МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ
ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ им. А. И. ГЕРЦЕНА»

Направление подготовки
09.03.01 Информатика и вычислительная техника

Направленность (профиль)
«Технологии разработки программного обеспечения»

Выпускная квалификационная работа

Разработка онлайн-тренажёра для обучения языку Python

Обучающегося 4 курса
очной формы обучения
Пузырева-Харьковского Владислава Евгеньевича

Руководитель выпускной квалификационной работы:
Кандидат педагогических наук, доцент
Государев Илья Борисович

Санкт-Петербург
2024

Оглавление

Введение

В современном мире программирование становится все более популярным и востребованным навыком, и язык Python является одним из наиболее популярных языков программирования. Однако, для успешного освоения этого языка требуется не только теоретические знания, но и навыки практического применения. Обучение программированию в современных школах и учебных заведениях становится все более актуальным и востребованным.

Несмотря на все преимущества языка Python, многие дети и начинающие программисты сталкиваются с рядом трудностей при его изучении. Одной из главных проблем является использование традиционных компиляторов и интегрированных сред разработки (IDE). Эти инструменты, хотя и мощные, часто оказываются слишком сложными и непонятными для новичков. Дети теряются в интерфейсе, не могут разобраться с ошибками компиляции и зачастую теряют мотивацию к дальнейшему обучению. Недостаток дружелюбного и интуитивно понятного интерфейса создает барьер на пути к эффективному обучению, что приводит к необходимости поиска альтернативных подходов.

Для успешного освоения Python требуется среда обучения, которая будет учитывать возрастные особенности и потребности учащихся. Такой тренажер должен обладать простым и понятным интерфейсом, помогать визуализировать результаты выполнения программ, предоставлять пошаговые инструкции и задания, обеспечивать обратную связь в случае ошибок, а также предоставлять быстрый доступ к обучающим материалам. Среда обучения должна быть интерактивной, мотивирующей и ориентированной на пользователя, чтобы сделать процесс обучения Python увлекательным и эффективным.

Практическая значимость данной работы заключается в создании интерактивного онлайн-тренажера, который не только облегчит процесс обучения программированию для детей, но и повысит их мотивацию и интерес к предмету.

Веб-приложение, разработанное для сети школ программирования Progame, будет способствовать более эффективному усвоению материала, предоставляя учащимся доступ к удобной, интуитивно понятной и интерактивной обучающей среде. Это повысит качество образовательного процесса и позволит детям быстрее приобретать необходимые практические навыки программирования. Коммерческая значимость проекта заключается в увеличении конкурентоспособности и привлекательности сети школ Progame на рынке образовательных услуг.

Целью данной **работы** является разработка веб-приложения для сети школ программирования Progame, которое будет включать в себя интерактивный онлайн-тренажер и курс по обучению языку программирования Python, адаптированный для детей. Приложение должно быть удобным, понятным и мотивирующим для учащихся, предоставляя быстрый доступ к обучающим материалам и интерактивным заданиям.

Для достижения поставленной цели были установлены следующие задачи:

1. Провести анализ существующих методов и инструментов обучения программированию для детей, выявить их сильные и слабые стороны.
2. Исследовать потребности и особенности целевой аудитории, включая возрастные и психологические аспекты восприятия информации.
3. Разработать структуру и содержание курса по Python, включающего теоретические материалы и практические задания, соответствующие уровню подготовки учащихся.
4. Спроектировать интерфейс веб-приложения, учитывающий требования к удобству использования и интерактивности для детей.
5. Разработать функционал онлайн-тренажера, который будет включать редактор кода и тестирование по материалу.
6. Провести тестирование прототипа приложения с участием реальных пользователей из целевой аудитории, собрать отзывы и внести необходимые коррективы.

7. Подготовить методические рекомендации по использованию веб-приложения для преподавателей сети школ Progame.

Результатом бакалаврской дипломной работы станет комплексное веб-приложение для сети школ программирования Progame.

Глава 1

1.1. Анализ существующих методов и инструментов обучения программированию для детей

Современные методы обучения программированию для детей разнообразны и зависят от уровня школы, возраста учеников и образовательных целей. Одним из популярных языков программирования, используемых для обучения детей, является Python. Этот язык выбирается за его простоту, читаемость и широкие возможности для реализации различных проектов

В некоторых общеобразовательных школах в программу уроков информатики могут быть включены занятия по Python. Это зачастую является признаком школы с продвинутым уровнем изучения информатики. В таких школах преподавание программирования начинается с основ, включая базовые концепции алгоритмов и структур данных, а затем постепенно усложняется, охватывая более сложные темы, такие как объектно-ориентированное программирование (ООП) и работа с библиотеками Python.

Для проверки усвоения знаний в школах используют тестирование и контрольные работы. В качестве инструментов для программирования часто применяют среду разработки IDLE. Например, в учебнике для углубленного изучения информатики "Информатика: 7-9 класс. Учебное пособие для 7-9 классов общеобразовательных учреждений с русским языком обучения" И.Н. Цыбуля в качестве среды для написания кода предлагается использовать IDLE. Преподаватель проводит обучение, опираясь на материалы учебника и презентации.

Внеурочные школы программирования для детей предоставляют уникальные возможности для углубленного изучения языков программирования, таких как Python. Эти школы предлагают специализированные курсы, которые

выходят за рамки стандартной школьной программы по информатике, давая ученикам возможность погружаться в программирование через различные подходы и методики.

В таких школах дети занимаются программированием на Python более углубленно, чем в обычных школах. Курсы часто начинаются с основ, таких как типы данных, условные операторы и циклы, и постепенно переходят к более сложным темам, таким как функции, модули, обработка ошибок и работа с файлами. Ученики изучают алгоритмы и структуры данных, что позволяет им писать более эффективный и оптимизированный код.

Обучение в подобных школах строится вокруг практических занятий. Дети не только изучают теорию, но и сразу применяют полученные знания на практике, создавая собственные проекты. Это могут быть игры, веб-приложения, скрипты для автоматизации задач и многое другое. Внеурочные школы программирования часто используют профессиональные инструменты для разработки программного обеспечения. Чаще всего обучение Python ведется с использованием сред разработки, таких как PyCharm. Эти инструменты предоставляют удобные функции для написания, тестирования и отладки кода, что позволяет ученикам работать в условиях, приближенных к реальным.

Обучение программированию часто происходит в игровых формах, что делает процесс более увлекательным и интересным для детей. Например, занятия могут включать программирование в Minecraft, где ученики используют Python для создания и модификации игровых элементов. Это позволяет сочетать изучение программирования с любимыми играми, что способствует повышению мотивации и интереса к предмету. Один из ключевых аспектов обучения в внеурочных школах программирования – это проектная деятельность. Ученики работают над своими собственными проектами, начиная от простых программ и заканчивая сложными многокомпонентными системами. Проекты могут быть индивидуальными или групповыми, что развивает навыки командной работы и сотрудничества.

