МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное учреждение высшего образования
«КРАСНОЯРСКИЙ ГОСУДАРСТВЕННЫЙ ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.П. АСТАФЬЕВА»
(КГПУ им. В.П. Астафьева)

Институт математики, физики и информатики
Выпускающая кафедра: информатики и информационных технологий в образовании

Ефимова Оксана Романовна

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Чат-бот как средство организации самостоятельной работы обучающихся 8-9 классов при изучении программирования

Направление подготовки:
44.03.05 Педагогическое образование (с двумя профилями подготовки)
Направленность (профиль) образовательной программы:
Математика и Информатика

ДОПУСКАЮ К ЗАЩИТЕ
Заведующий кафедрой
д-р пед. наук, профессор Н.И. Пак
10.06.2025
(дата, подпись)

Научный руководитель
канд. пед. наук, доцент И.А. Яшина

Дата защиты 18.06.2025

Обучающийся
О.Р. Ефимова

Оценка отлично
Прописью

Красноярск 2025

Содержание

Введение..... 3
Глава 1. Теоретические аспекты организации самостоятельной работы обучающихся при изучении программирования.....6
§ 1.1. Особенности организации учебного процесса при изучении программирования.....6
§ 1.2. Роль и условия осуществления самостоятельной работы обучающихся.....16
Глава 2. Методические аспекты использования чат-бота в процессе обучения программированию в 8-9 классах.....24
§ 2.1. Технологические и методические аспекты создания образовательного чат-бота в Telegram на платформе PyCharm.....24
§ 2.2. Методические рекомендации по интеграции чат-бота в учебный процесс для формирования навыков программирования.....32
Заключение..... 40
Библиографический список..... 42
Приложение 1.....46
Приложение 2.....51
Приложение 3.....54
Приложение 4.....56

Введение

Изучение программирования каждый год становится все более востребованным. Этот процесс обусловлен стремительной цифровизацией всех сфер жизни, когда даже в традиционно «некомпьютерных» профессиях появляется потребность в базовых навыках автоматизации процессов и работы с данными.

Особенно ярко важность программирования проявляется в школьном образовании, где оно стало неотъемлемой частью подготовки к государственным экзаменам (ОГЭ в 9 классе и ЕГЭ в 11 классе). По данным ФИПИ за 2024 год задание 16 из ОГЭ правильно решают 29,3% школьников.

Такой низкий процент выполнения задания объясняется не только сложностью задания, но и тем фактом, что значительная часть обучающихся вообще не приступает к его решению, что может свидетельствовать либо о недостаточной подготовке, либо о непонимании принципов алгоритмизации.

ЕГЭ по информатике проходит в форме компьютерного тестирования, поэтому большую часть заданий можно решать с использованием программирования, что дает преимущество обучающимся и больше времени на решение контрольно-измерительных материалов. Такой формат теоретически должен способствовать более практическому подходу к решению задач. Однако статистика показывает обратное: так каждое задания с 24-27, требующие написания программ на языке программирования, в 2024 году решило меньше 11% школьников.

На уроки информатики в средней школе отводится 1 час в неделю. Такого количества учебного времени катастрофически недостаточно не только для освоения программирования, но даже для формирования базовых алгоритмических навыков. Поэтому для успешной подготовки к экзаменам, обучающиеся должны заниматься самостоятельной подготовкой.

Самостоятельная работа в изучении программирования необходима, так как сложно приобрести и отточить необходимые навыки на уроке информатики за отведенное время. Разумеется, в данной ситуации помогают

факультативы, секции и кружки, но важнейшую роль играет самостоятельная работа обучающегося в изучении программирования.

Противоречие: при изучении программирования, обучающиеся должны выполнять задания, но низкий уровень самостоятельности и отсутствие мотивации препятствует этому.

Проблема: как повысить результативность самостоятельной работы в области изучения программирования.

Объект: процесс обучения программированию в школьном курсе информатики.

Предмет: чат-бот как средство организации и повышения результативности самостоятельной работы при обучении школьников программированию.

Цель работы: теоретически обосновать и разработать чат-бот в Telegram для организации самостоятельной работы при обучении программированию в 8-9 классах.

Для достижения цели поставлены следующие **задачи**:

1. Проанализировать особенности организации самостоятельной работы обучающихся при обучении программированию.
2. Выявить условия повышения результативности самостоятельной работы.
3. Спроектировать и разработать чат-бот в Telegram для изучения языка программирования Python.
4. Описать методические рекомендации по использованию чат-бота в обучении программированию.

Область применения полученных результатов:

Созданный чат-бот для изучения языка программирования Python может применяться как в традиционном очном формате, так и в дистанционном обучении. Он также может быть полезен для самостоятельной работы учеников. Удобство использования данного чат-бота

оценивается с помощью анкетирования школьников после использования чат-бота.

Структура выпускной квалификационной работы состоит из введения, двух глав, заключения, библиографического списка и приложений.

Глава 1. Теоретические аспекты организации самостоятельной работы обучающихся при изучении программирования

§ 1.1. Особенности организации учебного процесса при изучении программирования

В современном образовании программирование стало одним из ключевых аспектов. В школьной программе по информатике этому направлению уделяется большое внимание. В условиях стремительного развития информационных технологий одной из основных проблем в преподавании информатики является недостаточное количество времени, отведенное на изучение отдельных тем раздела «Алгоритмизация и программирование» [21].

Тематическое планирование, содержащееся в Примерных рабочих программах [26, 27, 28, 29] учебного предмета «Информатика», для 5–11 классов базового и углубленного уровней представлено в таблице 1.

Таблица 1 – Распределение учебного материала по разделу «Алгоритмизация и программирование»

Темы	*Базовый уровень*	*Углубленный уровень*
5 класс		
Алгоритмы и исполнители	-	8 часов
Работа в среде программирования	-	2 часа
Итого:	-	10 часов
6 класс		
Основные алгоритмические конструкции	-	8 часов
Темы	*Базовый уровень*	*Углубленный уровень*
Вспомогательные алгоритмы	-	4 часа
Итого:	-	12 часов

7 класс		
Исполнители и алгоритмы. Алгоритмические конструкции	-	16 часов
Компьютерная графика и анимация	-	8 часов
Итого:	-	24 часа
8 класс		
Исполнители и алгоритмы. Алгоритмические конструкции	10 часов	-
Язык программирования	9 часов	34 часа
Анализ алгоритмов	2 часа	-
Итого:	21 час	34 часа
9 класс		
Разработка алгоритмов и программ	6 часов	24 часа
Управление	2 часа	4 часа
Итого:	8 часов	28 часов
10 класс		
Введение в программирование	-	16 часов
Вспомогательные алгоритмы	-	8 часов
Численные методы	-	5 часов
Алгоритмы обработки символьных данных.	-	5 часов
Алгоритмы обработки массивов	-	10 часов
Итого:	-	44 часа
11 класс		
Алгоритмы и элементы программирования	11 часов	-
Элементы теории алгоритмов	-	6 часов
Алгоритмы и структуры данных	-	28 часов
Основы объектно-ориентированного программирования	-	16 часов
Итого:	11 часов	50 часов
Общее количество часов за весь раздел	**40 часов**	**202 часа**

«Алгоритмизация и программирование»		

На начальном этапе обучения, в пятых и шестых классах, знакомство с программированием происходит через систему внеурочных занятий, где особое внимание уделяется формированию фундаментальных понятий алгоритма и исполнителя. В качестве инструментов используются специализированные образовательные среды, такие как КуМир или Scratch, позволяющие в доступной визуальной форме освоить базовые принципы управления исполнителем и построения алгоритмических конструкций [22].

Изучение раздела на базовом уровне, как видно в таблице 1, начинается в 8 классе. На данном этапе обучающиеся переходят к текстовому программированию и изучают один из языков программирования (Python, C++, C#, Java, Паскаль, школьный алгоритмический язык, Бейсик).

В девятом классе программирование приобретает более системный и практико-ориентированный характер: учащиеся закрепляют базовые знания, полученные в восьмом классе, и переходят к освоению более сложных структур данных и алгоритмических приемов.

В старшей школе на базовом уровне язык программирования изучается для понимание ключевых принципов работы алгоритмов при выполнении стандартных задач, которые могут встретиться на едином государственном экзамене по информатике. На углубленном уровне освоение специализированного языка программирования — ключевая задача для тех, кто хочет участвовать в олимпиадах по программированию и развиваться в сфере информационных технологий.

Выстраивание непрерывного курса - одна из основных тенденций развития школьного курса информатики [30]. Однако, анализ содержания федеральных рабочих программ основного и среднего общего образования показывает, что на базовом уровне изучение программирования носит фрагментарный характер, в то время как углубленный курс предполагает

системную работу с профессиональными языками на протяжении нескольких лет.

Стоит отметить, что в базовых учебниках по информатике не представлен полный синтаксис какого-либо одного языка программирования, что оставляет учителю свободу в выборе инструмента обучения. Эта ситуация отражается и в структуре ОГЭ, где задания предлагаются на пяти различных языках: Basic, Python, алгоритмический язык, Pascal и C++. Такой подход, с одной стороны, позволяет учитывать специфику образовательного учреждения, с другой - создает определенные сложности при формировании единых образовательных стандартов.

В процессе изучения основ информатики для создания алгоритмов обучающимся предоставляется выбор между учебными (Python, Pascal, школьный алгоритмический язык) и профессиональными языками программирования (Python, C++, Java, C#). Однако при углубленном изучении информатики основное внимание уделяется разработке программ с использованием профессиональных языков программирования. Python становится универсальным языком, который можно использовать как на базовом, так и на углубленном уровне изучения информатики [2].

Изучение языка программирования Python обладает рядом значительных преимуществ по сравнению с другими языками [9]:

1. Обладает низким порогом входа благодаря простому и понятному синтаксису, что облегчает его изучение.
2. Является более высокоуровневым по сравнению с такими языками, как C и Pascal, и имеет обширную библиотеку, а также возможности для создания программ с графическим интерфейсом.
3. По данным многих интернет-ресурсов, Python является одним из самых популярных и быстрорастущих языков программирования, что подтверждается рейтингом TIOBE Index (рис. 1) [20].

4. Подходит для решения учебных задач по информатике, а также для реализации проектов в различных областях.

May 2025	May 2024	Change	Programming Language	Ratings
1	1		Python	25.35%
2	3	↑	C++	9.94%
3	2	↓	C	9.71%
4	4		Java	9.31%
5	5		C#	4.22%
6	6		JavaScript	3.68%
7	8	↑	Go	2.70%
8	7	↓	Visual Basic	2.62%
9	11	↑	Delphi/Object Pascal	2.29%
10	9	↓	SQL	1.90%
11	10	↓	Fortran	1.78%
12	24	⇈	R	1.46%
13	22	⇈	Ada	1.42%
14	17	↑	Scratch	1.35%
15	16	↑	PHP	1.22%
16	30	⇈	Perl	1.20%
17	14	↓	MATLAB	1.02%
18	12	⇊	Assembly language	0.97%
19	18	↓	Rust	0.94%
20	20		COBOL	0.88%

Рисунок 1 - Ежегодный рейтинг TIOBE Index for May 2025

Python во многом превосходит другие языки программирования, однако следует учитывать его недостатки:

1. Скорость выполнения программ на Python работают медленнее по сравнению с другими языками программирования.
2. При копировании кода из внешних источников происходит потеря отступов.

3. Размер программы увеличивается после преобразования кода в формат «exe».
4. Использование Unicode и кириллицы иногда приводит к появлению нечитаемых символов в выводе программы.

Следует отметить, что у любого языка программирования есть свои сильные и слабые стороны. Однако, наиболее важным аспектом является не столько выбор конкретного языка программирования, сколько методика его преподавания и достижение образовательных результатов, закрепленных в Федеральных государственных образовательных стандартах. Процесс обучения программированию представляет собой многоэтапную систему, где центральное место занимает не столько изучение синтаксиса конкретного языка, сколько освоение фундаментальных принципов алгоритмизации, развитие вычислительного мышления и способности к решению практических задач. В процессе обучения программированию в школе можно выделить несколько ключевых моментов:

1. Выбор языка программирования для изучения.
2. Организация учебного процесса.
3. Применение методов и технологий обучения программированию.

При организации учебного процесса учителю необходимо выбирать методы и формы обучения, а также учитывать принципы преподавания программирования (рис. 2) [22].

