. МГУ имени М.В. Ломоносова, 24-26 марта 2011г.: Тезисы докладов. – М.: МГУ, 2011. – С. 486-487.

89. Кириенко Д.П. Курс алгоритмизации с использованием исполнителей системы КуМир и автоматического тестирования: 2010-2011 гг. / Д.П. Кириенко // Портал школы 179: [Электронный ресурс]. – URL: http://server.179.ru/wiki/?page=DenisKirienko/Kumir (Дата обращения: 20.08.2024)

90. Кожевников В.А. Система автоматической проверки ответов на открытые вопросы на русском языке / В.А. Кожевников, О.Ю. Сабинин // Информатика, телекоммуникации и управление. – 2018. – №3. – С. 57-72. – DOI: 10.18721/JCSTCS.11306

91. Коменский Я.А. Избранные педагогические сочинения: Т. 1: Великая дидактика / Я.А. Коменский; Под ред. А.А. Красновского. – М.: Государственное учебно-педагогическое издательство НАРКОМПРОСА РСФСР, 1939. – 320 с. – (Педагогическая библиотека) .

92. Комлева Н.В. Цифровая платформа для создания персонализированных адаптивных онлайн курсов / Н.В. Комлева, Д.А. Вилявин // Открытое образование. – 2020. – Т. 24, № 2. – С. 65–72. – DOI: 10.21686/1818-4243-2020-2-65-72

93. Конференция по искусственному интеллекту 24 ноября 2022 г. // Президент России: [Офиц. сайт]. – URL: http://www.kremlin.ru/events/president/news/69927 (Дата обращения: 13.08.2024).

94. Концепция государственной языковой политики Российской Федерации: Утверждено Распоряжением Правительства Российской Федерации от 17.08.2024 № 1481-р // Официальное опубликование нормативных актов. – URL:

http://publication.pravo.gov.ru/document/0001202406140048?index=1 (Дата обращения: 20.08.2024)

95. Концепция информатизации образования / Б.Е. Алгинин, Б.Г. Киселев, С.К. Ландо [и др.] // Информатика и образование. – 1990. – № 1. – С. 3-9.

96. Кречетов И.А. Раскрываем потенциал адаптивного обучения: от разработки до внедрения / И.А. Кречетов, М.Ю. Дорофеева, А.В. Дегтярев // Elearning stakeholders and researchers: Москва, 05–06 декабря 2018 года: Материалы международной конференции. – Москва: Национальный исследовательский университет “Высшая школа экономики”, 2018. – С. 76-88.

97. Кречетов И.А. Реализация методов адаптивного обучения / И.А. Кречетов, В.В. Романенко // Вопросы образования. – 2020. – № 2. – С. 252-277.

98. КуМир: Система программирования: Стартовая страница отечественной цифровой образовательной среды «КуМир»: [Электронный ресурс]. – URL: https://www.niisi.ru/kumir/ (Accessed: 31.07.2024)

99. Кузнецов А.А. Современный курс информатики: От концепции к содержанию / А.А. Кузнецов, С.А. Бешенков, Е.А. Ракитина // Информатика и образование. – 2004. – №2. – С. 2-6.

100. Кузнецов А. Методическая система обучения ОИВТ: Структура и функции, состояние и перспективы / А. Кузнецов, В. Долматов // Информатика и образование. – 1989. – №1. – С. 3-8.

101. Курашева А. Доля маркетплейсов с внедренными чат-ботами достигла 75% / А. Курашева // Ведомости. Технологии. – 2023. – 21 июня. – URL: https://www.vedomosti.ru/technology/articles/2023/06/21/981513-dolya-marketpleisov-s-vnedrennimi-chat-botami (Дата обращения: 19.08.2024)

102. Курмангалиев А.Ч. Компьютерное мышление / А.Ч. Курмангалиев // Central Asian Journal of Art Studies. – 2019. – № 2. – С. 109-116.

103. Кушниренко А.Г. 12 лекций о том, для чего нужен школьный курс информатики и как его преподавать / А.Г. Кушниренко, Г.В. Лебедев // Информатика. – 1999. – №1. – С. 2-15.

104. Кушниренко А.Г. Выравнивающий курс Азы программирования для первокурсников научно-технических и педагогических специальностей / А.Г. Кушниренко, А.Г. Леонов, М.В. Райко // Свободное программное обеспечение в высшей школе: Одиннадцатая конференция: Материалы. – М.: Альт Линукс, 2016. – С. 50-59.

105. Кушниренко А. Г. Знакомим дошкольников и младших школьников с азами алгоритмики с помощью систем ПиктоМир и Кумир / А. Г. Кушниренко, А.Г. Леонов, М.А. Ройтберг // Труды НИИСИ РАН. – 2015. – Т. 5, № 1. – С.134-137.

106. Кушниренко А.Г. Информатика 7-9 классы: Учебник для общеобразовательных учебных заведений / А.Г. Кушниренко, Г.В. Лебедев, Я.Н. Зайдельман. – М: Дрофа, 2000, 2001, 2002, 2003.

107. Кушниренко А.Г. Методические указания по проведению цикла занятий «Алгоритмика» в подготовительных группах дошкольных образовательных учреждений с использованием свободно распространяемой учебной среды ПиктоМир: Версия от 10.08.2019 – занятия 1 – 30: [Электронный ресурс] / А.Г. Кушниренко, А.Г. Леонов, М.В. Райко. – 231 с. // ФГУ ФНЦ НИИСИ РАН: [Офиц. сайт]. – URL: https:// https://www.niisi.ru/piktomir/Алгоритмика%20для%20дошкольников.%2019.09.2019.pdf (Accessed: 31.07.2024)

108. Кушниренко А.Г. Основы информатики и вычислительной техники: Основы информатики и вычислительной техники: пробный учебник для средних учебных заведений: учебное издание / А.Г. Кушниренко, Г.В. Лебедев, Р.А. Сворень. – М.: Просвещение, 1990, 1991, 1993, 1996.

109. Кушниренко А.Г. Программирование для дошкольников и младших школьников / А.Г. Кушниренко, А.Г. Леонов // Первое сентября. Информатика. – 2011. – № 15. – С. 20-23.

110. Кушниренко А.Г. Программирование для математиков: учебное пособие для вузов по специальностям «Математика» и «Прикладная математика» / А.Г. Кушниренко, Г.В. Лебедев. – М.: Наука, 1988. – 384 с.

111. Кушниренко А.Г. Элементы цифровизации образовательного процесса на примере системы Мирера / А.Г. Кушниренко, М.А. Кузьменко, А.Г. Леонов // Свободное программное обеспечение в высшей школе: Сборник материалов Тринадцатой конференции. – М.: BaseaIt, 2018. – С. 66-68.

112. Ланда Л.Н. Алгоритмизация в обучении / Под общ. ред. Б.В. Гнеденко, Б.В. Бирюкова. – М.: Просвещение, 1966. – 523 с.

113. Ландо С.К. Информатика. Алгоритмика. 7 класс / С.К. Ландо, А.Л. Семенов, Н.М. Вялый. – М.: Просвещение, 2008. – 208 с.

114. Лаппо Л.Д. Домашняя работа по геометрии за 7 класс к учебнику А.В. Погорелова "Геометрия. 7-9 классы" / Л.Д. Лаппо, А.А. Сапожников. – М.: Экзамен, 2015. – 128 с. – (Решебник).

115. Леднев В.С. О теоретических основах содержания обучения информатике в общеобразовательной школе / В.С. Леднев, А.А. Кузнецов, С.А. Бешенков // Информатика и образование. – 2000. – № 2. – С. 13-16.

116. Леднев В.С. Перспективы изучения основ кибернетики в средней школе / В.С. Леднев, А.А. Кузнецов // Советская педагогика. – 1975. – № 6. – С. 3-7.

117. Леднев В.С. Состояние и перспективы развития курса информатики в общеобразовательной школе / В.С. Леднев, А.А. Кузнецов // Информатика и образование. – 1998. – №3. – С. 76-78.

118. Леонов А.Г. Алгоритмический язык - пропедевтика курса программирование в университетском образовании / А.Г. Леонов, А.А. Прилипко // Интеграция отечественной науки в мировую: Структурные преобразования и перспективные направления развития: Международная научно-практическая конференция: Санкт-Петербург, 30–31 мая 2016 г.: Сборник научных статей. – СПб.: КУЛЬТ-ИНФОРМ-ПРЕСС, 2016. – С. 206-211.

119. Леонов А.Г. Базы данных и электронные таблицы / А.Г. Леонов, М.Г. Эпиктетов // Информатика и образование. – 1996. – №3. – С. 15-26.

120. Леонов А.Г. Базы данных и электронные таблицы / А.Г. Леонов, М.Г. Эпиктетов // Информатика и образование. – 1996. – №4. – С. 5-16.

121. Леонов А.Г. Базы данных и электронные таблицы / А.Г. Леонов, М.Г. Эпиктетов // Информатика и образование. – 1996. –№6. – С. 21-26.

122. Леонов А.Г. Решение задачи автоматизации учебного процесса с помощью экспериментального поиска индивидуальной образовательное траектории / Леонов А. Г., Дьяченко М. С. // Информатика и образование. – 2024. – № 4 – С. 14-26.

123. Леонов А.Г. Качественные оценки эффективности методики обучения элементам информатики в пропедевтическом курсе / А.Г. Леонов, Ю.А. Первин // Ярославский педагогический вестник. – 2015. – № 5. – С. 92-96.

124. Леонов А.Г. КуМир вернулся! / А.Г. Леонов, А.Г. Кушниренко // Первое сентября. Информатика. – 2009. – №6. – С.17-21.

125. Леонов А.Г. Методика преподавания основ алгоритмизации на базе системы “КуМир”: Лекция 1-8 / А.Г. Леонов, А.Г. Кушниренко. – М., 2010. – 78 с.

126. Леонов А.Г. Методы интеграции цифровых образовательных сред в цифровую образовательную платформу Мирера / А.Г. Леонов, А.Е. Орловский // Труды НИИСИ РАН. Математическое и компьютерное моделирование сложных систем: теоретические и прикладные аспекты. – 2021. – Т. 11, № 3. – С. 59-65.

127. Леонов А.Г. Навигатор к учебно-методическому комплекту «Алгоритмика для дошкольников и учащихся начальных классов с использованием робототехнического образовательного набора и цифровой образовательной среды ПиктоМир» / А.Г. Леонов, М.В. Райко. – Самара: Самарский областной институт повышения квалификации и переподготовки работников образования, 2021. – 69 с.

128. Леонов А.Г. Об опыте построения априорной оценки успеваемости студентов в системе Мирера с использованием нейронных сетей / А.Г. Леонов, М.А. Матюшин, М.С. Дьяченко // Успехи кибернетики. – 2021. – Т. 2, № 4. – С. 49-59. – DOI: 10.51790/2712-9942-2021-2-4-5

129. Леонов А. Г. О вариантах решения задачи распознавания табличной структуры по изображению в условиях отсутствия априорной информации / Бесшапошников Н. О., Леонов А. Г., Матюшин М. А. // Научная визуализация. – 2020. – Т. 12, № 5. – С. 1–12.

130. Леонов А.Г. Освой КуМир за 6 часов: Первые два часа занятий / А.Г. Леонов // Первое сентября. Информатика. – 2010. – № 24. – С. 15-19.

131. Леонов А.Г. Освой КуМир за 6 часов: Последние четыре часа занятий / А.Г. Леонов // Первое сентября. Информатика. – 2011. – № 2. – С. 25-32.

132. Леонов А.Г. Переход от непосредственного управления исполнителями к составлению программ в пропедевтическом курсе информатики / А.Г. Леонов, Ю.А. Первин // Ярославский педагогический вестник. – 2013. – Т. 3, №. 3. – С. 17-30.

133. Леонов А. Г. Проведение годового цикла занятий алгоритмика для дошкольников в подготовительных группах ДОУ / А.Г. Леонов, А.Г. Кушниренко, М.В. Райко // Воспитание и обучение детей младшего возраста. – 2018. – №7. – С. 174-175.

134. Леонов А.Г. Программные исполнители в цифровых образовательных средах «ПиктоМир», «Роботландия» и «КуМир» / А.Г. Леонов, Ю.А. Первин, Я.Н. Зайдельман // Информатика в школе. – 2019. – №9. – С. 54-61.

135. Леонов А.Г. Программный инструментарий дошкольного и младшего школьного обучения информатике / А.Г. Леонов, Ю.А. Первин // Информатика в школе. – 2018 – № 8 (141). – С. 112-118.

136. Леонов А.Г. Разработка и внедрение компьютерных практикумов в учебные курсы программирования в школе и вузе / А.Г. Леонов, А.А. Прилипко // Инновации в формировании стратегического вектора развития фундаментальных и прикладных научных исследований: Международная научно-практическая конференция: Санкт-Петербург, 20–21 ноября 2015 года: Сборник научных статей. – СПб.: КУЛЬТ-ИНФОРМ-ПРЕСС, 2015. – С. 116-120.

137. Леонов А.Г. Система КуМир в непрерывном школьном курсе информатики / А.Г. Леонов, Ю.А. Первин // Ярославский педагогический вестник. – 2012. – Т. 3, № 4. – С. 28-44.

138. Леонов А.Г. Учебные и тестовые логические задачи в пропедевтическом курсе информатики / А.Г. Леонов, Ю.А. Первин // Информатика и образование. – 2015. – № 9. – С.32-36.

139. Леонов А.Г. Цифровой мир дошкольника / А.Г. Леонов, Т.В. Тимофеева // The world of academia: Culture, Education. – 2024. – № 1. – С. 31-39.

140. Леонов А.Г. ЭВМ-практикум / А.Г. Леонов // Первое сентября. Информатика. – 2011. – № 11. – С. 9-13.

141. Лобеева Е.Г. Дополнительная общеразвивающая программа технической направленности «Робомышь» для детей 5-7 лет: (Срок реализации 1 год): [Утверждено приказом №161 А.С. Юнусовой от 25 сентября 2023 г.]: [Электронный ресурс] / Е.Г. Лобеева; МБДОУ ДСКВ № 46. – URL: https://www.детсад46-братск.рф/images/23-24/doc/obr/dop.programma_robomysh.pdf (Дата обращения: 20.08.2024)

142. Лутц М. Программирование на Python / М. Лутц; Пер. с англ. А. Киселева. – 4-е изд. – СПб.: Символ-плюс, 2011. – 1280 с.

143. Мазный Г. Л. Программирование на БЭСМ-6 в системе «Дубна» / Г.Л. Мазный; Под ред. Н.Н. Говоруна. – М.: Наука, 1978. – (Библиотечка программиста).

144. Международная конвенция о ликвидации всех форм расовой дискриминации: Принята резолюцией 2106 (XX) Генеральной Ассамблеи ООН от 21 декабря 1965 г.: [Электронный ресурс] // Treaties and internationala greements registered or filed and recorded with the Secretariat of the United Nations: Treaty Series. – 1971. – Vol. 660. – Pp. 240-266. – URL: https://www.un.org/ru/documents/decl_conv/conventions/raceconv.shtml (Дата обращения: 20.08.2024)

145. МетаМир – система для проведения индивидуальных и командных олимпиад по алгоритмике и программированию для дошкольников и младших

школьников / А.Г. Леонов, Н.О. Бесшапошников, К.А. Мащенко, А.А. Прилипко // Наука нового времени: Сохраняя прошлое – создаем будущее: Международная научно-практическая конференция: 22-23 декабря 2017 г., Санкт-Петербург: Сборник научных статей. – Спб.: КультИнформПресс, 2017. – С. 49-55.

146. Мирера: Высокотехнологичная мультиформатная образовательная платформа: Стартовая страница отечественной цифровой образовательной платформы: [Электронный ресурс]. – URL: https://mirera.ru (Accessed: 31.07.2024)

147. Мирера - система поддержки непрерываемого образования / А.Г. Леонов, М.В. Райко, Н.О. Бесшапошников, Д.Б. Ерёмин // Свободное программное обеспечение в высшей школе: Сборник материалов Двенадцатой конференции, Переславль, 26–28 января 2018 г. – М.: МАКС-ПРЕСС, 2017. – С. 47-50.

148. Мир информатики: 1–2-й год обучения: [Электронный ресурс] / Под рук. А.В. Могилева. – М.: Кирилл и Мефодий, 2002. – 1 электрон, опт. диск (CD-ROM).

149. Монтессори М. Антропология / М. Монтессори. – М.: АСТ, 2007. – 606 с.

150. Монтессори М. Впитывающий разум ребенка / М. Монтессори. – СПб.: Благотворительный фонд «Волонтёры», 2011. – 320 с.

151. Монтессори М. Мой метод: руководство по воспитанию детей от 3 до 6 лет / М. Монтессори; [Пер. с англ. А. В. Колесниковой, Е. А. Рязанцевой]. – М.: Центрполиграф, 2011. – 415 с.

152. Монтессори М. Подготовка учителя: методическое пособие / М. Монтессори; Пер. с англ. И. Дичковской // МАМА: Альманах: Научно-методическое издание Межрегиональной альтернативной Монтессори-ассоциации: Выпуск 1. – М.: Межрегиональная альтернативная Монтессори-ассоциация, 1994. – 354 с.

153. Мухаметзянов И.Ш. Цифровое пространство в образовании: ожидания, возможности, риски, угрозы / И.Ш. Мухаметзянов // Россия: тенденции и перспективы развития. – 2020. – №15-1. – С. 571-574.

154. Начальное общее образование: Федеральный государственный образовательный стандарт: Утверждено Приказом Минобрнауки России от 06.10.2009 N 373: (Ред. от 11.12.2020) // ФГОС / Национальная ассоциация развития образования и науки. – URL: https://fgos.ru/fgos/fgos-noo/ (Accessed: 31.07.2024)

155. Некоторые вопросы эффективности детерминированных алгоритмов распознавания образов с помощью библиотеки OpenCV / Н.О. Бесшапошников, М.А. Кузьменко, А.Г. Леонов, М.А. Матюшин // Труды НИИСИ РАН. – 2018. – Т.8, №2. – С.65-69.

156. Немов Р.С. Психология: учебное для студентов высших пед. учеб. Заведений / Р.С. Немов // Общие основы психологии.: В 3 т. Т.1. – 4 изд. – М.: ВЛАДОС, 2003. – 688 с.

157. Новые информационные технологии в дошкольном образовании / Ю.М. Горвиц, Е.В. Зворыгина, Н.Н. Поддьяков, Л.Д. Чайнова [и др.]. – М.: ЛИНКА-ПРЕСС, 1998. – 328 с.