На мой взгляд, IDLE является непривлекательной средой для написания кода для детей из-за слишком простого дизайна и отсутствия интерактивных элементов. Вместо этого, детям может быть более интересно использовать PyCharm, который предоставляет более широкие возможности для разработки и имеет более привлекательный интерфейс. Однако, как программа, PyCharm может быть слишком требовательной к ресурсам компьютера, что может создать проблемы для учеников с менее мощными устройствами. В этом случае, лучше рассмотреть альтернативные среды разработки, которые меньше нагружают компьютер.

Что касается методов обучения, использование игровых форматов и проектной деятельности кажется более привлекательным и эффективным для детей, чем традиционные уроки с теоретическим материалом и контрольными работами. Проверка знаний может быть более эффективной, если осуществлять её небольшими порциями в течение обучения, что позволит детям лучше усваивать материал и чувствовать себя более уверенно.

1.2. Исследование потребностей и особенностей целевой аудитории

При создании образовательного продукта для детей важно учитывать их возрастные особенности восприятия информации. Дети на разных этапах своего развития имеют различные способы восприятия и усвоения знаний.

Для младших школьников, например, важно использовать игровые методики и яркие иллюстрации, так как они еще развивают свои навыки чтения и анализа. Краткие и интересные задания, которые позволяют детям активно участвовать в уроке, часто более эффективны, чем длительные лекции.

Старшеклассники уже более готовы к более сложной информации и могут обучаться более формальным методам, таким как чтение учебников и самостоятельное выполнение заданий. Однако даже для них важно предоставлять интересные и актуальные материалы, которые могут мотивировать их к обучению.

Учитывая эти особенности, образовательный продукт должен быть разработан с учетом возраста целевой аудитории, чтобы максимально эффективно передать знания и убедить детей в интересности изучаемого материала.

Исследования в области психологии обучения программированию подтверждают, что дети в возрасте около 12 лет лучше всего усваивают текстовое программирование и демонстрируют более глубокое понимание его концепций. Этот возрастной период характеризуется активным развитием абстрактного мышления и когнитивных способностей, что способствует более глубокому пониманию программных конструкций и алгоритмов.

Одно из исследований, проведенных в Университете Стэнфорда, показало, что дети в возрасте около 12 лет, занимающиеся текстовым программированием, имеют более высокий уровень концентрации и мотивации, чем при использовании графических средств программирования. Они также продемонстрировали более высокий уровень креативности и способность к решению проблем.

Эти результаты указывают на то, что текстовое программирование в данном возрасте не только эффективно с точки зрения усвоения материала, но также способствует развитию критического мышления, логического мышления и решения проблем. Поэтому разработчики образовательных программ и приложений должны учитывать эти психологические аспекты и предоставлять детям возможность изучать текстовое программирование в удобной и доступной форме.

Психологические аспекты обучения детей программированию имеют важное значение при разработке образовательных программ и методик. Исследования показывают, что дети в возрасте около 12 лет находятся в ключевом периоде развития когнитивных способностей, что делает их особенно подходящими для усвоения текстового программирования.

Одно исследование, проведенное в университете Стэнфорда, обнаружило, что дети в возрасте 12 лет лучше всего усваивают текстовое программирование и проявляют более высокий уровень понимания алгоритмов и структур данных по сравнению с младшими или старшими возрастными группами.

Это объясняется тем, что дети в этом возрасте находятся в стадии развития абстрактного мышления, что способствует их способности к анализу и пониманию сложных концепций, включая программирование. Кроме того, дети в этом возрасте обычно обладают достаточной концентрацией и мотивацией для освоения новых навыков.

Интерактивность: Продукты должны быть интерактивными и привлекательными для детей. Включение игровых элементов, анимаций и возможности взаимодействия может сделать обучение более увлекательным и эффективным.

Удобство использования: Интерфейс образовательного продукта должен быть интуитивно понятным и легко навигируемым даже для маленьких детей. Это поможет сделать процесс обучения более комфортным и приятным.

Делимость информации: Материалы должны быть структурированы таким образом, чтобы информация подавалась небольшими частями, поэтапно. Это поможет детям усваивать знания более эффективно и предотвратит перегрузку информацией.

Проверки знаний: В образовательных продуктах должны быть встроены механизмы проверки знаний, такие как викторины, тесты или задания, чтобы дети могли оценить свой прогресс и усвоение материала.

1.3. Разработка структуры и содержания курса по Python

Разработка курса по Python для детей от 12 лет и старше включает в себя несколько ключевых этапов:

1. Определение учебных целей и задач
2. Составление теоретического материала
3. Разработка практических заданий и упражнений.

Основные цели курса включают введение учеников в мир текстового программирования, отличного от визуального программирования, с которым они могли сталкиваться ранее. Курс нацелен на обучение основам языка Python, включая его синтаксис, структуры данных и основные библиотеки. Важной задачей является развитие навыков написания кода для решения базовых задач и алгоритмов, а также формирование способности к самостоятельному решению задач и развитию логического мышления через программирование. В конечном итоге курс должен подготовить учеников к дальнейшему изучению программирования и освоению более сложных языков или концепций.

При составлении плана теоретического материала я опирался на несколько авторитетных источников. В их числе онлайн-курс на сайте Code Basics, учебник "Python для детей" Джейсона Бриггса, а также онлайн-курс от freeCodeCamp.

Онлайн-курсы на Code Basics и freeCodeCamp содержат проверенные временем теоретические материалы, которые успешно использовались для обучения программированию различных возрастных групп. В частности, freeCodeCamp представляет собой сообщество людей по всему миру, которые вместе учатся кодить, что позволяет постоянно обновлять и совершенствовать учебные материалы, опираясь на коллективный опыт и обратную связь от пользователей.

Учебник "Python для детей" Джейсона Бриггса предоставляет структурированный и доступный для детей подход к изучению языка программирования Python, что делает его идеальным источником для составления теоретических материалов курса. В этой книге особое внимание уделяется методам обучения детей программированию, так как она специально написана для данной возрастной группы. Эти методы были использованы при написании текстов к теории, чтобы сделать материал максимально понятным и интересным для учеников. Учебник включает в себя понятные объяснения, яркие иллюстрации и увлекательные примеры, что помогает учащимся лучше усваивать материал и сохранять интерес к обучению.

Таким образом, использование этих источников позволяет создать структурированный, доступный и эффективный курс по изучению Python, ориентированный на детей от 12 лет и старше.