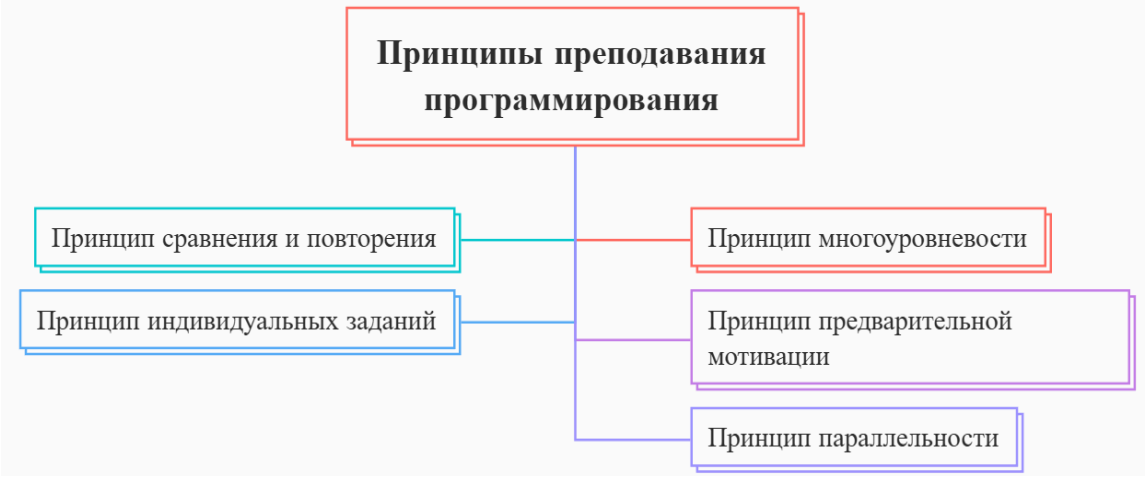

Рисунок 2 - Принципы преподавания программирования

Принцип многоуровневости связан с тем, что в разделе «Алгоритмизация и программирование» темы и элементы языка тесно связаны друг с другом, поэтому возникает сложность в установлении очередности изучения определений понятий и конструкций языка. Суть данного принципа заключается в разделении учебного материалы на этапы. Базовый уровень – простейшие элементы языка (целочисленные/вещественные типы, арифметические операции, стандартные функции), позволяющие быстро создавать простые программы. Далее следует второй этап или основной уровень – углубленное изучение конструкций языка и их возможностей. И продвинутый уровень – дополнительные, редко используемые возможности (для профильных классов или факультативов).

Принцип предварительной мотивации состоит в том, что тему лучше изучать, используя метод индукции, т.е. «от частного к общему». А также использовать схему, представленную на рисунке 3.

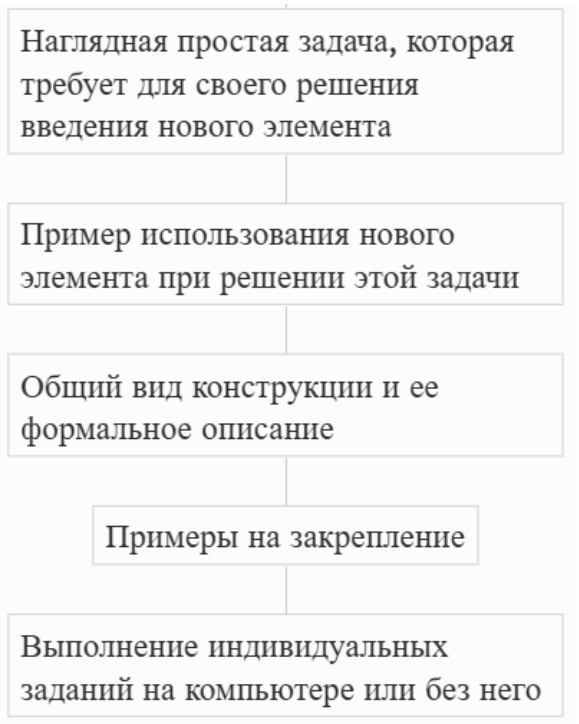

Рисунок 3 - Схема введения элементов языка программирования по принципу предварительной мотивации

В процессе освоения программирования важно помнить о принципе сопоставления и повторения. Он подразумевает анализ разных способов решения одной задачи и выбор наиболее эффективного из них.

Рассмотрим и сравним предполагаемые предметные результаты по разделу «Алгоритмизация и программирование» на углублѐнном и базовом уровне изучения дисциплины «Информатика» в рамках общеобразовательной школы согласно Федеральным рабочим программам (Таблица 2).

Таблица 2 – Предметные результаты раздела «Алгоритмизация и программирование» в школьном курсе информатики

Класс/тема	**Предметные результаты (по уровням обучения)**	
	Базовый уровень	*Углубленный уровень*
7	Раздел не изучается	Раскрывать смысл понятий «исполнитель», «алгоритм», «программа», понимая разницу между употреблением этих терминов в обыденной речи и в информатике. Описывать алгоритм решения задачи различными способами, в том числе в виде блок-схемы. Разбивать задачи на подзадачи. Составлять, выполнять вручную и на компьютере несложные алгоритмы с использованием ветвлений, циклов и вспомогательных алгоритмов для управления исполнителями, такими как Робот, Черепашка, Чертёжник
8	Раскрывать смысл понятий «исполнитель», «алгоритм», «программа». Описывать алгоритм решения задачи различными способами, в том числе в виде блок-схемы. Составлять, выполнять вручную и на компьютере несложные алгоритмы с использованием ветвлений и циклов для управления исполнителями, такими как Робот, Черепашка, Чертёжник. Использовать константы и переменные различных типов. Создавать и отлаживать программы на одном из языков	Уметь выбирать подходящий алгоритм для решения задачи. Свободно оперировать понятиями: переменная, тип данных, операция присваивания, арифметические и логические операции, включая операции целочисленного деления и остатка от деления. Записывать логические выражения на изучаемом языке программирования. Анализировать предложенные алгоритмы, в том числе определять, какие результаты возможны при заданном множестве исходных

	Базовый уровень	*Углубленный уровень*
8	программирования (Python, C++, Паскаль, Java, C#, Школьный Алгоритмический Язык). Разбивать задачи на подзадачи.	значений. Определять возможные входные данные, приводящие к определенному результату. Создавать и отлаживать программы на современном языке програм-я общего назначения (Python, C++, Java, C#). Разбивать задачи на подзадачи
9	Составлять, выполнять вручную и на компьютере несложные алгоритмы с использованием ветвлений, циклов и вспомогательных алгоритмов для управления исполнителями, такими как Робот, Черепашка, Чертёжник. Составлять и отлаживать программы, реализующие типовые алгоритмы обработки числовых последовательностей на одном из языков программирования (Python, C++, Паскаль, Java, C#, Школьный Алгоритмический Язык)	Создавать и отлаживать программы на современном языке программирования общего назначения (Python, C++, Java, C#), реализующие алгоритмы обработки числовых данных с использованием подпрограмм (процедур, функций). Составлять и отлаживать программы на современном языке программирования общего назначения из приведенного выше списка, реализующие: несложные рекурсивные алгоритмы; алгоритмы сортировки массивов, двоичного поиска в упорядоченном массиве; основные алгоритмы обработки двумерных массивов; простые приемы динамического программирования
10	Раздел не изучается	Понимание базовых алгоритмов обработки числовой и текстовой информации (запись чисел в позиционной системе счисления; нахождение всех простых чисел в заданном диапазоне. Обработка многоразрядных целых чисел. Анализ символьных строк и др.), алгоритмов поиска и сортировки. Умение определять сложность изучаемых в курсе базовых алгоритмов (суммирование элементов массива, сортировка массива, переборные алгоритмы, двоичный поиск) и приводить примеры нескольких алгоритмов разной сложности для решения одной задачи.
11	Умение читать и понимать программы, реализующие несложные алгоритмы обработки числовых и	Владение универсальным языком программирования высокого уровня (Python, Java, C++, C#),

	Базовый уровень	*Углубленный уровень*
11	текстовых данных (в том числе массивов и символьных строк) на выбранном для изучения универсальном языке программирования высокого уровня (Паскаль, Python, Java, C++, C#). Определять без использования компьютера результаты выполнения несложных программ, включающих циклы, ветвления и подпрограммы, при заданных исходных данных.	представлениями о базовых типах данных и структурах данных. Умение использовать основные управляющие конструкции. Умение осуществлять анализ предложенной программы: определять результаты работы программы при заданных исходных данных. Определять, при каких исходных данных возможно получение указанных результатов; выявлять данные, которые могут привести к ошибке в работе программы. Умение разрабатывать и реализовывать в виде программ базовые алгоритмы. Знать функциональные возможности инструментальных средств среды разработки

Предполагаемые предметные результаты углублѐнного уровня изучения информатики ставят более расширенные требования перед педагогами и обучающимися в разделе «Алгоритмизация и программирование». Количество предполагаемых результатов между углублѐнным и базовым уровнем изучения данной дисциплины отличается в 1,5 раза.

В рамках курса информатики для учеников 10–11 классов на базовом уровне, по сути, продолжается изучение программирования, которое началось в основной школе. На углубленном уровне отличается разнообразием вопросов.

Важно понимать, что цель обучения — не просто изучение языка программирования или подготовка специалистов высокого уровня. Задача — освоить принципы и методы программирования, научиться создавать алгоритмы и решать задачи с помощью кода [15].

§ 1.2. Роль и условия осуществления самостоятельной работы обучающихся

Одним из ключевых вопросов методики преподавания является вопрос самостоятельной работы обучающихся. В педагогике нет единого мнения о том, что такое самостоятельная работа. В разные годы учёные предлагали свои трактовки, подчеркивая активность учащихся, роль педагога, необходимость специальной организации и осознание целей задания [23].

Обратимся к понятию «самостоятельная работа». Рассмотрим различные трактовки авторов в таблице 3.

Таблица 3 – Сравнения определений понятий

Термин	**Определение**	**Автор/источник**
Самостоятельность	Результат большой внутренней работы человека, его способность ставить не только индивидуальные цели и задачи, но и определять направление своей деятельности	С.Л. Рубинштейн [17]
	Волевое качество, которое проявляется в действиях человека и выражается в том, что он готов самостоятельно принимать решения, не поддаваясь чужому влиянию, и обладает самокритичностью.	В.В. Богословский [18]
Самостоятельная работа	Способность обучающихся овладевать знаниями и умениями, недостающими для решения учебных задач	Г.А. Цукерман и А.Л. Венгер [31]
	Учебное задание, выполнение которого приводит к получению новых знаний, умений и навыков. Внутренним содержанием самостоятельной работы является познавательная задача, содержащая противоречие между данным и искомым, что стимулирует мыслительную активность	П.И. Пидкасистый [14]
	Самостоятельная работа сочетает функции формы организации обучения, системы заданий и	А.И. Хамитова [34]

	деятельности по их выполнению	
	Многообразные виды индивидуальной и коллективной учебной деятельности школьников, осуществляемой ими на классных и внеклассных занятиях или дома по заданиям без непосредственного участия учителей	Российской педагогическая энциклопедия [16]

А.В. Усова рассматривают самостоятельную работу как метод обучения, а В.А. Сластенин — как особую форму обучения, предполагающую самостоятельность ученика [25].

Кроме того, самостоятельная деятельность обучающихся – это средство обучения, которое [7]:

- отвечает поставленной дидактической цели и задаче урока в данной ситуации;
- формирует компетенции и умения для решения определённых задач в сфере познания, а также способствует повышению уровня знаний на этапе перехода от незнания к пониманию;
- выступает в качестве инструмента, который позволяет учащемуся управлять своей деятельностью, а также служит для педагога способом контроля и руководства в процессе обучения;
- является одним из ключевых элементов в развитии мотивации к постоянному и осознанному расширению кругозора и навыков, которые позволяют анализировать и структурировать информацию для успешного преодоления сложных ситуаций и решения новых познавательных задач.

Самостоятельная работа – это один из этапов и составляющих элементов процесса обучения. Он используется во ФГОСах, рабочих программах, тематических планах. К сожалению, часто этот вид работы не используется оптимально педагогом для достижения поставленных целей и

задач в процессе обучения и игнорируется самими обучающимися, что негативно сказывается на результате обучения [32].

Кроме того, правильно организованная и контролируемая самостоятельная работа может стать дополнительным инструментом в процессе обучения, а также основой для развития целого ряда навыков, необходимых для успешной работы в выбранной сфере деятельности.

Контроль за качеством самостоятельной работы обучающихся представляет собой значимую часть учебного процесса. Одним из ключевых условий его эффективной организации является наличие технических средств. Это может быть оборудование учителя (как домашнее, так и рабочее), навыки работы с различными программами, а также обеспеченность обучающихся необходимой техникой. Также важно, чтобы обучение проходило на платформе, которая позволяет эффективно взаимодействовать между учителем и учениками.

Не менее важно соблюдать принцип системности в обучении. Он подразумевает под собой не только выполнение заданий, но и регулярную проверку и контроль со стороны педагогов. Иначе теряется смысл самостоятельной работы, который прописан в учебных программах.

В процессе обучения важно, чтобы преподаватель давал оценку работе ученика. Это своего рода обратная связь в процессе обучения. В этом процессе важно использовать личностно-ориентированный подход: если ученик хорошо подготовлен, ему можно дать более сложное задание. Если же ученик недостаточно подготовлен, то оценка его работы должна помочь ему вырасти в своих глазах и в глазах других учеников.

Задача самостоятельной учебной работы — сформировать у учащихся способность к самообразованию и самообучению, которые будут необходимы им на протяжении всей жизни.