158. Носкова Т.Н. Дидактика цифровой среды. / Носкова Т. Н. // Российский государственный педагогический университет им. А. И. Герцена. — Санкт-Петербург : Изд-во РГПУ им. А. И. Герцена, 2020. — 382, [2] с. ил., табл.; 23. — ISBN 978-5-8064-2981-1.

159. Носкова Т.Н. Цифровая образовательная среда: методологический аспект запуска инноваций / Т.Н. Носкова // Информатика и образование. –2023. – №38(6). – С. 45-51. – DOI: 10.32517/0234-0453-2023-38-6-45-51

160. Об образовании в Российской Федерации: Глава 11, Статья 77. Организация получения образования лицами, проявившими выдающиеся способности: Федеральный закон от 29.12.2012 № 273-ФЗ: Принят Государственной Думой 21 декабря 2012 г: Одобрен Советом Федерации 26 декабря 2012 г.: (Ред. от 25.12.2018) // Официальный интернет-портал правовой информации. – URL: https://http://pravo.gov.ru/proxy/ips/?docbody=&nd=102162745&ysclid=lxzibok5jz369839508 (Accessed: 31.07.2024)

161. Об образовании в Российской Федерации: Федеральный закон от 29.12.2012 N 273-ФЗ (ред. от 08.08.2024): Принят Государственной Думой 21 декабря 2012 г.; Одобрен Советом Федерации 26 декабря 2012 г. // Официальный интернет-портал правовой информации. –URL: http://pravo.gov.ru/proxy/ips/?docbody=&nd=102162745 (Дата обращения: 25.08.2024).

162. Об организации образовательной деятельности в организациях, реализующих образовательные программы высшего образования и соответствующие дополнительные профессиональные программы, в условиях предупреждения распространения новой коронавирусной инфекции на территории Российской Федерации: Приказ Министерства науки и высшего образования Российской Федерации от 14 марта 2020 г. № 397 // Министерство науки и высшего образования РФ: [Офиц. сайт]. – URL: https://minobrnauki.gov.ru/documents/?ELEMENT_ID=18515 (Дата обращения: 20.08.2024)

163. О внесении изменений в Федеральный закон «Об образовании в Российской Федерации: и статью 1 Федерального закона «Об обязательных требованиях в Российской Федерации»: Федеральный закон от 24.09.2022 г. № 371-ФЗ // Президент России [Офиц. сайт]. – URL: http://www.kremlin.ru/acts/bank/48313 (Дата обращения: 28.08.2024)

164. О внесении изменений в некоторые приказы Министерства образования и науки Российской Федерации и Министерства просвещения Российской Федерации, касающиеся федеральных государственных образовательных стандартов общего образования и образования обучающихся с ограниченными возможностями здоровья и умственной отсталостью (интеллектуальными нарушениями): Приказ Министерства просвещения Российской Федерации от 08.11.2022 № 955 // Официальное опубликование нормативных документов. – URL:

http://publication.pravo.gov.ru/Document/View/0001202302060059 (Дата обращения: 28.08.2024)

165. Огилько И. По Москве запустили первые такси-роботы. Какие впечатления у прокатившихся / И. Огулько // Российская газета. – 2023. – 08 июня // RG.RU [Офиц. сайт]. – URL: https://rg.ru/2023/06/08/reg-cfo/progulka-s-intellektom.html?ysclid=lxrmlh6uxn340574888 (Дата обращения: 20.08.2024)

166. О мерах по обеспечению компьютерной грамотности учащихся средних учебных заведений и широкого внедрения электронно-вычислительной техники в учебный процесс: Постановление ЦК КПСС и Совета министров СССР от 28 марта 1985 года № 271 // Вопросы образования. – 2005. – № 3. – С. 341-346.

167. О персональных данных: Федеральный закон Российской Федерации от 27.07.2006 г. № 152-ФЗ: Принят Государственной Думой 8 июля 2006 г.: Одобрен Советом Федерации 14 июля 2006 г. // Президент России: [Офиц. сайт]. – URL: http://www.kremlin.ru/acts/bank/24154 (Дата обращения: 20.08.2024)

168. О подходах к построению надежной цифровой образовательной платформы / М.С. Дьяченко, М.А. Кузьменко, А.Г. Кушниренко, Г.О. Райко, И.Г. Райко // Труды НИИСИ РАН. – 2022. – Том 12, № 3. – С. 20-25.

169. О проведении эксперимента по внедрению цифровой образовательной среды: Постановление Правительства Российской Федерации от 07.12.2020 № 2040 // Официальное опубликование правовых актов. – URL: http://publication.pravo.gov.ru/file/pdf?eoNumber=0001202012090002 (Дата обращения: 20.08.2024)

170. Основное общее образование: Федеральный государственный образовательный стандарт: Утверждено Приказом Минобрнауки России от 17.12.2010 N 1897: (Ред. от 11.12.2020) // ФГОС / Национальная ассоциация развития образования и науки. – URL: https://fgos.ru/fgos/fgos-ooo/ (Accessed: 31.07.2024)

171. Основной государственный экзамен по информатике: Демонстрационный вариант контрольных измерительных материалов основного

государственного экзамена 2024 года по информатике / Федеральный институт педагогических измерений // ФИПИ: [Электронный ресурс]. – URL: https://doc.fipi.ru/oge/demoversii-specifikacii-kodifikatory/2024/inf_9_2024.zip (Дата обращения: 24.08.2024)

172. Основы государственной политики по сохранению и укреплению традиционных российских духовно-нравственных ценностей: Утверждено Указом Президента Российской Федерации от 09.11.2022 г. № 809 // Президент России [Офиц. сайт]. – URL: http://www.kremlin.ru/acts/bank/48502 (Дата обращения: 28.08.2024)

173. Основы информатики и вычислительной техники: Пробное учебное пособие для сред. учеб. заведений. В 2-х ч. Ч. 2 / А.П. Ершов, В.М. Монахов, А.А. Кузнецов [и др.]; Под ред. А.П. Ершова, В.М. Монахова. – М.: Просвещение, 1986. – 143 с.

174. Основы информатики и вычислительной техники: Пробный учебник для средних учебных заведений / А.П. Ершов, А.Г. Кушниренко, Г.В. Лебедев, А.Л. Семенов, А. Х. Шень. – М.: Просвещение, 1988. – 207 с.

175. Основы информатики и вычислительной техники: Программа для средних учебных заведений / Академия педагогических наук СССР, НИИ содержания и методов обучения, Сибирское отделение АН СССР, Вычислительный центр; Под ред. А. А. Кузнецова [и др.]. – М.: Просвещение, 1985. – 102 с.

176. Пантина Н.С. Игра и детские взаимоотношения / Н.С. Пантина // Дошкольное воспитание. – 1967.– № 12. – С.22-26.

177. Пейперт С. Переворот в сознании: Дети, компьютеры и плодотворные идеи / С. Пейперт. – М.: Педагогика, 1989. – 224 с.

178. Первин Ю.А. Алгоритмические и логические задачи начального курса информатики: (Из опыта дистанционного обучения) / Ю.А. Первин, И.А. Кустова. – Берлин: Palmarium Academic Publisher, 2013. – 385 с.

179. Первин Ю.А. Большой Московский семинар по методике раннего обучения информатике как продолжение традиций педагогической школы А.П. Ершова / Ю.А. Первин // VII Международная конференция памяти академика А.П. Ершова, секция «Информатика и образование» – Новосибирск, Академгородок, 15-19 июня 2009. – Новосибирск: Академгородок, 2009. – 7 с.

180. Первин Ю.А. Все лучшее детям? / Ю.А. Первин // Всероссийский съезд учителей информатики Москва, МГУ имени М.В. Ломоносова, 24-26 марта 2011 г.: Тезисы докладов. – М.: МГУ, 2011. – 730 с.

181. Первин Ю.А. Методика раннего обучения информатике / Ю.А. Первин. – 2-е изд. – М.: Бином, Лаборатория базовых знаний, 2008. – 288 с.

182. Первин Ю.А. От операционного стиля мышления через педагогические компетенции к универсальным учебным действиям / Ю.А. Первин // Методика раннего обучения информатики. Т.2. – М.: Российский государственный социальный университет, 2011. – С. 115-127.

183. Песталоцци И.Г. Избранные педагогические сочинения: В 2-х т. Т. 1 / Под ред. В.А. Ротенберг, В.М. Кларина. – М.: Педагогика, 1981. – 336 с. – С. 61-212.

184. Песталоцци И.Г. Как Гертруда учит своих детей / И.Г. Песталоцци // Краткая хрестоматия по истории зарубежной и отечественной педагогической мысли: учеб. пособие для вузов / Сост. И.П. Андриади, Е.Б. Куракина, С.Н. Ромашова, [и др.]; Под общ. ред. И.П. Андриади; Правительство Москвы, Департамент образования г. Москвы. – М., 2007. – С. 190-194.

185. Пиаже Ж. Избранные психологические труды: Психология интеллекта: Генезис числа у ребенка: Логика и психология / Ж. Пиаже; [Пер. с фр.]. – М.: Просвещение, 1969. – 659 с.

186. Пиаже Ж. Речь и мышление ребенка / Ж. Пиаже; [Сост., пер. с фр. Вал. А. Лукова, Вл. А. Лукова]. – М.: Педагогика-пресс, 1994. – 527 с. – (Психология: классические труды).

187. ПиктоМир: Основы программирования для детей: Стартовая страница отечественной цифровой образовательной среды «ПиктоМир»: [Электронный ресурс]. – URL: https://piktomir.ru (Accessed: 31.07.2024)

188. Пирс Ч.С. Что такое знак? / Ч.С. Пирс // Вестник Томского государственного университета. Философия. Социология. Политология. – 2009. – №3 (7). – С. 88-95.

189. Плаксин М.А. Информатика для малышей. Модуль 2: Словари. Каталоги. Организация текста / М.А. Плаксин // Информатика в младших классах: Приложение к журналу «Информатика и образование». Серия «Информатика в школе». – 1998. – № 1, 2. – С.48-56.

190. По Э. Философия творчеств / Э. По // Стихотворения. Новеллы. Повесть о приключениях Артура Гордона Пима. Эссе. – М.: Аст, 2003. – с. 707-709.

191. Поддьяков А.Н. Исследовательское поведение: Стратегии, познания, помощь, противодействие, конфликт / А.Н. Поддьяков. – М.: Национальное образование, 2016. – 301 с.

192. Поддьяков А.Н. Комбинаторное экспериментирование дошкольников с различными объектами / А.Н. Поддьяков // Вестник Московского университета. Психология. – 1991. – №4. – С. 34-41.

193. Поддьяков А.Н. Обучение дошкольников комбинаторному экспериментированию / А.Н. Поддьяков // Вопросы психологии. – 1991. – № 4. – С. 29-34.

194. Поддьяков А.Н. Развитие исследовательской инициативности в детском возрасте: диссертация на соискание ученой степени доктора психологических наук / А. Н. Поддьяков. Москва, 2001. 320 с.

195. Поддьяков Н.Н. Мышление дошкольника / Н.Н. Поддьяков. – М.: Педагогика, 1977. – 271 с.

196. Поддьяков Н.Н. Некоторые общие вопросы развития мышления дошкольников // Развитие мышления и умственное воспитание дошкольника / Н.Н.

Поддьяков; Под ред. Н.Н. Поддьякова, А.Ф. Говорковой. – М.: Педагогика, 1985. – С. 5-28.

197. Поддьяков Н.Н. Новые подходы к исследованию мышления дошкольников / Н.Н. Поддьяков // Вопросы психологии. – 1985. – № 2. – С. 103-117.

198. Подходы к учету посещаемости студентов в цифровой образовательной платформе Мирера / А.Г. Леонов, К.А. Мащенко, А.В. Шляхов, А.А. Холькина // Труды НИИСИ РАН. – 2022. – Т.12, № 3. – С. 26-32.

199. Пойа Д. Как решать задачу / Д. Пойа; Ред. Ю.М. Гайдук, пер. В.Г. Звонарева, Д.Н. Белл. – М.: Государственное учебно-педагогическое издательство Министерства просвещения РСФСР, 1959. – 207 с.

200. Поляков К.Ю. Информатика. 10 класс: Углубленный уровень: В 2 ч.: Ч.2. / К.Ю. Поляков, Е.А. Еремин. – М.: БИНОМ: Лаборатория знаний, 2019. – 648 с.

201. Поляков К.Ю. КуМир и школьная информатика / К. Поляков // Мысли вслух: Блог: Пятница, 8 апреля 2011 г.: [Электронный ресурс]. – URL: http://kpolyakov.blogspot.ru/2011/04/blog-post_5678.html (Дата обращения: 20.08.2024)

202. Поляков К.Ю. Программы на школьном алгоритмическом языке, приведённые в учебнике «Информатика. Углублённый уровень» для 10 класса К.Ю. Полякова и Е.А. Еремина / К. Поляков // Мысли вслух: Блог: [Электронный ресурс]. – URL: https://kpolyakov.spb.ru/school/probook/kumir.htm (Дата обращения: 20.08.2024)

203. Проблемы применения чат-ботов в естественно-научных курсах / А.Г. Леонов, М.С. Дьяченко, К.А. Мащенко, Н.О. Бесшапошников // Информатизация образования и методика электронного образования: Цифровые технологии в образовании: IV Международная научная конференция: Красноярск, 6-9 октября 2020: Материалы: В 2-х ч. Ч.1. – Красноярск: СФУ, 2020. – С. 393-396.

204. Проведение цикла занятий «Алгоритмика» в летнем лагере для дошкольников и младшеклассников / А.Г. Кушниренко, А.Г. Леонов, И.Н. Грибанова, М.В. Райко // Труды НИИСИ РАН. – 2018. – Т. 8. № 4. – С. 181-185.

205. Пышкало А.М. Методическая система обучения геометрии в начальной школе: авторский доклад по монографии «Методика обучения элементам геометрии в начальных классах», представленной на соискание ученой степени д-ра пед. наук / А.М. Пышкало. – М.: Академия пед. наук СССР, 1975. – 60 с.

206. Развитие психологических новообразований старших дошкольников в процессе обучения программированию на базе цифровой образовательной среды «ПиктоМир» / А.Г. Кушниренко, А.Г. Леонов, О.В. Собакинских, Л.В. Шибаева // Труды НИИСИ РАН. – 2019. – Т.9, № 6. – С. 21-24.

207. Развитие цифровой образовательной среды в Российской Федерации: механизмы развития и возможные риски / С.Д. Каракозов, Л.Р. Пикалова, Е.П. Седова, О.Н. Титова // Ростовский научный журнал. – 2018. – №11. – С. 85-100.

208. Раскин Д. Интерфейс: Новые направления в проектировании компьютерных систем / Д. Раскин; [Пер. Ю. Асотова]. – М.: Символ-Плюс, 2005. – 272 с.

209. Реализация нового типа задач «электронные таблицы» в цифровой образовательной платформе Мирера / И.А. Васильев, А.Г. Леонов, К.А. Мащенко, А.В. Шляхов. // Труды НИИСИ РАН. – Т.12, № 3. – С. 33-38.

210. Результаты освоения годовой программы «Алгоритмика для дошколят» подготовительными группами муниципального ДОУ / А.Г. Леонов, М.В. Райко, О.В. Собакинских, Н.В. Собянина // Труды НИИСИ РАН. –2020. – Т.10, № 5. – С.195-199.

211. Роберт И.В. Дидактика периода информатизации образования / И.В. Роберт // Педагогическое образование в России. – 2014. – № 8. – С. 110-119.

212. Роберт И.В. Дидактика эпохи цифровых информационных технологий / И.В. Роберт // Профессиональное образование. Столица. – 2019. – № 3. – С. 16-26.

213. Роботландия: Негосударственное образовательное учреждение: [Сайт]. – URL: http://www.robotlandia.ru/ (Дата обращения: 20.08.2024)

214. Руссо Ж.-Ж. Педагогические сочинения: В 2-х т. Т.1. Эмиль, или О воспитании / Под ред. Г.Н. Джиладзе; Сост. А.Н. Джурински. – М.: Педагогика, 1981. – 656 с. – (Педагогика).

215. Салмина Н.Г. Знак и символ в обучении / Н.Г. Салмина. – М.: МГУ, 1988. – 287 с.

216. Санитарно-эпидемиологические требования к организациям воспитания и обучения, отдыха и оздоровления детей и молодежи: Об утверждении санитарных правил СП 2.4. 3648-20: Постановление Главного государственного санитарного врача Российской Федерации от 28.09.2020 № 28 // Официальное опубликование нормативных актов. – URL: http://publication.pravo.gov.ru/Document/View/0001202012210122 (Accessed: 31.07.2024)

217. Сборник психологических тестов. Часть II: Пособие / Сост. Е.Е. Миронова. – М.: Женский институт ЭНВИЛА, 2006. – 146 с.

218. Себеста Роберт В. Величайший проект в истории: язык Ada / Роберт В. Себеста // Основные концепции языков программирования. – М.: Вильямс, 2001. – с. 103-108.

219. Себеста Роберт У. Объектно-ориентированное программирование: язык Smalltalk / Роберт У. Себеста // Основные концепции языков программирования. – М.: Вильямс, 2001. – с.109-111.

220. Семенов А.Л. Задачи, которые «неизвестно-как-решать», в цифровом мире / А.Л. Семенов, Т.А. Рудченко // Информатизация образования и методика электронного обучения: Цифровые технологии в образовании: VII Международная научная конференция: Красноярск, 19–22 сентября 2023 г.: Материалы. –

Красноярск: Красноярский государственный педагогический университет им. В.П. Астафьева, 2023. – С. 881-884.

221. Семенов А.Л. Концептуальные проблемы информатики, алгоритмики и программирования в школе / А.Л. Семенов // Вестник кибернетики. – 2016. – № 22. – С. 11-15.

222. Семенов А.Л. Образование, информатика, компьютеры / А.Л. Семенов // Информатика и образование. – 1995. – №5. – С. 6-11.

223. Семенов А.Л. Симор Паперт и мы. Конструкционизм – образовательная философия XXI века / А.Л. Семенов // Вопросы образования. – 2017. – №1. – С. 44-49.

224. Семенов А.Л. Тридцать лет – это все-таки мало / А.Л. Семенов, А.Ю. Уваров // Информатика и образование. – 2015. – № 7 (266). – С. 6-8.