Составленный план курса включает следующие темы:

1. Введение в Python. Переменные;
2. Операторы и операнды. Порядок операций;
3. Строки. Специальные символы;
4. Ввод входных данных;
5. Типы данных и их преобразование;
6. Комментарии;
7. Логические выражения и логические операторы;

8. Условное исполнение;
9. Продвинутые строки. Элементы, срезы и методы;
10. Цикл while. Бесконечный цикл;
11. Модули;
12. Цикл for;
13. Списки и кортежи. Преобразование коллекций;
14. Списковые включения (List Comprehension);
15. Словари, их методы и оператор in;
16. Функции и аргументы;
17. Значения по умолчанию и возврат результата;
18. Области видимости;
19. Обработка исключений;
20. Чтение и запись файлов;
21. Работа с датой и временем;
22. Функции высших порядков;
23. Классы и объекты.

Было осознанно принято решение давать меньше теории по объектно-ориентированному программированию (ООП) и сфокусировать внимание на базовых концепциях. ООП может быть сложной темой, и для неподготовленного ученика может создать лишние трудности. Чтобы не перегружать учащихся и удерживать их внимание, информация подается небольшими частями. Общее знакомство с ООП дается только в самом конце курса, обеспечивая мягкий переход и подготовку к более сложным темам в будущем.

К каждой теме курса по Python были разработаны практические задания и тесты, которые не только закрепляют знания по теме, но и постепенно становятся сложнее. Такой подход позволяет ученикам поэтапно развивать свои навыки программирования, обеспечивая плавное и уверенное освоение материала.

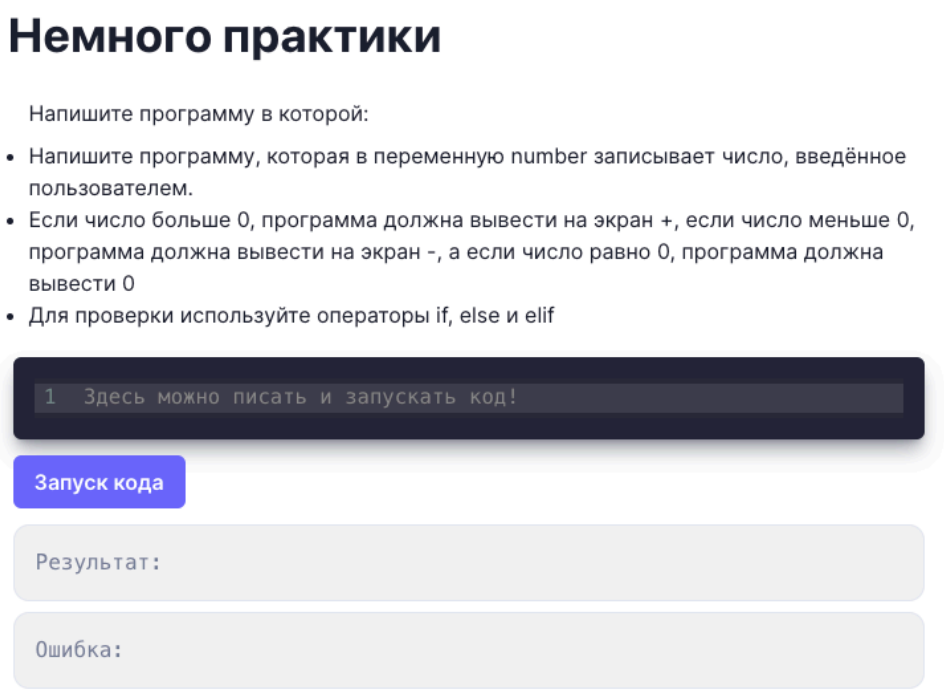

Рисунок 1.1 Пример практической задачи из курса

Практические задания охватывают все ключевые темы курса, начиная с введения в программирование и заканчивая сложными концепциями, такими как функции высших порядков и объекты. Например, на начальных этапах задания могут включать простые упражнения по созданию и использованию переменных, выполнению базовых операций и работе с основными типами данных. По мере продвижения курса задания усложняются, требуя от учащихся разработки более комплексных алгоритмов, работы с различными структурами данных и использования модулей.

Кроме того, разработаны тесты-квизы для тем, к которым трудно придумать практическую задачу на Python. Эти квизы направлены на закрепление небольших тем и являются одним из способов проверки теоретических знаний и понимания концепций, таких как имена переменных, общий синтаксис, порядок операций, и логические выражения. Квизы помогают укрепить усвоение теоретических аспектов курса, обеспечивая всестороннюю проверку знаний учащихся.

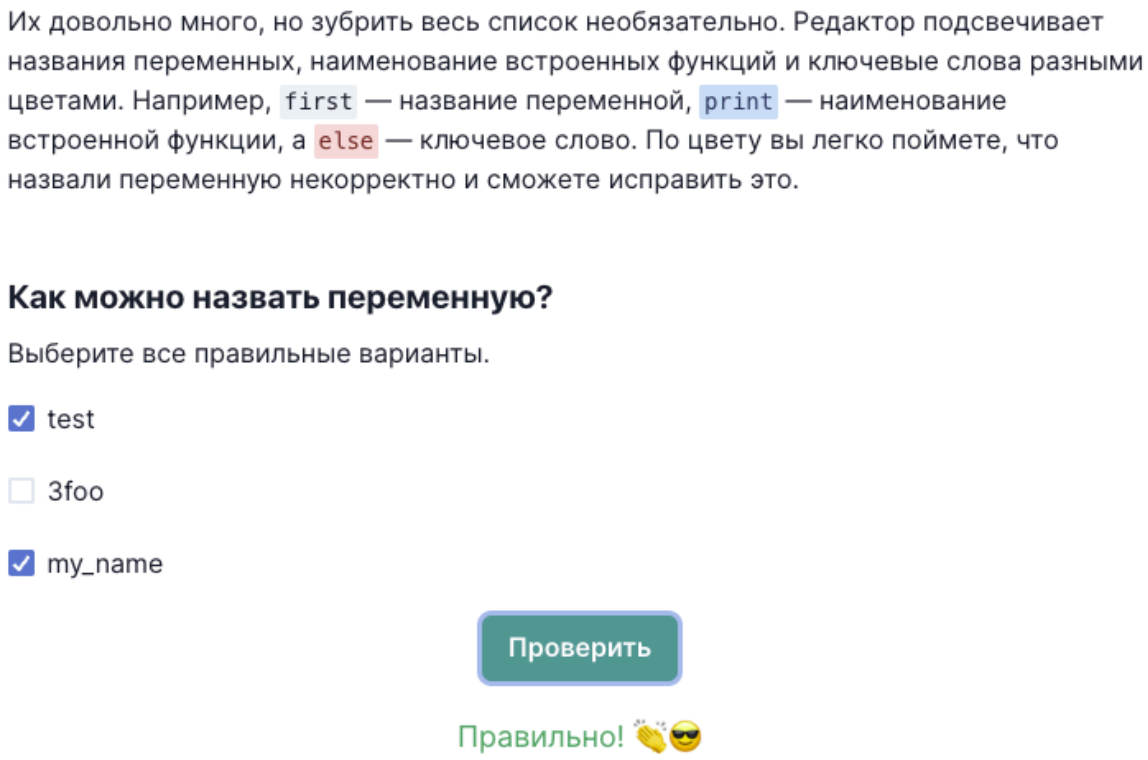

Рисунок 1.2 Пример теста из курса

Таким образом, тщательно разработанные практические задания и тесты способствуют эффективному обучению, позволяя учащимся не только закреплять знания, но и развивать свои навыки программирования в логической и последовательной форме.