В ходе самостоятельной работы ученики повторяют, закрепляют, систематизируют, обобщают и углубляют свои знания, умения и навыки, а также осваивают новые способы деятельности. Всё это помогает им

формировать универсальные учебные действия в соответствии с учебной программой. При этом контроль за самостоятельной работой не является главной целью.

В процессе организации самостоятельной работы на уроках следует придерживаться определенных правил. Вот как должна быть устроена система самостоятельной работы обучающихся [33]:

1. Самостоятельная работа должна помогать в решении основных образовательных задач. Ученики должны не только получать знания, но и развивать свои познавательные способности, чтобы применять их на практике.
2. Методы, формы и инструменты самостоятельной работы должны соответствовать основным принципам обучения. Она должна быть тесно связана с теорией и практикой, а также с принципами обучения.
3. Самостоятельная работа должна быть разнообразной по своим целям и содержанию. Она должна способствовать развитию различных умений и навыков у младших школьников.
4. Основой самостоятельной работы является последовательное выполнение домашних и классных заданий. Домашние и классные работы должны быть связаны между собой, вытекать из предыдущих заданий и готовить учеников к последующим.

Самостоятельная работа может быть организована в различных видах и формах, каждая из которых имеет свои особенности и преимущества (рис. 4).

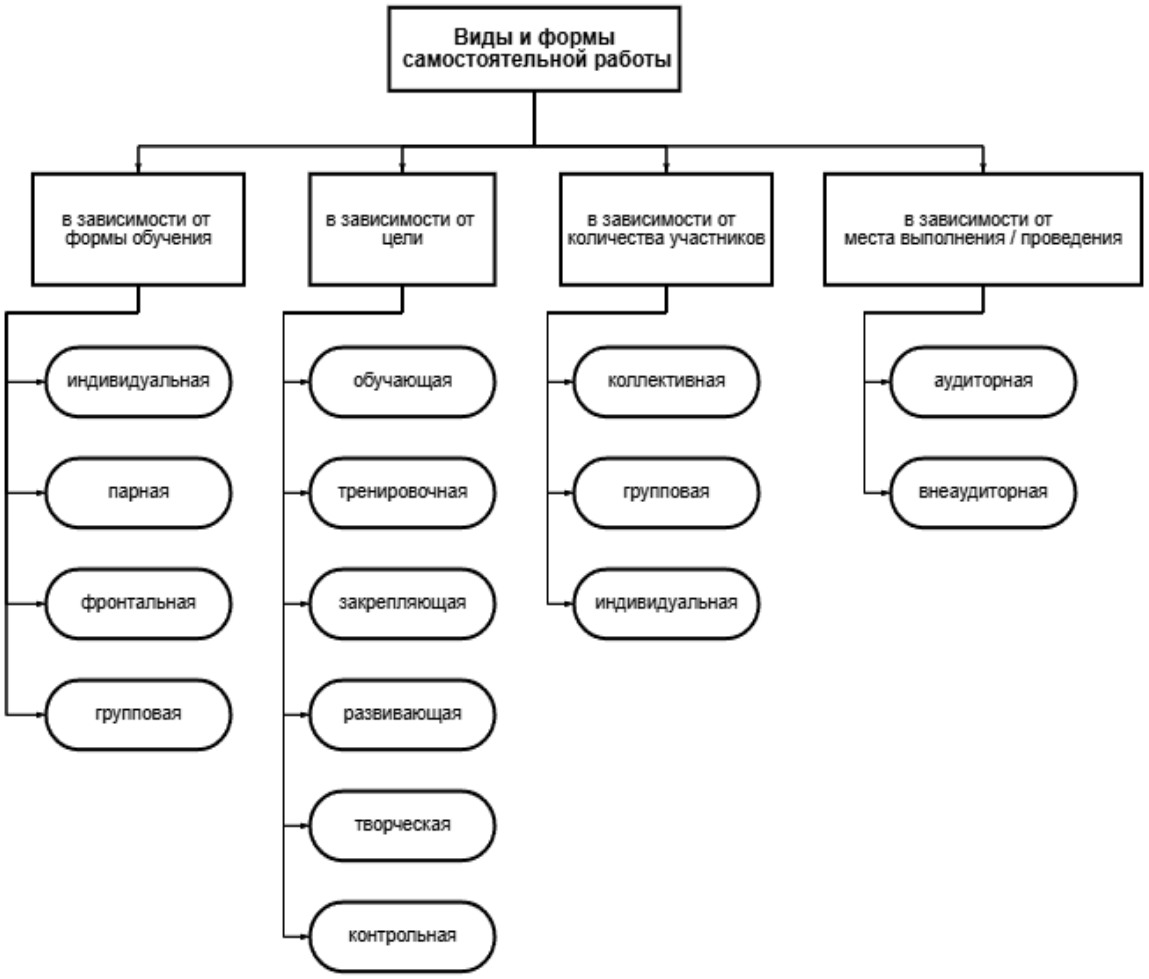

Рисунок 4 - Классификация видов и форм самостоятельной работы

Рассмотрим роль самостоятельной работы в зависимости от формы обучения. Индивидуальная форма предполагает выполнение заданий лично каждым учащимся, что позволяет развивать самостоятельность, концентрацию и персональную ответственность за результат. Фронтальная форма подразумевает одновременное выполнение одинаковых заданий всеми участниками учебного процесса, что способствует синхронизации обучения и облегчает контроль со стороны преподавателя. Групповая форма основана на совместной работе в небольших коллективах, где учащиеся взаимодействуют друг с другом, обмениваются идеями и учатся работать в команде. Каждая из этих форм способствует формированию различных компетенций и может быть эффективно использована в зависимости от целей обучения и особенностей учебного процесса.

Выделим основные виды самостоятельной работы при изучении информатики [18].

Домашнее задание можно считать основным и систематическим видом самостоятельной деятельности. В этом контексте под домашним заданием подразумевается комплекс задач и упражнений, которые помогают закрепить материал, изученный на уроке. Эти задания выдаются ученикам почти на каждом уроке, и уровень мотивации учащихся не оказывает значительного влияния на то, как контролируется и поддерживается эта самостоятельная работа.

Ещё один вид самостоятельной работы — это выполнение **индивидуальных задач**, например, по программированию или освоению специализированного ПО. В этом случае важно обеспечить методическую поддержку, установить сроки выполнения, организовать промежуточный контроль и предоставить консультации.

Еще одной формой самостоятельной работы выступает написание **реферата** по заданной теме. Реферат — это не просто устное выступление, рекомендуется дополнить его презентацией и примерами.

Значимым заданием является составление краткого **конспекта** по изученной теме. Это творческое задание, которое помогает не только систематизировать и выделить главное в материале, но и развить навыки самоорганизации и умение работать с дополнительной научной литературой.

Работа над **индивидуальным или групповым проектом** — это отличный способ развить навыки самообучения. Ученик должен самостоятельно определить цель проекта, спланировать использование ресурсов для достижения целей и отслеживать выполнение отдельных шагов в процессе работы.

Особое внимание в самостоятельной работе стоит уделить учебным материалам. Учебные материалы классифицируются на три основные группы (Таблица 4) [13].

Таблица 4 – Виды учебных материалов и их функции

Учебные материалы	Вид учебных материалов	Функции
Печатные	Учебники, пособия, хрестоматии, задачники, рабочие тетради и др	Они традиционно фиксируют содержание дисциплины, но должны также способствовать личностному развитию обучающихся. Важно наличие не только информативного, но и репродуктивного, творческого, эмоционально-ценностного компонентов
Аудиовизуальные	Звукозаписи, фильмы, телепередачи	Используются как дополнительные источники информации, способные наглядно раскрывать содержание дисциплины. Их эффективность возрастает при включении заданий на анализ, синтез, применение знаний
Электронные	Электронные учебники, энциклопедии, обучающие игры, автоматизированные системы, тренажеры, мультимедийные презентации, образовательные веб-сайты, чат-боты	Электронные ресурсы предоставляют свободу выбора времени и формата обучения, способствуют развитию информационной компетентности, навыков самоконтроля и интерактивного общения

Вывод по главе:

В первой главе было изучено и проанализировано место раздела «Алгоритмизация и программирование» в школьном курсе информатики. Согласно данным проанализированных источников на базовом уровне изучение программирования носит фрагментарный характер, в то время как

углубленный курс предполагает системную работу с профессиональными языками на протяжении нескольких лет.

В рамках изучения данного раздела важно организовать самостоятельную работу обучающихся, так как она формирует критическое мышление и навыки решения задач. Данный формат работы позволит обучающимся повторить пройденный материал, закрепить знания и отработать практические навыки.

Глава 2. Методические аспекты использования чат-бота в процессе обучения программированию в 8-9 классах

§2.1. Технологические и методические аспекты создания образовательного чат-бота в Telegram на платформе PyCharm

В наши дни жизнь человека невозможно вообразить без интернета. В сети мы получаем знания, которые используем в нашей повседневной жизни. Но самая главная функция интернета — общение. Сегодня для этого чаще всего применяют социальные сети и мессенджеры.

Особого внимания заслуживает Telegram — бесплатный кроссплатформенный мессенджер, поддерживающий обмен текстовыми, голосовыми и видеосообщениями. Его ключевыми преимуществами являются:

- надежное шифрование всех сообщений
- наличие секретных чатов с функцией самоуничтожения
- поддержка всех операционных систем
- высокая скорость передачи данных
- возможность использования чат-ботов

Благодаря этим особенностям Telegram остается одним из самых востребованных инструментов для безопасного и удобного общения в современном цифровом мире.

Чат-бот — это программное приложение, виртуальный робот-собеседник, предназначенный для интерактивного общения с одним или несколькими пользователями с помощью текста или преобразования текста в речь вместо обеспечения прямого контакта с живым агентом-человеком [4].

Существует два подхода к реализации программного обеспечения для разработки чат-бота:

- однокомпонентная архитектура. В одном приложении, адаптированном под каждую платформу, содержится вся

необходимая информация. Однако оно не предусматривает возможности социального взаимодействия между пользователями.

- многокомпонентная архитектура или клиент-сервер. Описывает взаимодействие между пользователем и сервером. Пользователь выполняет определенные действия, после чего отправляет запрос. После получения запроса сервер обрабатывает его и отправляет пользователю результат. [11]

В контексте нашей задачи для создания чат-бота оптимальным решением является использование многокомпонентной архитектуры (клиент-сервер). (Рис. 5).

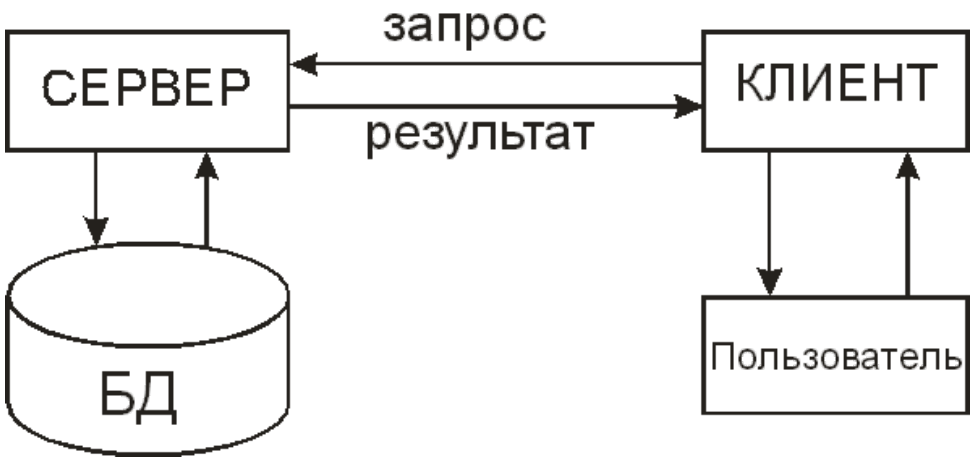

Рисунок 5 - Многокомпонентная архитектура взаимодействия

На рисунке 6 представлена блок-схема, описывающая режим работы чат-бота. Алгоритм начинается с получения входных данных (вопрос или фраза пользователя). Сначала бот пытается найти ответ в своей базе данных. Если ответ существует, то он воспроизводится, если нет, то он переходит в режим обучения. В данном режиме чат-бот принимает не только входные данные, но и соответствующий ответ и сохраняет его в базу данных.

После выполнения одного из режима - либо выдачи готового ответа, либо сохранение нового, бот завершает работу.