225. Семёнов А.Л. О продолжении российского математического образования в XXI веке / А.Л. Семёнов // Вестник Московского университета. Педагогическое образование. – 2023. – Т. 21, №2. – С. 7-45. – DOI: 10.55959/MSU2073-2635-2023-21-2-7-45

226. Система ejudge – EjudgeWiki: [Электронный ресурс]. – URL: https://ejudge.ru/wiki/index.php/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_ejudge (Accessed: 31.07.2024)

227. Система программирования PascalABC.NET // PascalABC.NET: Современное программирование на языке Pascal: [Электронный ресурс]. – URL: https://pascalabc.net (Accessed: 31.07.2024)

228. Смолянинова О.Г. Обзор практик обеспечения электронной поддержки образовательного процесса средствами LMS Moodle: Опыт российских вузов / О.Г. Смолянинова, Н.А. Иванов // АНИ: педагогика и психология. – 2019. – Т.8, №2 (27). – С.228-232.

229. Современная цифровая образовательная среда в Российской Федерации: Паспорт приоритетного проекта [25.10.2016 г. - 01.02.2021 г.]: Утверждено президиумом Совета при Президенте Российской Федерации по

стратегическому развитию и приоритетным проектам 25 октября 2016 г. // Правительство России: [Офиц. сайт]. – URL: http://static.government.ru/media/files/8SiLmMBgjAN89vZbUUtmuF5lZYfTvOAG.pdf (Дата обращения: 20.08.2024)

230. Список языков России в итогах всероссийской переписи населения / Министерство науки и высшего образования Российской Федерации, Институт языкознания Российской Академии наук // Росстат: Федеральная служба государственной статистики: [Офиц. сайт]. – URL: https://rosstat.gov.ru/storage/mediabank/Tom5_Spisok_yazykov.doc (Дата обращения: 20.08.2024)

231. Справочный центр Stepic: [Электронный ресурс]. – URL: https://help.stepik.org/collection/5642 (Accessed: 31.07.2024)

232. Среднее общее образование: Федеральный государственный образовательный стандарт: Утверждено Приказом Минобрнауки России от 17.05.2012 N 413: (Ред. от 11.12.2020) // ФГОС / Национальная ассоциация развития образования и науки. – URL: https://fgos.ru/fgos/fgos-soo/ (Accessed: 31.07.2024)

233. Стандарт оснащения государственных и муниципальных общеобразовательных организаций, осуществляющих образовательную деятельность в субъектах Российской Федерации, на территории которых проводится эксперимент по внедрению цифровой образовательной среды, компьютерным, мультимедийным, презентационным оборудованием и программным обеспечением: Утверждено Приказом Министерства просвещения Российской Федерации, Министерства цифрового развития, связи и массовых коммуникаций Российской Федерации от 08.09.2021 № 634/925 // Официальное опубликование нормативных актов. – URL: http://publication.pravo.gov.ru/file/pdf?eoNumber=0001202112160056 (Accessed: 31.07.2024)

234. Стась А.Н. Развитие алгоритмического мышления в процессе обучения будущих учителей информатики / А.Н. Стась, Н.Ф. Долганова // Вестник ТГПУ. – 2012. – №7 (122). – С. 241-244.

235. Страуструп Б. Дизайн и эволюция языка C++ / Б. Страуструп. – М.: ДМК Пресс; СПб.: Питер, 2006. – 448 с.

236. Сусов Р.В. Цифровая вычислительная техника в советском народном хозяйстве // ТРУДЫ SORUCOM-2017: Развитие вычислительной техники в России и странах бывшего СССР: история и перспективы: Четвертая Международная конференция: 3–5 октября 2017 г.: Москва, Зеленоград: Материалы. – Саратов: Российский экономический университет им. Г.В. Плеханова, 2017. – С. 353-357.

237. Талызина Н.Ф. Теоретические проблемы программированного обучения / Н.Ф. Талызина. – М.: МГУ, 1969. – 133 с.

238. Толстой Л.Н. Воспитание и образование / Л.Н. Толстой // Полное собрание сочинений = Oeuvres complètes : юбилейн. изд. (1828-1928) / Л.Н. Толстой; Под общ. ред. В.Г. Черткова. – М.: Худож. лит., 1957. – С. 342-365.

239. Уваров А.Ю. Информатизация школы: вчера, сегодня, завтра / А.Ю. Уваров. – М.: БИНОМ: Лаборатория знаний, 2011. – 484 с.

240. Уваров А.Ю. Об описании компетенций XXI века / А.Ю. Уваров // Образовательная политика. – 2014. – № 1 (63). – С. 13-30.

241. Уваров А.Ю. О развитии естественно-научного образования в западных странах / А.Ю. Уваров. – М.: ВЦ РАН, 2013. – 104 с.

242. Ушинский К.Д. О народности в общественном воспитании / К.Д. Ушинский // Собрание сочинений в 11 т.: Т.2: Педагогические статьи 1857–1861 гг. – М.: Педгиз, 1948. – С. 69-166.

243. Федеральная адаптированная образовательная программа дошкольного образования для обучающихся с ограниченными возможностями здоровья: Утверждено Приказом Министерства просвещения Российской Федерации от 24.11.2022 № 1022 // Официальное опубликование нормативных документов. – URL:

http://publication.pravo.gov.ru/Document/View/0001202301270036 (Дата обращения: 28.08.2024)

244. Федеральная образовательная программа дошкольного образования: Утверждено Приказом Министерства просвещения Российской Федерации от 25.11.2022 № 1028 // Официальное опубликование нормативных актов. – URL: http://publication.pravo.gov.ru/Document/View/0001202212280044?ysclid=lz7ziutd57291276108 (Дата обращения: 20.08.2024)

245. Федеральная образовательная программа начального общего образования: Утверждено Приказом Министерства просвещения Российской Федерации от 18.05.2023 № 372 // Официальное опубликование нормативных актов. – URL: http://publication.pravo.gov.ru/document/0001202307130044?ysclid=lz7zfkuhjz26691087 (Дата обращения: 20.08.2024)

246. Федеральная образовательная программа основного общего образования: Утверждено Приказом Министерства просвещения Российской Федерации от 18.05.2023 № 370 // Официальное опубликование нормативных актов. – URL: http://publication.pravo.gov.ru/document/0001202307140040?ysclid=lxd7ie51oz40598804 (Дата обращения: 20.08.2024)

247. Федеральный базисный учебный план и примерные учебные планы для образовательных учреждений Российской Федерации, реализующих программы общего образования : Утверждено Приказом Минобразования РФ от 09.03.2004 № 1312 // Электронный фонд правовых и нормативно-технических документов. – URL: https://docs.cntd.ru/document/901895864 (Дата обращения: 18.08.2024)

248. Фейгенберг И.М. Человек Достроенный и этика: Цивилизация как этап развития жизни Земли / И.М. Фейгенберг. – М.: Медицинское информационное агентство, 2011. – 128 с.

249. Филимонова М. И. Сравнительный анализ методик Кубики Кооса и Цветные прогрессивные матрицы Равена в диагностике коэффициента интеллекта детей от 4-х до 8-ми лет / М.И. Филимонова // Журнал психолого-педагогических исследований. – 2023. – № 4(4). – С. 40-46.

250. Флейвелл Дж. Х. Генетическая психология Жана Пиаже / Дж. Флейвелл. – М.: Просвещение, 1967. – 623 с.

251. Формирование универсальных учебных действий в Ф79 основной школе: от действия к мысли: Система заданий: пособие для учителя / А.Г. Асмолов, Г.В. Бурменская, И.А. Володарская, О. А. Карабанова, Н. Г. Салмина, С. В. Молчанов; Под ред. А.Г. Асмолова. – М.: Просвещение, 2010. – 159 с.

252. Фултон Х. Путь Ruby / Х. Фултон, А. Арко; Пер. с англ. – 3-е изд. – М.: ДМК Пресс, 2016. – С. 33-38.

253. Харина Г.В. Среда обитания человека как предмет экологического образования: социально-философский аспект / Г.В. Харина, А.В. Чернов // Образование и наука. – 2008. – №2 (50). – С. 39-47.

254. Хеннер Е.К. Вычислительное мышление / Е.К. Хеннер // Образование и наука. – 2016. – №2 (131). – С. 18-33.

255. Хохлова Н.И. Принципы исследования комбинаторики у детей дошкольного возраста // Вестник РГГУ. Педагогика. Психология. 2012, Вып. 1, С. 97-105.

256. Цветкова М.С. Изобразительное искусство и информационные технологии: Интегрированный курс / М.С. Цветкова // Информатика и образование. – 2001. – № 9. – С. 74-81.

257. Целевая модель цифровой образовательной среды: Утверждено Приказом Министерства просвещения РФ от 2 декабря 2019 г. № 649 // Официальное опубликование правовых актов. – URL: http://publication.pravo.gov.ru/Document/View/0001201912250047 (Дата обращения: 20.08.2024)

258. Цифровая образовательная среда «ПиктоМир»: Опыт разработки и массового внедрения годового курса программирования для дошкольников / Н.О. Бесшапошников, А.Г. Кушниренко, А.Г. Леонов, М.В. Райко, О.В. Собакинских // Информатика и образование. – 2020. – №10. – С. 28-40.

259. Цифровая экономика Российской Федерации: Программа: Утверждено распоряжением Правительства Российской Федерации от 28 июля 2017 г. № 1632-р // Правительство России: [Офиц. сайт]. – URL: http://static.government.ru/media/files/9gFM4FHj4PsB79I5v7yLVuPgu4bvR7M0.pdf (Дата обращения: 19.08.2024)

260. Что такое Интернет? / А.Г. Кушниренко, А.Г. Леонов, М.А. Кузьменко, Б.А. Назаров, Н.А. Подольская, С.Б. Ханжин // Информатика и образование. – 1998. – №5, №6, №7. – С. 91-101, С.113-122, С.85-93.

261. Чуркина Н.А. «Цифровой след» в аспекте электронного обучения: [Электронный ресурс] / Н.А. Чуркина // Международный научно-исследовательский журнал. – 2022. – №11 (125). – URL: https://research-journal.org/archive/11-125-2022-november/10.23670/IRJ.2022.125.35 (Дата обращения: 10.07.2024). – DOI: 10.23670/IRJ.2022.125.35

262. Шагалова Е. Н. Словарь новейших иностранных слов / Е. Н. Шагалова. – М.: АСТ-Пресс Книга, 2017. – 576 с. – (Настольные словари русского языка).

263. Шамсутдинова Т.М. Формирование индивидуальной образовательной траектории в адаптивных системах управления обучением / Т.М. Шамсутдинова // Открытое образование. – 2021. – Т. 25, № 6. – С. 36–44. – DOI: 10.21686/1818-4243-2021-6-36-44

264. Шапкин С.А. Компьютерная игра: Новая область психологических исследований / С.А. Шапкин // Психологический журнал. – 1999. – № 1. – С. 86-102.

265. Шилдт Г. C# 4.0: полное руководство = C# 4.0 The Complete Reference / Г. Шилдт; Пер. с англ. – М.: Вильямс, 2010. – 1056 с.

266. Щедровицкий Г.П. О двух типах отношений руководства в групповой деятельности детей / Г.П. Щедровицкий, Р.Г. Надежина // Вопросы психологии. – 1973. – № 5. – С.74-84.

267. Эккель Б. Философия Java = Thinking in Java / Б. Эккель. – 3-е изд. – СПб.: Питер, 2003. – 976 с.

268. Эльконин Д.Б. Дневники 1948–1954 гг.: Научные дневники: 1948 // Антропопраксис. Ежегодник гуманитарных исследований. – 2010. – Т. 2. – С. 197-217.

269. Эльконин Д.Б. Игра / Д.Б. Эльконин // Философский энциклопедический словарь. – М.: Сов. энцикл., 1989. – С. 202.

270. Эльконин Д.Б. Психология игры / Д.Б. Эльконин. – М.: Педагогика, 1978. – 304 с.

271. Яковлев Е.В. Педагогический эксперимент в диссертационных исследованиях / Е.В. Яковлев, Н.О. Яковлева // Современная высшая школа: инновационный аспект. – 2011. – № 1. – С. 52-63.

272. 6 Ways artificial intelligence and chatbots are changing education // The future of learning. Discussions and thoughts: [Electronic resource]. – URL: https://chatbotsmagazine.com/six-ways-a-i-and-chatbots-are-changing-education-c22e2d319bbf (Accessed: 19.08.2024)

273. Abdelrahman G. Knowledge tracing: A survey / G. Abdelrahman, Q. Wang, B. Pereira Nunes // ACM Computing Surveys. – 2022. – Vol. 55 (11). – P.1-37. – DOI: 10.1145/3569576

274. Abufardeh S. Culturalization of software architecture: Issues and challenges / S. Abufardeh, K. Magel // International Conference on Computer Science and Software Engineering: Hubei, China, 12-14 December: Proceedings. – Hubei: The Institute of Electrical and Electronics Engineers, 2008. – Pp. 436-439. – DOI: 10.1109/CSSE.2008.1414

275. AdaptKT: A domain adaptable method for knowledge tracing / S. Cheng, Q. Liu, E. Chen, K. Zhang, Z. Huang, Y. Yin, X. Huang, Y. Su // The Fifteenth ACM

International Conference on Web Search and Data Mining: Proceedings. – New York, USA: Association for Computing Machinery, 2022. – Pp. 123-131. – DOI:10.1145/3488560.3498379

276. A data-driven market simulator for small data environments / H. Buhler, B. Horvath, T. Lyons, I. P. Arribas, B. Wood // Social Science Research Network. – 2020. – URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3632431 (Accessed: 19.08.2024). – DOI: 10.2139/ssrn.3632431

277. Advantages and constraints of a hybrid Model K-12 E-Learning assistant chatbot / E. Wu, C.-H. Lin, Y.Y. Ou, C.-Z. Liu, W.-K. Wang, C.-Y. Chao // IEEE Access. – 2020. – Vol.8. – Pp. 77788-77801. – DOI: 10.1109/access.2020.2988252.

278. Aggarwal N. Faculty development in a flexible learning context / N. Aggarwal // Procedia – Social and Behavioral Sciences: 3rd World Conference on Learning, Teaching and Educational Leadership (WCLTA-2012): 21 October 2013. Vol. 93. – Amsterdam: Elsevier Ltd, 2013. – Pp. 1329-1332.

279. Alkhatlan A. Intelligent tutoring systems: A comprehensive historical survey with recent developments / A. Alkhatlan, J. K. Kalita // International Journal of Computer Applications. – 2019. – Vol. 181, №43. – Pp. 975-8887.

280. Application development in programming environment: Third class in general senior high school of Greece / A. Bakali, I. Giannopoulos, N. Ioannidis, C. Kilias, K. Malamas, J. Manolopoulos, P. Politis. – 5th edition. – Athens: Organization of School Books Publications, 2004. – Pp. 124-135.

281. Araki M. Programming education in elementary school: Using a music box as a theme with programming system “Kotodama on Squeak” / M. Araki, K. Okada, Oiwamotoet // Information Processing Society of Japan SIG: [Technical Report]: 2017: [Electronic resource]. – URL: https://ci.nii.ac.jp/naid/110006225064/ (Accessed: 01.08.2024)

282. Arsac J. (1985) LSE 83 / J. Arcas // LE BULLETIN DE L'EPI. – 1985. – № 38. – Pp. 116-137. – URL: http://www.epi.asso.fr/fic_pdf/b38p116.pdf (Accessed: 01.08.2024)

283. Back to the future: the story of Squeak, a practical Smalltalk written in itself. / D. Ingalls, T. Kaehler, J. Maloney, S. Wallace, A.Kay // The 12th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications (OOPSLA '97): Proceedings. – New York, NY, USA: Association for Computing Machinery, 1997. – Pp. 318–326. – DOI: 10.1145/263698.263754

284. Baron G.-L. La prise en compte de l'informatique par le système scolaire, genèse du champ informatique pédagogique, 1970-1980 / G.-L. Baron // Actes du premier colloque sur l'histoire de l'informatique en France: 3, 4, 5 mai 1994. Vol 2. – Grenoble, 1994. – Pp. 45-59.

285. Baron G.-L. L'informatique et ses usagers dans l'éducation / G.-L. Baron // Education. – Paris: Université René Descartes, 1994. – Pp. 215-231.

286. Barr V. Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? / V. Barr, C. Stephenson // ACM Inroads. – 2011. – Vol. 2(1). – Pp. 48-54.

287. Basic programming concepts as explained for preschoolers / V.B. Betelin, A.G. Kushnirenko, A.G Leonov, [et al.] // International Journal of Education and Information Technologies (NAUN). – 2021. – Vol.15. – Pp. 245-255. – DOI: 10.46300/9109.2021.15.25

288. Baudé J. Le système LSE: [Electronic resource] / J. Baude // EpiNet. – 2016. – Vol.182. – Pp.41-56. – URL: https://epi.asso.fr/revue/histo/h70-lse_jb-2015.htm (Accessed: 01.08.2024).

289. Berche S. Pogrammer en LSE / S. Berche, Y. Noyelle. – Paris : Editions du P.S.I., 1979. – Pp. 334-351.

290. Bernard F. Les technologies de l'information et de la communication: de l'introduction de l'informatique à l'École aux pratiques actuelles des jeunes / F. Bernard, R. Ailincai, D. Baur. – CRDP de Guyane, 2010. – Pp. 45-55.

291. Bers M.U. Help Your Kids Learn to Code: The Official ScratchJr Book / M.U. Bers, M. Resnick. – No Starch Press, 2015. – Pp. 15-19.

292. Besshaposhnikov N. Pictomir: How and why do we teach textless programming for preschoolers, first graders and students of pedagogical universities / N. Besshaposhnikov, A. Kushnirenko, A. Leonov // CEE-SECR'17: Proceedings of the 13th Central & Eastern European Software Engineering Conference in Russia, October. – 2017. – № 21. – Pp. 1-7.

293. Biermann A.W. Toward Natural Language Computation / A.W. Biermann, B.W. Ballard // Computational Linguistics. – 1980. – №6 (2). – Pp. 71-86.

294. Borrella I. Predict and intervene: Addressing the dropout problem in a MOOC-based Program / I. Borrella, S. Caballero-Caballero, E. Ponce-Cueto // Sixth ACM Conference on Learning: June 24-25, 2019: Proceedings. – New York: Association for Computing Machinery, 2019. – Pp. 1-9. – DOI: 10.1145/3330430.3333634

295. Bowman R.F. A Pac-Man theory of motivation: Tactical implications for classroom instruction / R.F. Bowman // Educational Technology. – 1982. – №22 (9). – Pp.14-17.