Выводы к первой главе

1. Были раскрыты основные понятия, связанные с темой выпускной квалификационной работы.
2. Проведен анализ существующих методов и инструментов обучения программированию для детей, включая школы программирования. В результате этого анализа были выявлены сильные и слабые стороны различных подходов.
3. Исследованы потребности и особенности целевой аудитории, включающие возрастные и психологические аспекты восприятия информации детьми от 12 лет и старше.
4. Разработана структура и содержание курса по Python, основанные на авторитетных источниках и адаптированные для детей. Особое внимание уделено последовательному и логическому изложению материала.
5. Разработаны практические задания и тесты для каждой темы курса.

Глава 2. Разработка онлайн-тренажёра для обучения языку Python

2.1. Проектирование интерфейса веб-приложения

Проектирование интерфейса веб-приложения является ключевым этапом разработки, так как удобство использования и интерактивность интерфейса напрямую влияют на эффективность обучения и удовлетворенность пользователей. Данный раздел описывает процесс проектирования интерфейса, начиная с анализа требований и заканчивая созданием и оценкой макетов.

В работе "Требования к современным пользовательским интерфейсам" Демидова Д.Г. и Костаковой Е.С. были выделены следующие требования к современным пользовательским интерфейсам:

1. Простота: Интерфейс должен быть простым и интуитивно понятным для пользователя. «Простое должно оставаться простым»
2. Ориентация на пользователя: Интерфейс должен быть разработан с учетом потребностей и сценариев использования конечными пользователями.
3. Унификация: Основные функции и внешний вид интерфейса должны быть унифицированы для создания привычных привычек у пользователей.
4. Запоминаемость и уникальность: Интерфейс должен иметь уникальные черты (логотип, цвет), которые помогут пользователю запомнить его.

Анализ требований основывается на современных принципах юзабилити и включает сбор данных от потенциальных пользователей и экспертов в области педагогики.

Учитывая целевую группу — детей 12 лет, интерфейс должен быть однозначным и понятным, чтобы исключить возможность путаницы. Назначение кнопок и окон должно быть интуитивно понятно, а сам интерфейс должен быть защищен от возможности случайной "поломки". На экране должно быть только

основное — без "визуального мусора", который бы отвлекал и рассеивал внимание.

При разработке макетов интерфейса воспользовались инструментом Figma. На первоначальном этапе, рассматривались различные варианты, в которых редактор кода занимал половину экрана, а вторая половина - предназначалась для теоретических материалов, подобно тому, как это реализовано на платформах freeCodeCamp и code-basics.com. Однако, учитывая особенности процесса обучения детей, стало ясно, что постоянно занятая половина экрана редактором кода создает ограничения на отображение контента и усложняет восприятие информации.

На основании этого было принято решение о модификации подхода к размещению редактора кода. Опыт использования инструмента Google Colab в преподавательской практике по обучению и практике языка Python послужил стимулом для нового подхода. В конечном итоге, редакторы кода были размещены в ячейках непосредственно под материалами задач. Это позволило сделать интерфейс более компактным и удобным для пользователей. Конечный дизайн таким образом больше напоминает классический учебник, где теория и практика следуют друг за другом в логической последовательности.

При разработке макетов интерфейса воспользовались инструментом Figma. Изначально рассматривались варианты с более яркими цветами, которые ассоциировались со сладостями и игрушками, чтобы сделать обучение более привлекательным для детей.

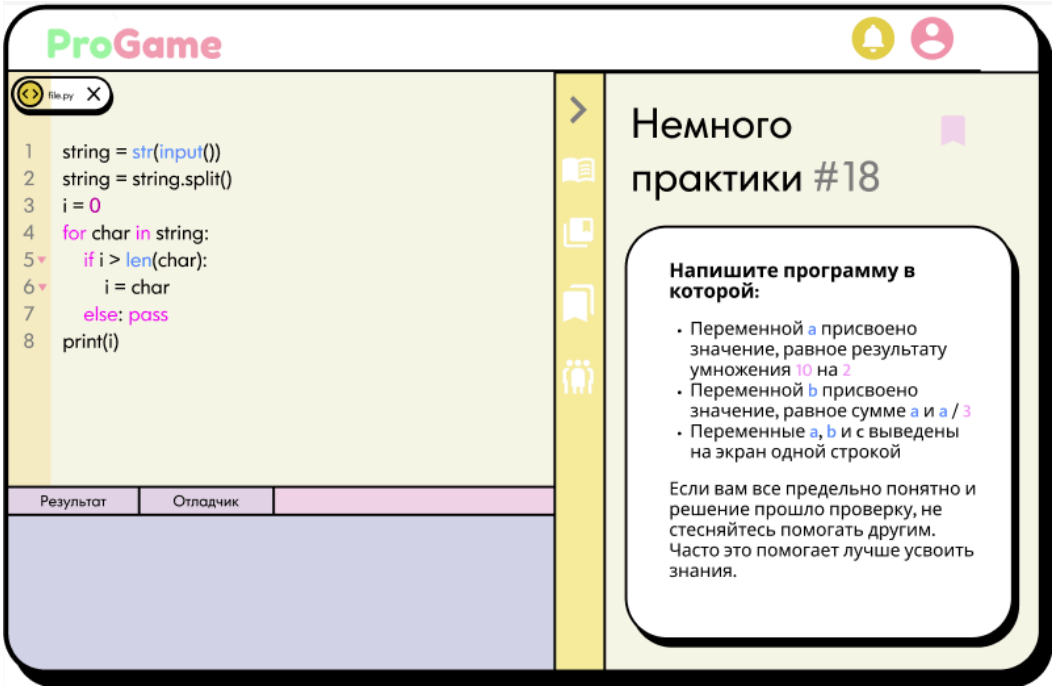

Рисунок 2.1 Первый прототип интерфейса

Однако, в процессе обдумывания и анализа пришел к выводу, что такой дизайн может не подойти каждому пользователю и может отвлекать от учебного процесса. Поэтому было принято решение остановиться на более простом и нейтральном дизайне с контрастными цветами, который не только не отвлекает внимание, но и более подходит для широкой аудитории. Этот выбор направлен на обеспечение комфортного и эффективного обучения пользователя.