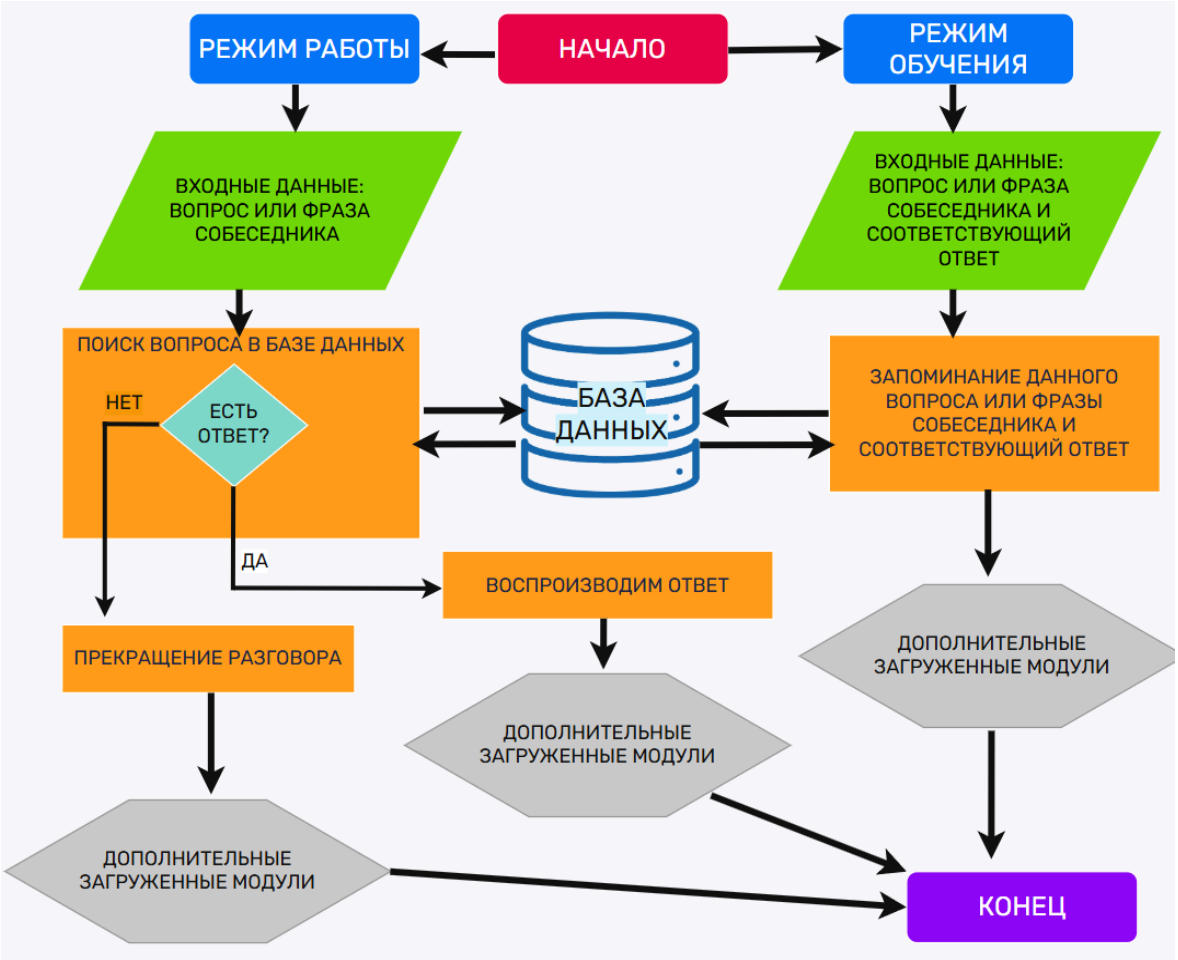

Рисунок 6 - Основа чат-бота

Разработка Telegram бота состоит из нескольких этапов:

1. **Идея и планирование.** Прежде всего, нужно понять, для чего создаётся чат-бот. Затем следует разработать основные команды, сценарии взаимодействия с пользователями и возможные ответы.
2. **Выбор средств и среды разработки.** Для создания чат-бота можно использовать специальные онлайн-сервисы. Но если вы хотите получить полный контроль над функционалом бота и не зависеть от ограничений, то лучше заняться самостоятельным программированием. Далее необходимо выбрать язык программирования для написания кода (Python, Java, PHP или другие языки).
3. **Разработка чат-бота.** Для создания и управления чат-ботом в Telegram нужно зарегистрировать его в системе. После этого вы

получите уникальный токен, который свяжет бота с вашим аккаунтом.

4. **Настройка сервера**. Чтобы чат-бот работал круглосуточно необходимо запустить его на сервере. Для этого можно использовать различные облачные платформы.
5. **Публикация.**

Разберем подробнее этапы создания Telegram-бота на примере бота для изучения языка программирования Python.

Основной идеей является создание Telegram бота для изучения Python, где обучающимся будут предоставляться:

1. Теоретический материал по основным темам программирования на языке Python.
2. Практические задания в текстовом виде.
3. Проверка кода с помощью GPT, который анализирует его и дает обратную связь.

В качестве мессенджера для реализации чат-бота была выбрана платформа Telegram, поскольку она предоставляет удобный API для разработки ботов. Telegram бот создан в интегрированной среде разработки PyCharm Professional Edition - редактор кода, поддерживающий множество языков программирования, например, Python, JavaScript, CSS и др. Данная среда разработки является популярным выбором среди разработчиков Python, благодаря своим удобным функциям и возможностям, таким как автодополнение кода, отладка и интеграция с системой контроля версий.

В PyCharm можно легко работать с разными базами данных прямо из приложения. На рисунке 7 представлен интерфейс IDE PyCharm.

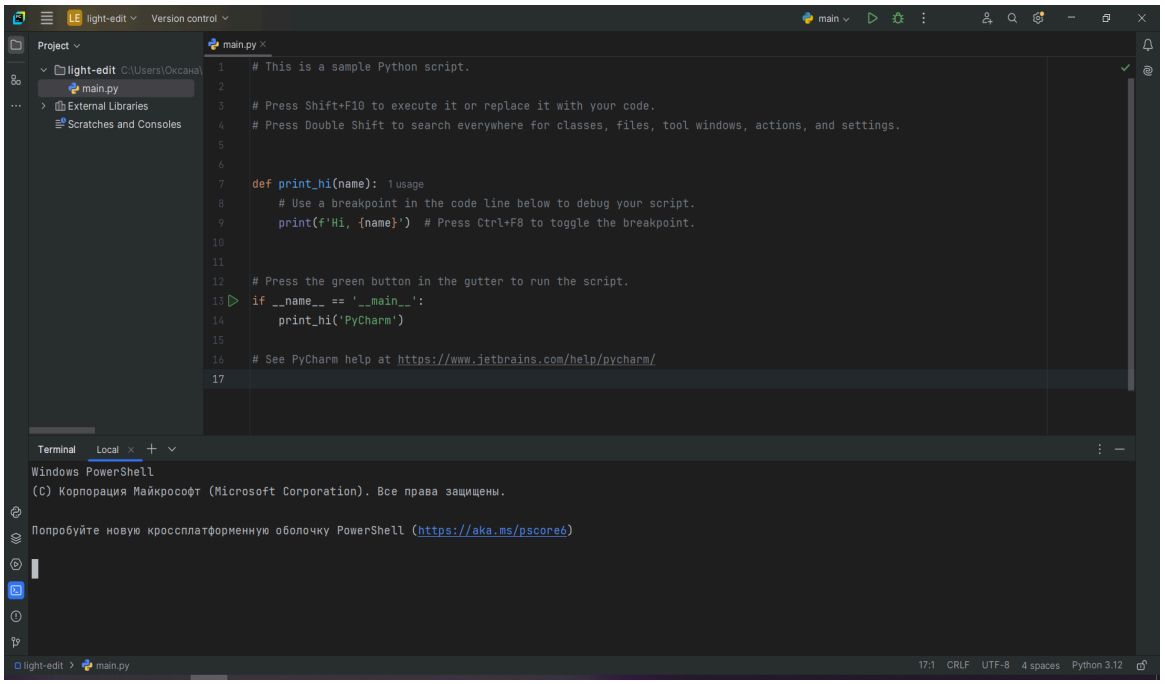

Рисунок 7 - Интерфейс IDE PyCharm

Бот включает в себя использование интерпретатора Python и все необходимые функции для поддержки основных концептов и синтаксиса Python, упражнений для практики и расширенных ресурсов.

Для создания Telegram бота необходимо зарегистрировать его в системе и получить токен для привязки чат-бота [24].

Код разработанный чат-бота состоит из трех основных модулей: Bot.py, Gpt.py и Util.py. Каждый из этих модулей выполняет определенные функции и обеспечивает слаженную работу Telegram бота.

Модуль Bot (Приложение 1) отвечает за взаимодействие с пользователем через Telegram API, обрабатывает входящие сообщения и команды (/start, /lesson, /gpt и др.), а также координирует работу с другими модулями. Для реализации модуля используется библиотека python-telegram-bot.

Основная задача модуля Gpt - генерировать ответы на вопросы пользователя, а именно проверяет корректность отправленных кодов и отсутствии в них ошибок (Приложение 2).

Модуль Util содержит вспомогательные функции и утилиты, необходимые для работы бота. В разработанном чат-боте реализован широкий спектр функциональных возможностей (Приложение 3).

Бот способен отправлять в чат различные типы сообщений, включая HTML-сообщения, фотографии и текстовые сообщения с возможностью добавления интерактивных кнопок. Также система обладает функционалом конвертировать объект пользователя в текстовую строку.

Важной особенностью Telegram бота является возможность отображать команду и главное меню, обеспечивая удобную навигацию для пользователей. Кроме того, бот умеет загружать сообщения и промпты из указанной папки.

Для добавления искусственного интеллекта в Telegram бот необходимо получить API-токен у выбранной ИИ-модели и импортировать все необходимые модули в проекте PyCharm. Далее разрабатывается основная логика обработки сообщений, где будет происходить взаимодействие с ИИ-моделью: сообщения от пользователя отправляются в ИИ-модель и возвращается сгенерированный ответ обратно в чат.

При первом запуске Telegram бота для изучения языка программирования Python открывается главное меню и список команд (рис.8).

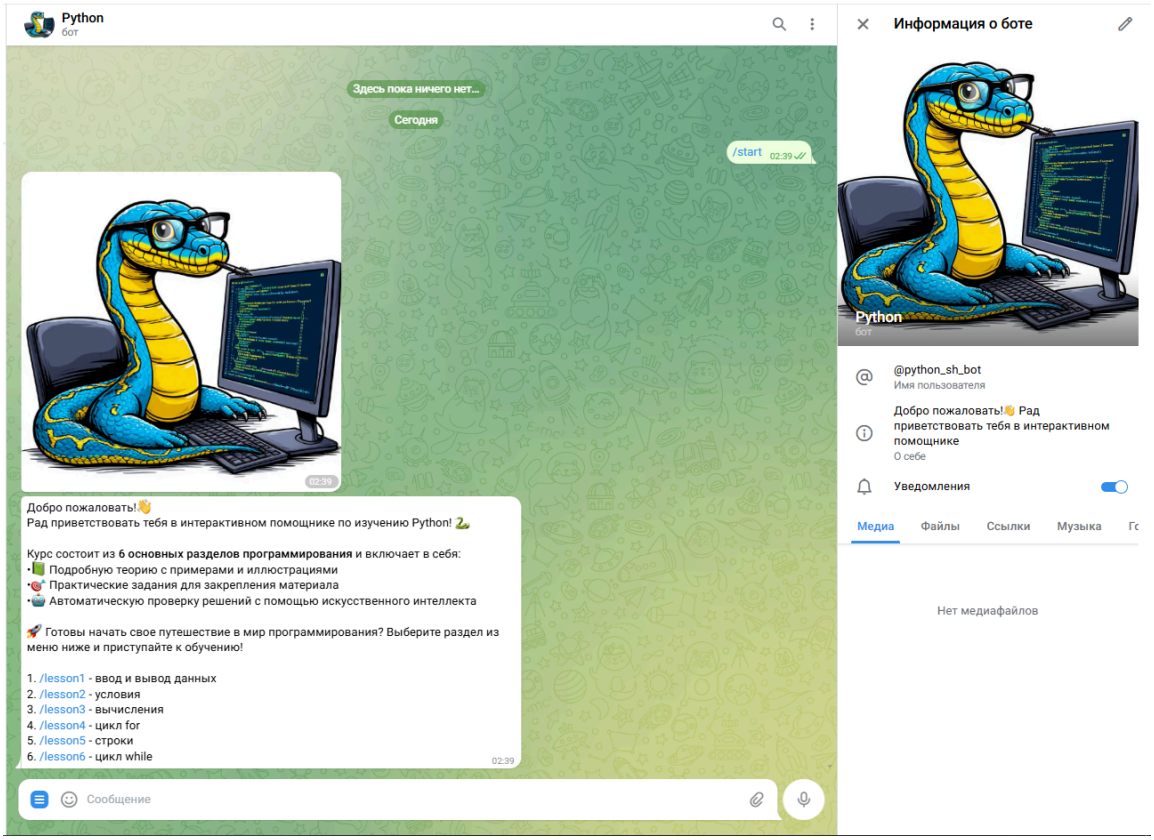

Рисунок 8 - Запуск Telegram бота

Бот состоит из 6 тем для изучения языка программирования Python:

1. Ввод и вывод данных
2. Условия
3. Вычисления
4. Цикл for
5. Строки
6. Цикл while

Каждая тема включает в себя следующие разделы:

- Теорию. Краткое объяснение теоретических основ Python, а также примеры кода, чтобы лучше понять материал.
- Практику. После изучения теории бот предложит задания разной сложности. Эти задания помогут закрепить полученные знания и развить навыки программирования.
- Проверку кода. После выполнения задания необходимо отправить код для проверки в чат-бот. Используя возможности GPT, бот

автоматически проверит код на наличие ошибок и даст обратную связь. (рис.9)

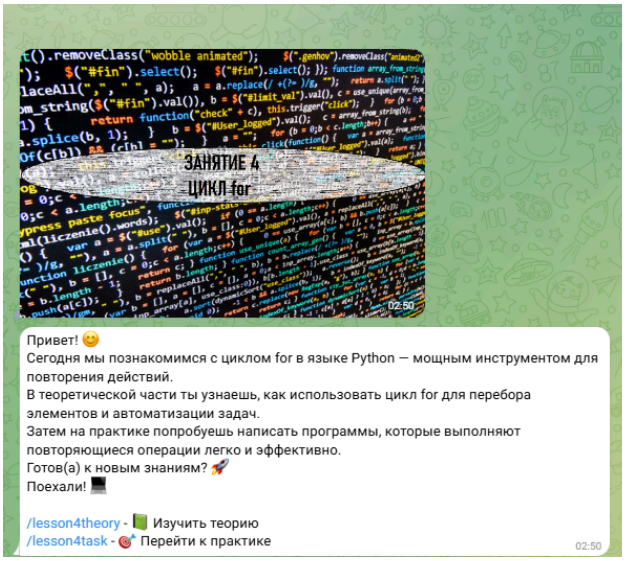

Рисунок 9 - Выбор действий Telegram бота

Пользователи могут также перемещаться между функциями бота через навигацию из списка команд (рис.10).