296. Brusilovsky P. Adaptive and intelligent web-based educational systems / P. Brusilovsky, C. Peylo // International Journal of Artificial Intelligence in Education (IJAIED). – 2003. – №13. – С. 159-172.

297. Burbeck S. Applications programming in Smalltalk-80 (TM): How to use Model-View-Controller (1987): [Electronic resource] / S. Burbeck. – URL: http://st-www.cs.uiuc.edu/users/smarch/st-docs/mvc.html (Accessed: 01.08.2024)

298. Cantu M. Object Pascal Handbook / M. Cantu. – 1st. ed. – North Charleston, SC, USA: CreateSpace Independent Publishing Platform, 2015. – 534 p.

299. Carbonell J. AI in CAI: An Artificial-Intelligence approach to Computer-Assisted Instruction / J. Carbonell // IEEE Transactions on Man-Machine Systems. – 1970. – № 4. – Pp. 190-202. – DOI:10.1109/TMMS.1970.299942

300. Cardelli L. Squeak: A language for communicating with Mice / L. Cardelli, R. Pike // SIGGRAPH Computer Graphics, 1985. – №19 (3). – Pp. 199-204.

301. Chatzopoulou D.I. Adaptive assessment of students: Knowledge in programming courses / D.I. Chatzopoulou, A.A. Economides // Journal of Computer

Assisted Learning. – 2010. – № 26 (4). – Pp. 258-269. – DOI: 10.1111/j.1365-2729.2010.00363.x

302. Chen K.-T. Integration of paths, geometric invariants and a generalized Baker-Hausdorff formula / K.-T. Chen // Ann. of Math. – 1957. – № 65. – Pp. 163-178.

303. Chen K.-T. Integration of paths – a faithful representation of paths by non-commutative formal power series / K.-T. Chen // Trans. Amer. Math. Soc. –1958. – № 89. – Pp. 395-407.

304. Chevyrev I. A Primer on the signature method in machine learning / I. Chevyrev, A. Kormilitzin // Arxiv. – 2019. – DOI: 10.48550/arXiv.1603.03788

305. Clayton J.L. Theoretical implications of contemporary brain science for Japanese EFL Learning / J.L. Clayton // TESOL Journal. – 2015. – №6 (3). – Pp. 554-578.

306. Clements D.H. The future of educational computing research: The case of computer programming / D.H. Clements // Information Technology in Childhood Education Annual. – 1999. – №3. – Pp. 147-179.

307. Code Your Imagination – Robbo – Code Your Imagination: [Electronic resource]. – URL: https://www.robbo.world/?ysclid=lzbj4w1tx0716437153 (Accessed: 31.07.2024)

308. Coding set: Entry Level Hands-on Coding Robot Set for Age 4-9 // Matatalab: [Electronic resource]. – URL: https://matatalab.com/en/coding-set (Accessed: 01.08.2024)

309. Computational Thinking – a guide for teacher / A. Csizmadia, P. Curzon, M. Dorling, S. Humphreys, T. Ng, C. Selby, J. Woollard // Computing at School. Charlote BCS. The Chartered Institue for IT. – 2015. – URL: https://www.researchgate.net/publication/327302966_Computational_thinking_-_a_guide_for_teachers (Accessed: 18.08.2024)

310. Computational thinking for youth: White Paper for the ITEST Small Working Group on Computational Thinking (CT): 2010 / W. Allan, B. Coulter, J. Denner, J. Erickson, I. Lee, J. Malyn-Smith, F. Martin // Baskin School of Engineering:

[Electronic resource]. – URL: https://users.soe.ucsc.edu/~linda/pubs/ACMInroads.pdf (Accessed: 18.08.2024)

311. Computational thinking in compulsory education: Towards an agenda for research and practice / J. Voogt, P. Fisser, J. Good, P. Mishra, A. Yadav // Education and Information Technologies. – 2015. – Vol. 20, №4. – Pp. 715-728.

312. Computational thinking in elementary and secondary teacher education / A.Yadav, C. Mayfield, N. Zhou, S. Hambrusch, J.T. Korb // ACM Transactions on Computing Education (TOCE). – 2014. – Vol. 14(1). – Pp. 5-16.

313. Computer gaming and interactive simulations for learning: A Meta-analysis / J.J. Vogel, D.S. Vogel, J. Cannon-Bowers, C.A. Bowers., K. Muse, M. Wright // Journal of Educational Computing Research. – 2006. – Vol. 34(3). – Pp. 229-243.

314. Coursera – доступное онлайн-образование // Навигатор образования: Сетевое издание: [Электронный ресурс]. – URL: https://fulledu.ru/articles/1410_coursera-dostupnoe-onlain-obrazovanie.html (Accessed: 31.07.2024)

315. Cuny J.M. Demystifying computational thinking for noncomputer scientists. Unpublished manuscript in progress / J. Cuny, L. Snyder, J.M. Wing // Carnegie Mellon University: School of Computer Science: [Electronic resource]. – URL: http://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf (Accessed: 18.08.2024)

316. Dahl O.-J. Class and subclass declarations / O.-J. Dahl, K. Nygaard K. // Simulation Programming Languages: Proceedings from the IFIP Working Conference in Oslo, May 1967 / J. Buxton. – Oslo: North Holland, 1968. – Pp.1201-1211.

317. Dahl O.-J. SIMULA 67 Common Base Language / O.-J. Dahl, B. Myhrhaug, K. Nygaard. – Oslo: Norwegian Computing Center, 1968. – 99 p.

318. Darsac J. Premières leçons de programmation / J. Darsar. – Paris: CEDIC/FERNAND Nathan, 1980. – 221 p.

319. Deep knowledge tracing / C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. Guibas, J. Sohl-Dickstein // Neural Information Processing Systems :

Advances in neural information processing systems 28: 4-9 December 2017, Long Beach, California, USA. – Long Beach: Curran Associates, Inc. 2015. – P.1-12. – DOI: 10.48550/arXiv.1506.05908

320. Delorey D.S. Productivity? A case study using data from open source projects / D.S. Delorey // First International Workshop on Emerging Trends in FLOSS Research and Development (FLOSS'07: ICSE Workshops 2007), 20-26 May. – Minneapolis, MN, USA, 2007. – Pp. 1-5. – DOI: 10.1109/FLOSS.2007.5

321. Denning P. Beyond Computational Thinking: A CACM IT Profession Column / P. Denning // Communications of the ACM. – 2009. – Vol. 52(6). – Pp. 28-30.

322. Designing games for emotional health / R. Vacca, M. Bromley, J. Leyrer, M. Sprung, B. Homer // Learning, Education and Games / Ed. K. Schrier. – Richland, WA: ETC PressEditors, 2014. – Pp.123-140.

323. Drake C. Object-Oriented Programming with C++ and Smalltalk / C. Drake. – New Jersey: Prentice Hall, 1998. – 1028 p.

324. Ebashi T. Proto Mahjong: Mahjong tiles in the 19th Century / T. Ebashi // Mahjong Museum Report. – 2005. – Vol. 5, № 2, Is. 9. – Pp. 14-17.

325. Enseignement de l'informatique dans le secondaire en France. Un retourde balancier ? / G.-L. Baron, B. Drot-Delange, M. Grandbastien, F. Tort // Informatique en éducation : Perspectives curriculaires et didactiques. – Clermont-Ferrand, France: Presses universitaires Blaise-Pascal, 2015. – Pp. 83-104.

326. Gamut L.T.F. Logic, language and meaning: Volume 2: Intensional logic and logical grammar / L.T.F. Gamut. – London: The University of Chicago Press, 1982. – 182 p.

327. Gobbo F. From universal to programming languages / F. Gobbo, H. Durnova. – Amsterdam: University of Amsterdam, Amsterdam Center for Language and Communication, 2014. – 302 p.

328. Goldberg A. Smalltalk-80: The language and its implementation / A. Goldberg, D. Robson. – Boston: Addison-Wesley, 1983. – 744 p.

329. Graham B. Sparse arrays of signatures for online character recognition / B. Graham // Computer Vision and Pattern Recognition. – 2013. – URL: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.768.8229&rep=rep1&type=pdf (Accessed: 19.08.2024). – DOI: 10.48550/arXiv.1308.0371

330. Heift T. Web delivery of adaptive and interactive language tutoring: Revisited / Heift T. // International Journal of Artificial Intelligence in Education. – 2016. – Vol.26. – Pp. 489-503. – DOI: 10.1007/s40593-015-0061-0

331. Homer M. Combining tiled and textual views of code / M. Homer, J. Noble // 2nd IEEE Working Conference on Software Visualization (VISSOFT 2014): Victoria, British Columbia, Canada: 9-30 September 2014: Proceedings. – Victoria: IEEE, 2014. – Pp. 1-10. – DOI:10.1109/VISSOFT.2014.11

332. Hopper G.M. History of programming: ACM SIGPLAN: Keynote address // History of Programming Languages. – New York: ACM, 1978. – Pp. 8.

333. Hoyrup J. History of Mathematics Education in the European Middle: [Electronic resource] / J. Hoyrup // Handbook on History of Mathematics Education / Eds G. Schubring, A. Karp. – New York: APA, 2012. – URL: https://www.academia.edu/79503379/History_of_Mathematics_Education_in_the_European_Middle&nav_from=c218a98a-0f76-4a4f-94e3-6b39ac4b2212&rw_pos=0 (Accessed: 01.08.2024)

334. Hutchins J. Petr Petrovich Troyanskii (1894–1950): A forgotten pioneer of mechanical translation / J. Hutchins, E. Lovtskii // Machine Translation. – 2000. – №15. – Pp.187–221.

335. IDE и редактор кода для разработчиков // Visual Studio: [Электронный ресурс]. –URL: https://visualstudio.microsoft.com/ru/ (Accessed: 31.07.2024)

336. IPA Declaration of the Child's Right to Play // International Play Association (IPA World): [Electronic resource]. – URL: https://ipaworld.org/about-us/declaration/ipa-declaration-of-the-childs-right-to-play/ (Accessed: 01.08.2024)

337. Implementation of individual learning trajectories in LMS Moodle / F. Bensalah, M.P. Daniel, I. Patra, D.S. García, S. Irgasheva, R. Tsarev // Data Analytics in

System Engineering: CoMeSySo 2023: Lecture Notes in Networks and Systems. Vol 935. – Switzerland: Springer, Cham, 2024. – Pp. 159-174. – DOI: 10.1007/978-3-031-54820-8_14

338. Ingalls D. Design principles behind Smalltalk / D. Ingalls // Byte. – 1981. – №8. – Pp. 286-298.

339. Introduction to English Linguistics / I. Plag, M. Braun, S. Lappe, M. Schramm. – Berlin: De Gruyter, 2015. – 272 p.

340. Isozaki H. HPSG-Based preprocessing for English-to-Japanese Translation / H. Isozaki, K. Sudoh, H. Tsukada, K. DuhACM // Transactions on Asian Language Information Processing (TALIP). – 2012. – №11 (3). – Pp. 1-16. – DOI: 10.1145/2334801.2334802

341. Kay A. Microelectronics and the personal computer / A. Kay // Scientific American. –1977. – Vol. 237, Is. 3. – P. 230-244. – DOI: 10.1038/SCIENTIFICAMERICAN0977-230

342. Ke F. A Qualitative meta-analysis of computer games as learning tools / F. Ke // Handbook of research on effective electronic gaming in education. – 2009. – №1. – Pp. 1-32. – DOI:10.4018/978-1-59904-808-6.CH001

343. Kersten G.E. Application software and culture: Beyond the surface of software interface / G.E. Kersten, M.A. Kersten, W. Rakowski // Journal of Global Information Management. – 2002. – Vol. 10, Is. 4. – Pp.16. – DOI: 10.4018/jgim.2002100105

344. Knoll R. Pegasus: First steps toward a naturalistic programming language / R. Knoll, M. Mezini // The 21st ACM SIGPLAN Symposium on Object-oriented Programming Systems, Languages, and Applications (OOPSLA'06): Portland, Oregon, USA, 22-26 October: Proceedings. – Portland, 2006. – Pp. 543-559. – DOI: 10.1145/1176617.1176628

345. Kosugi H. Performative language and power: Japanese and Swearing / H. Kosugi: [Electronic resource] // SoLLs INTEC International Conference: Putrajaya,

Malaysia, 2009, 5-6 May. – Putrajaya, 2009. – Pp.1-15. – URL: http://www.ukm.my/solls09/Proceeding/PDF/Hana.pdf (Accessed: 01.08.2024)

346. LEGO®Education WeDo 2.0: Комплект учебных проектов: [Электронный ресурс]. – URL: https://education.lego.com/v3/assets/blt293eea581807678a/blteb267366ce34fc6b/5f880486f4f4cf0fa39d304d/teacherguide-ru-ru-v1.pdf (Дата обращения: 20.08.2024)

347. Lightbot: Solve Puzzles using Programming: [Electronic resource]. – URL: http://lightbot.com (Accessed: 20.08.2024)

348. Logical thinking and programming: General information: 1949-1950: [Electronic resource] / Keio University. – 48 p. – URL: http://crew-lab.sfc.keio.ac.jp/lectures/2011s_ronpro/pukiwiki/index.php?2011s (Accessed: 01.08.2024)

349. Luk W.S. ELFS: English Langauge from SQL / W.S. Luk, S. Kloster // ACM Transactions on Database Systems (TODS). – 1986. – №11 (4). – Pp. 447-472. – DOI: 10.1145/7239.384276

350. MOOC dropout prediction using a hybrid algorithm based on decision tree and extreme learning machine / J. Chen, J. Feng, X. Sun, N. Wu, Z. Yang, S. Chen // Mathematical Problems in Engineering. – 2019. – Vol.1. – Pp.1-11. – DOI: 10.1155/2019/8404653

351. Matsuzawa Y. Kotodama on Squeak: de manabu ronri shikō to puroguramingu / Y. Matsuzawa. – Tokyo: Keio Unviersity, 2008. – 205 p.

352. Mini-languages: A way to learn programming principles / P. Brusilovsky, E. Calabrese, J. Hvorecky, A. Kouchnirenko, P. Miller // Education and Information Technologies. – 1997. – №2 (1). – Pp. 65-83.

353. Myers B. Natural programming languages and environments / B. Myers, J. Pane, A. Ko // Communications of the ACM. – 2004. – Vol. 47, №9. – Pp. 47-52. – DOI: 10.1145/1015864.1015888

354. Noyelle Y. La Saga du LSE et de sa famille (LSD/LSG/LST) / Y. Noyelle // ACM SIGPLAN Notices. – 1995. – Vol. 30, Is.12. – Pp. 43-50. – DOI: 10.1145/219726.219747

355. Oancea B. Predicting students' results in higher education using neural networks / B. Oancea, R. Dragoescu, S. Ciucu // Applied Information and Communication Technologies (AICT2013), 25-26 April, 2013: International Conference. – Jelgava, Latvia: Latvia University of Agriculture, 2013. – Pp. 190-193.

356. Object oriented programming in Pascal: A graphical approach / D. Niguidula, A.Van Dam, D. Higuidula, C. Brookshire. – Boston: Addison-Wesley, 2002. – 616 p.

357. Okada K. Kotodama on squeak as attempt at educational programming language: [Electronic resource] / K. Okada. – URL: http://www.cmc.osaka-u.ac.jp/publication/for-2006/17-22.pdf (Accessed: 01.08.2024)

358. Operational definitions of computational thinking // ISTE, CSTA: [Electronic resource]. – URL: https://c.ymcdn.com/sites/www.csteachers.org/resource/resmgr/CompThinkingFlyer.pdf (Accessed: 18.08.2024)

359. Pakpahan F. Theory of cognitive development by Jean Piaget / F. Pakpahan, M. Saragih // Journal of Applied Linguistics. – 2022. – №2. – Pp.55-60. – DOI: 10.52622/joal.v2i2.79.

360. Pane J.F. A programming system for children that is designed for usability: Ph.D. thesis: [Electronic resource] / J.F. Pane. – Pittsburgh, PA: Carnegie Mellon University, 2002. – 190 p. – URL: https://john.pane.net/pdf/PaneThesis.pdf (Accessed: 01.08.2024)

361. Pane J.F. Studying the Language and Structure in Non-Programmers: Solutions to Programming Problems / J.F. Pane // International Journal of Human-Computer Studies. – 2001. – Vol., 54(2). – Pp. 237-264. – DOI: 10.1006/ijhc.2000.0410

362. Papert S. A Critique of Technocentrism in Thinking About the School of the Future // Educational Researcher. – 1987. – Vol. 16, №1. – C. 32-38.

363. Papert S. An exploration in the space of mathematics educations / S. Papert // International Journal of Com-600 puters for Mathematical Learning. – 1996. – Vol. 1(1). – Pp. 95-123.

364. Papert S. Mindstorms: Children, Computers and Powerful Ideas / S.Papert. – New York, USA: Basic Books Inc. Publishers, 1980. – 252 p.

365. Paz T. Introducing Computer Science to Educationally Disadvantaged High School Students / T. Paz, D. Levy // Science & Technological Education. – 2005. – № 23 (2). – Pp. 229-244.

366. Piaget J. Play, dreams and imitation in childhood / J. Piaget. – London: Heinemann, 1945. – 316 p.

367. Pisareva O.M. Adaptive digital educational environments as drivers of remote staff training technologies / O.M. Pisareva, A.G. Leonov, M.S. Dyachenko // Smart Nations: Global Trends In The Digital Economy: Proceedings of the International Scientific Conference. Vol. 398. – Cham, Switzerland: Springer International Publishing AG, 2022. — Pp. 360–366.

368. Problems of early learning to program: How to bridge the gap between pictographic and textual programming styles / D.B. Agliamutdinova, N.O. Besshaposhnikov, A.G. Kushnirenko, A.G. Leonov, M.V. Raiko // International Journal of Education and Information Technologies (NAUN). – 2021. – Vol.15. – Pp. 331-343.

369. Programming in Japanese for Literacy Education / K. Okada, M. Sugiura, Y. Matsuzawa, M. Araki, H. Ohiwa // History of Computing and Education 3 (HCE3): IFIP 20th World Computer Congress, Proceedings of the Third IFIP Conference on the History of Computing and Education WG 9.7/TC9, History of Computing, September 7–10, 2008, Milano, Italy. – Mödling: IFIP, 2008. – Pp.1 71-176. – DOI: 10.1007/978-0-387-09657-5_12

370. Proposal for a universal language for the description of computing processes / F.L. Bauer, H. Bottenbruch, H. Rutishauser, K. Samelson // Computer Programming and Artificial Intelligence / J.W. Carr. – Michigan: University of Michigan UP, 1958. – Pp. 355-373.