Для отображения редактора кода в приложении был использован популярный npm пакет react-codemirror. Этот пакет обеспечивает простой и эффективный способ встраивания многофункционального редактора кода в веб-приложения. У этого пакета богатый функционал — подсветка синтаксиса для различных языков программирования, автодополнение, поддержка различных тем оформления и многое другое.

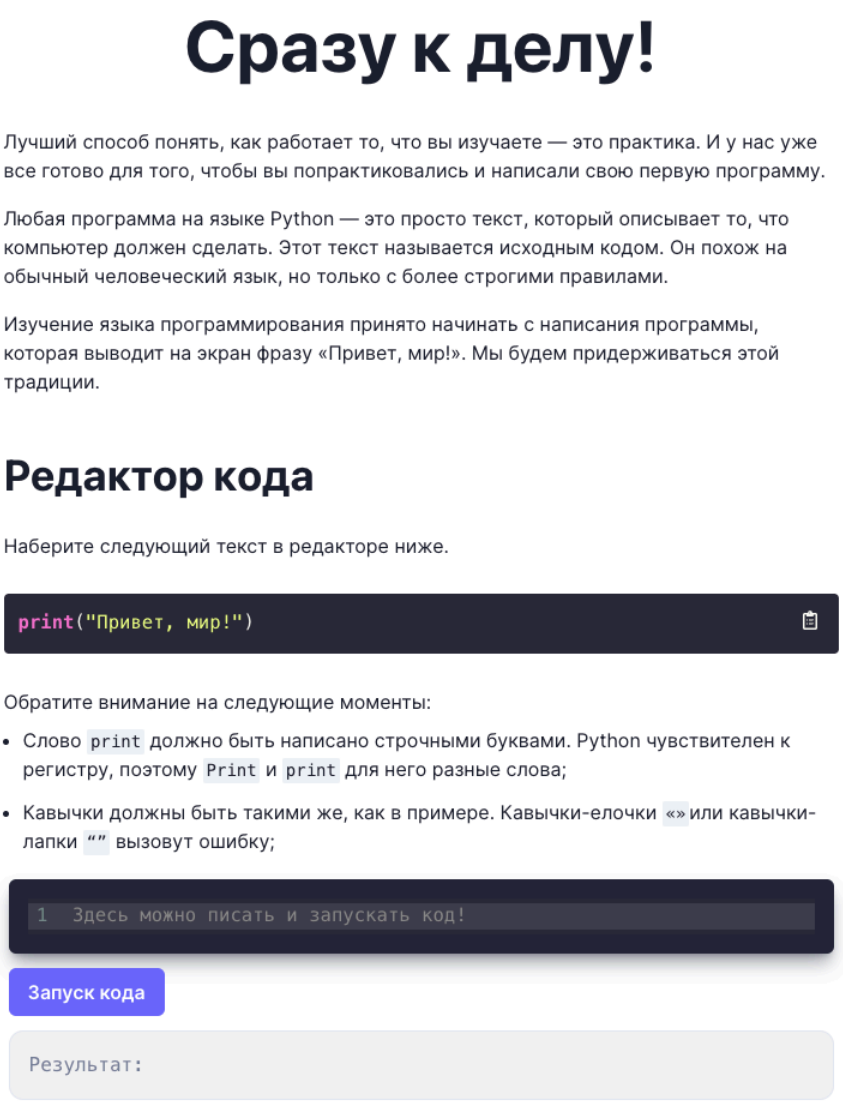

Рисунок 2.2 Последний прототип интерфейса

Приложение разработано на React - это JavaScript-библиотека для создания пользовательских интерфейсов. React позволяет разрабатывать масштабируемые и быстрые веб-приложения с использованием компонентного подхода. Он используется для создания интерактивных пользовательских интерфейсов, которые реагируют на изменения данных без необходимости перезагрузки страницы.

Next.js, в свою очередь, является метафреймворком для React. Он предоставляет дополнительные инструменты и возможности для разработки веб-приложений, такие как предварительная загрузка страниц, статическая

генерация, рендеринг на стороне сервера и динамическая маршрутизация. Next.js упрощает создание универсальных приложений, которые могут быть оптимизированы как для клиентской, так и для серверной части.

Разработка на React и Next.js осуществляется на языке JavaScript, который является одним из наиболее широко используемых языков программирования веб-разработки. Он обеспечивает возможность создания динамических и интерактивных веб-приложений, а также имеет богатую экосистему инструментов и библиотек для разработчиков.

При разработке приложения на React с использованием Next.js я выбрал язык TypeScript. TypeScript - это язык программирования, который является строгим надмножеством JavaScript. Он добавляет статическую типизацию к JavaScript, что позволяет обнаруживать и предотвращать ошибки на этапе разработки, а также улучшает процесс поддержки кода и его читаемость.

Одной из основных причин выбора TypeScript является повышение безопасности и надежности кода за счет статической типизации. TypeScript помогает выявлять ошибки на ранних стадиях разработки, что уменьшает количество ошибок и упрощает рефакторинг кода. Использование TypeScript позволяет создавать более надежные и поддерживаемые приложения на базе React и Next.js, повышая эффективность разработки и качество конечного продукта.

Для стилизации приложения использовались библиотеки Chakra UI, Tailwind CSS и framer-motion для создания анимаций. Chakra UI позволяет создавать стилизованные компоненты с помощью предварительно определенных тем, что обеспечивает гибкость и простоту в изменении дизайна страниц. Это позволяет легко адаптировать дизайн к различным потребностям и контекстам, обеспечивает легкость в настройке переключения между светлой и темной темами, сохраняя единый стиль приложения. Это позволяет пользователям выбирать режим отображения, который наиболее комфортен для них, что особенно важно для детей.

```
import { extendTheme } from "@chakra-ui/react";

const mainTheme = extendTheme({
  styles: {
    global: {
      body: {
        fontFamily: "body",
        color: "gray.600",
        bg: "gray.50",
      },
    },
  },
  fonts: {
    body: "system-ui, sans-serif",
    heading: "system-ui, sans-serif",
    mono: "Menlo, monospace",
  },
  components: {
    Container: {
      baseStyle: {
        maxW: "container.md",
      },
    },
    Heading: {
      baseStyle: {
        mt: "1em",
        mb: "0.5em",
        textAlign: "center",
        size: "3xl",
      },
    },
    Text: {
      baseStyle: {
        py: 2,
      },
    },
    CodeFragment: {
      baseStyle: {
        py: 2,
      }
    },
  },
```

```
});

export default mainTheme;
```