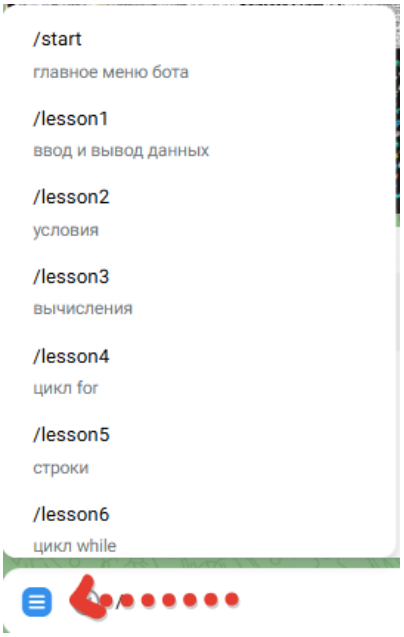

Рисунок 10 - Список команд Telegram бота

Для того, чтобы пользователь мог в любой момент получить доступ к Telegram боту, он был загружен на удаленный сервер ИМФИ КГПУ.

§2.2. Методические рекомендации по интеграции чат-бота в учебный процесс для формирования навыков программирования

Данный Telegram бот для изучения Python можно использовать во время самостоятельной работы обучающихся на уроках информатики в 8–9 классах. При подготовки к занятию педагогу необходимо тщательно продумать, как организовать практическую и самостоятельную деятельность учеников. Это позволит им применить полученные знания на практике и развить необходимые умения.

На уроке открытия нового знания учитель может начать с краткого объяснения новой темы, после чего ученики обращаются к Telegram боту для углубленного изучения материала. После изучения теории ученики переходят к практике: бот предлагает задания разного уровня сложности, которые можно решать прямо в чате или в специализированных средах разработки. После они отправляют свои решения боту, который с помощью GPT анализирует код, находит ошибки, предлагает оптимизации и даёт обратную связь. В процессе анализа ошибок учитель совместно с учениками обсуждает наиболее распространенные ошибки, выявленные чат-ботом, и дает рекомендации по их предотвращению в дальнейшем.

Применение чат-бота не ограничивается работой на уроке. Дома ученики могут продолжать изучать Python с помощью чат-бота, проходя новые темы или закрепляя пройденный материал. Бот выдаёт конспекты, практические задачи и даже проверяет домашние задания, оценивая качество кода и давая рекомендации.

Еще одним вариантом интеграции Telegram бота в процесс обучения является использование технологии «Перевернутый класс». Ученики самостоятельно изучают теоретический материал дома и решают задачи, используя чат-бот. На уроке же больше времени уделяется обсуждению возникших вопросов, практическому применению знаний и углубленному изучению темы под руководством преподавателя.

Чат-бота может стать эффективным инструментом для индивидуальной работы с учениками, которые испытывают сложности в освоении темы. Чтобы помочь ученикам восполнить пробелы в знаниях, учитель может предложить им использовать чат-бота в качестве дополнительного материала для самостоятельного изучения. Это позволит ученикам проработать проблемные вопросы и выполнить индивидуальные задания.

Ученики могут обращаться к чат-боту в любое удобное для них время. Они смогут повторить пройденный материал, закрепить знания и отработать практические навыки.

В процессе самостоятельного решения задач по программированию чат-бот становится персональным помощником ученика, который не только проверяет правильность написанного кода, но и анализирует типичные ошибки.

Для оценки удобства использования разработанного Telegram бота была создана анкета обратной связи (Приложение 4), с целью выявить проблемы и недостатки, улучшения функционала бота, а также определить общее впечатление о продукте.

Апробация проводилась на студентах 1 курса института математики, физики и информатики КГПУ им. В.П. Астафьева в количестве 16 человек.

Анкета состояла из вопросов, позволяющих оценить понятность и качество материала, преимущества и недостатки использования Telegram бота.

В целом, больше половины опрошенных оценило пройденный курс на высшую оценку, лишь 2 человека оценили его на три и менее баллов (Рис.11.1). На основе представленных данных можно сделать вывод, что большинство участников курса программирования остались довольны обучением.

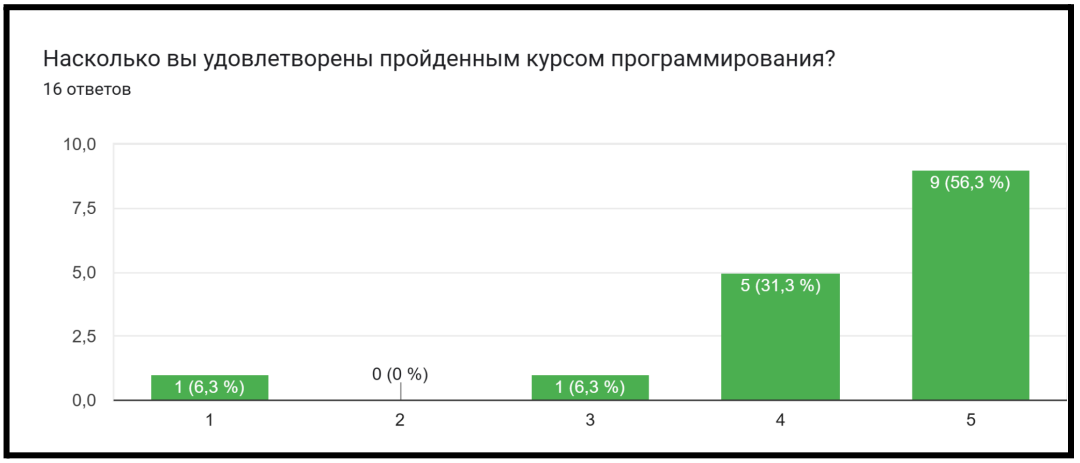

Рисунок 11.1 - Результат опроса «Насколько вы удовлетворены пройденным курсом программирования?»

На основе представленных данных опроса видно, что почти 70% опрошенных (11 человек) частично достигли своих целей в изучении языка программирования (рис. 11.2).

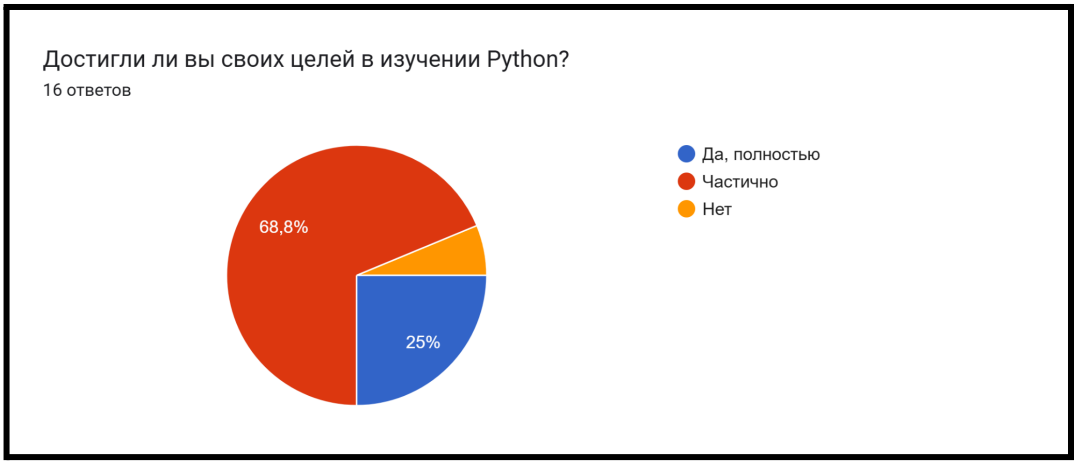

Рисунок 11.2 - Результат опроса «Достигли ли вы своих целей в изучении Python?»

Для большинства участников все темы курса не вызывали особых трудностей, так как они оценили их сложность на «Удовлетворительно». Самыми сложными темами, по мнению опрошенных, оказались «Циклы for и while», а также «Списки» (рис. 11.3). Возможно, эти темы воспринимались, как наиболее проблемные из-за нехватки теоретического материала и чрезвычайно сложности заданий. Данные опроса представлены на рисунках 11.4 и 11.5.

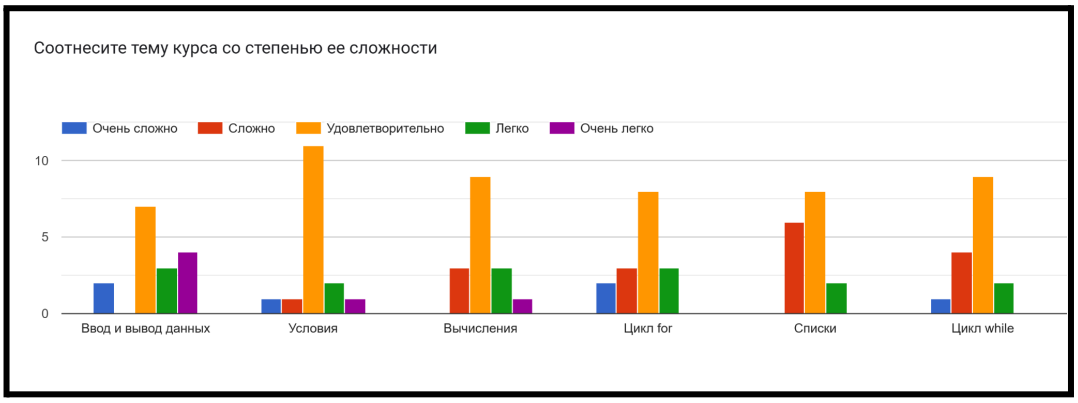

Рисунок 11.3 - Результат опроса «Соотнесите тему курса со степенью ее сложности»

Рисунок 11.4 - Результат опроса «Достаточно ли в было теоретического материала в ТГ-боте?»

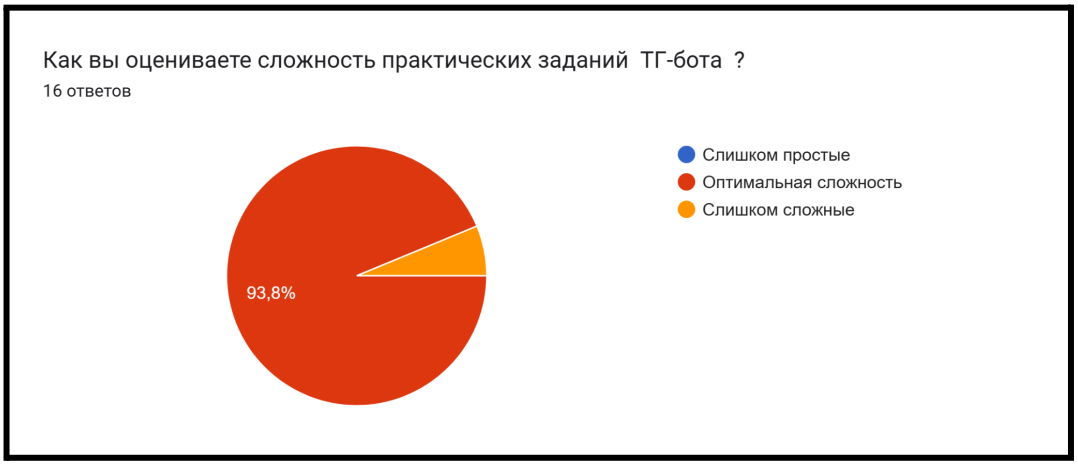

Рисунок 11.5 - Результат опроса «Как вы оцениваете сложность практических заданий ТГ-бота?»