371. Protopsaltis S. Does online education live up to its promise? A look at the evidence and implications for federal policy / Protopsaltis S., Baum S. // Researchgate: [Electronic resource]. – URL: https://www.researchgate.net/publication/330442019_Does_Online_Education_Live_Up_to_Its_Promise_A_Look_at_the_Evidence_and_Implications_for_Federal_Policy (Accessed: 20.08.2024)

372. Pulido-Prieto O. A Survey of Naturalistic Programming Technologies / O. Pulido-Prieto, U. Juárez-Martínez // Computing Surveys (CSUR). – 2017. – Vol. 50 (5). – Pp.1-35. – DOI: 10.1145/3109481

373. PyCharm IDE для Data Science и веб-разработки на Python: [Электронный ресурс]. – URL: https://www.jetbrains.com/ru-ru/pycharm/ (Accessed: 31.07.2024)

374. Pélisse E. Pour une histoire de l'informatique dans l'enseignement français / E. Pélisse: [Electronic resource] // Système éducatif et révolution informatique: Collection Recherches, Les cahiers de la FEN. – 1985. – 192 p. – URL: https://edutice.archives-ouvertes.fr/file/index/docid/276158/filename/h85ep.htm (Accessed: 01.08.2024).

375. Report of a workshop on the scope and nature of computational thinking / National Academies of Science. – Washington DC: National Academies Press, 2010. – 108 p. – DOI: 10.17226/12840.

376. Rogozhkina I.B. PiktoMir: Teaching Programming Concepts to Preschoolers with a New Tutorial Environment / I.B. Rogozhkina, A.G. Kushnirenko // Procedia – Social and Behavioral Sciences. – 2011. – Vol. 28. – Pp. 601-605. – DOI:10.1016/j.sbspro.2011.11.114

377. SIF, Communiqué: La spécialité «Numérique et sciences informatiques» au lycée en 2022–2023: en progrès, mais peut et doit mieux faire // Bulletin. – 2023. – Vol. 1024. – Pp. 7-11. – DOI: 10.48556/SIF.1024.22.7

378. Savolainen R. Approaches to Socio-Cultural Barriers to Information Seeking / R. Savolainen // Library and Information Science Research. – 2016. – Vol. 38(1). – Pp. 52-59.

379. Scratch Jr: Coding for young children: [Electronic resource]. – URL: https://www.scratchjr.org (Accessed: 17.08.2024)

380. Scratch: Programming for all / M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk // Communications of the ACM. – 2009. – Vol. 52, №11. – Pp. 60-67. – DOI: 10.1145/1592761.1592779

381. Seargeant P. Idea of English in Japan: Ideology and the Evolution of a Global Language / P. Seargent. – Bristol: Channel View Publications, 2009. – 188 p. – DOI:10.1080/13488678.2010.10801275

382. Shockey N. Toward a new word order: Early Twentieth Century orthographic reform and its discontents: [Electronic resource] / N. Shockey // Japanese Language and Literature. – 2016. – Vol. 50 (2). – Pp. 303-345. – URL: http://www.jstor.org.ezproxy.staffs.ac.uk/stable/24892015 (Accessed: 01.08.2024).

383. Shut down or restart? The way forward for computing in UK schools. Technical report, London, 2012: The Royal Society / S. Furber [et al.]. – URL: http://royalsociety.org/uploadedFiles/Royal_Society_Content/education/policy/computing-in-schools/2012-01-12-Computing-in-Schools.pdf (Accessed: 18.08.2024)

384. Siani A. BYOD strategies in higher education: Current knowledge, students' perspectives, and challenges / A. Siani // New Directions in the Teaching of Physical Sciences. – 2017. – Vol.12 (12). – DOI:10.29311/ndtps.v0i12.824

385. Simon J.-C. L'éducation et l'informatisation de la société: Rapport remis au Président de la République le 31 août 1980 / J.-C. Simon. – Paris: La Documentation Française, 1980. – 275 p.

386. Stefik A. An Emprical Investigation into programming langauge Syntax / A. Stefik, S. Siebert // Transactions on Computing Education (TOCE). – 2013. – Vol. 13(4). – Pp.1-40. – DOI: 10.1145/2534973

387. Streib, J. T. Guide to assembly language: A concise introduction undergraduate topics in computer science / J. T. Streib. – Vol. 2. – Berlin: Springer Nature, 2020. – 344 p.

388. Stroustrup B. The C++ programming language / B. Stroustrup. – Boston: Addison-Wesley, 1986. – 720 p.

389. Supporting creation of FAQ dataset for E-Learning chatbot / Y. Sumikawa, M. Fujiyoshi, H. Hatakeyama, M. Nagai // Intelligent Decision Technologies. Smart Innovation, Systems and Technologies: Vol 142. – Singapore: Springer, 2019. – Pp.3-13. – DOI: 10.1007/978-981-13-8311-3_1

390. The Community for Open Collaboration and Innovation // Eclipse Foundation : [Electronic resource]. – URL: https://www.eclipse.org (Accessed: 17.08.2024)

391. The Most Spoken Languages 2023 // Statistics and Data: [Electronic resource]. – URL: https://statisticsanddata.org/data/the-most-spoken-languages-2023/ (Accessed: 17.08.2024)

392. The role of gamification in education a literature review / G. Surendeleg, V. Murwa, H. K. Yun, Y. S. Kim // Contemporary Engineering Sciences. – 2014. – Vol. 7, № 2932. – Pp.1609–1616.

393. The signature-based model for early detection of sepsis from electronic health records in the intensive care unit / J. Morrill, A. Kormilitzin, A. Nevado-Holgado, S. Swaminathan, S. Howison, T.J. Lyons // IEEE Conference Computing in Cardiology. – 2019. – Vol.46. – DOI: 10.22489/CinC.2019.014

394. Thomazo A. Statistiques et probabilités avec l'ordinateur / A. Thomazo. – Paris: Broché, 1981. – 305 p.

395. Valli H. Using Coursera for Campus in Your Teaching / Duke: Learning Innovation and Lifetime Education: [Electronic resource]. – URL: https://learninginnovation.duke.edu/blog/2020/07/using-coursera-for-campus-in-your-teaching/ (Accessed: 17.08.2024)

396. Wanner T. Personalising learning: Exploring student and teacher perceptions about flexible learning and assessment in a Flipped University Course / T. Wanner, E. Palmer // Computers & Education. – 2015. – Vol. 88. – Pp. 354-369.

397. Weintrop D. Comparing block-based and text-based programming in high school computer science classrooms / D. Weintrop, U. Wilensky // ACM Transactions on Computing Education. – 2018. – Vol. 18(1). – Pp. 1-25. – DOI: 10.1145/3089799

398. Wing J. Computational thinking / J.Wing // Communications of the ACM. – 2006. – Vol. 49. № 3. – P. 33-35.

399. Wing J.M. Computational thinking benefits society / J. Wing // TERC Blog: [Electronic resource]. – URL: https://blog.terc.edu/what-is-computational-thinking (Accessed: 18.08.2024)

400. Wing J.M. Research Notebook: Computational thinking – what and why? / J.M. Wing // The Link Magazine, 2011: Carnegie Mellon University: School of Computer Science: [Electronic resource]. – URL: https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why (Accessed: 18.08.2024)

401. Wolfram S. How to teach computational thinking / S. Wolfram // Stephen Wolfram: [Electronic resource]. – URL: http://blog.stephenwolfram.com/2016/09/how-to-teach-computational-thinking/ (Accessed: 18.08.2024)

402. Wolfram S. We've come a long way in 30 years (But you haven't seen anything yet!) // Stephen Wolfram: [Electronic resource]. – URL: https://writings.stephenwolfram.com/2018/06/weve-come-a-long-way-in-30-years-but-you-havent-seen-anything-yet/ (Accessed: 18.08.2024)

403. Wouters P. A Meta-analytic review of the role of instructional support in game-based learning / P. Wouters, H. Van Oostendorp // Computers & Education. – 2013. – Vol. 60(1). – Pp. 412-425.

404. Yakushi K. Kotodama in Ancient Sociolinguistic Concepts: [Electronic resource] / K. Yakushi // Mejiro University. – 2013. – Vol. 9. – Pp. 141-150. – URL: https://ci.nii.ac.jp/els/contentscinii_20180127054949.pdf?id=ART0010078403 (Accessed: 01.08.2024)

405. Yano Y. World Englishes in 2000 and Beyond / Y. Yano // World Englishes. – 2001. – Vol.20. – Pp. 119-132. – DOI:10.1111/1467-971X.00204

406. Zuse K. Über den allgemeinen Plankalkül als Mittel zur Formulierung schematisch-kombinativer Aufgaben / K. Zuse // Arch. Math. – 1948/49. – Vol. 1. – Pp. 441-449.

Приложение №1

С П И С О К
организаций, входящих в состав сетевой инновационной площадки по теме «Апробация и внедрение основ алгоритмизации и программирования для дошкольников и младших школьников в цифровой образовательной среде ПиктоМир» на основании приказа от 18.11.2020 № П-188

Руководитель рабочей группы – Леонов А.Г., заведующий кафедрой ДПО

Самарская область

1. МБОУ Школа № 12 г.Самара
2. МБДОУ «Детский сад № 2» г.о.Самара
3. МБДОУ «Детский сад № 96» г.о.Самара
4. МБДОУ «Детский сад №244» г.Самара
5. МБДОУ «Детский сад № 365» г.о. Самара
6. МАДОУ «Центр развития ребенка - детский сад № 374» г.Самара
7. ГБОУ СОШ ОЦ «Южный город» СП «Детский сад «Лукоморье» м.р.Волжский, п.Придорожный, Самарской обл.
8. ГБОУ СОШ «ОЦ «Южный город» СП «Детский сад «Семицветик» п.Придорожный, Самарской обл.
9. ЧОУ СОШ «Общеобразовательный центр «Школа» г.Тольятти, Самарской обл.
10. МБУ детский сад № 54 «Алёнка» г.Тольятти, Самарской обл.
11. МБУ детский сад №76 «Куколка» г.Тольятти, Самарской обл.
12. МБДОУ детский сад № 125 "Росточек" г.Тольятти, Самарской обл.
13. МБУ детский сад № 138 «Дубравушка» г.Тольятти, Самарской обл.
14. МАДОУ детский сад № 200 «Волшебный башмачок г.о. Тольятти, Самарской обл.

15. ГБОУ СОШ № 3 СП - детский сад №19 «Колокольчик» г.о.Чапаевск, Самарской обл.
16. ГБОУ СОШ № 22 СП – детский сад № 28 «Ёлочка» г.Чапаевск, Самарской обл.
17. ГБОУ СОШ № 13 СП - детский сад № 29 «Кораблик» г.Чапаевск, Самарской обл.
18. СП «Детский сад «Аленький цветочек»» ГБОУ СОШ №2 с.Приволжье, Самарской обл.
19. СП ГБОУ СОШ "Образовательный центр" детский сал "Чайка" с.Утевка, Самарской обл.
20. ГБОУ лицей №16 СП детский сад «Вишенка» г.Жигулевск, Самарской обл.
21. ГБОУ лицей №16 СП детский сад "Красная шапочка" г.Жигулевск, Самарской обл.
22. ГБОУ СОШ №1 СП- детский сад «Золотой ключик» п.г.т Суходол, Самарской обл.
23. ГБОУ СОШ №1 СП – детский сад «Аленушка» п.г.т. Суходол, Самарской обл.
24. МАДОУ «Детский сад №403» г. Самара
25. Детский сад «Мозаика» при АНО Православная классическая гимназия г. Самара
26. АНО ДО «Планета детства «Лада» детский сад № 97 «Хрусталик» г. Тольятти
27. АНО ДО «Планета детства «Лада» детский сад № 164 «Весточка» г. Тольятти
28. АНО ДО «Планета детства «Лада» детский сад №157 «Светлячок» г. Тольятти
29. АНО ДО «Планета детства «Лада» детский сад № 137 «Чижик» г. Тольятти
30. АНО ДО «Планета детства «Лада» детский сад №207 "Эдельвейс" г. Тольятти
31. МАОУ д/с №210 "Ладушки" г. Тольятти
32. МБУ детский сад № 43 «Гнездышко» г. Тольятти

33. МБУ детский сад №76 "Куколка" г. Тольятти
34. ГБОУ СОШ №10 СПДС "Алёнушка" г. Жигулевск Самарская область
35. СП ДС «Ветерок» ГБОУ СОШ «ОЦ» п. Серноводск, Самарская область
36. ГБОУ СО СОШ № 10 имени Героя России Сергея Анатольевича Хихина, г.о.Чапаевск
37. СП «Детский сад «Чудо-Град» ГБОУ СОШ «ОЦ «Южный город» пос. Придорожный
38. СП «Детский сад Крепыш» ГБОУ СОШ № 3 г. Похвистнево
39. МБДОУ «Детский сад № 110» г. о. Самара
40. МАДОУ «Детский сад № 172» г. о. Самара
41. МАДОУ «ЦРР – детский сад № 375» г. о. Самара
42. МБДОУ «Детский сад № 4» г.о. Самара
43. МБДОУ «Детский сад №149» г.о.Самара
44. МБДОУ «Детский сад общеразвивающего вида №290» г.о.Самара
45. ГБОУ СОШ № 10 СП «детский сад «Березка», город Чапаевск
46. МБУ «Школа № 89», г. Тольятти
47. ГБОУ СОШ № 1 «ОЦ» пгт Смышляевка СП «Детский сад «Янтарик»
48. МБУ детский сад № 196 «Маячок» г.о. Тольятти
49. ГБОУ гимназия №1 г. Новокуйбышевска
50. МБДОУ «Детский сад № 153» г. о. Самара
51. МБДОУ «Детский сад № 33» г.о. Самара
52. МБДОУ «Детский сад № 229» г.о. Самара
53. МБДОУ «Детский сад № 296» г.о. Самара
54. МАДОУ «Детский сад № 50» г.о.Самара
55. МБДОУ «Детский сад № 61» г.о. Самара
56. МБОУ Школа №154 г.о. Самара
57. ГБОУ СОШ №1 «ОЦ» с. Борское
58. МБДОУ «Детский сад №12» г.о. Самара
59. МБДОУ «Детский сад № 411» г.о. Самара

60. СП ГБОУ СОШ №9 г.о. Чапаевск д/с №10 «Планета детства»
61. МБДОУ «Детский сад № 365» г.о. Самара
62. ГБОУ НШ с. Красноармейское
63. МБДОУ «Детский сад комбинированного вида № 255» г.о. Самара

Кемеровская область

64. МАДОУ «Детский сад № 10» "Аленький цветочек" г.Прокопьевск, Кемеровской обл.
65. МБОУ «СОШ № 8», г. Новокузнецк
66. МБОУ «Средняя общеобразовательная школа №9», г. Таштагол
67. МАДОУ Новосафоновский детский сад "Солнышко", п. Новосафоновский
68. МБДОУ детский сад № 99 города Белово
69. МБ ДОУ «Детский сад № 120», г. Новокузнецк

Московская область

70. МАДОУ «Образовательный центр «Успех» г.Троицк, Московской обл.
71. МБДОУ Детский сад №49 п. Красногорск
72. МДОУ Детский сад № 75 комбинированного вида г. Орехово-Зуево
73. МДОУ Детский сад п. Мирный
74. МДОУ Детский сад №43 «Лучик» г. Подольск
75. МДОУ Детский сад №55 «Кузнечик» г. Подольск
76. МДОУ Детский сад комбинированного вида № 2 «Ромашка» г. Талдом
77. МБОУ начальная общеобразовательная школа № 17 г. Красногорск
78. МДОУ центр развития ребёнка - детский сад № 39 «Золотой ключик» г. Серпухов
79. МДОУ Детский сад комбинированного вида № 3 г. Фрязино
80. МДОУ Детский сад № 53 г. Орехово-Зуево

81. МДОУ Липицкий детский сад комбинированного вида «Колосок» г. Серпухов
82. МДОУ Детский сад комбинированного вида №58 «Радость» г.о. Люберцы
83. МАДОУ Детский сад №5 «Непоседы» г. Домодедово
84. МАДОУ «Детский сад комбинированного вида №2 «Радуга» г. Зарайск
85. МАДОУ «Детский сад №1 «Лесная полянка» г. Зарайск
86. МАДОУ «Детский сад №12 «Ягодка» г. Зарайск
87. МАДОУ детский сад № 23 «Золотой ключик», г. Домодедово
88. МБДОУ «Детский сад комбинированного вида № 2 «Василек», г.о. Реутов
89. МДОУ детский сад № 94, г.о. Люберцы
90. МОУ Индустринская ООШ г.о. Коломна
91. МАДОУ детский сад № 4 комбинированного вида г. Звенигород
92. МБДОУ детский сад № 27 «Тополёк», Коломенский г.о.
93. МБДОУ детский сад комбинированного вида № 14 «Радуга», г. Лобня
94. МБДОУ детский сад № 21, п. Отрадное
95. МБДОУ Центр развития ребенка - детский сад № 1 «Чайка», г. Лобня
96. МОУ СОШ №4, Орехово-Зуевский г.о.
97. МБОУ Лицей №1 им. Г.С. Титова, г. Краснознаменск
98. МБОУ «Гимназия», г.о. Протвино
99. МОУ Гимназия №1, г.о. Жуковский
100. МБОУ СОШ «Горки-Х», пос. Горки-10
101. МОУ гимназия № 14, Орехово-Зуевский г.о.
102. МБДОУ «Детский сад N97 «Солнышко», Богородский г.о.
103. МОУ школа № 2, г. Жуковский
104. МБОУ «Начальная школа - детский сад 14», г. Зарайск
105. МБОУ СОШ №3 имени Героя Советского Союза В.А. Борисова, г. Лобня
106. МБОУ НОШ №17, г. Красногорск
107. МБОУ СОШ № 14, Коломенский г.о.