Рисунок 2.3 — Код темы для Chakra UI

Для эффективной демонстрации теоретических материалов по программированию необходимо представлять код в читаемой и выделенной форме. Одним из наиболее эффективных способов достижения этой цели является размещение кода в контрастных блоках с подсветкой синтаксиса и удобным для чтения шрифтом, а также добавлением кнопки "скопировать" для удобства использования. Для реализации таких блоков я воспользовался пакетом npm (Node Package Manager) под названием react-code-blocks. npm - это менеджер пакетов для языка JavaScript, который позволяет управлять зависимостями проекта и устанавливать необходимые библиотеки и инструменты. React-code-blocks обладает множеством функций, включая поддержку различных тем оформления, что позволяет выбирать наиболее подходящий стиль для представления кода. При использовании этого компонента, через его пропсы, передается сам код, который следует отобразить, язык программирования, на котором этот код написан, а также указывается, нужно ли отображать номера строк кода для более удобной навигации.

```
<CodeFragment
            code={`x = 1
y = 2
long_variable = 3`}
            language={"python"}
            showLineNumbers={false}
          ></CodeFragment>
```

Рисунок 2.4 — Пример использования блока с кодом

2.2 Разработка функционала онлайн-тренажера

Изначально рассматривалась архитектура приложения, включающая клиентскую и серверную части (фронтенд и бэкенд). Первоначальный прототип предусматривал создание бэкенд-сервера на базе Node.js, с использованием библиотек fs, cors, express и PythonShell. Fs (File System) - это модуль Node.js, который предоставляет API для работы с файловой системой операционной системы. cors (Cross-Origin Resource Sharing) - это пакет Node.js, который обеспечивает механизм для безопасного обмена ресурсами между разными источниками (доменами) веб-приложений. express - это веб-фреймворк для Node.js, который упрощает создание веб-приложений и API. Он предоставляет множество готовых функций для маршрутизации запросов, обработки запросов и ответов, управления сеансами и многое другое. PythonShell - это модуль Node.js, который предоставляет интерфейс для запуска и взаимодействия с процессами Python из Node.js. Он позволяет выполнять код Python из Node.js и обмениваться данными между двумя языками программирования.

Пользовательский код, введенный в интерфейсе, передавался на сервер для выполнения, затем результат возвращался и отображался на клиентской части. Однако такой подход привел к ряду технических сложностей, связанных с передачей данных и управлением серверной частью приложения.

В результате проведенного анализа было принято решение пересмотреть стратегию разработки и сосредоточиться на создании клиентской части приложения. Для этого был выбран метафреймворк Next.js, который позволяет строить веб-приложения с использованием только клиентской логики. Это сокращает сложности развертывания приложения и упрощает взаимодействие с браузером пользователя.

В качестве ключевого инструмента для выполнения кода на Python была выбрана библиотека Pyodide. Pyodide позволяет выполнять код на Python прямо в

браузере, что исключает необходимость в наличии серверной части и упрощает процесс обработки данных на стороне клиента.

Таким образом, в результате пересмотра архитектуры приложения было решено разрабатывать онлайн-тренажер исключительно на клиентской части с использованием Next.js и библиотеки Pyodide. Этот подход позволяет упростить развертывание и повысить доступность приложения для пользователей, а также снизить сложность взаимодействия между компонентами приложения.

Основным компонентом редактора кода в данном приложении является react-py. Это библиотека для React, предоставляющая удобный интерфейс для выполнения кода на Python прямо в браузере. React-py базируется на Pyodide - это Python-интерпретатор, который был скомпилирован в WebAssembly. Он позволяет запускать код на Python непосредственно в браузере без необходимости использования серверного окружения. На основе react-py построен онлайн-компилятор PyRepl.io. Этот сервис также позволяет пользователям писать и запускать код на Python прямо в браузере, не требуя для этого установки Python.

Для данного проекта использование react-py было единственным найденным решением для реализации запуска кода на Python без необходимости наличия серверной части.

```
import { useState } from 'react'
import { usePython } from 'react-py'

function Codeblock() {
 const [input, setInput] = useState('')

 const { runPython, stdout, stderr, isLoading, isRunning
} = usePython()

 return (
   <>
     {isLoading ? <p>Loading...</p> : <p>Ready!</p>}
     <form>
```

```
        <textarea
          onChange={(e) => setInput(e.target.value)}
          placeholder="Enter your code here"
        />
        <input
          type="submit"
          value={!isRunning ? 'Run' : 'Running...'}
          disabled={isLoading || isRunning}
          onClick={(e) => {
            e.preventDefault()
            runPython(input)
          }}
        />
      </form>
      <p>Output</p>
      <pre>
        <code>{stdout}</code>
      </pre>
      <p>Error</p>
      <pre>
        <code>{stderr}</code>
      </pre>
    </>
  )
}
```

Рисунок 2.5 — Базовый код для компонента редактора кода

Рассмотрим основные конструкции данного кода:

1. Для создания состояния input, которое хранит текст, введенный пользователем используется хук useState. Хук — это специальная функция в React, которая позволяет «подцепиться» к функциональностям React (таким как состояние или жизненные циклы компонентов) в функциональных компонентах.
2. Функции состояния для выполнения Python-кода предоставляет кастомный хук из библиотеки react-py — usePython.
3. В textarea используется обработчик событий onChange, который обновляет состояние input при каждом изменении текста.

4. В input типа submit используется обработчик событий onClick, который предотвращает стандартное поведение формы (перезагрузку страницы) и запускает выполнение Python-кода с помощью функции runPython.
5. { runPython, stdout, stderr, isLoading, isRunning } — деструктуризация объекта, возвращаемого хуком usePython. Это позволяет легко получить доступ к различным состояниям и функциям, предоставляемым react-py.
6. {isLoading ? Loading... : Ready! } используется для отображения состояния загрузки.
7. <textarea> используется для ввода кода.
8. <input type="submit"> используется для запуска кода.
9. Обработчик onClick на кнопке "Run" вызывает runPython с текущим значением input.

Для работы редактора кода необходимо обернуть тег с компонентом react-py в PythonProvider для предоставления контекста выполнения Python-кода внутри этого компонента. PythonProvider является контекстным провайдером, который передает информацию о среде выполнения Python-кода во всех вложенных компонентах.

Это позволяет компоненту react-py получить доступ к необходимым ресурсам и функциям, таким как интерпретатор Python и методы выполнения кода. Обычно внутри PythonProvider также устанавливаются дополнительные настройки и параметры для выполнения Python-кода, такие как настройки безопасности или параметры компиляции.

Для реализации функционала квизов был создан простой компонент Quiz. Ключевая функциональность реализована с использованием React Hooks, в частности, useState, который отслеживает выбранные пользователем варианты ответов и результат проверки. При изменении выбора опций вызывается функция handleChange, которая обновляет состояние selectedOptions, добавляя или удаляя

выбранные варианты. При нажатии кнопки "Проверить" вызывается функция handleSubmit, которая анализирует выбранные ответы и определяет, являются ли они правильными. Результат проверки отображается с помощью состояния result. Компонент принимает следующие параметры:

1. question: текст вопроса квиза;
2. options: массив строк, содержащий варианты ответов;
3. correctAnswers: массив строк с правильными ответами.

После выбора пользователем вариантов ответа и нажатия кнопки "Проверить", программа анализирует выбранные ответы и отображает результат.

2.3 Тестирование прототипа приложения

В ходе тестирования прототипа приложения были проведены три урока в рамках школы программирования Progame и ее дочерней организации Iteezy. Уроки были организованы для двух возрастных категорий: от 10 до 12 лет и от 12 до 14 лет, при участии взрослых преподавателей, которые внимательно наблюдали за усвоением материала. Дети в обеих группах внимательно слушали материал уроков и, опираясь на конспект в веб-приложении, выполняли задания. Показанный уровень понимания нового материала был примерно одинаков в обеих возрастных категориях.

По результатам тестирования получены положительные отзывы от учащихся, которые одобрили использование веб-приложения для обучения. Они отметили, что работа с приложением ощущается гораздо удобнее и комфортнее по сравнению с выполнением практических заданий в отдельном редакторе кода, при котором приходится отвлекаться на текст задания, выводимый на экран доски в классе.

Дети также отметили, что удобно иметь возможность просматривать материалы урока и практиковаться с веб-приложениями дома. Они отметили, что

записывание материалов в тетрадь занимает много времени, и использование своих конспектов не всегда эффективно. Веб-приложение предоставляет им доступ к материалам и позволяет практиковаться без необходимости полагаться только на свои записи.

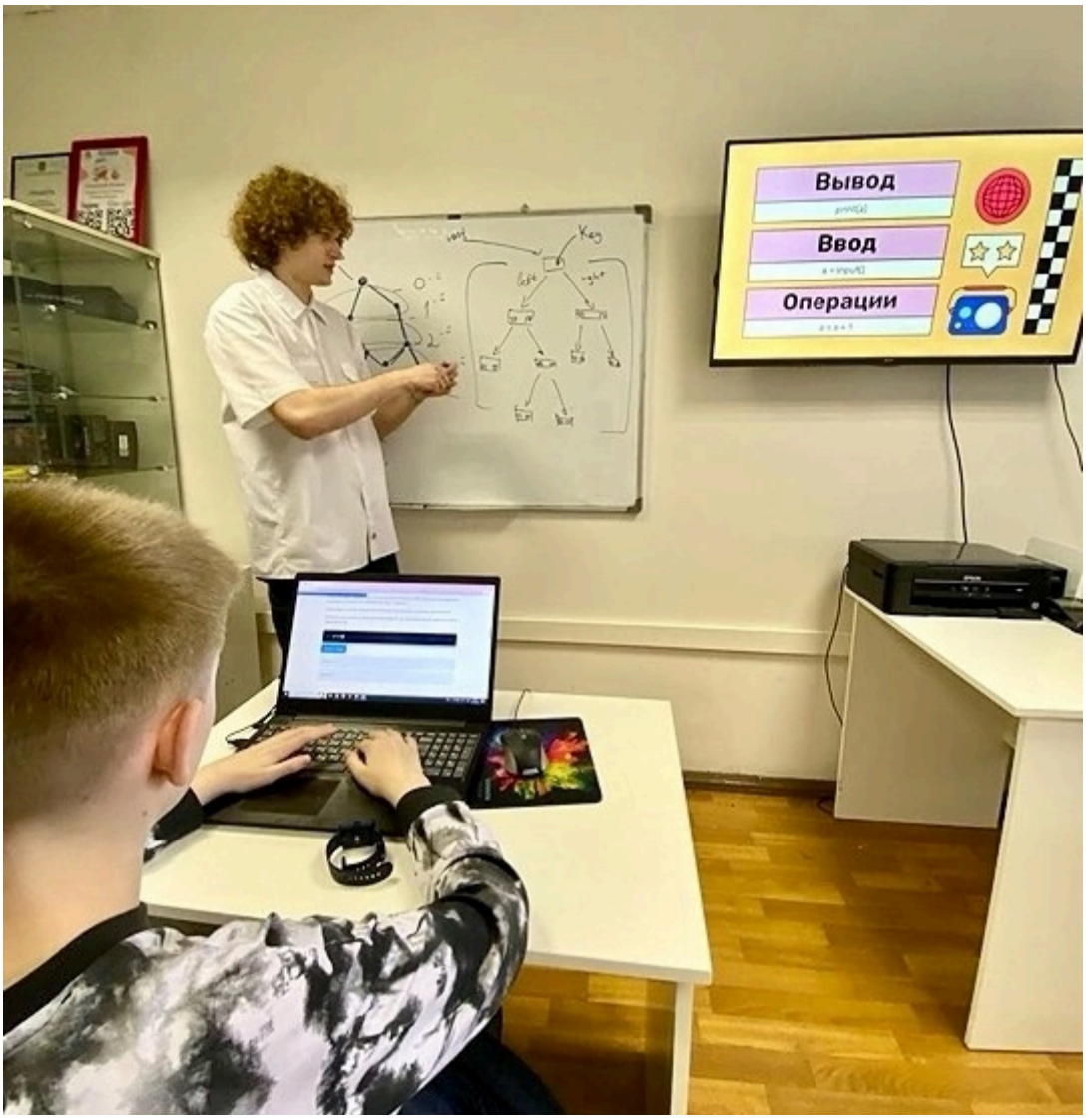

Рисунок 2.6 — Проведение урока для проверки веб-приложения

В результате тестирования были внесены следующие изменения в приложение:

1. Материал по объектно-ориентированному программированию (ООП) был значительно урезан, так как его усвоение вызывало трудности у учащихся.
2. Повышение сложности заданий было сделано более планомерным, чтобы дети могли более эффективно усваивать новый материал.
3. Шрифт в приложении был увеличен для лучшей читаемости, что сделало его более удобным для использования детьми.
4. Добавлена темная тема интерфейса, поскольку некоторым детям так комфортнее для глаз.
5. В приложение был добавлен элемент - полоса, показывающая прогресс пройденной темы. Это позволяет учащимся видеть, на каком они этапе обучения, не отвлекаясь от урока, и дает представление о том, сколько еще осталось до завершения темы.
6. Текст в приложении был упрощен и сделан более простым и детским, чтобы его легче воспринимали дети.

2.4 Разработка функционала онлайн-тренажера

Разрабатывая учебный продукт, важно учитывать не только удобство и эффективность обучения для детей, но и простоту использования для преподавателей. Веб-приложение должно быть интуитивно понятным и легким в освоении для взрослых преподавателей, чтобы они могли эффективно использовать его в учебном процессе.

Поскольку разрабатываемое приложение имеет простой дизайн, его легко интегрировать в учебную деятельность. Все функции и элементы интерфейса продуманы так, чтобы не вызывать вопросов. Тем не менее, я составил короткое руководство для преподавателей, включающее ключевые моменты.

Выводы к главе 2

В ходе разработки функционала онлайн-тренажера были достигнуты следующие результаты:

1. Был создан макет и прототип интерфейса на основании анализа требований к удобству использования и интерактивности.
2. Реализован функционал редактора кода Python в браузере.
3. Разработан функционал тестирования по материалам.
4. Организовано тестирование прототипа приложения с участием целевой аудитории
5. Внесены корректировки основе полученной обратной связи.

Заключение

В ходе выполнения выпускной работы был разработан программный продукт "Онлайн-тренажер для обучения языку Python", представляющий собой значимый инструмент для сеть школ Progame,. Продукт обеспечивает ученикам и преподавателям возможность улучшить процесс обучения, делая его более удобным и эффективным.

В рамках работы были выполнены следующие задачи:

1. Анализ существующих методов и инструментов обучения программированию для детей: Был проведен анализ рынка и отобраны наиболее эффективные и перспективные подходы к обучению программированию для детей.
2. Исследование потребностей и особенностей целевой аудитории: Проведенный анализ позволил понять потребности и особенности целевой аудитории, что было важным шагом при разработке функционала приложения.
3. Разработка структуры и содержания курса по Python: Создана структура и содержание курса, ориентированного на эффективное обучение программированию на языке Python с учетом особенностей целевой аудитории.
4. Проектирование интерфейса веб-приложения: Был разработан пользовательский интерфейс, который обеспечивает удобство и интуитивную понятность для пользователей.
5. Разработка функционала онлайн-тренажера: Реализован функционал онлайн-тренажера, позволяющий пользователям выполнять практические задания и получать обратную связь в режиме онлайн.
6. Тестирование прототипа приложения: Проведено тестирование разработанного прототипа приложения в реальных условиях с

участием целевой аудитории, что позволило выявить и исправить выявленные проблемы.

Литература

1. Капуста, И. О. Рабочая программа учебного курса по информатике в 7–9 классах / И. О. Капуста. — 2021-2022 учебный год. — URL: https://s-sin.k-edu.ru/sites/s-sin.k-edu.ru/files/rabochaya_programma_po_informatike_7-9_klassy.pdf (дата обращения: 22.05.2024). — Доступ: открытый.
2. А кодером можно стать? Чему учат (и не учат) на школьных уроках информатики // Редакция Mel. — 2022. — URL: https://mel.fm/ucheba/shkola/3194078-a-koderom-mozhno-stat-chemu-uchat-i-ne-uchat-na-shkolnykh-urokakh-informatiki (дата обращения: 22.05.2024). — Доступ: открытый.
3. Цыбуля, И. Н., Самыкбаева, Л. А., Беляев, А. А., Осипова, Н. Н., Мамбетакунов, У. Э. Информатика: 7-9 класс. Учебное пособие для 7-9 классов общеобразовательных учреждений с русским языком обучения / И. Н. Цыбуля, Л. А. Самыкбаева, А. А. Беляев, Н. Н. Осипова, У. Э. Мамбетакунов. — 1-е изд. — Фонд Сорос-Кыргызстан, 2019. — 205 с. — ISBN 978-5-94074-863-2. — URL: https://goo.su/7khJPT (дата обращения: 22.05.2024). — Доступ: открытый.
4. Требования к современным пользовательским интерфейсам // Вестник Московского государственного университета печати. — 2016. — URL: https://cyberleninka.ru/article/n/trebovaniya-k-sovremennym-polzovatelskim-interfeysam (дата обращения: 22.05.2024). — Доступ: открытый.
5. Преобразование кадров для детей от рождения до 8 лет: Объединяющая основа / Национальная академия наук США. — 2015. — URL: https://www.ncbi.nlm.nih.gov/books/NBK310550/ (дата обращения: 22.05.2024). — Доступ: открытый.
6. Бриггс, Джейсон. Python для детей. Самоучитель по программированию / Джейсон Бриггс ; переводчик Станислав Ломакин ; редактор Дарья Абрамова. — 6-е изд. — Москва : Манн, Иванов и Фербер, 2022. — ISBN 978-5-94074-863-2. — URL: https://www.litres.ru/book/jason-r-briggs-21070454/python-dlya-detey-samouchitel-po-programmirovaniu-23781112/ (дата обращения: 22.05.2024). — Доступ: открытый.
7. React Documentation // Meta Open Source. — 2024. — URL: https://react.dev/learn (дата обращения: 22.05.2024). — Доступ: открытый.
8. Next.js Documentation // Vercel. — URL: https://nextjs.org/docs (дата обращения: 22.05.2024). — Доступ: открытый.
9. Pyodide Documentation // Pyodide Community. — URL: https://pyodide.org/en/stable/ (дата обращения: 22.05.2024). — Доступ: открытый.

10. Python Documentation // Python Software Foundation. — URL: https://docs.python.org/3/ (дата обращения: 22.05.2024). — Доступ: открытый.
11. Chakra UI Documentation // Chakra UI Team. — URL: https://chakra-ui.com/docs (дата обращения: 22.05.2024). — Доступ: открытый.
12. Tailwind CSS Documentation // Tailwind Labs. — URL: https://tailwindcss.com/docs (дата обращения: 22.05.2024). — Доступ: открытый.
13. Framer Motion Documentation // Framer. — URL: https://www.framer.com/motion/ (дата обращения: 22.05.2024). — Доступ: открытый.
14. React Code Blocks Documentation // npm. — URL: https://www.npmjs.com/package/react-code-blocks (дата обращения: 22.05.2024). — Доступ: открытый.
15. React Codemirror Documentation // npm. — URL: https://www.npmjs.com/package/@uiw/react-codemirror (дата обращения: 22.05.2024). — Доступ: открытый.
16. TypeScript Documentation // Microsoft. — URL: https://www.typescriptlang.org/docs/ (дата обращения: 22.05.2024). — Доступ: открытый.