Большинство опрошенных считают, что Telegram бот предоставляет учебные материалы по программированию в удобной форме. При этом 66,7% опрошенных считают, что бот наиболее полезен в качестве полноценного сопровождения в течение всего периода обучения, тогда как 53,3% видят в нём инструмент для подготовки к контрольным. Это свидетельствует о том, что обучающиеся рассматривают бота не как разовое решение, а как часть образовательного процесса. Интересно отметить, что 40% участников опроса видят в боте помощника для самостоятельной работы в течение семестра, что подчеркивает его роль в поддержке непрерывного обучения (рис. 11.6-11.7).

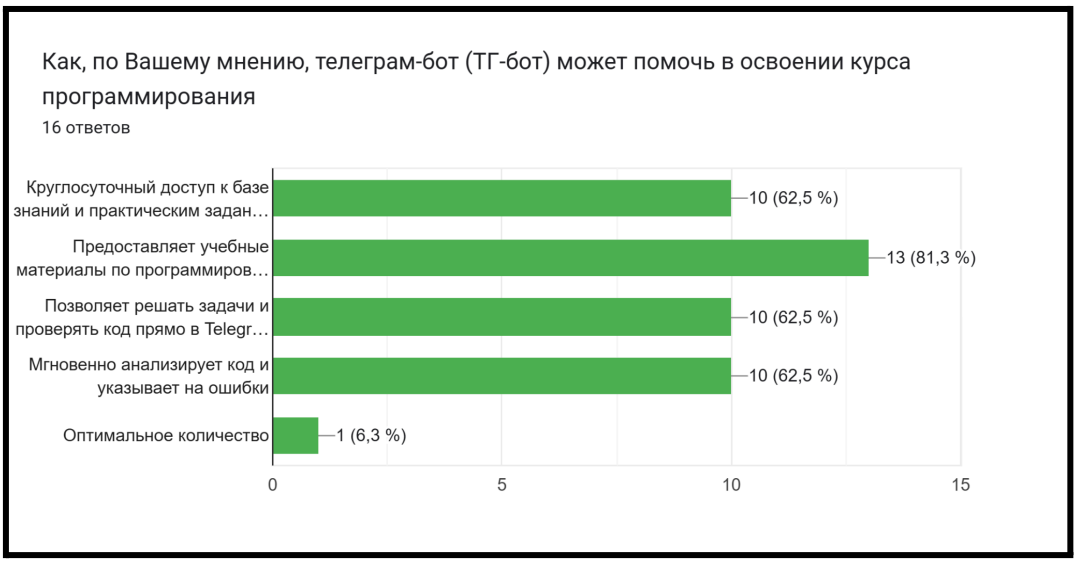

Рисунок 11.6 - Результат опроса «Как, по Вашему мнению, телеграм-бот (ТГ-бот) может помочь в освоении курса программирования?«

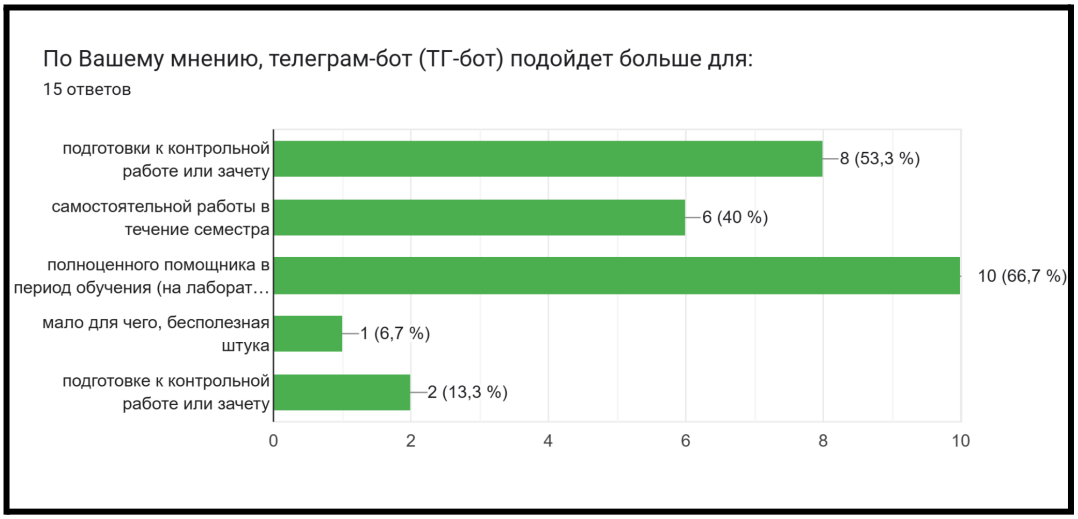

Рисунок 11.7 - Результат опроса «По Вашему мнению, ТГ-бот подойдет больше для:»

На основе анализа трех вопросах о применении GPT в разработанном Telegram боте, показанных на рисунка 11.8-11.9 можно сделать следующие выводы.

Большинство опрошенных оценили полезность автоматической проверки заданий через GPT на высший балл.

Формат обучения через Telegram-бота также получил положительную оценку: 37,5% учащихся нашли его очень удобным, а 31,3% — просто удобным. Треть опрошенных (31,3%) отнеслись к этому нейтрально, но важно отметить, что ни один из респондентов не выбрал варианты «неудобно» или «очень неудобно».

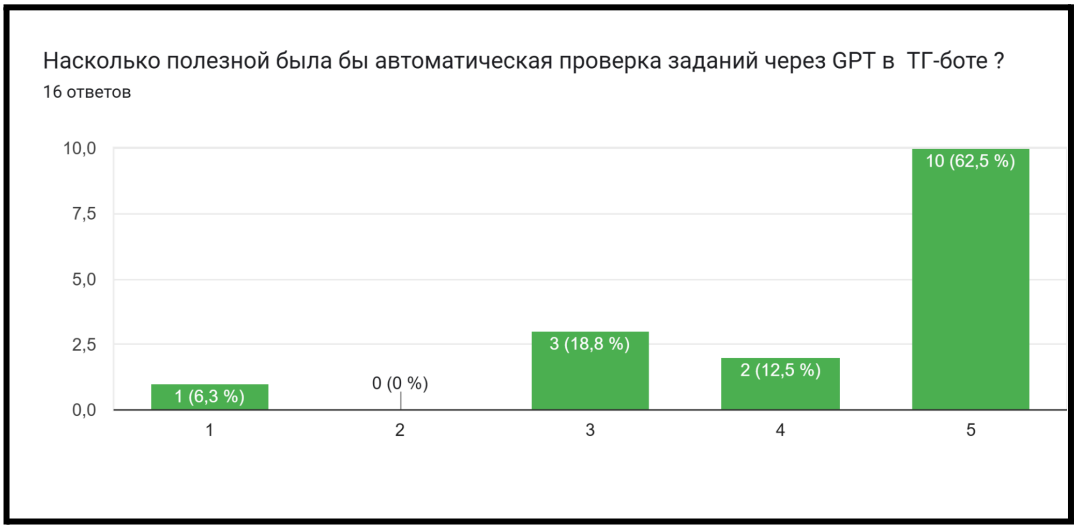

Рисунок 11.8 - Результат опроса «Насколько полезной была бы автоматическая проверка заданий через GPT в ТГ-боте?»

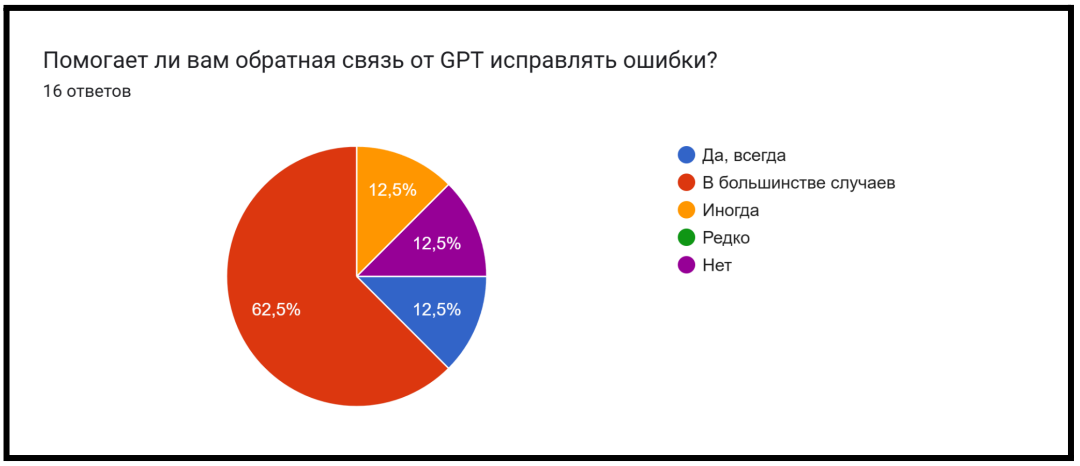

Рисунок 11.9 - Результат опроса «Помогает ли вам обратная связь от GPT исправлять ошибки?»

На основе данных опроса видно (рис.11.10-11.11), что формат обучения через бота высоко ценится учащимися за его доступность и удобство, что подтверждается 81,3% положительных ответов. Половина респондентов отмечает структурированный доступ к материалам и ценность практических заданий с мгновенной обратной связью как важное преимущество, а 43,8% подчеркивают.

Большинство опрошенных положительно оценивают такой формат обучения: 37,5% респондентов считают его очень удобным, а 31,3% — просто удобным.

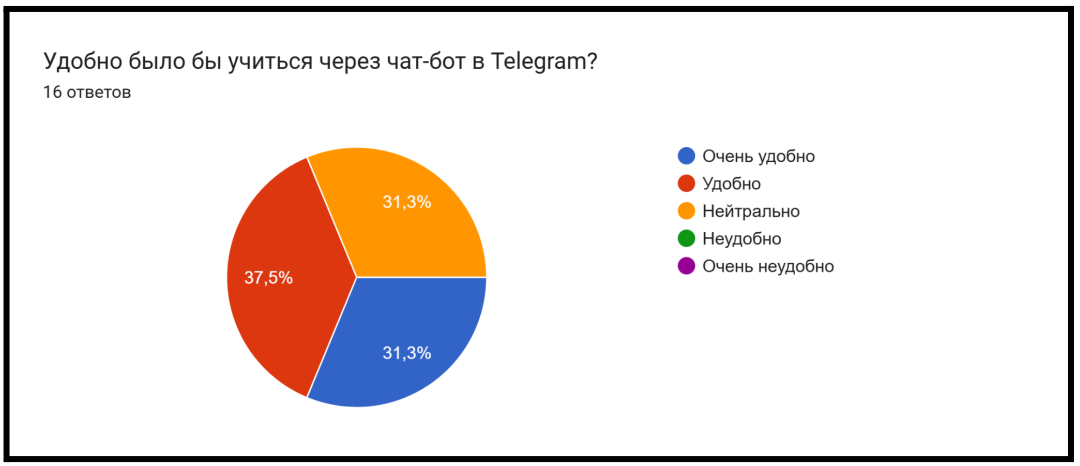

Рисунок 11.10 - Результат опроса «Удобно было бы учиться через чат-бот в Telegram?»

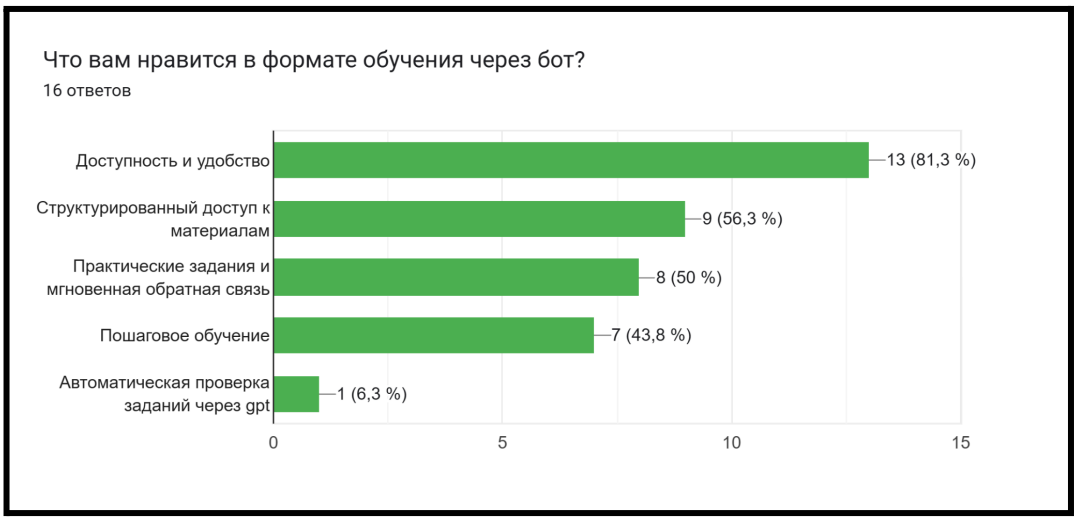

Рисунок 11.11 - Результат опроса «Что вам нравится в формате обучения через бот?»