108. МБОУ «Средняя общеобразовательная школа №25», Сергиево-Посадский г.о.
109. МБОУ «Лицей №5», г. Зарайск
110. МБДОУ «Детский сад № 97 «Солнышко», г. Ногинск-9
111. МБОУ «Лицей», г. Протвино
112. МБОУ Лицей «Серпухов», Серпухов
113. МБОУ СОШ № 10, Коломна
114. МАДОУ № 56 «Ромашка», г.о. Химки
115. МБОУ Опалиховская гимназия, г.о. Красногорск
116. МАОУ «Лицей №15», г. Мытищи
117. ГАОУ МО «Балашихинский лицей», г. Балашиха
118. МАОУ Домодедовская СОШ №1, г. Домодедово
119. МАООУ «СЛШ Полянка», г.Балашиха
120. МДОУ д/с №25 компенсирующего вида, Орехово-Зуевский г.о.
121. ЧДОУ «Детский сад № 36 ОАО «РЖД», г. Москва
122. МОУ СОШ №4 с угл. изучением отд. предм. г.Фрязино МО
123. МАДОУ «Детский сад комбинированного вида №10 "Улыбка"», г. Зарайск
124. МБОУ «Фруктовская СОШ», г.о. Луховицы
125. МБОУ г. Красноармейск «СОШ №1»
126. МДОУ детский сад № 24, г.Орехово-Зуево
127. МБДОУ «Детский сад № 2», г. Кашира
128. МОУСОШ пос. МИС, г.о. Подольск
129. МОУ Кабановская СОШ, Орехово-Зуевский г.о.
130. МОУ СОШ № 15, Жуковский
131. МАДОУ «Детский сад №19 «Зёрнышко», пос. Зарайский, г. о. Зарайск
132. МАОУ «Гимназия №3», Балашиха
133. МАДОУ «Детский сад комбинированного вида №10 "Улыбка"», г.о. Зарайск
134. МБОО «Гимназия» г.о. Фрязино МО

135. МАОУ СОШ №8 ГОЩ, г. Щёлково
136. МБОУ «Гимназия», г. Протвино
137. МБОУ ЦО № 83, г. Ногинск-9
138. МАОУ Домодедовская СОШ №8 (дошкольное отделение «Золотой ключик»), г. Домодедово
139. МБОУ СОШ №8 имени Героя Советского Союза Будника Г.Д. ДО «Радуга», г. Лобня
140. МБОУ СОШ №19 имени Романа Катасонова, г.о. Серпухов
141. МОУ Песковская СОШ, г. Коломна
142. МБОО СОШ №3 имени Героя Советского Союза А.Г. Дудкина г.о. Фрязино МО
143. МОУ СОШ №2 г. Талдома
144. МОУ СОШ №22, г. о. Подольск
145. МАОУ СОШ №3 с УИОП, г.о. Балашиха
146. МОУ СОШ № 22 Орехово-Зуевского городского округа
147. МОУ «Запрудненская гимназия», Талдомский городской округ, г.п. Запрудня
148. МБОУ «Гимназия №2», г. Зарайск
149. МБОУ «Гимназия №5» (дошкольное отделение), г.о. Королёв
150. МАОУ «ВХТЛ», г. Видное
151. МОУ Квашёнковская СОШ, Талдомский городской округ, с. Квашёнки
152. МОУ СОШ №36, г.о. Подольск
153. МБОУ СОШ №9, г. Лобня
154. МАОУ Востряковский лицей №1, г. Домодедово
155. МБОУ СОШ №12, город Коломна
156. МОУ СОШ №22, г. Орехово-Зуево
157. МБОУ Лицей №6 им. академика Г.Н. Флерова, г. Дубна
158. МБОУ «СОШ №3», г. Протвино
159. МАОУ Домодедовская СОШ №12 им. полного кавалера ордена Славы В.Д. Преснова

160. Раменская средняя школа № 5, г. Раменское
161. МОУ СОШ № 11, г. Подольск
162. МОУ гимназия № 16 «Интерес», Люберцы
163. МОУ «Средняя общеобразовательная школа №33» г.о. Подольск
164. МОУ Средняя общеобразовательная школа №10, г.о. Орехово-Зуевский
165. МБОУ СОШ № 1, городской округ Лосино-Петровский
166. МОУ СОШ №33, г. Подольск
167. МБОУ «Гимназия №9», г. Балашиха
168. МБОУ «Луховицкая СОШ № 2» (дошкольное отделение «Колокольчик») г. Луховицы

Чувашская республика

169. МБДОУ "Детский сад № 160 общеразвивающего вида с приоритетным осуществлением деятельности по художественно-эстетическому развитию детей» г. Чебоксары Чувашской Республики
170. МБДОУ «Детский сад общеразвивающего вида с приоритетным осуществлением деятельности по социально-личностному развитию детей № 40 «Радость» г. Новочебоксарск Чувашской Республики
171. МБОУ «Детский сад № 16 «Красная Шапочка» г. Новочебоксарск Чувашской Республики
172. МБДОУ «Детский сад №113» г. Чебоксары
173. МАДОУ «Детский сад №200» г. Чебоксары
174. МБДОУ №Детский сад №206 "Антошка" г. Чебоксары
175. МБДОУ «Детский сад № 208» г. Чебоксары
176. МБДОУ «Детский сад № 209 «Эврика» г. Чебоксары
177. МБДОУ «Детский сад 210» г. Чебоксары
178. МБДОУ «Детский сад общеразвивающего вида № 2 «Калинка» г. Новочебоксарск

179. МБДОУ «Детский сад общеразвивающего вида с приоритетным ОД по художественно-эстетическому развитию детей №7 «Березка» г. Новочебоксарск
180. МБДОУ «Детский сад № 52 «Телей» г. Новочебоксарск
181. МБДОО «Детский сад №7 «Солнечный город» г. Цивильск
182. МБДОУ "Детский сад №6 "Сказка" Цивильского района
183. МАОУ «СОШ №61» г. Чебоксары
184. МБДОУ «Детский сад комбинированного вида № 48 «Журавлик», г. Новочебоксарск
185. МБДОУ «Детский сад №1 «Золотой ключик» города Шумерля
186. Вурнарский сельскохозяйственный техникум Минобразования Чувашии, пгт. Вурнары
187. МБОУ «СОШ № 57» г. Чебоксары
188. ЦЦОД «IT-Куб. Ядрин», г. Ядрин
189. МБДОУ «Детский сад №207» г. Чебоксары
190. МБДОУ «Детский сад №89» г. Чебоксары
191. МБДОУ "Детский сад №209 "Эврика" г. Чебоксары
192. МБДОУ «Детский сад №205 «Новоград» города Чебоксары
193. МБДОУ «Детский сад №130» г.Чебоксары
194. МБДОУ «ЦРР - детский сад № 185» г. Чебоксары

Свердловская область

195. МАДОУ «Детский сад № 3» г.Североуральск, Свердловской обл.
196. МАДОУ "Детский сад №38 "Теремок" г. Сысерть, Свердловская область
197. МАДОУ "Детский сад № 70" г. Первоуральск, Свердловская область,
198. Филиал МАДОУ "Детский сад № 70"- "Детский сад № 38", г. Первоуральск Свердловская область
199. МАДОУ Детский сад № 5 г. Первоуральск

200. МАДОУ Детский сад «Росинка» г. Новоуральск
201. МАДОУ детский сад «Умка», г. Карпинск
202. МБОУ ДО «ЦДО», г. Нижняя Тура
203. МАДОУ «Детский сад комбинированного вида № 25» АГО
204. МАДОУ «Радость» структурное подразделение — детский сад № 82, г. Нижний Тагил
205. МАОУ Политехническая гимназия, г. Нижний Тагил
206. МАДОУ «Радость» структурное подразделение — детский сад № 202, г. Нижний Тагил
207. МАОУ СОШ № 56, г. Артемовский
208. МАОУ «Лицей № 5», г. Камышлов
209. МАДОУ д/с «Детство» - структурное подразделение д/с № 194, г. Нижний Тагилсан
210. МКДОУ «Детский сад № 32 «Малыш», г. Асбест
211. МБДОУ № 28 «Колокольчик», г. Реж
212. МАОУ «СОШ № 10», г. Ревда
213. МАДОУ «Детский сад №2», г. Тавда
214. МАДОУ № 38 «Теремок», г. Сысерть
215. МДОУ «ЦРР - детский сад «Улыбка», г. Качканар
216. МБДОУ НГО «Детский сад №17 «Солнышко», Новолялинский ГО, п.Лобва
217. МБДОУ ДС № 88 г. Каменск-Уральский
218. МБДОУ ДС № 10 г. Алапаевск

Новосибирская область

219. МАОУ «Образовательный центр – гимназия № 6 «Горностай» г.Новосибирск
220. МБДОУ "Детский сад комбинированного вида №7 "Радуга" Барабинского района Новосибирской обл.
221. ДОУ «Родничок» г. Куйбышев, Новосибирская области

222. МКДОУ детский сад «Солнышко» Кыштовского района, Новосибирская область
223. МАОУ СОШ № 216, г. Новосибирск
224. МКДОУ д/с № 494, г. Новосибирск
225. МБОУ-ЛИЦЕЙ, г. Татарск
226. МБОУ СОШ № 96 с углубленным изучением английского языка, г. Новосибирск
227. МКДОУ - детский сад «Лукоморье», п. Элитный
228. МБОУ СОШ № 77, г. Новосибирск
229. МКОУ Краснозерский лицей № 2 имени Ф.И. Анисичкина, рп Краснозерское
230. МАОУ ЦО «Развитие», г. Новосибирск
231. Гимназия №1 им. А.Л. Кузнецовой, г. Куйбышев
232. МКДОУ д/с №478 «Белоснежка», г. Новосибирск
233. МБДОУ №8, г. Барабинск
234. МБДОУ д/с № 420, г. Новосибирск
235. ДОУ «Жемчужинка», г. Куйбышев
236. МБДОУ д/с № 448, г. Новосибирск

Республика Башкортостан

237. МАДОУ детский сад № 6 г. Нефтекамск Республики Башкортостан
238. МБДОУ Детский сад №234 г. Уфа
239. МАДОУ Детский сад №212 г. Уфа
240. МБДОУ Детский сад № 209 г. Уфа
241. МБДОУ Детский сад № 205 г. Уфа
242. МДОБУ ЦРР – детский сад «Айгуль» г. Сибай
243. МДОБУ детский сад «Ласточка» г. Сибай
244. МДОБУ ЦРР – детский сад «Светлячок» г. Сибай

245. МБДОУ «Центр развития ребенка - детский сад «Аленушка» г. Сибай
246. МАДОУ детский сад № 43 г. Нефтекамск
247. МАДОУ «Детский сад №2» г. Стерлитамак
248. МДОБУ ЦРР Детский сад «Березка» г. Сибай
249. МАДОУ «Детский сад № 56» г. Стерлитамак РБ
250. МОАУ «Гимназия №1», г. Нефтекамск
251. МБДОУ Д/с №1 «Ромашка», г. Давлеканово
252. МАДОУ «Детский сад №34» – городского округа г. Стерлитамак РБ
253. МДОБУ детский сад «Фантазия» д. Подымалово
254. МАДОУ д/с № 27 «Колокольчик», г. Кумертау
255. МАДОУ «Детский сад №64» г. Стерлитамак РБ

Ленинградская область

256. МБДОУ «Детский сад №51 комбинированного вида» г. Гатчина Ленинградской обл.
257. МБДОУ «Детский сад комбинированного вида № 37» г. Кировск
258. МБДОУ «Детский сад №52 комбинированного вида», г. Гатчина
259. МДОУ «Детский сад Рябинка», г. Тихвин
260. МДОУ «Сланцевский детский сад №7», г. Сланцы
261. МДОУ №16 с. Копорье
262. МДОУ «Детский сад № 24», г. Кириши
263. МБДОУ «Детский сад №44 комбинированного вида», пос.Новый Учхоз
264. МБДОУ № 4, г. Кировск
265. МОБУ «СОШ» Агалатовский ЦО», д. Агалатово
266. МКДОУ №24, Кировский район, с. Путилово

Республика Бурятия

267. МБДОУ «Детский сад № 27 «Сэсэг» г. Улан-Удэ»

268. МБОУ детский сад № 29 «Искорка» комбинированного вида г. Улан-Удэ
269. МБДОУ "Детский сад № 35 "Алые паруса" г. Улан-Удэ
270. МАДОУ детский сад № 41 "Ласточка" г. Улан-Удэ
271. МАДОУ Детский сад № 59 «Золотой ключик»" г. Улан-Удэ
272. МАДОУ "Детский Сад № 62 "Малыш" комбинированного вида г. Улан-Удэ
273. МБДОУ детский сад № 71 «Огонек» г. Улан-Удэ
274. МБДОУ детский сад № 72 «Алёнушка» комбинированного вида г. Улан - Удэ
275. МБДОУ «Детский сад №89 «Журавленок» компенсирующего вида г. Улан-Удэ
276. МБДОУ «Детский сад № 104 «Зорька» комбинированного вида г. Улан-Удэ
277. МБДОУ "Детский сад № 112 "Сибирячок" комбинированного вида" г. Улан-Удэ
278. МБДОУ детский сад № 8 «Огонёк» общеразвивающего вида Селенгинский район, г. Гусиноозерск
279. МБДОУ Центр развития ребенка – детский сад № 13 «Радуга» г. Гусиноозерск
280. МАОУ "Санагинская средняя общеобразовательная школа" Закаменский район, с. Санага
281. МАОУ "Саган-Нурская средняя общеобразовательная школа" п. Саган - Нур Мухоршибирский район
282. МБДОУ Онохойский детский сад "Солнышко" Заиграевский район, п. Онохой
283. МБДОУ Заиграевский центр развития ребенка - детский сад "Улыбка" Заиграевский район, п. Заиграево
284. МАОУ "Кабанская средняя общеобразовательная школа" Кабанский район, село Кабанск
285. МБДОУ детский сад № 1 «Солнышко» Селенгинский район, улус Тохой
286. ЧДОУ «Детский сад № 230 ОАО «РЖД» г. Северобайкальск
287. МАДОУ Детский Сад № 64 «Колокольчик» г. Улан-Удэ

288. МБДОУ Детский сад «Теремок», с. Бичура
289. МБДОУ № 38 г. Улан-Удэ
290. МБДОУ № 88 «Ладушки», г. Улан-Удэ
291. МАДОУ Детский сад «Радуга», с. Сосново-Озерское
292. МБДОУ № 112 «Сибирячок», г. Улан-Удэ
293. МБДОУ Детский сад «Огонек», с. Бичура

Тюменская область

294. МАДОУ «Детский сад «Березка» г. Белоярский
295. МАДОУ д/с № 153 города Тюмени
296. МАОУ «СОШ №4» г. Ялуторовск
297. МАОУ «Нижнеаремзянская СОШ», д. Нижние Аремзяны
298. МАДОУ «Детский сад № 40 — ЦРР» г. Тобольска
299. МАОУ СОШ № 16 имени В.П. Неймышева, г. Тобольск
300. МАОУ «Байкаловская СОШ» — детский сад «Василёк» с. Байкалово
301. МАДОУ д/с № 186 города Тюмени
302. МАОУ «Бизинская СОШ», с.Бизино
303. МАДОУ Голышмановский ЦРР – д/с № 4 «Ёлочка», р.п. Голышманово
304. МАОУ СОШ № 88 г. Тюмени

Иркутская область

305. МБДОУ «Детский сад комбинированного вида № 115» МО г. Братск
306. МБДОУ Детский сад № 15 «Ручеек» г. Усть-Илимск
307. МБДОУ Детский сад № 22 «Искорка» г. Усть-Илимск
308. МБДОУ «Центр развития ребенка - детский сад «Гармония», г. Тулун
309. МБДОУ «Детский сад № 17 «Сказка», г. Усть-Илимск
310. МБДОУ детский сад № 78, г. Иркутск

311. МБДОУ детский сад № 184, г. Иркутск
312. МОБУ НОШ № 24 р.п. Чунский
313. МБОУ г. Иркутска СОШ №77
314. МДОУ № 24 г. Черемхово
315. МБОУ г. Иркутска СОШ №17
316. МБДОУ г. Иркутска детский сад №31
317. МКОУ «Школа-интернат № 26 г. Нижнеудинск»
318. МБОУ «СОШ № 17», г. Ангарск
319. МДОУ ИРМО «Хомутовский Детский Сад №4», с. Хомутово
320. МОУ «Эдучанская СОШ», пос. Эдучанка
321. МБОУ «СОШ № 42», г. Братск
322. МБДОУ «Лузгиновский детский сад», Осинский район, д. Лузгина
323. МБДОУ № 86, г. Ангарск
324. МДОУ "Малыш", Усть-Илимский район, р.п.Железнодорожный
325. МДОУ ДС общеразвивающего вида № 39 УКМО
326. МБДОУ детский сад №48, Ангарский район, р.п. Мегет
327. МДОУ ЦРР Детский сад №24, г. Усть – Кут
328. МБДОУ «ЦРР-ДС № 97», г. Братск
329. МДОУ ДС № 3 УКМО
330. МБДОУ «Детский сад № 28», г. Усолье-Сибирское
331. МКДОУ детского сада общеразвивающего вида «Умка», Братский район, г. Вихоревка
332. МБОУ СОШ № 45 г. Братск
333. МБДОУ ДС № 108 г. Ангарск

Ростовская область

334. МБДОУ «Детский сад № 70» г. Ростов-на-Дону
335. МБДОУ Детский сад № 30 г. Батайск
336. МБДОУ Детский сад № 25 г. Батайск

337. МБДОУ Детский сад № 44 г. Новочеркасск
338. МБДОУ Детский сад № 1 г. Новочеркасск
339. МАДОУ «Детский сад № 315», г. Ростов-на-Дону
340. МБДОУ «Детский сад № 292», г. Ростов-на-Дону
341. МБДОУ детский сад № 24, г. Ростов-на-Дону
342. МБОУ «Лицей № 103», г. Ростов-на-Дону
343. МБДОУ «Детский сад № 121», г. Ростова-на-Дону
344. МБДОУ № 266, г. Ростов-на-Дону
345. МБДОУ «Детский сад №111», г. Ростов-на-Дону
346. МБДОУ №29, г. Ростов-на-Дону
347. МАОУ «Юридическая гимназия № 9 имени М.М. Сперанского», г. Ростов-на-Дону
348. МБДОУ ДС «Аленький цветочек» г. Волгодонска
349. МАДОУ №1, г. Ростов-на-Дону
350. МБДОУ ДС «Казачок» г. Волгодонска
351. МБДОУ № 85, г. Ростов-на-Дону
352. МАДОУ № 301, г. Ростов-на-Дону
353. МБДОУ № 21, г. Ростов-на-Дону
354. МБДОУ № 219, г. Ростов-на-Дону
355. МБДОУ № 137, г. Ростов-на-Дону
356. МАДОУ № 232, г. Ростов-на-Дону
357. МБДОУ № 312, г. Ростов-на-Дону
358. МБДОУ № 5, г. Ростов-на-Дону
359. МАДОУ № 49, г. Ростов-на-Дону
360. МБДОУ № 7, г. Ростов-на-Дону
361. МБДОУ № 207, г. Ростов-на-Дону
362. МБДОУ № 254, г. Ростов-на-Дону
363. МАДОУ № 272, г. Ростов-на-Дону
364. МБДОУ № 210, г. Ростов-на-Дону

365. МБДОУ № 284, г. Ростов-на-Дону
366. МБДОУ № 237, г. Ростов-на-Дону
367. МБДОУ № 63, г. Ростов-на-Дону
368. МБДОУ № 22, г. Ростов-на-Дону
369. МБДОУ № 107, г. Ростов-на-Дону
370. МАДОУ № 267, г. Ростов-на-Дону
371. МБДОУ № 36, г. Ростов-на-Дону
372. МБДОУ № 74, г. Ростов-на-Дону
373. МБДОУ № 225, г. Ростов-на-Дону
374. МБДОУ № 263, г. Ростов-на-Дону
375. МБДОУ № 229, г. Ростов-на-Дону
376. МБОУ «Гимназия № 117», г. Ростов-на-Дону
377. МАДОУ 108, г. Ростов-на-Дону
378. МБДОУ № 8, г. Ростов-на-Дону
379. МБДОУ № 278, г. Ростов-на-Дону
380. МБДОУ № 142, г. Ростов-на-Дону
381. МБДОУ д/с «Сказка» г. Зернограда
382. МБДОУ № 2, г. Ростов-на-Дону
383. МАДОУ №42, г. Ростов-на-Дону
384. МБДОУ Д/с «Сказка», г. Семикаракорск
385. МБДОУ «Детский сад №215», г. Ростов-на-Дону
386. МБДОУ Детский сад «Родничок», г. Семикаракорск
387. МБУ ДО ДТДМ, г. Ростов-на-Дону
388. МБДОУ № 37, г. Ростов-на-Дону
389. МБДОУ № 275, г. Ростов-на-Дону
390. МБДОУ № 25, г. Ростов-на-Дону
391. МБДОУ д/с «Здоровый ребенок», г. Таганрог
392. МБДОУ № 56, г. Ростов-на-Дону
393. МБДОУ № 276, г. Ростов-на-Дону

394. МАДОУ № 59, г. Ростов-на-Дону
395. МБДОУ № 251, г. Ростов-на-Дону
396. МБДОУ ДС № 198 г. Ростов-на-Дону

Челябинская область

397. МАДОУ Детский сад № 453 г. Челябинск
398. МАДОУ Детский сад комбинированного вида №73 г. Златоуст
399. МБДОУ Детский сад № 472 г. Челябинск
400. МДОУ «ЦРР - д/с № 137» г. Магнитогорска
401. МДОУ «Д/с № 49 о.в.» г. Магнитогорска
402. МОУ «СОШ №3», г. Кыштым
403. МБДОУ д/с № 1 г.Куса
404. МБДОУ «ДС №447 г. Челябинска»
405. МДОУ «Д/с №3 «Золотой ключик», г. Южноуральск
406. ЧОУ «Начальная школа - детский сад №67 ОАО «РЖД», г. Челябинск
407. МАДОУ «Детский сад №155 г. Челябинска»
408. МАДОУ «ДС № 23 г. Челябинска»
409. МАДОУ ДС № 449 Олимпиец г. Челябинск

Орловская область

410. МБДОУ Детский сад № 96 г. Орел
411. МБДОУ детский сад № 92, г. Орла
412. МБОУ- школа № 52 г.Орла
413. БОУ ОО «Мезенский лицей», с.Плещеево

Тамбовская область

414. МБДОУ Детский сад №59 «Ягодка» г. Тамбов
415. МАУ ДОО Детский сад №9 комбинированного вида г. Рассказово
416. МАДОУ «Детский сад № 12 Ягодка», г. Тамбов
417. МБДОУ «Детский сад № 52 «Маячок», г. Тамбов

Ямало-Ненецкий автономный округ

418. МАДОУ Детский сад «Калинка» г. Новый Уренгой
419. МБДОУ «Золотая рыбка» г. Ноябрьск
420. МБДОУ «Колокольчик» г. Ноябрьск
421. МАДОУ «Надежда», г. Ноябрьск
422. МБДОУ «ДС «Снегурочка», г. Новый Уренгой
423. МБДОУ «Крепыш», г. Ноябрьск
424. МАДОУ «Детский сад «Радуга», г. Новый Уренгой
425. МДОУ «ЦЕНТР РАЗВИТИЯ РЕБЕНКА «ДЕТСКИЙ САД «УМКА» г. Надыма
426. МДОУ «Детский сад «Медвежонок» г. Надыма»
427. МДОУ «Детский сад "Газовичок" г. Надыма»
428. МАДОУ «Лукоморье», г. Ноябрьск
429. МБДОУ «Детский сад «Северная сказка», г. Новый Уренгой
430. МБДОУ Детский сад «Оленёнок», г. Салехард
431. МАОУ "СОШ №4", г. Губкинский
432. МДОУ "ДС "Буратино" г. Надыма"

Калининградская область

433. МАДОУ Центр развития ребенка – детский сад №23 «Сказка» г. Зеленоградск
434. МАДОУ ЦРР – детский сад № 4, г. Зеленоградск

435. МАДОУ «Детский сад № 11 Центр развития ребенка», г. Гусев
436. МАДОУ детский сад № 3, г. Зеленоградск
437. МАДОУ д/с №51, г. Калининград
438. МАДОУ д/с № 129, г. Калининград
439. МАДОУ детский сад №135, г. Калининград
440. МАДОУ МО «СГО» -д/с №9 «Улыбка», г. Светлый
441. МАДОУ ЦРР д/с №122, г. Калининград
442. МАДОУ ЦРР д/с № 76, г. Калининград
443. МАДОУ «Детский сад №2 «Мозаика», Гурьевский район, п. Васильково

Удмуртская республика

444. МБДОУ Детский сад №37 г. Сарапул
445. МБОУ СОШ №17, г.Ижевск
446. ГБОУ УР «Лицей №14», г. Ижевск
447. МБДОУ «Игринский детский сад №10», п. Игра
448. АНО «Центр цифрового образования детей «IT-куб», г. Ижевск
449. МБОУ Нечкинская СОШ, с. Нечкино
450. МБДОУ №179, г. Ижевск
451. АНО ДПО "Центр повышения квалификации в сфере информационных технологий" г.Ижевск

Костромская область

452. МБДОУ Детский сад № 73 «Алёнушка» г. Шарья
453. МОУ СОШ №2 г. Буя
454. МБДОУ детский сад №3 «Ромашка» комбинированного вида Мантуровского муниципального округа Костромской области

Пермский край

455. МАДОУ первой категории Центр развития ребенка «Добрянский детский сад №16 «Березка» г. Добрянка
456. МАДОУ «Детский сад № 312», г. Пермь
457. МБДОУ ЦРР - детский сад № 16 г. Нытва
458. МБДОУ «ЦРР - Детский Сад № 14», г. Чернушка
459. МАДОУ «ЦРР-детский сад № 210», г. Пермь
460. МАДОУ «Детский сад «Электроник», г. Пермь
461. МБДОУ «Детский сад № 13», г. Чернушка
462. МАДОУ ЦРР-д/с «Лира», г. Оса
463. МБДОУ «Центр развития ребенка - детский сад №8», г. Чернушка
464. МБДОУ «Детский сад «Наукоград», г. Чусовой
465. МБДОУ детский сад № 4 «Березка», г. Чайковский
466. МАДОУ «Детский сад №22», г. Пермь
467. МАДОУ «Детский сад №11» г. Березники
468. МАОУ «Фроловская средняя школа «Навигатор», с. Фролы
469. МАОУ «Гимназия № 4 имени братьев Каменских» г. Пермь
470. МАОУ «Гимназия», г. Чайковский
471. МАДОУ «Детский сад № 11», г. Краснокамск
472. МБДОУ «ДДС №16 «ПроУспех», г. Добрянка

Нижегородская область

473. МАДОУ Детский сад № 26 «Антошка» г. Бор
474. МАДОУ «Детский сад №46», г. Нижний Новгород
475. МАДОУ ЦРР № 22 «Колокольчик», г. Бор
476. МБДОУ «Детский сад № 18 «Волшебница», г. Нижний Новгород
477. МАДОУ детский сад № 9 «Золотой ключик», г. Бор

478. МБОУ «Лицей», г. Арзамас
479. МБОУ Больше-Рыбушкинская СОШ имени А.С.Садекова, с.Большое Рыбушкино
480. МБОУ «Средняя школа № 18», г. Дзержинск
481. МБОУ школа № 27, г. Дзержинск
482. МБОУ «Школа №154» г. Нижнего Новгорода
483. МАОУ «Школа № 5», г.Богородск
484. МБДОУ «Детский сад 126», г. Дзержинск
485. МБДОУ «Детский сад № 88», г. Нижний Новгород
486. МБОУ СШ №18, г. Заволжье
487. МАДОУ детский сад «Капелька», г. Бор
488. МБДОУ Детский сад №61, г. Дзержинск
489. МБДОУ «Детский сад №82», г. Дзержинск
490. МБОУДО «ДДТ» городской округ Навашинский Нижегородской области
491. МОУ «Шатковская средняя школа», Шатковский округ, р.п. Шатки
492. МАДОУ детский сад «Белоснежка», г. Бор
493. МБДОУ «Детский сад №13», р.п. Гидроторф
494. МБДОУ «Детский сад №318», г. Нижний Новгород
495. МАДОУ «Колокольчик», г. Первомайск
496. МБДОУ д/с №41, г. Шахунья
497. МБОУ "Школа № 26", г. Нижний Новгород
498. МБОУ "Школа № 93", г. Нижний Новгород
499. МБОУ «Школа № 37», г. Нижний Новгород
500. МБОУ «ШКОЛА №170», г. Нижний Новгород
501. МБДОУ «Детский сад № 101», г. Нижний Новгород
502. МБДОУ «Детский сад № 27», г. Нижний Новгород
503. МБДОУ «Детский сад № 27» г. Балахна
504. МДОУ Детский сад «Светлячок» д. Гавриловка
505. МБДОУ Детский сад № 364 «Звёздочка», г. Нижний Новгород

506. МБДОУ «Детский сад № 20» г. Балахна
507. МБДОУ «Детский сад № 132», г. Нижний Новгород
508. МБДОУ детский сад № 9 «Ромашка», г. Семенов
509. МАДОУ детский сад № 11 г. Павлово
510. МБДОУ ДС № 32 г. Заволжье

Вологодская область

511. БДОУ «Детский сад комбинированного вида «Березка» с. Кичменгский Городок
512. БДОУ СМР «Детский сад № 10», г. Сокол
513. МАДОУ «Детский сад № 118 «Звездочка», г. Вологда
514. МАДОУ «Детский сад № 115 «Акварель», г. Вологда
515. МБДОУ «Детский сад № 39 «Ленок», г. Вологда
516. МБОУ ВМР «Семенковская основная школа имени С.В.Солодягина», п. Семенково
517. МБДОУ ВМР «Семенковский детский сад общеразвивающего вида», п. Семенково
518. МБОУ ВМР «Федотовская средняя школа», п. Федотово
519. БДОУ ВМР «Детский сад «Гармония», г. Вытегра

Воронежская область

520. МБДОУ «Детский сад" Колокольчик» Каменского м.р.
521. МБОУЛ «ВУВК им. А.П. Киселева», г. Воронеж
522. МБОУ «Каменская СОШ №2», пгт. Каменка
523. МБДОУ «Бобровский детский сад №3 «Солнышко», г. Бобров
524. МБОУ Бобровский образовательный центр «Лидер» имени А.В. Гордеева, г. Бобров
525. МКДОУ БГО Детский сад №20 комбинированного вида, г. Борисоглебск

526. МБОУ «Елань-Коленовская СОШ № 2», р.п. Елань-Коленовский
527. МБОУ БГО СОШ №4, г. Борисоглебск
528. МБОУ Лицей №7, г. Воронеж
529. МКДОУ БГО Детский сад № 1 комбинированного вида, г. Борисоглебск
530. МБДОУ «Детский сад общеразвивающего вида № 52», г. Воронеж
531. МБОУ Аннинская СОШ №3, п.г.т. Анна
532. МКОУ Архангельская СОШ, с. Архангельское
533. МКОУ Бродовская СОШ - структурное подразделение детский сад, с. Бродовое
534. МКДОУ Рождественский детский сад, с.Ямное
535. МБДОУ «Центр развития ребенка – детский сад № 196», г. Воронеж
536. МБДОУ «Детский сад общеразвивающего вида № 74», г. Воронеж
537. МБОУ гимназия им. академика Н.Г. Басова, г. Воронеж
538. МКДОУ БГО Детский сад № 21 комбинированного вида, г. Борисоглебск
539. МБДОУ «Детский сад общеразвивающего вида №170», г. Воронеж
540. МБДОУ «ЦРР-ДЕТСКИЙ САД 188», г. Воронеж
541. МКОУ Бродовская СОШ, с. Бродовое
542. МБОУ БГО СОШ № 13, г. Борисоглебск
543. МБДОУ «Детский Сад Комбинированного Вида № 80», г. Воронеж
544. МБДОУ «Детский сад «Развитие», с. Новая Усмань
545. МБДОУ д/с «Колокольчик», с. Хохол

Томская область

546. МАДОУ детский сад комбинированного вида № 16 «Солнышко», г. Асино
547. МАОУ СОШ «Интеграция», п. Зональная Станция
548. МАДОУ № 13, г. Томск
549. МАДОУ № 8, г. Томск
550. МБДОУ № 18. г. Томск

551. МАДОУ № 61, г. Томск
552. МАДОУ № 45, г. Томск
553. МБДОУ № 21, г. Томск
554. МАДОУ № 28, г. Томск
555. МАДОУ № 63, г. Томск
556. МБДОУ № 35 г. Томска
557. МБОУ Беляйская ООШ, п. Беляй
558. МАДОУ № 100, г. Томск
559. МАДОУ № 82, г. Томск
560. МАДОУ № 54, г. Томск
561. МБДОУ № 23, г. Томск
562. МБДОУ «Детский сад № 53», г. Северск
563. МОУ «СОШ №5», г.о. Стрежевой
564. МБОУ «Озеренская СОШ», с.Озерное
565. МБДОУ «Детский сад «Северный парк» Томского района
566. МАДОУ «Детский сад «Малышок», с. Александровское
567. МАДОУ № 66 г. Томска
568. МАОУ СОШ № 16 г. Томска
569. МАДОУ ДС № 104 Г. Томск
570. МДОУ ДС Стрежевой. Структурное подразделение «Золотой ключик»

Алтайский край

571. Детский сад № 183 ОАО «РЖД», г. Барнаул
572. Центр цифрового образования детей «IT-куб» - с.п. КАУ ДПО «Алтайский институт цифровых технологий и оценки качества образования им. О.Р. Львова», г. Барнаул
573. МАДОУ «ЦРР д/с №1 «Жар-птица», г. Рубцовск

574. МБОУ «Гимназия № 1», г. Бийск
575. МБДОУ «Детский сад «Сказка», г. Белокуриха

Кабардино-Балкарская республика

576. МКОУ «Прогимназия №3 г. Баксана», г. Баксан

Краснодарский край

577. МБДОУ детский сад комбинированного вида № 1, станица Ленинградская
578. МАДОУ центр развития ребенка-детский сад № 82 «Сказка» муниципального образования, г. Новороссийск
579. МБДОУ детский сад №18 «Солнышко», г. Приморско-Ахтарск
580. МАДОУ детский сад общеразвивающего вида № 23, г. Новороссийск
581. МБДОУ детский сад № 60, г. Новороссийск
582. МАДОУ № 4, г. Армавир
583. МБОУ СОШ № 2, ст. Медведовская
584. МБОУ СОШ № 14 имени А.И. Покрышкина, ст. Кавказская
585. МАДОУ ЦРР детский сад № 4, ст. Павловская
586. МАДОУ МО г. Краснодар «Детский сад №192»
587. МАДОУ ДС № 1 «Белоснежка», ст. Переясловская
588. МАДОУ ЦРР – детский сад №2, г. Усть-Лабинск
589. МАДОУ ЦРР-д/с № 33, ст. Кавказская
590. МБДОУ д/с №22, ст. Петровская
591. МБДОУ детский сад компенсирующего вида № 34, ст. Ленинградская
592. МАДОУ ДС № 11 "Колокольчик", ст. Брюховецкая
593. МБДОУ детский сад № 9 «Улыбка», станица Ловлинская
594. МБДОУ № 79, г. Новороссийск
595. ЧДОУ Детский сад №99 ОАО «РЖД», г. Новороссийск
596. МБДОУ МО г. Краснодар «Детский сад №30»

597. МАДОУ №18, г. Армавир
598. МБДОУ центр развития ребенка – детский сад №13, г. Новороссийск
599. МБДОУ д/с №8 «Буратино», г. Геленджик
600. МАДОУ № 34 станицы Ленинградской муниципального образования Ленинградский район

Республика Хакасия

601. МБДОУ «Центр развития ребёнка - детский сад «Калинка», г. Абакан
602. МБДОУ «д/с «Журавлик», г. Абакан
603. МБДОУ «Детский сад «Подснежник», г. Абакан
604. МБДОУ «ЦРР - д/с «Кристаллик», г. Абакан
605. МБДОУ «ЦРР-д/с «Золушка», г. Абакан
606. МБДОУ детский сад «Березка», п. Малые Арбаты
607. МБДОУ «Д/с «Филиппок», г. Абакан
608. МБДОУ «ЦРР - д/с «Дельфин», г. Абакан
609. МБОУ «СОШ № 9», г. Абакан
610. МБДОУ Детский сад № 6 «Жемчужинка», п. Колодезный
611. Гимназия №1 им. А.Л. Кузнецовой, г. Куйбышев
612. МБОУ «СОШ №9», г. Абакан
613. МАДОУ «Золотая рыбка», г. Черногорск
614. МБОУ «Опытненская СОШ», с. Зелёное, Усть-Абаканский район

Белгородская область

615. МБДОУ «Детский сад «Колобок», с. Засосна
616. МБДОУ «Ровеньский детский сад №1 комбинированного вида», п. Ровеньки
617. МДОУ детский сад №1 комбинированного вида, п. Вейделевка
618. МБДОУ «Детский сад «Колокольчик» комбинированного вида, п. Чернянка
619. МДОУ «Детский сад № 22», п. Северный

620. МБДОУ детский сад № 70 «Центр развития ребенка «Светлячок», г. Белгород
621. МБДОУ «Детский сад комбинированного вида № 3», г. Алексеевка
622. МБДОУ д/с №49 г. Белгорода
623. МБДОУ «Волоконовский детский сад №3 «Родничок», п. Волоконовка
624. МБДОУ «Детский сад общеразвивающего вида №19 «Светлячок» г. Губкина
625. МБДОУ «Борисовский детский сад «Ягодка», п. Борисовка
626. МБДОУ д/с № 88, г. Белгород
627. МДОУ ДС №4 «Калинка» комбинированного вида г. Валуйки
628. МБДОУ «Детский сад «Ромашка» села Белянка Шебекинского района Белгородской области»
629. МБДОУ Детский сад №19 «Антошка» г. Белгорода
630. МАДОУ Детский сад №73 «Мишутка» Старооскольского городского округа

Тульская область

631. МБДОУ «Детский сад № 40 «Ладошки», г. Новомосковск
632. МБОУ «Центр образования 36», г. Тула
633. МБДОУ «Црр – д/с 37», г. Новомосковск
634. МБОУ «ЦО 52 им.В.В.Лапина» п. Рассвет
635. МБДОУ ЦРР №5 «Мир детства», г. Тула
636. МБДОУ ЦРР - д/с №6, г. Тула
637. МБОУ ЦО № 7 имени Героя Советского Союза Сергея Николаевича Судейского, г. Тула
638. МБОУ ЦО № 51 с. Алешня

Сахалинская область

639. МАДОУ «Детский сад № 30 «Кораблик», г. Корсаков
640. МАОУ «НОШ № 5» г.Корсаков
641. МАДОУ № 5 «Полянка» г. Южно-Сахалинска

642. МБДОУ № 2 г. Поронайска
643. МБДОУ Детский сад № 5 пгт. Тымовское
644. МАОУ СОШ №2, г. Корсаков
645. МАДОУ Детский сад № 8, г. Корсаков
646. МБДОУ № 1 г. Углегорск
647. МБДОУ № 22 с. Бошняково
648. МАДОУ № 47 «Ягодка» г.Южно-Сахалинска
649. МАДОУ ДС № 4 Лебедушка Г. Южно-Сахалинск

Республика Саха (Якутия)

650. МДОУ № 58 «Красная шапочка», г. Нерюнгри
651. МБОУ С(К) НШ - ДС №3 г.Нерюнгри
652. МБДОУ «Центр развития ребенка – детский сад №25 «Кустук», с. Огородтах
653. МБДОУ «ЦРР-д/с «Чуораанчык» с. Чурапча
654. ЧОУ «Точка развития», г. Якутск
655. МБДОУ «Детский сад №17 «Чуораанчык», с. Мастах
656. МБДОУ ДС № 1 Солнышко с. Намцы

г. Севастополь

657. ГБУ «Центр психолого-педагогической, медицинской и социальной помощи», г. Севастополь
658. ГБОУ «Образовательный центр «Бухта Казачья»

Омская область

659. ДОО ЧУ ДО «Детский сад «Бейбиленд», г. Омск

Владимирская область

660. МБОУ «Курловская СОШ», г. Курлово
661. МБОУ «Лицей им.ак. И.А.Бакулова» пос.Вольгинский
662. МБУ ДО «ЦДО «Исток», г. Суздаль
663. МАОУ «СОШ № 36», г. Владимир
664. МБДОУ № 44, г. Ковров
665. МБДОУ «Детский сад № 112», г. Владимир

Мурманская область

666. МБДОУ №46 п.г.т. Молочный
667. МДОУ детский сад КВ № 2 «Радуга», г. Заозерск
668. МАДОУ № 1, г. Мончегорск
669. МАДОУ №5 «Теремок», г. Ковдор
670. МБОУ «ЛСОШ», с. Ловозеро
671. МАДОУ г. Мурманска № 45
672. МБДОУ № 14, п.г.т. Зеленоборский
673. МБДОУ г. Мурманска №80
674. МБДОУ №14, п.г.т. Зеленоборский
675. МБДОУ г. Мурманска № 2
676. ГАНОУ МО «ЦО «Лапландия», г. Мурманск

Магаданская область

677. МБДОУ №53, г. Магадан
678. МАДОУ №58, г. Магадан

Республика Коми

679. МОУ «Гимназия № 6» г. Воркуты
680. МАДОУ «ЦРР – Д/с № 114», г. Сыктывкар

Курская область

681. МБОУ «СОШ №59», г. Курск
682. МБДОУ «Детский сад комбинированного вида 9», г. Курск
683. МБДОУ «Детский сад комбинированного вида «Родничок» Курского района Курской области, п. Маршала Жукова

Амурская область

684. МДОАУ детский сад № 38 г. Свободного
685. МАОУ «Алексеевская гимназия г. Благовещенска»

Ульяновская область

686. МБДОУ ЦРР-детский сад № 242 «Садко», г. Ульяновск

Красноярский край

687. МАОУ Гимназия № 13 «Академ», г. Красноярск
688. МБОУ СШ № 36, г. Красноярск
689. МАОУ Гимназия № 14, г. Красноярск
690. МАДОУ № 238, г. Красноярск
691. МБОУ «Гимназия № 164», г. Зеленогорск
692. МБДОУ № 8 «Иволга», гп. Северо-Енисейский
693. МБДОУ № 22, г. Красноярск

694. МАОУ СШ № 34, г. Красноярск
695. МАОУ СШ № 23, г. Красноярск
696. МАДОУ «Детский сад № 300 комбинированного вида», г. Красноярск

Саратовская область

697. МАОУ «Гимназия №1», г. Саратов
698. МОУ СОШ №1 им. З.К. Пряхиной р.п. Мокроус
699. ГАОУ СО «Гимназия №1», г. Саратов
700. МАДОУ «Центр развития ребёнка – детский сад №1 «Солнечный зайчик» г. Саратова
701. МОУ «Гимназия №89» Ленинского района г. Саратова
702. МБДОУ «Детский сад № 16 г. Красноармейска»
703. МОУ ООШ п. Прибрежный
704. МОУ СОШ № 3 г. Ершов

Хабаровский край

705. МБДОУ Детский Сад № 7 РП Переяславка
706. МАДОУ г. Хабаровска «Детский сад комбинированного вида № 192», г. Хабаровск

Приморский край

707. МБДОУ ДС ОВ №32, пос. Раздольное

Республика Мордовия

708. МОУ «Лицей № 7», г. Саранск

Ханты-Мансийский автономный округ – Югра

709. МАДОУ «Детский сад «Звездочка» г. Белоярский»
710. МАДОУ города Нижневартовска ДС № 69 «Светофорчик»
711. МБДОУ №20 «Югорка», г. Сургут
712. МБДОУ ДС №47 «Успех», г. Нижневартовск
713. МАДОУ города Нижневартовска ДС №21 «Звездочка»
714. МАДОУ ДС №68 «Ромашка», г. Нижневартовск
715. МБДОУ ДС № 67 «Умка», г. Нижневартовск
716. НРМ ДОБУ «Д/с «Солнышко», гп. Пойковский
717. МБДОУ «Детский сад №23 «Брусничка», г. Ханты-Мансийск

Ульяновская область

718. ОГАОУ «Гимназия № 2», г. Ульяновск

Республика Татарстан

719. МАОУ «Лицей - инженерный центр», г. Казань
720. МБОУ «Гимназия №7», г. Казань
721. МБДОУ «Билингвальный детский сад №7 комбинированного вида», г. Казань
722. МАДОУ «Детский сад № 401», г. Казань
723. МБДОУ «Детский сад № 6», г. Чистополь
724. МАДОУ «Детский сад №108 «Счастливое детство», г. Набережные Челны
725. МБДОУ Пестречинский детский сад №1 «Колокольчик», с. Пестрецы
726. МАДОУ «Детский сад №185», г. Казань

Камчатский край

727. МБОУ «Средняя школа №17 им. В.С. Завойко», г. Петропавловск-Камчатский
728. МБОУ Николаевская СШ, г. Елизово

729. МБОУ «Раздольненская СШ», п. Раздольный
730. МБОУ СШ № 6, Усть – Камчатский муниципальный район, п. Козыревск

г. Санкт-Петербург

731. ГБДОУ детский сад № 43 Колпинского района СПб
732. ГАДОУ Детский сад № 15 Колпинского района СПб
733. ГБДОУ детский сад № 30 Петродворцового района СПб, г. Петергоф
734. ГБДОУ детский сад №73 комбинированного вида Фрунзенского района Санкт-Петербурга
735. ГБДОУ детский сад №112 Центрального района Санкт-Петербурга
736. ГБДОУ детский сад №91 Красносельского района Санкт-Петербурга
737. ГБДОУ детский сад № 1 Московского района Санкт-Петербурга
738. ГБОУ СОШ № 422 Кронштадтского района Санкт-Петербурга
739. ГБДОУ детский сад №9 Колпинского района Санкт-Петербурга
740. ГБДОУ детский сад №87 Калининского района Санкт-Петербурга
741. ГБДОУ №17 комбинированного вида Колпинского района СПб
742. ГБДОУ детский сад №20 Колпинского района Санкт-Петербурга
743. ГБДОУ детский сад № 20 Колпинского района СПб
744. ГБДОУ детский сад №130 Невского района Санкт-Петербурга
745. ГБДОУ детский сад №21 комбинированного вида Петродворцового района Санкт-Петербурга
746. ГБДОУ детский сад № 18 Приморского района Санкт-Петербурга
747. ГБОУ школа № 606 Пушкинского района Санкт-Петербурга
748. ГБДОУ ДС № 48

Брянская область

749. МБОУ СОШ №71 ОДО Детский сад «Гармония», г. Брянск
750. МБОУ «Центр образования «Перспектива» г. Брянска
751. МБДОУ № 1 Тюльпанчик г.Брянска

Волгоградская область

752. МОУ детский сад №283, г. Волгоград
753. МБДОУ ДС 3 «Колокольчик», г. Котельниково
754. МОУ НШ № 2, г. Волгоград
755. МБОУ «Новониколаевская СШ №2», рп Новониколаевский
756. МДОУ д/с № 8, г. Волжский
757. МОУ детский сад №247, г. Волгоград
758. МБДОУ ДС № 3 Колокольчик г. Котельниково

Ставропольский край

759. ГБОУ СК «Лицей №14 им. Героя РФ В.В.Нургалиева», г. Ставрополь
760. МБДОУ № 24 г. Невинномысска

Оренбургская область

761. МДОБУ «Детский сад № 21», г. Бузулук

Ярославская область

762. МДОУ «Детский сад № 3 Золотая рыбка», г. Ростов
763. СОШ № 23, г. Рыбинск
764. МДОУ «Детский сад №125» г. Ярославля
765. МДОУ №27 «Светлячок» ЯМР, пос. Щедрино
766. МДОУ «Детский сад № 3 «Солнышко», г. Гаврилов-Ям
767. МДОУ «Детский сад № 120», г. Ярославль
768. МДОУ «Детский сад №193», г. Ярославль
769. МДОУ «Детский сад №233» города Ярославля

г. Москва

770. ГБОУ Школа № 1391
771. ГБОУ Школа № 2115
772. ГБОУ города Москвы «Школа № 2083»

Калужская область

773. МБДОУ «Детство» «ЦРР» г. Калуги
774. МКДОУ детский сад «Ромашка», г. Кондрово
775. Детский сад «Родничок» г. Кондрово
776. МДОУ «Детский сад №12 «Маленькая страна», г. Балабаново

Смоленская область

777. МБДОУ д/с № 1 г. Вязьмы Смоленской области
778. СОГБОУ «Екимовичская средняя школа-интернат для обучающихся с ограниченными возможностями здоровья», село Екимовичи
779. МБДОУ «Детский сад №56 «Загадка», г. Смоленск
780. ОГБПОУ «Смоленская областная технологическая академия»

Курганская область

781. МБДОУ «Детский сад № 106», г. Курган

Рязанская область

782. МБОУ «Школа № 3», г. Рязань
783. МАДОУ «ЦРР — детский сад № 27», г. Рязань
784. МАДОУ «Детский сад №34», г. Рязань

Республика Карелия

785. МДОУ «Детский сад №38», г. Петрозаводск

Республика Тыва

786. МАДОУ Детский сад №15 «Страна детства», г. Кызыл

Липецкая область

787. МАДОУ детский сад «Петушок» д. Копцевы Хутора

Кировская область

788. МКОУ СОШ с УИОП № 10, г. Кирово-Чепецк

Республика Крым

789. МБДОУ г. Керчи РК «Детский сад комбинированного вида №53 «Звоночек»

Архангельская область

790. МБДОУ Детский сад №183, г. Архангельск
791. МБДОУ ЦРР - детский сад № 173, г. Архангельск

Донецкая Народная Республика

792. МБОУ «Школа - интернат ДО», г. Горловка

Забайкальский край

793. МОУ СОШ с.Маккавеево

Приложение №2

Назначение, цели и требования к цифровой образовательной платформы Мирера

ЦОП Мирера - отечественная образовательная платформа, оптимизированная под российскую систему организации высшего и среднего образования. ЦОП Мирера ориентирована на разработку и проведение курсов, в которых сочетаются онлайн-технологии и аудиторные методы ведения учебного процесса. Платформа снимает с преподавателя значительную часть нагрузки путем автоматизации большинства рутинных элементов образовательного процесса. В ЦОП Мирера реализована автоматическая проверка разных форматов заданий и разных уровней сложности из различных предметных областей. В этих и в других задачах платформа использует методы ИИ, при помощи которых показала высокую эффективность при поддержке IT курсов и курсов по другим STEM дисциплинам. ЦОП Мирера сохраняет полный цифровой след работы студентов и преподавателей в проводимых средствами системы курсах, а также автоматически формирует и предоставляет в реальном времени детальную аналитику, необходимую преподавателю и деканату вуза.

ЦОП Мирера направлена на снятие с преподавателя значительной части нагрузки путем достижения максимальной степени автоматизации всех рутинных элементов цифрового образовательного процесса: автоматический контроль посещаемости при очном и дистанционном обучении, контроль самостоятельности выполнения заданий, автоматизированная помощь (HELP-боты), оперативный анализ результатов прохождения студентом типовой образовательной траектории прохождения курса и выдача студенту рекомендации по индивидуальному изменению этой траектории, автоматический сбор и предоставление в реальном времени информации о ходе прохождения всех курсов отдельными студентами, группами, потоками.

ЦОП Мирера, что особенно важно при работе с возможно недостаточно эмоционально зрелыми старшеклассниками и младшекурсниками, предоставляет разработчикам и организаторам курсов инструменты для психологически комфортного побуждения обучаемых к своевременному выполнению заданий курса, возможности смягчения организационных последствий нарушения дедлайнов, возможности ликвидации студентами отставаний от графика прохождения курса, какими бы причинами эти отставания не были вызваны.

ЦОП Мирера обобщает и обогащает мировую практику современного обучения программированию, в том числе и многоязыкового как для новичков, так и для профессионалов.

Платформа предназначена для автоматизации процесса обучения по следующим направлениям:

1. Планирование, проведение и контроль учебной деятельности с использованием автоматизированных адаптивных траекторий образовательного процесса.

2. Разработка электронных курсов, методических материалов, сопровождение очных, смешанных и дистанционных форм обучения в режиме ВКС конференций с сохранением цифрового следа.

3. Выполнение студентами заданий, в том числе удаленно, с автоматической проверкой, с использованием при проверке нейросетевых технологий и технологий искусственного интеллекта.

4. Формирование портфеля компетенций обучаемого с сохранением его достижений в рамках пройденных курсов.

5. Автоматизированное консультирование обучаемых с использованием нейросетевых интеллектуальных чат-ботов.

6. Сопровождение образовательного процесса с использованием социальной сети ВКонтакте.

7. Сопровождение образовательного процесса с использованием чат-ботов Telegram.

Платформа создавалась для достижения следующих целей:

1. Цифровая трансформация образовательного процесса с поддержкой очного, дистанционного и смешанного форматов обучения.

2. Повышение вовлеченности обучаемых за счет непрерываемого учебного процесса.

3. Переориентация преподавательской деятельности на индивидуальный подход обучения каждого слушателя за счет снятия с преподавателя рутинной нагрузки путем автоматизации контроля прохождения курса с использованием нейросетевых технологий и искусственного интеллекта.

4. Непрерывное и удобное создание, развитие и тиражирование курсов, методических наработок в платформе.

5. Исключение рисков ограничения доступа к системе дистанционного обучения за счет размещения системы на ресурсах, расположенных на территории РФ.

Требования к цифровой образовательной платформе Мирера.

1. Обучение проводится в очном, дистанционном или смешанном формате обучения.

2. К системе имеют доступ администраторы, преподаватели, обучаемые и специалисты деканата/учебной части.

2.1. Для преподавателей Платформа обеспечивает возможность разрабатывать учебные материалы с использованием автоматизированных систем проверки учебных заданий. Доступна возможность проводить лекции, семинары с использованием аудиосвязи, видеосвязи, виртуальной доски, демонстрации экрана, демонстрации презентационных материалов и проверочных заданий с последующей автоматической проверкой.

2.2. Для обучаемых Платформа предоставляет возможность просматривать учебные материалы, выполнять проверочные задания и получать консультации от интеллектуального чат-бота.

2.3. Для специалистов учебной части Платформа предоставляет возможность просматривать журналы успеваемости обучаемых.

2.4. Для администратора Платформа предоставляет возможность управлять пользователями и курсами.

3. Автоматизации подлежит учебный процесс, состоящий из лекций, семинаров, промежуточных испытаний/мероприятий и финальных испытаний (зачетов/экзаменов). Учебный курс может длиться один семестр и более, ограничение на длительность отсутствует.

4. Основные автоматизируемые сценарии:

4.1. Разработка новых методических материалов или повторное использование ранее созданных учебных материалов.

4.2. Разработка проверочных заданий для семинарских занятий, домашних работ, контрольных работ и экзаменов.

4.3. Ведение расписания занятий и проверочных работ.

4.4. Проведение занятий в режиме аудио и видео-конференц-связи (ВКС).

4.5. Распределение вариантов заданий между обучаемыми.

4.6. Поддержка выполнения заданий обучаемыми.

4.7. Подготовка индивидуальных учебных траекторий обучаемых.

4.8. Ответы на вопросы обучаемых в процессе самостоятельной работы.

4.9. Уведомление обучаемых о начале и конце проверок и о статусе выполнения проверочных заданий.

4.10. Проверка результатов выполнения заданий обучаемыми.

4.11. Подготовка отчетной документации по результатам проверки знаний обучаемых.