Вывод по главе:

Во второй главе рассмотрены теоретические аспекты создания чат-бота и этапы его разработки. Выбрана среда разработки PyCharm, и с помощью данного сервиса спроектирован и разработан чат-бот Telegram для изучения языка программирования Python.

Также разработаны методические рекомендации по интеграции Telegram бота в процесс самостоятельной работы обучающихся 8-9 классов для изучения программирования. По результатам анкетирования, можно сделать вывод об удобстве, эффективности и перспективности использования Telegram-бота с интеграцией GPT для обучения программированию. Чат-бот оказался удобным, интерактивным и обеспечил быструю обратную связь.

Заключение

В заключении можно сказать, что цель исследования, которая заключается в разработке чат-бота в Telegram и методических рекомендаций по его использованию для организации самостоятельной работы при обучении программированию в основной школе была достигнута.

Для достижения данной цели были выполнены следующие задачи:

1. Проанализированы научно-педагогические источники и выявлены особенности организации самостоятельной работы обучающихся при обучении программированию.
2. Анализ источников помог определить такие понятия, как «самостоятельная работа», «чат-бот». Выявлены условия повышения результативности самостоятельной работы, а также ее роль, задачи, вид и формы в процессе обучения программированию.
3. Спроектирован и разработан чат-бот Telegram для организации самостоятельной работы при обучении программированию в 8-9 классах. Чат-бот состоит из 6 тем изучения языка программирования Python: «Ввод и вывод данных», «Условия», «Вычисления», «Цикл for», «Строки», «Цикл while».

Каждая тема включает в себя теоретические материалы, практические задания и их проверку. В разделе теории были использованы примеры кода. Проверка заданий реализуется через автоматическую проверку с использованием GPT-технологии.

4. Разработаны методические рекомендации по использованию данного Telegram бота в процессе обучения программированию в основной школе. Описаны способы использования чат-бота для активизации самостоятельной деятельности обучающихся как на уроках информатики в школе, так и для внеурочной деятельности.

С целью оценки удобства Telegram бота для самостоятельного изучения языка программирования Python была проведена его апробация и опрос студентов 1 курса ИМФИ КГПУ. Оценка разработанного чат-бота и

проанализированные результаты, которые в большей степени являются положительными и доказывают возможность использования Telegram бота в процессе активизации самостоятельной деятельности обучающихся при изучении языка программирования Python.

Полученные результаты исследования могут быть использованы в рамках очного и дистанционного обучения основам программирования на языке Python. Чат-бот также может использоваться в вузе при обучении основам программирования на языке Python.

Библиографический список

1. Абрамов Д. А., Гаев Л. В. Роль чат-ботов с искусственным интеллектом в современной индустрии программирования // Международный журнал гуманитарных и естественных наук. 2024. № 5-1 (92). URL: https://cyberleninka.ru/article/n/rol-chat-botov-s-iskusstvennym-intellektom-v-sovremennoy-industrii-programmirovaniya (дата обращения: 12.04.2025).

2. Завьялова О. А., Маркелов В. К. Преподавание программирования в школе как барьер в профессиональном выборе будущего учителя информатики // Научный поиск: личность, образование, культура. 2022. № 2. С. 44.

3. Интерактивный учебник языка Питон // Питонтьютор. URL: https://pythontutor.ru/ (дата обращения: 04.04.2025).

4. Кадеева О. Е., Сырицына В. Н. Чат-боты и особенности их использования в образовании // Информатика в школе. 2020. № 10. С. 45–53.

5. Кондратьева В. А. Обучение основам программирования на языке Python в школьном курсе информатики // Вестник МГПУ. Серия «Информатика и информатизация образования». 2021. № 1 (55). URL: https://cyberleninka.ru/article/n/obuchenie-osnovam-programmirovaniya-na-yazyke-python-v-shkolnom-kurse-informatiki (дата обращения: 10.04.2025).

6. Крапивко А. С. Использование чат-ботов в рамках организации информационного образовательного пространства // От цифровизации к цифровой трансформации: Материалы VI Междунар. науч.-практ. конф. Челябинск: Челябинский институт развития проф. образования, 2022. С. 160–163. URL: https://elibrary.ru/item.asp?id=48701024 (дата обращения: 03.04.2025).

7. Кулевская Е. С., Воронина М. В. Методика применения интерактивных заданий на уроках информатики в рамках самостоятельной работы учащихся // Вопросы педагогики. 2019. № 12-2. С. 183–186. URL: https://elibrary.ru/item.asp?id=41671289 (дата обращения: 17.04.2025).

8. Куликова Н. Ю., Данильчук Е. В., Малова А. И. Обучение информатике в образовательных онлайн-сообществах школьников с использованием чат-ботов // Известия ВГПУ. 2022. № 9 (172). URL: https://cyberleninka.ru/article/n/obuchenie-informatike-v-obrazovatelnyh-onlayn-soobschestvah-shkolnikov-s-ispolzovaniem-chat-botov (дата обращения: 10.04.2025).

9. Куликова Н. Ю., Катруш Г. В. Особенности организации учебной деятельности в онлайн-сообществе учащихся школ при обучении

программированию с использованием визуальных сред и конструкторов роботов // Известия ВГПУ. 2023. № 3 (176). URL: https://cyberleninka.ru/article/n/osobennosti-organizatsii-uchebnoy-deyatelnosti-v-onlayn-soobschestve-uchaschihsya-shkol-pri-obuchenii-programmirovaniyu-s (дата обращения: 03.04.2025).

10. Михлюк А. А., Кунакова Р. С. Использование нейросетей и чат-ботов в современном программировании // Вестник науки. 2024. № 1 (70). URL: https://cyberleninka.ru/article/n/ispolzovanie-neyrosetey-i-chat-botov-v-sovremennom-programmirovanii (дата обращения: 12.04.2025).

11. Мнацаканян В. В., Малофеев В. А., Чеботарева Е. Р. Разработка функционального чат-бота как способ обучения программированию школьников // Наука. Управление. Образование. РФ. 2023. № 2 (10). С. 60–64. URL: https://elibrary.ru/item.asp?id=53935082 (дата обращения: 12.04.2025).

12. Могилевская Н. С., Самойленко Г. П. Чат-бот для тестирования знаний на платформе мессенджера Telegram // Молодой исследователь Дона. 2022. № 3 (36). URL: https://cyberleninka.ru/article/n/chat-bot-dlya-testirovaniya-znaniy-na-platforme-messendzhera-telegram (дата обращения: 12.04.2025).

13. Осадчук О. Л. Роль учебных материалов в активизации самостоятельной работы обучающихся // Начальная школа плюс До и После. 2009. № 12. С. 58–61.

14. Пидкасистый П. И. Самостоятельная познавательная деятельность школьников в обучении. М., 1980. 146 с.

15. Родыгин Е. Ф. Методические рекомендации обучения программированию в школе // Вестник Марийского государственного университета. 2011. № 7. С. 20–22.

16. Российская педагогическая энциклопедия: в 2 т. / гл. ред. В. В. Давыдов. М.: Большая Российская энциклопедия, 1993. Т. 1. 608 с.

17. Рубинштейн С. Л. Принцип творческой самодеятельности // Вопросы философии. 1989. № 4. С. 89–95.

18. Самостоятельная учебная деятельность школьника: методическое пособие / под ред. И. В. Усковой. М.: Институт стратегии развития образования, 2023. 180 с.

19. Самылкина Н. Н., Сидорова А. И. Создание чат-ботов в Telegram как практико-ориентированное задание по программированию в углубленном

курсе информатики // Актуальные проблемы обучения математике в школе и вузе: Материалы VII Междунар. науч.-практ. конф. М., 2022. С. 666–679. URL: https://elibrary.ru/download/elibrary_52691041_61091592.pdf (дата обращения: 12.04.2025).

20. Смельчаков А. В. Преимущества изучения в образовательных учреждениях языка программирования Python // Современные тенденции в науке и образовании: новый взгляд. 2020. С. 8–12.

21. Сомова Н. Ю. Роль программирования в современном образовании // Ученый, педагог, наставник: материалы Всерос. науч.-практ. конф. Тула, 2023. С. 118–121.

22. Софронова Н. В. Теория и методика обучения информатике: учебник для СПО / Н. В. Софронова, А. А. Бельчусов. 3-е изд., перераб. и доп. М.: Юрайт, 2025. 469 с.

23. Суслова А. В. К вопросу о сущности понятия «самостоятельная работа» в образовательном процессе // Педагогические науки. 2021. С. 249.

24. Telegram Bot API. URL: https://core.telegram.org/bots/api (дата обращения: 12.10.2024).

25. Усова А. В. Самостоятельная работа учащихся по физике в средней школе. М., 1981. 5 с.

26. Федеральная рабочая программа основного общего образования. Информатика. Базовый уровень (для 7–9 классов). М., 2023. URL: https://edsoo.ru/wp-content/uploads/2023/08/15_ФРП-Информатика-7-9-классы_база.pdf (дата обращения: 10.05.2025).

27. Федеральная рабочая программа основного общего образования. Информатика. Базовый уровень (для 10–11 классов). М., 2023. URL: https://edsoo.ru/wp-content/uploads/2023/08/21_ФРП-Информатика_10-11-классы_база.pdf (дата обращения: 10.05.2025).

28. Федеральная рабочая программа основного общего образования. Информатика. Углубленный уровень (для 7–9 классов). М., 2023. URL: https://edsoo.ru/wp-content/uploads/2023/08/16_ФРП_Информатика_7-9-классы_угл.pdf (дата обращения: 10.05.2025).

29. Федеральная рабочая программа основного общего образования. Информатика. Углубленный уровень (для 10–11 классов). М., 2023. URL: https://edsoo.ru/wp-content/uploads/2023/08/22_ФРП_Информатика-10-11-классы_угл.pdf (дата обращения: 10.05.2025).

30. Фоминых И. А., Довранов А. Р. О разработке курса по выбору «Основы программирования на Python» для предпрофильной подготовки школьников // Информатика в школе. 2022. № 1. С. 22–29.

31. Цукерман Г. А., Венгер А. Л. Развитие учебной самостоятельности средствами школьного образования // Психологическая наука и образование. 2010. Т. 15. № 4. С. 77–90.

32. Шныпко В. С., Дьяченко Н. В. Содержание и формы самостоятельной подготовки обучающихся в условиях дистанционного обучения // Культура и безопасность. 2021. № 1. С. 57–61.

33. Щербакова Е. В., Щербакова Т. Н. Историческое развитие самостоятельной работы как вида деятельности школьников // Colloquium-journal. 2019. № 6 (30). С. 58–63.

Приложение 1

Модуль Bot

bot.py × gpt.py util.py

```python
from telegram.ext import ApplicationBuilder, MessageHandler, f

from gpt import *
from util import *

import logging
logging.basicConfig(level=logging.INFO)

async def start(update, context):  1 usage
    dialog.mode = "main"
    text = load_message("main")
    await send_photo(update, context,  name: "main")
    await send_text(update, context, text)

    await show_main_menu(update, context,  commands: {
        "start": "главное меню бота",
        "lesson1": "ввод и вывод данных",
        "lesson2": "условия",
        "lesson3": "вычисления",
        "lesson4": "цикл for",
        "lesson5": "строки",
        "lesson6": "цикл while",
        "check": "проверка заданий"
    })

async def lesson1(update, context):  1 usage
    await send_photo(update, context,  name: "pic.1")
    text = load_message("text1")
    await send_text(update, context, text)

async def lesson1_theory(update, context):  1 usage
    await send_photo(update, context,  name: "picture 1.1")
    await send_photo(update, context,  name: "picture 1.2")
    await send_photo(update, context,  name: "picture 1.3")
    await send_photo(update, context,  name: "picture 1.4")
    await send_photo(update, context,  name: "picture 1.5")
    text = load_message("lesson1")
    await send_text(update, context, text)

async def lesson1_task(update, context):  1 usage
    text1 = load_message("task1")
    await send_text(update, context, text1)


async def lesson1_dialog(update, context):
    text = update.message.text
    dialog.list.append(text)

```

```python
async def lesson2(update, context):  1 usage
    await send_photo(update, context, name: "pic.2")
    text = load_message("text2")
    await send_text(update, context, text)

async def lesson2_theory(update, context):  1 usage
    text = load_message("lesson2")
    text1 = load_message("lesson2.1")
    await send_photo(update, context, name: "picture 2.1")
    await send_photo(update, context, name: "picture 2.2")
    await send_photo(update, context, name: "picture 2.3")
    await send_photo(update, context, name: "picture 2.4")
    await send_photo(update, context, name: "picture 2.5")
    await send_text(update, context, text)
    await send_text(update, context, text1)

async def lesson2_task(update, context):  1 usage
    text1 = load_message("task2")
    await send_text(update, context, text1)

async def lesson2_dialog(update, context):
    text = update.message.text
    dialog.list.append(text)

async def lesson3(update, context):  1 usage
    await send_photo(update, context, name: "pic.3")
    text = load_message("text3")
    await send_text(update, context, text)

async def lesson3_theory(update, context):  1 usage
    text = load_message("lesson3")
    text1 = load_message("lesson3.1")
    await send_text(update, context, text)
    await send_text(update, context, text1)
    await send_photo(update, context, name: "picture 3.1")
    await send_photo(update, context, name: "picture 3.2")
    await send_photo(update, context, name: "picture 3.3")
    await send_photo(update, context, name: "picture 3.4")
    await send_photo(update, context, name: "picture 3.5")


async def lesson3_task(update, context):  1 usage
    text1 = load_message("task3")
    await send_text(update, context, text1)

async def lesson3_dialog(update, context):
    text = update.message.text
    dialog.list.append(text)
```

```python

async def lesson4(update, context):  1 usage
    await send_photo(update, context, name: "pic.4")
    text = load_message("text4")
    await send_text(update, context, text)

async def lesson4_theory(update, context):  1 usage
    text = load_message("lesson4")
    text1 = load_message("lesson4.1")
    await send_photo(update, context, name: "picture 4.1")
    await send_photo(update, context, name: "picture 4.2")
    await send_photo(update, context, name: "picture 4.3")
    await send_photo(update, context, name: "picture 4.4")
    await send_photo(update, context, name: "picture 4.5")
    await send_text(update, context, text)
    await send_text(update, context, text1)

async def lesson4_task(update, context):  1 usage
    text1 = load_message("task4")
    await send_text(update, context, text1)

async def lesson4_dialog(update, context):
    text = update.message.text
    dialog.list.append(text)

async def lesson5(update, context):  1 usage
    await send_photo(update, context, name: "pic.5")
    text = load_message("text5")
    await send_text(update, context, text)

async def lesson5_theory(update, context):  1 usage
    text = load_message("lesson5")
    text1 = load_message("lesson5.1")
    text2 = load_message("lesson5.2")
    await send_photo(update, context, name: "picture 5.1")
    await send_photo(update, context, name: "picture 5.2")
    await send_photo(update, context, name: "picture 5.3")
    await send_photo(update, context, name: "picture 5.4")
    await send_photo(update, context, name: "picture 5.5")
    await send_photo(update, context, name: "picture 5.6")
    await send_photo(update, context, name: "picture 5.7")
    await send_text(update, context, text)
    await send_text(update, context, text1)
    await send_text(update, context, text2)

async def lesson5_task(update, context):  1 usage
    text1 = load_message("task5")
    await send_text(update, context, text1)
```

```python

async def lesson5_dialog(update, context):
    text = update.message.text
    dialog.list.append(text)

async def lesson6(update, context):  1 usage
    await send_photo(update, context,  name: "pic.6")
    text = load_message("text6")
    await send_text(update, context, text)

async def lesson6_theory(update, context):  1 usage
    text = load_message("lesson6")
    text1 = load_message("lesson6.1")
    await send_photo(update, context,  name: "picture 6.1")
    await send_photo(update, context,  name: "picture 6.2")
    await send_photo(update, context,  name: "picture 6.3")
    await send_photo(update, context,  name: "picture 6.4")
    await send_photo(update, context,  name: "picture 6.5")
    await send_photo(update, context,  name: "picture 6.6")
    await send_photo(update, context,  name: "picture 6.7")
    await send_photo(update, context,  name: "picture 6.8")
    await send_text(update, context, text)
    await send_text(update, context, text1)

async def lesson6_task(update, context):  1 usage
    text1 = load_message("task6")
    await send_text(update, context, text1)

async def lesson6_dialog(update, context):
    text = update.message.text
    dialog.list.append(text)

async def check(update, context):  2 usages
    dialog.mode = "check"
    text = load_message("check")
    await send_photo(update, context,  name: "check")
    await send_text(update, context, text)

async def check_dialog(update, context):
    my_message = await send_text(update, context,  text: "ChatGPT дума
    prompt = load_prompt("check")
    text = update.message.text
    answer = await chatgpt.send_question(prompt, text)
    await my_message.edit_text(answer)


dialog = Dialog()
dialog.mode = None
dialog.list = []
dialog.count = 0
dialog.user = {}

```

```
chatgpt = DeepSeekService(api_key="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")


app = ApplicationBuilder().token("8120968718:AAHQNnzO6Me9W9D9QOx0aBiIb8CtTwzrTfE").build()
app.add_handler(CommandHandler( command: "start", start))
app.add_handler(CommandHandler( command: "lesson1", lesson1))
app.add_handler(CommandHandler( command: "lesson1theory", lesson1_theory))
app.add_handler(CommandHandler( command: "lesson1task", lesson1_task))
app.add_handler(CommandHandler( command: "lesson2", lesson2))
app.add_handler(CommandHandler( command: "lesson2theory", lesson2_theory))
app.add_handler(CommandHandler( command: "lesson2task", lesson2_task))
app.add_handler(CommandHandler( command: "lesson3", lesson3))
app.add_handler(CommandHandler( command: "lesson3theory", lesson3_theory))
app.add_handler(CommandHandler( command: "lesson3task", lesson3_task))

app.add_handler(CommandHandler( command: "lesson4", lesson4))
app.add_handler(CommandHandler( command: "lesson4theory", lesson4_theory))
app.add_handler(CommandHandler( command: "lesson4task", lesson4_task))
app.add_handler(CommandHandler( command: "lesson5", lesson5))
app.add_handler(CommandHandler( command: "lesson5theory", lesson5_theory))
app.add_handler(CommandHandler( command: "lesson5task", lesson5_task))
app.add_handler(CommandHandler( command: "lesson6", lesson6))
app.add_handler(CommandHandler( command: "lesson6theory", lesson6_theory))
app.add_handler(CommandHandler( command: "lesson6task", lesson6_task))
app.add_handler(CommandHandler( command: "check", check))

app.add_handler(MessageHandler(filters.TEXT & ~filters.COMMAND, check))

app.run_polling()
```

Приложение 2

Модуль Gpt

bot.py gpt.py util.py

```python
import json
import requests
import aiohttp

API_KEY = "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"
MODEL = "deepseek/deepseek-r1"


class DeepSeekService:  2 usages
    def __init__(self, api_key: str, model: str = MODEL):
        self.API_KEY = api_key
        self.MODEL = model
        self.message_list = []

    def load_prompt(prompt_name: str) -> str:
        prompts = {
            "check": "Ты — помощник для проверки кода. Анализируй код и давай чёткие исправления.",
            "gpt": "Ты — полезный ИИ-ассистент. Отвечай на вопросы развёрнуто."
        }
        return prompts.get(prompt_name, "")

    def set_prompt(self, prompt_text: str) -> None:
        self.message_list.clear()
        self.message_list.append({"role": "system", "content": prompt_text})

    async def add_message(self, message_text: str) -> None:
        self.message_list.append({"role": "user", "content": message_text})
        return await self.send_message_list()

    async def send_question(self, prompt_text: str, message_text: str) -> None:  1 usage
        self.message_list.clear()
        self.message_list.append({"role": "system", "content": prompt_text})
        self.message_list.append({"role": "user", "content": message_text})
        return await self.send_message_list()

    async def send_message_list(self):  6 usages (4 dynamic)
        headers = {
            "Authorization": f"Bearer {self.API_KEY}",
            "Content-Type": "application/json"
        }

        data = {
            "model": self.MODEL,
            "messages": self.message_list
        }
```

```
46
47          async with aiohttp.ClientSession() as session:
48              async with session.post(
49                      "https://openrouter.ai/api/v1/chat/completions",
50                      headers=headers,
51                      json=data
52              ) as response:
53                  if response.status != 200:
54                      return "Ошибка API"
55
56                  result = await response.json()
57                  return result["choices"][0]["message"]["content"]
58
59      def process_content(self, content: str) -> str:  1 usage
60          return content.replace( _old: '<think>', _new: '').replace( _old: '</think>', _new: '')
61
62      def chat_stream(self, prompt: str) -> str:  1 usage
63          headers = {
64              "Authorization": f"Bearer {self.API_KEY}",
65              "Content-Type": "application/json"
66          }
67
68          data = {
69              "model": self.MODEL,
70              "messages": [{"role": "user", "content": prompt}],
71              "stream": True
72          }
73
74          with requests.post(
75                  url: "https://openrouter.ai/api/v1/chat/completions",
76              headers=headers,
77              json=data,
78              stream=True
79          ) as response:
80              if response.status_code != 200:
81                  print("Ошибка API:", response.status_code)
82                  return ""
83
84              full_response = []
85              for chunk in response.iter_lines():
86                  if chunk:
87                      chunk_str = chunk.decode('utf-8').replace('data: ', '')
88                      try:
89                          chunk_json = json.loads(chunk_str)
90                          if "choices" in chunk_json:
91                              content = chunk_json["choices"][0]["delta"].get("content", "")
92                              if content:
```

```
                                cleaned = self.process_content(content)
                                print(cleaned, end='', flush=True)
                                full_response.append(cleaned)
                        except:
                            pass

            print()  # Перенос строки
            return ''.join(full_response)

def main():  1 usage
    service = DeepSeekService(API_KEY)
    print("Чат с DeepSeek-R1 (by Antric)\nДля выхода введите 'exit'\n")

    while True:
        user_input = input("Вы: ")

        if user_input.lower() == 'exit':
            print("Завершение работы...")
            break

        print("DeepSeek-R1:", end=' ', flush=True)
        service.chat_stream(user_input)

if __name__ == "__main__":
    main()
```

Приложение 3

Модуль Util

bot.py gpt.py util.py

```python
from telegram import InlineKeyboardButton, InlineKeyboardMarkup, Message, BotCommand, MenuButtonCommands, BotCommandScope
from telegram import Update
from telegram.constants import ParseMode
from telegram.ext import ContextTypes

# конвертирует объект user в строку
def dialog_user_info_to_str(user) -> str:
    result = ""
    for key, name in map.items():
        if key in user:
            result += name + ": " + user[key] + "\n"
    return result

# посылает в чат текстовое сообщение
async def send_text(update: Update, context: ContextTypes.DEFAULT_TYPE, text: str) -> Message:  27 usages
    if text.count('_') % 2 != 0:
        message = f"Строка '{text}' является невалидной с точки зрения markdown. Воспользуйтесь методом send_html()"
        print(message)
        return await update.message.reply_text(message)

    text = text.encode( encoding: 'utf16', errors='surrogatepass').decode('utf16')
    return await context.bot.send_message(chat_id=update.effective_chat.id, text=text, parse_mode=ParseMode.MARKDOWN)

# посылает в чат html сообщение
async def send_html(update: Update, context: ContextTypes.DEFAULT_TYPE, text: str) -> Message:
    text = text.encode( encoding: 'utf16', errors='surrogatepass').decode('utf16')
    return await context.bot.send_message(chat_id=update.effective_chat.id, text=text, parse_mode=ParseMode.HTML)

# посылает в чат текстовое сообщение, и добавляет к нему кнопки
async def send_text_buttons(update: Update, context: ContextTypes.DEFAULT_TYPE, text: str, buttons: dict) -> Message:
    text = text.encode( encoding: 'utf16', errors='surrogatepass').decode('utf16')
    keyboard = []
    for key, value in buttons.items():
        button = InlineKeyboardButton(str(value), callback_data=str(key))
        keyboard.append([button])
    reply_markup = InlineKeyboardMarkup(keyboard)
    return await update.message.reply_text(text, reply_markup=reply_markup, parse_mode=ParseMode.MARKDOWN)

# посылает в чат фото
async def send_photo(update: Update, context: ContextTypes.DEFAULT_TYPE, name: str) -> Message:  45 usages (2 dynamic)
    with open('resources/images/' + name + ".jpg", 'rb') as photo:
        return await context.bot.send_photo(chat_id=update.effective_chat.id, photo=photo)

# отображает команду и главное меню
async def show_main_menu(update: Update, context: ContextTypes.DEFAULT_TYPE, commands: dict):  1 usage
    command_list = [BotCommand(key, value) for key, value in commands.items()]
    await context.bot.set_my_commands(command_list, scope=BotCommandScopeChat(chat_id=update.effective_chat.id))
    await context.bot.set_chat_menu_button(menu_button=MenuButtonCommands(), chat_id=update.effective_chat.id)

# Удаляем команды для конкретного чата
async def hide_main_menu(update: Update, context: ContextTypes.DEFAULT_TYPE):
    await context.bot.delete_my_commands(scope=BotCommandScopeChat(chat_id=update.effective_chat.id))
    await context.bot.set_chat_menu_button(menu_button=MenuButtonDefault(), chat_id=update.effective_chat.id)

# загружает сообщение из папки  /resources/messages/
def load_message(name):  26 usages
    with open("resources/messages/" + name + ".txt", "r", encoding="utf8") as file:
        return file.read()
```

```
59
60  # загружает промпт из папки  /resources/messages/
61  def load_prompt(name):  1 usage
62      with open("resources/prompts/" + name + ".txt", "r", encoding="utf8") as file:
63          return file.read()
64
65  class Dialog:  1 usage
66      pass
```

Приложение 4

Анкета обратной связи по ТГ-боту для обучения основам Python: