

Министерство просвещения Российской Федерации
федеральное государственное бюджетное
образовательное учреждение высшего образования
«Уральский государственный педагогический университет»
Институт математики, физики, информатики
Кафедра информатики, информационных технологий и
методики обучения информатике

ОБУЧЕНИЕ ШКОЛЬНИКОВ ПРОГРАММИРОВАНИЮ НА ЯЗЫКЕ PYTHON В КОМПЬЮТЕРНЫХ ИГРАХ

Выпускная квалификационная работа

Направление «44.03.05 – Педагогическое образование»
Профиль «Математика и информатика»

Работа допущена к
защите:

Дата	Подпись
Оценка	

Исполнитель:

Панов Н. Е.,
студент группы МИ-1931

Научный руководитель:

Газейкина А. И.,
кандидат педагогических
наук, доцент кафедры
информатики,
информационных технологий
и методики обучения
информатике

Содержание

ВВЕДЕНИЕ	3
Глава 1. Теоретические основы обучения программированию в компьютерных играх	6
1.1. Анализ опыта обучения программированию в компьютерных играх	6
1.2. Программирование на языке Python в компьютерных играх	13
1.3. Лабораторные работы как средство обучения программированию	18
Глава 2. Методика обучения программированию на языке Python в компьютерных играх	23
2.1. Цели и содержание обучения программированию на языке Python в компьютерной игре Minecraft	23
2.2. Комплекс лабораторных работ для обучения программированию на языке Python в компьютерной игре Minecraft	24
2.2. Апробация разработанных материалов методом экспертных оценок.	44
ЗАКЛЮЧЕНИЕ	48
ЛИТЕРАТУРА	49
ПРИЛОЖЕНИЯ	54

ВВЕДЕНИЕ

Одна из проблемных тенденций в современном образовании – снижение мотивации школьников к обучению. Поэтому актуальной задачей для учителя становится поиск новых методик обучения и образовательных технологий, способствующих решению указанной проблемы

С учетом требований Федерального государственного образовательного стандарта [22] третьего поколения, актуальность освоения предметных результатов обучения по информатике приобретает особое значение. Развитие алгоритмического мышления становится неотъемлемой составляющей профессиональной компетентности в современном мире. Понимание сущности алгоритма и его свойств является ключевым фундаментом для способности создавать и отлаживать программы на различных языках программирования. Овладение навыками написания программ, реализующих простейшие алгоритмы обработки числовых данных, при помощи циклов и ветвлений, а также умение разбивать задачи на подзадачи — важный этап формирования компетентности в области информационных технологий. Использование констант, переменных и выражений различных типов (числовых, логических, символьных) позволяет учащимся успешно анализировать предложенные алгоритмы и определять возможные результаты при различных вариантах входных данных. Формирование этих предметных результатов не только развивает умение программировать на конкретных языках, таких как Python, C++, Паскаль, Java, C#, но и способствует формированию критического мышления, необходимого для успешного решения задач в различных сферах деятельности. Актуализация данных компетенций сегодня не просто важна — она становится обязательной составляющей подготовки к успешной профессиональной реализации в условиях быстро

меняющегося информационного общества.

Python – это универсальный язык программирования, созданный нидерландским программистом Гвидо Ван Россумом в 1992г., с помощью которого можно создавать любые приложения в диапазоне от интернет-сайтов до роботов и системных сервисов. Отличается он простотой синтаксиса и быстрым написанием кода, быстро осваивается новичками в области программирования. Питер Норвиг, директор по исследованиям в корпорации Google говорит о Python следующее: «Python был важной частью Google с самого начала, и останется таким до тех пор, пока система растет и развивается. На сегодняшний день десятки инженеров Google используют Python, и мы ищем большое количество людей с навыками в работе на этом языке».

Опираясь на данные положения, можно сделать вывод о том, что Python является на сегодняшний день достаточно востребованным языком программирования, который подходит для обучения основам программирования школьников.

В настоящее время компьютерные развлечения приобретают всю большую популярность. В условиях цифровой трансформации каждой сферы жизни человека перед современным педагогом встаёт задача не просто внедрить в образовательный процесс игровые техники, а использовать в обучении компьютерные игры. В настоящее время существуют сотни компьютерных игр и приложений, предназначенных для использования в образовательном процессе и позволяющих повысить мотивацию школьников к обучению.

Все вышеперечисленное обосновывает актуальность данной выпускной квалифицированной работы.

Объект исследования: процесс обучения программированию учащихся основной школы.

Предмет исследования: обучение учащихся основной школы программированию на языке Python.

Цель исследования: разработать комплекс лабораторных работ для обучения школьников программированию на языке Python в компьютерной игре Minecraft.

На основе цели исследования были сформулированы следующие **задачи:**

1. Проанализировать опыт применения компьютерных игр в образовательном процессе для обучения школьников программированию.

2. Обосновать возможность и целесообразность внедрения обучения программированию на языке Python в компьютерных играх в процессе обучения информатике в средней школе.

3. Проанализировать опыт использования лабораторных работ при обучении школьников программированию.

4. Разработать комплекс лабораторных работ, посвященных обучению школьников программированию на языке Python в компьютерной игре Minecraft.

5. Сформулировать методические рекомендации по применению разработанных лабораторных работ в учебном процессе, провести апробацию разработанных материалов.

Методы исследования: анализ психолого-педагогической и научно-методической литературы, сравнение и обобщение полученной информации.

Глава 1. Теоретические основы обучения программированию в компьютерных играх

1.1. АНАЛИЗ ОПЫТА ОБУЧЕНИЯ ПРОГРАММИРОВАНИЮ В КОМПЬЮТЕРНЫХ ИГРАХ

В наше время образование находится в процессе активной модернизации. Этот процесс предъявляет к учителям и образовательным учреждениям новые требования, ставит новые задачи, среди которых основополагающей является задача повышения мотивации учащихся к обучению. Для ее решения становится все более важным применение нетрадиционных форм проведения уроков и применение нетрадиционных средств обучения. Среди таких средств видеоигры выделяются как одно из наиболее эффективных решений, способное привлечь и заинтересовать учащихся.

В современном мире компьютерные игры не только развлекают, но и создают уникальную образовательную среду, наполненную интерактивностью и увлекательными сюжетами. Компьютерные игры обладают потенциалом сделать учебный процесс увлекательным, доступным и эффективным.

Игровая деятельность является одной из наиболее эффективных видов деятельности для повышения мотивации, развития познавательных интересов и раскрытия творческого потенциала у обучающихся. В жизни в качестве мотивации к выполнению тех или иных действий выступают интерес и возможность получить некий практический результат, а также достижение цели. Такая мотивация называется внутренней. В обучении часто возникают ситуации, когда достижение практического результата от выполнения данных действий значительно отдалено во времени. В таком случае у обучаемых может наблюдаться потеря интереса к действию, и чтобы его поддерживать, можно использовать игровой подход к обучению. [7].

К положительным аспектам использования игрового подхода в обучении относятся снятие стресса, способствование раскрепощению,

создание положительного эмоционального фона на занятии. При этом игры дают наибольший эффект, если они подобраны в соответствии с целью и материалом конкретного учебного занятия. Существует связь между игровыми технологиями в образовательном процессе и игровой формой взаимодействия педагога и учащихся, достигающаяся через реализацию определенного сюжета. При этом образовательные задачи включаются в содержание игры. Реализация игровых приёмов и ситуаций при урочной форме занятий происходит по таким основным направлениям, как:

1. постановка дидактической цели перед учащимися в форме игровой задачи;
2. подчинение учебной деятельности правилам игры;
3. использование учебного материала в качестве средства игры и введение элемента соревнования в учебную деятельность для перевода дидактической задачи в игровую;
4. связывание успешного выполнения дидактического задания с игровым результатом [10].

В образовательном процессе игровой метод может быть реализован в виде различных игр, в частности, в виде компьютерных.

Наряду с уже привычными графическими и текстовыми редакторами, облачными технологиями, интерактивными презентациями в образовательный процесс внедряются подходы микрообучения, m-learning и геймификации [25].

Один из первых, кто предложил использовать компьютерные технологии для обучения школьников, был Сеймур Пэйперт. Он был математиком и одним из ранних пионеров в области робототехники. В 1960-х годах Паперт разработал язык программирования LOGO и создал "Логотоши", робототехнические устройства, предназначенные для обучения программированию детей. Паперт поддерживал идею, что дети должны иметь доступ к средствам выражения своих идей, а не только к приемам получения знаний. Он видел потенциал компьютерных технологий и

предпосылки для изменения образовательной парадигмы. Пэйперт заметил, что использование новых на тот момент технологий делает обучение более интерактивным и увлекательным для детей. В своей книге, "Mindstorms: Children, Computers, and Powerful Ideas», Пэйперт изложил свои идеи о том, как компьютеры и робототехника могут стать мощными инструментами для обучения и стимулирования мышления детей. Его работы и идеи внесли значительный вклад в развитие образовательных технологий и стали одним из первых примеров использования компьютерных технологий в образовании [3].

Мир информационных технологий и языков программирования претерпел заметные изменения. Сегодня новые языки программирования стали доступны для обучения и использования как никогда, открывая более широкие возможности для обучения. В настоящее время существует обширный выбор компьютерных игр, которые могут быть эффективно использованы в образовательных целях, особенно для обучения программированию. Эти игры предлагают учащимся интерактивные и увлекательные способы изучения основ программирования, алгоритмов и компьютерной логики, что делает процесс обучения более увлекательным и доступным.

Онлайн школа CODDY разработала курс «Python и JavaScript»: игровое программирование в CodeCombat». CodeCombat – это многопользовательская браузерная игра, позволяющая научиться программировать. Курс подойдет детям с 10 лет, длительность прохождения курса 4-6 месяцев, а периодичность занятий составляет 1 раз в неделю по 2 часа. В результатах обучения заявлено, что, после освоения курса ребенок научится [29]:

1. грамотно и качественно писать программы на Python и/или JavaScript;
2. сотрудничать и работать как самостоятельно, так и в команде;
3. решать нестандартные задачи.

CodeMonkey - это образовательная платформа, разработанная для

обучения детей основам программирования через интерактивные игры и задачи. Она предоставляет ученикам возможность изучать программирование, играя в разнообразные игровые сценарии и решая задачи на таких языках программирования, как CoffeeScript и JavaScript. CodeMonkey предлагает серию интерактивных курсов, разработанных для детей от 5 до 14 лет. Курсы разделены на уровни сложности, позволяя учащимся начинать с базовых концепций и постепенно переходить к более сложным задачам. Учащиеся изучают программирование, решая различные игровые задачи и головоломки. Каждая задача представляет собой новый игровой уровень, который требует написания кода на JavaScript для решения конкретной задачи или достижения цели. Платформа предоставляет ученикам встроенный редактор кода, который позволяет им писать, исправлять и тестировать свой код непосредственно в браузере. Это обеспечивает моментальную обратную связь и позволяет учащимся наблюдать результаты своей работы в реальном времени [12].

Онлайн школа «Пиксель» разработала курс программирования на основе платформы Scratch. Scratch - это интерактивная платформа, предназначенная для обучения программированию детей с помощью готовых блоков кода. Она позволяет детям создавать анимации, игры и интерактивные истории, используя простой и интуитивно понятный интерфейс. Курс рассчитан для детей возрастом 8-12 лет и состоит из 3 модулей:

1. знакомство по Scratch;
2. интерактивность в Scratch;
3. создание игра в Scratch.

В среде программирования Scratch дети 8–12 лет учатся программировать при помощи разноцветных блоков. Цветные элементы выполняют роль команд: умение писать скрипты не пригодится на курсах Scratch для детей онлайн и офлайн. Визуальная разработка игр на Scratch станет первым шагом ребенка в мире IT. В результате программирования в

среде Scratch будущие разработчики научатся делать компьютерные видеоигры и анимации, осуществлять арифметические операции, работать с координатной системой [30].

Программированию на языке Python в игре Minecraft так же учат в школе программирования «Пиксель». Курс представляет собой уникальную образовательную программу, целью которой является обучение программированию с использованием популярной игры Minecraft. Курс предлагает разнообразные учебные материалы, включая теорию с примерами, пошаговые инструкции по созданию проектов и задачи для закрепления материала. Преподаватели используют методические разработки школы и делятся своим опытом и примерами из практики для помощи детям в усвоении материала. Программирование в Minecraft на языке Python подходит для любителей видеоигр и поклонников этой популярной вселенной. Курсы предназначены для детей в возрасте от 9 до 13 лет. Курс состоит из 12 модулей по 90 минут. Занятия проходят в формате очных или онлайн занятий в группе, либо индивидуально в режиме онлайн [30].

В московской школе № 1517 разработали для учащихся 5 и 6 классов программу дополнительного образования "Программирование в среде ROBLOX". В ходе обучения ученики изучат типы данных программирования и основные алгоритмы на языке Lua, используемом в среде ROBLOX. Стоимость курса составляет 4500 рублей в месяц. Очные занятия проходят в кружках и секциях в пределах групп, сформированных по параллелям классов. Этот курс предоставляет уникальную возможность для учеников начать изучение программирования и создания игр в популярной среде ROBLOX [23].

Чтобы вдохновить обучающихся осваивать программирование, с 2014 года при поддержке ведущих компаний IT-отрасли, Министерства связи и массовых коммуникаций и Министерства образования и науки России проводится тематическая образовательная акция «Час кода», в ходе которой преподавателям и всем, кто работают с детьми и молодёжью 6-18 лет,

предлагается провести яркие тематические уроки, занятия и различные мероприятия, направленные на формирование и поддержку интереса молодых людей к изучению информатики, информационных технологий и повышение престижности ИТ-специальностей. Задачи акции:

- создать условия для повышения мотивации и увлечь как можно большее число обучающихся изучением информатики и программирования (в том числе тех, кто пока не приступал к изучению программирования в рамках школьной программы);
- познакомить обучающихся с основными направлениями и возможностями развития информационных технологий и программирования;
- предоставить обучающимся информацию о работе ИТ-компаний и ИТ-специалистов, необходимых им ресурсов, знаний, умений и 51 навыков, а также возможностей для самообразования и развития в данном направлении;
- показать, что работа в ИТ-индустрии и профессия программиста – это интересно, увлекательно, престижно и перспективно;
- создать ситуацию успеха (положительный опыт) изучения программирования для каждого участника акции [8].

Реализация обучающей системы параллельному программированию на базе языка Кубик разрабатывается в модульно объектно-ориентированной динамической среде с интегрированным приложением виртуального «3Д-мира Second Life».

Second Life - это трехмерный виртуальный мир с элементами социальной сети, его пользователи могут создавать трехмерные фигуры, управлять их поведением, общаться с другими пользователями с помощью текстовых и голосовых инструментов.

Дополнительные возможности курсов в Second Life, в первую очередь, заключаются в предоставлении виртуального пространства для встреч студентов, временно находящихся для обучения или стажировки за рубежом,

и потому не способными встретиться для обсуждения проблем и обмена опытом. Но возможности Second Life в обучении не ограничиваются виртуализацией совместной работы в группах географически удаленных друг от друга людей. Виртуальные миры предлагают и другие важные преимущества: симуляторы, деловую информацию и контакты, а также доступ к другим учебным заведениям. Каждая встреча в Second Life - всегда испытание в полевых условиях и экскурсия одновременно, а богатство виртуальных предложений в этом ориентированном почти исключительно на социальное взаимодействие мира в мире явилось одной из основных причин выбора Second life в качестве платформы для обучения. LSL - скриптовый язык программирования, используемый при создании контента резидентами (пользователями) виртуального мира Second Life, созданного компанией Linden Lab. С помощью LSL-скрипов можно управлять поведением объектов виртуального мира. LSL позволяет объектам взаимодействовать с миром Second life и Интернетом посредством электронной почты [9].

Исследования показывают, что процесс мотивации можно контролировать. Ученые подтверждают: чем выше уровень мотивации учащегося, будь то внутренняя или внешняя, к изучению определенного предмета, тем более эффективным становится его обучение [5].

В настоящее время компьютерные игры становятся все более распространенным инструментом образования, внедряясь в учебные процессы. Использование игровых технологий направлено на решение различных образовательных задач, среди которых формирование позиции учащегося в отношении собственной деятельности, общения и саморазвития.

Компьютерные игры охватывают широкий спектр жанров: от игр с приключениями до стратегических и многопользовательских игр. Обучающие компьютерные игры классифицируются по содержанию и целям игры, подстраиваясь под дидактические задачи.

В исследовании [19] отмечается эффективность использования программы Macromedia Flash для создания мотивирующих обучающих игр в

области информатики для школьников. Результаты экспериментального исследования показывают, что применение компьютерных игр в обучении информатике для школьников 8-9 классов способствует повышению мотивации и эффективности учебного процесса, а также качества обучения в целом.

На основе проведённого исследования можно сделать вывод, что программированию в компьютерных играх учат в основном в специальных школах программирования. С развитием информационных технологий открываются новые возможности для обучения. Сегодняшние образовательные практики включают использование компьютерных игр для обучения программированию как эффективного способа повышения мотивации учащихся к обучению. Данное направление является развивающимся и тем самым представляет интерес в рамках проводимого исследования.

1.2. ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ PYTHON В КОМПЬЮТЕРНЫХ ИГРАХ

Исследователи В. В. Бабенко и Ю. В. Гольчевский подчёркивают важность правильного выбора языка программирования для обучения специалистов в области прикладной информатики. Этот язык должен быть достаточно мощным, соответствовать учебным и профессиональным задачам, быть удобным для изучения и использования [4].

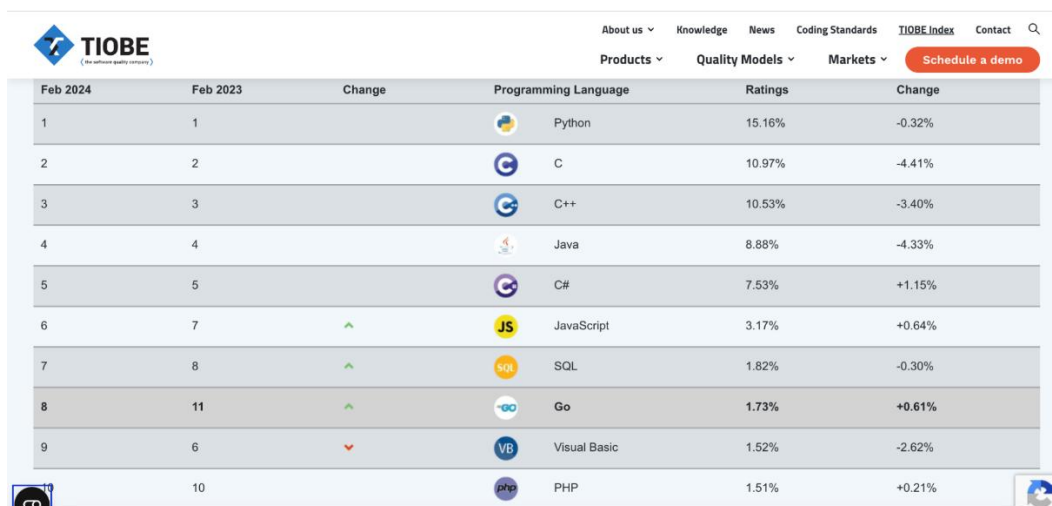
Выбор языка программирования также зависит от возрастных особенностей старшеклассников, поэтому язык программирования должен быть простым для освоения молодыми людьми и обладать широким функционалом для решения задач из школьного курса информатики, а также для успешной сдачи ОГЭ и ЕГЭ.

Многие исследователи, в том числе Л. Самыкбаева, А. Беляев, А. Палитаев, И. Ташиев, С. Маматов подтверждают, что Python является одним из популярных языков программирования [24].

Язык программирования Python широко используется во всём мире

благодаря своей универсальности: он подходит для решения разнообразных задач, начиная с простых операций и заканчивая разработкой сложных автоматизированных систем. Этот язык легко осваивают новички, поэтому его следует применять при обучении школьников программированию.

Согласно анализу Tiobe, Python занимает первое место по популярности среди таких языков, как C++, C#, Visual Basic, JavaScript, Assembly, SQL и PHP в 2023 году [15].



Feb 2024	Feb 2023	Change	Programming Language	Ratings	Change
1	1		Python	15.16%	-0.32%
2	2		C	10.97%	-4.41%
3	3		C++	10.53%	-3.40%
4	4		Java	8.88%	-4.33%
5	5		C#	7.53%	+1.15%
6	7	▲	JavaScript	3.17%	+0.64%
7	8	▲	SQL	1.82%	-0.30%
8	11	▲	Go	1.73%	+0.61%
9	6	▼	Visual Basic	1.52%	-2.62%
10	10		PHP	1.51%	+0.21%

Рис. 1. Индекс tiobe за май 2024

При обучении программированию на языке Python прежде всего необходимо обратить внимание на то, что язык Python является объектно-ориентированным языком, но следует школьников следует обучать процедурному стилю программирования в рамках базового курса информатики, так как это позволяет продолжать линию развития алгоритмических умений школьников путем обучения записи алгоритмов на языке программирования.

Особенности синтаксиса и типизации языка Python таковы, что можно научиться структурному программированию на этом языке, не имея знаний в области объектно-ориентированного программирования.

В своей статье исследователь Е. Климекова отмечает, что правильный выбор первого языка программирования оказывает значительное влияние на формирование навыков программирования у обучающихся, а также на усвоение и понимание соответствующих алгоритмов [16].

Были выявлены следующие преимущества языка Python:

- Синтаксис языка прост, поэтому многие начинают изучать программирование с него.
- Python является лидером по популярности среди языков программирования.
- Множество созданных библиотек и расширений, ответы на любые возникающие вопросы зачастую сразу можно найти в интернете.

Сервис CodeCombat позволяет приступить к изучению программирования на языке Python в игровой форме на начальном этапе основной школы. Данный ресурс не является курсом уроков, а представляет собой полноценную мультиплеерную игру. Это одно из немногих игровых приложений для изучения программирования, где предполагается работа с кодом в «чистом» виде. Данный сервис русифицирован, а интерфейс интуитивно понятен для пользователей любого возраста.

По ходу прохождения различных по сложности уровней учащийся изучает основные правила синтаксиса, реализацию алгоритмических структур и напрямую работает с кодом. CodeCombat охватывает множество аспектов программирования: строки, переменные, вызов метода, векторную графику и многое другое. Обучение ведется на языках программирования Python, JavaScript и нескольких экспериментальных версиях JavaScript. В начале 2014 года основатели CodeCombat опубликовали исходный код проекта. Теперь все желающие могут вносить свои изменения, разрабатывать новые уровни. Таким образом, сервис стремительно развивается и совершенствуется. И если учитель обладает достаточными навыками в области разработки, он может не только руководить игровым процессом, но и принимать участие в создании игрового мира и привлекать к этому учащихся. Принципиальным отличием сервиса является его ориентированность на форму отношения «учитель–ученик». Преподаватель может объединять учеников в группы («кланы») и контролировать их прогресс в игре с помощью статистики. Помимо этого, разработчики игры

оказывают содействие педагогам по электронной почте, предоставляют обучающие видео и другие виды поощрений. У сервиса существует развитое пользовательское сообщество, поэтому недостатка в поддержке и единомышленниках не ощущается [26].

Check iO – игра в жанре «приключение», с помощью которой можно изучать язык программирования Python. Суть игры заключается в том, что пользователь имеет определенный набор задач, связанных с написанием кода, которые он должен выполнить для того, чтобы продвигаться дальше, постепенно улучшая навыки программирования, осваивая новые знания в этой области. Содержательно игра обучает основам программирования и учит работать: с разными типами данных и переменными, условными операторами (if, else, elif), циклами (for, while), с функциями в Python [11].

Codemonkey – игра в жанре «приключение». Игроку предлагается играть маленькой обезьянкой, собирающей бананы на поле. Для этого необходимо создать последовательность команд в правой текстовой панели на языке Python и запустить их. Способ программирования представляет собой реализацию scratch технологии, когда код составляется из уже готовых блоков. Сайт игры доступен на русском языке. Начиная с простых действий, дети научатся создавать сложные алгоритмы. Codemonkey предлагает серию заданий на Python, посвященных созданию простого чат-бота [12].

Roblox — популярная игра в жанре песочницы. В Roblox также есть возможность использовать Python для создания скриптов и программ [14].

В Roblox ученики могут создавать свои собственные игры и взаимодействовать с игровым миром, используя встроенный язык программирования Lua. Использование Roblox Studio в качестве интегрированной среды разработки, позволяет ученикам изучать основы программирования через создание собственных игровых миров, персонажей и скриптов. Это предоставляет им возможность не только понять основы программирования, но и применить их на практике, создавая уникальный контент внутри игровой платформы. Однако Roblox не является игрой с

открытым исходным кодом, ее адаптация для программирования на языке Python не популярна [14].

Minecraft — еще одна популярная игра, которая позволяет игрокам создавать свои собственные миры и взаимодействовать с ними. Для Minecraft разработана библиотека Minecraft Python API. С помощью этой библиотеки игроки могут создавать различные скрипты и программы, которые будут взаимодействовать с миром Minecraft [13].

Minecraft является простой игрой, которую легко освоить даже начинающим игрокам. Это делает её идеальной платформой для изучения основ программирования на Python. Здесь можно создавать множество различных миров и сценариев, что позволяет ученикам изучать различные аспекты программирования, такие как работа с переменными, функциями, классами и т. д. Ученики могут сразу же видеть результаты своих действий в игре, полученные навыки программирования ученики могут применить в реальной жизни, что делает процесс обучения более увлекательным и мотивирующим.

В Minecraft ученики могут изучать следующие разделы программирования на Python [13]:

1. Основы синтаксиса Python. Ученики изучают основные конструкции языка Python, такие как переменные, функции, циклы и условия.
2. Работа с библиотеками и модулями Python. Ученики учатся использовать библиотеки и модули Python для взаимодействия с игровым миром.
3. Создание собственных программ и скриптов. Ученики могут создавать свои собственные программы и скрипты, которые будут взаимодействовать с игровым миром Minecraft, добавлять в него новые возможности.

Рассмотрение особенностей программирования на Python в приведенных играх позволяет сделать вывод, что игры, в которых можно

обучать программированию можно разделить на две категории: игровые онлайн сервисы и песочницы.

Игровые онлайн сервисы при обучении программированию на Python, как правило, уже имеют разработанный учебный курс, посвященный обучению основам программирования на языке и весьма ограничены.

Игры песочницы же напротив, дают неограниченные возможности по обучению школьников программированию, так как предоставляют большой инструментарий для разработки.

Из рассмотренных песочниц, для обучения школьников программированию на Python, выберем Minecraft, так как эта игра имеет открытый исходный код, и огромная фанбаза уже разработала для нее библиотеку Minecraft Python API, которая позволяет взаимодействовать с миром Minecraft на язык Python.

Таким образом, выбор языка Python обоснован его актуальностью, поскольку он является лидером по популярности среди других языков программирования на сегодняшний день. В качестве компьютерной игры для обучения школьников программированию была выбрана игра Minecraft, так как эта игра имеет открытый исходный код и для нее уже разработана библиотека Minecraft Python API, которая позволяет взаимодействовать с миром Minecraft на языке Python, а также предоставляет широкие возможности по обучению школьников программированию на этом языке.

В данной работе предлагается разработать комплекс лабораторных работ по обучению школьников программированию на Python в компьютерной игре Minecraft.

1.3. ЛАБОРАТОРНЫЕ РАБОТЫ КАК СРЕДСТВО ОБУЧЕНИЯ ПРОГРАММИРОВАНИЮ

Лабораторная работа – это средство обучения, при котором учащиеся под руководством учителя и по заранее намеченному плану проделывают опыты или выполняют определённые практические задания и в процессе их выполнения воспринимают и осмысливают новый учебный материал [21].

Значение слова «лаборатория» (от латинского labor - труд, работа, трудность) указывает на сложившиеся в далекие времена понятия, связанные с применением умственных способностей и трудовых усилий для разрешения возникших научных и жизненных задач. Лабораторные работы позволяют интегрировать теоретические знания, практические умения и навыки в едином процессе учебной деятельности по предмету.

Важной составляющей лабораторных работ являются практические задания. Основа практических заданий - пример, который разбирается с позиций теоретических знаний, развитых посредством изучения материала в учебнике. Как правило, основное внимание уделяется формированию конкретных умений, навыков, что и определяет содержание деятельности учащихся - решение задач, графические работы, являющиеся предпосылкой правильного мышления и речи.

Лабораторные работы занимают промежуточное положение между теоретическим и практическим обучением и выступают в роли связующего звена теории и практики. С одной стороны, это закрепление и совершенствование знаний, а с другой - формирование умений, которое применяется в процессе дальнейшего обучения. Значимость лабораторных работ для развития учебно-познавательных компетенций объясняется тем, что они наиболее широко и глубоко охватывают практическую сторону предмета информатика и ИКТ в школе, предоставляют учащимся возможность самостоятельно открывать известные научные факты в интересной, увлекательной форме, развивают экспериментальные, конструктивные, коммуникативные, исследовательские умения учащихся, творческий подход [3].

По форме организации лабораторных работ выделяют [28]:

- фронтальные (все учащиеся выполняют одну и ту же работу под руководством учителя);
- групповые (одна и та же работа выполняется группами учащихся по 2–5 человек, далее результаты объединяются или анализируются по группам);

– индивидуальные (каждый учащийся самостоятельно выполняет свою работу).

Лабораторные работы бывают:

- исследовательские (исследование процессов и явлений);
- иллюстративные (составление графиков, схем и т.д.);
- фронтальные (учащиеся работают на своих рабочих местах с программными средствами).

Е. А. Широкова выделяет 3 типа лабораторных работ с использованием информационно-коммуникационных технологий (ИКТ) по степени самостоятельной активности учащихся по их выполнению [27]:

1) демонстрационные – учитель сам выполняет работу с помощью ИКТ, ученики лишь наблюдают за её выполнением и делают самостоятельные выводы;

2) фронтальные – учитель показывает ученикам, как нужно выполнять работу, затем учащиеся выполняют её самостоятельно с использованием аналогичных моделей, после чего обсуждаются результаты и делаются выводы;

3) самостоятельные – ученики полностью самостоятельно выполняют работу в качестве творческого или зачётного задания.

Структура лабораторных работ:

- цели и задачи обучения;
- теоретические сведения по теме;
- демонстрационные примеры;
- самостоятельные задания.

Лабораторные работы как метод обучения во многом носят исследовательский характер и могут относиться к числу методов, активизирующих и мотивирующих учебно-познавательную деятельность учащихся, ведь во время их выполнения они являются активными участниками учебного процесса и сами добывают новые знания или

закрепляют уже полученные.

Лабораторные работы активно применяются на уроках информатики. Например, «Компьютерный практикум» в УМК Л.Л. Босовой для 7-9 классов [6]; методическое пособие, которое содержит 10 лабораторных работ по разделу информатики «Алгоритмизация и программирование» учителя информатики из г. Инты [18]; лабораторные работы по обучению школьников 9 классов языку программирования Python [17]. В лабораторных работах есть готовые решения задач, также прилагаются задания для самостоятельной работы. В конце каждой лабораторной работы приводятся вопросы для самоконтроля и защиты.

Лабораторные работы помогают развивать у обучающихся:

- творческие способности в ходе выполнения самостоятельных творческих заданий;
- навыки использования информационных технологий и различных источников информации для решения познавательных задач;
- умение вести индивидуальную работу, умение самостоятельного поиска решения новой задачи;
- помогает формировать интерес к информатике.

Во время проведения лабораторной работы учителю необходимо отслеживать ее выполнение каждым учеником и при необходимости оказывать оперативную помощь. При этом удобно использовать локальную компьютерную сеть. В случае, если будет обнаружено, что часть учащихся допускает однотипную ошибку, учитель должен остановить работу всех учеников, указать на нее и объяснить правильный ход работы. Проведение лабораторной работы происходит в форме инструктирования (вводного и текущего), основной задачей которого является создание у учащихся ориентировочной основы деятельности для наиболее эффективного выполнения заданий. На лабораторных работах доминирующей составляющей является процесс формирования экспериментальных умений учащихся.

Кроме всего прочего, такой тип урока обладает следующими основными достоинствами: практическое применение теоретических знаний; развитие экспериментальных умений; развитие самостоятельности учащихся; возможность деления учащихся на группы; возможность ранжировать учебный материал; использование дополнительных средств обучения [3].

Лабораторные работы позволяют обучающимся быстро и эффективно усваивать учебный материал. Поэтому было принято решение о создании лабораторных работ по программированию на языке Python в компьютерной игре Minecraft. В процессе выполнения лабораторных работ, обучающиеся смогут углубить знания основ программирования, решая сложные задачи по созданию игровых сценариев и симуляций, они научатся разрабатывать программы.

Таким образом, лабораторные работы по программированию на языке Python в Minecraft могут развить у обучающихся не только навыки программирования, но и творческие способности, логическое мышление.

Глава 2. Методика обучения программированию на языке Python в компьютерных играх

2.1. Цели и содержание обучения программированию на языке Python в компьютерной игре Minecraft

В процессе разработки собственной учебной программы учитель имеет возможность интегрировать обучение программированию на языке Python с компьютерными играми, такими как Minecraft. Это позволит создать увлекательную и интерактивную среду обучения, где учащиеся могут применять свои навыки программирования непосредственно внутри игровой среды. Такой подход стимулирует интерес учащихся к изучению программирования, поскольку он делает процесс обучения более увлекательным.

Целями обучения являются:

1. развитие алгоритмического мышления;
2. развитие творческого потенциала;
3. углубление знаний синтаксиса языка программирования Python.

Содержание:

1. Необходимое программное обеспечение и его запуск. Игра Minecraft.
 - 1.1. Запуск необходимого программного обеспечения.
 - 1.2. Первый запуск программы.
2. Система координат. Установка блоков.
 - 1.1. Система координат.
 - 1.2. Установка блоков.
3. Создание шахматной доски.
 - 1.1. Простые и вложенные циклы for.
 - 1.2. Математические операции: сложение, нахождение остатка от деления.
 - 1.3. Определение является ли число четным.

- 1.4. Определение и вызов функций.
4. Размещение шашек на доске.
 - 1.1. Работа со словарями.
 - 1.2. Работа со списками.
 - 1.3. Работа с кортежами.
 - 1.4. Аргументы функции.
5. Реализация основным игровых механик.
 - 1.1. Нахождение средней точки между двумя заданными точками в трехмерной системе координат.
 - 1.2. Оператор return в функциях.
 - 1.3. Константа None.
 - 1.4. Функция time.sleep()
6. Реализация управления игровым процессом через игровой чат.
 - 1.1. Бесконечный цикл while
 - 1.2. Вложенные конструкции if и циклы while
 - 1.3. Конструкция while-else.
 - 1.4. Обработка исключений в Python.
 - 1.5. Функция split().

Для обучения программированию на языке Python в компьютерной игре Minecraft было принято решение использовать лабораторный практикум, состоящий из 6 лабораторных работ. Они включают в себя самостоятельную работу, но с учётом того, что она будет иметь репродуктивный характер.

2.2. КОМПЛЕКС ЛАБОРАТОРНЫХ РАБОТ ДЛЯ ОБУЧЕНИЯ ПРОГРАММИРОВАНИЮ НА ЯЗЫКЕ PYTHON В КОМПЬЮТЕРНОЙ ИГРЕ MINECRAFT

Лабораторные работы будут строиться в соответствии со следующими принципами:

1. теоретический материал не выделяется в качестве отдельного блока, а органичным образом сочетается с практической деятельностью;

2. в процессе выполнения каждой лабораторной работы обучающимся предлагаются задания, которые могут служить для наилучшего осознания теоретического материала, отработки и закрепления учебных навыков, самостоятельной работы под руководством учителя, контроля;
3. каждая лабораторная работа является логическим продолжением предыдущей,
4. материал, как теоретический, так и практический, представляется от простого к сложному,

В рамках работы были разработаны 6 лабораторных работ по информатике.

Таблица 1.

№ Л/Р	Тема	Цель работы	Краткое описание
1	Запуск и настройка программного обеспечения	Научиться запускать необходимое программное обеспечение. Знакомство с первой функцией библиотеки Minecraft Pi Edition API, <code>mc.postToChat()</code>	В первой лабораторной работе содержится информация о том, как правильно и в какой последовательности необходимо запускать программное обеспечение. В конце лабораторной работы учащийся сможет написать первый код, в котором он реализует отправку сообщения в чат Minecraft.
2	Система координат. Установка блоков	Познакомиться с системой координат мира Minecraft, научиться устанавливать блоки	Во второй работе обучающиеся получают представление о системе координат в игре и освоят установку блоков разных типов по заданным координатам
3	Создание шахматной доски	Научиться создавать сложные структуры из блоков в игровом мире Minecraft, познакомиться с вложенными циклами и функциями в Python.	В третьей лабораторной работе обучающиеся познакомятся с вложенными циклами и функциями в Python, разработают функции <code>clear()</code> и <code>def create_chessboard(x, y, z, boardsize)</code> , позволяющие очищать заданную область от блоков и строить шахматную доску с заданными параметрами.
4	Размещение	Научиться устанавливать	В четвертой лабораторной работе

	шашек на доске	шашки на игровой доске, разработать стартовую конфигурацию игровой доски для игры в шашки, познакомиться со словарями, кортежами и списками в Python.	обучающиеся познакомятся со словарями, кортежами и списками в Python, научатся устанавливать шашки на игровое поле. С помощью словаря MineToChess от трехмерной системы координат мира Minecraft будет реализован переход к двухмерной системе координат, используемой в шашках, будет разработана функция <code>start_game()</code> , задающую стартовую конфигурацию расположения шашек.
5	Реализация основных игровых механик	Разработать функции для реализации перемещения и захвата шашек, познакомиться с оператором <code>return</code> , константой <code>none</code> и функцией <code>time.sleep()</code> в Python	В пятой лабораторной работе обучающиеся познакомятся с оператором <code>return</code> , константой <code>None</code> и функцией <code>time.sleep()</code> в Python, разработают функции <code>move(start, end)</code> и <code>find_middle(start, end)</code> , реализующие перемещение и захват шашек на игровом поле.
6	Реализация управления игровым процессом через игровой чат	разработать цикл для обработки игровых событий, реализовать управление игровым процессом через чат игры, познакомиться с обработкой исключений в Python и функцией <code>split()</code> .	В шестой лабораторной работе обучающиеся познакомятся с обработкой исключений и функцией <code>split()</code> в Python, разработают цикл для обработки игровых событий, реализуют управление игровым процессом через чат игры. В результате выполнения лабораторной работы будет получен финальный код, реализующий игру в шашки в мире Minecraft.

Содержание каждой лабораторной работы включает в себя:

- изложение теоретического материала, в том числе в виде заданий с пошаговым выполнением;
- задания для самостоятельного решения;
- необходимый иллюстративный материал;
- поясняющий текст, который предоставляет учащемуся возможности для обдумывания, разбора и выполнения самостоятельных действий.

Выполнение лабораторных работ предусматривает демонстрацию

преподавателю выполненной работы и воспроизведение каких-либо действий по просьбе преподавателя. Практическая проверка позволит выявить насколько хорошо обучающиеся умеют применять полученные умения на практике.

Лабораторные работы должны проводиться в специально оборудованных кабинетах, а именно в компьютерных классах. На персональных работах должны быть установлено следующее программное обеспечение:

1. интерпретатор Python;
2. среда разработки Visual Studio Code;
3. ProgMine Launcher.

Самостоятельная работа обучающихся при проведении лабораторного практикума занимают большую часть времени. Самостоятельная работа обучающихся начинается с того что, обучающиеся открывают лабораторную работу, изучают теоретическую часть, знакомятся с поясняющим текстом и предложенными практическими заданиями.

Разработанные лабораторные работы позволят углубить знание языка программирования Python, в результате выполнения лабораторных работ будет реализована возможность играть в шашки в компьютерной игре Minecrft. В процессе разработки, обучающиеся познакомятся с вложенными циклами, списками, кортежами, словарями и функциями в Python.

В Лабораторных работах 3-6 последовательно создается необходимый функционал для реализации игры в шашки, каждая следующая лабораторная работа включает в себя необходимость использовать ранее разработанные функции.

Рассмотрим для примера лабораторную работу №3 и лабораторную работу №6.

Лабораторная работа №3

Создание шахматной доски

Цель: Научиться создавать сложные структуры из блоков в игровом мире Minecraft, познакомиться с вложенными циклами и функциями в Python.

Создайте файл lab3.py в своей папке. Откройте его с помощью VSC.



Шахматная доска - это классическое игровое поле для шахмат/шашек, состоящее из квадратных клеток, укладываемых в 8 рядов и 8 столбцов. Каждая клетка доски имеет уникальные координаты, которые обычно обозначаются буквами от "a" до "h" для столбцов и цифрами от 1 до 8 для рядов. Например, клетка в нижнем левом углу доски обозначается как "a1", а клетка в верхнем правом углу - как "h8".

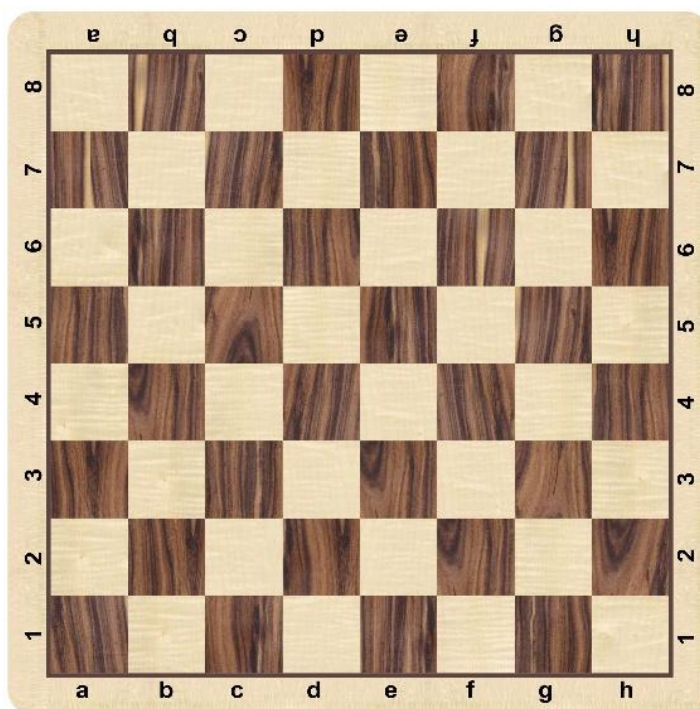


Рис. 1. Шахматная доска

Для того, чтобы научиться создавать шахматную доску в игре Minecraft с помощью Python, необходимо:

1. Уметь создавать плоские фигуры из заданных блоков;
2. уметь использовать циклы и ветвление для автоматизации задач;
3. уметь определять четность/нечетность заданного числа.



Система координат Minecraft состоит из трех осей: X, Y и Z (см Лаб.2). определим за точку отсчета или «начальный блок» для построения

шахматной доски блок (0;3;0).

```
mc.player.setPos(0, 4, 0)
```

Для начала, телепортируемся в заданные координаты.

Не забудь, что если необходимо оказаться над желаемым блоком, то необходимо добавить единицу к координате Y!

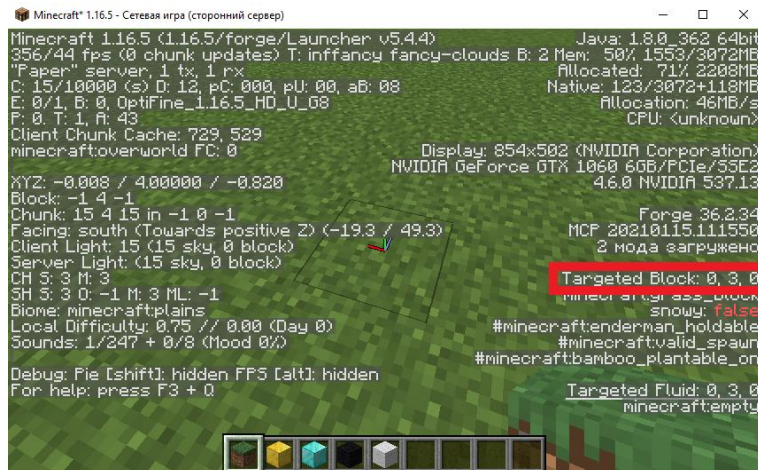


Рис. 2. Точка отсчета

Задание 1. Используя `mc.setBlock()`, постройте квадрат 3 на 3, как показано на рисунке 3.

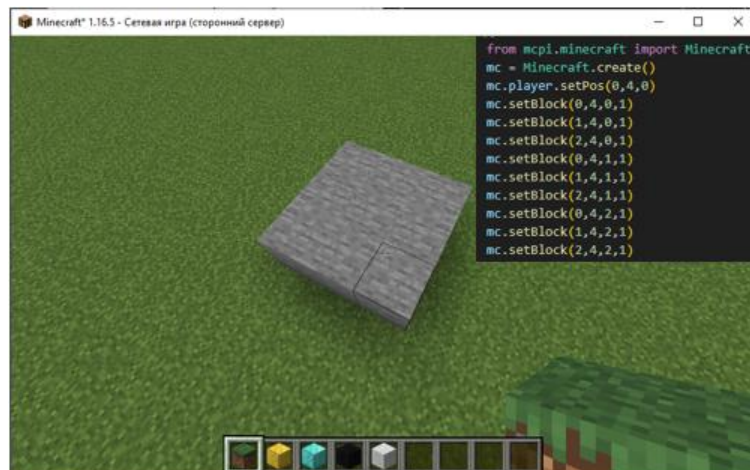


Рис. 3. Квадрат 3x3 из камня

После демонстрации учителю построенной фигуры, вручную сломайте все установленные блоки.

В данном примере каждая строка кода `mc.setBlock()` используется для установки блока в определенных координатах. Однако эти координаты увеличиваются на 1 для каждого блока.

Вместо того чтобы вручную указывать каждый набор координат,

необходимо использовать циклы для автоматического увеличения значений координат.

Обратим внимание, что меняются только координаты X и Z, значит потребуется два счетчика.



```
for i in range(3):  
    for j in range(3):  
        mc.setBlock(x + i, y, z + j, 1)
```



Цикл, находящийся внутри другого цикла, называется вложенным!

Цикл «for i in range(3)» и «for j in range(3)» выполнится ровно три раза, при этом счетчики i и j будут принимать значения от 0 до 2!



Задание 2. Объявите 3 переменные «x», «y», «z» и задайте им значение координат «начального блока». Постройте фигуру из задания 1, используя цикл, как показано на рисунке.

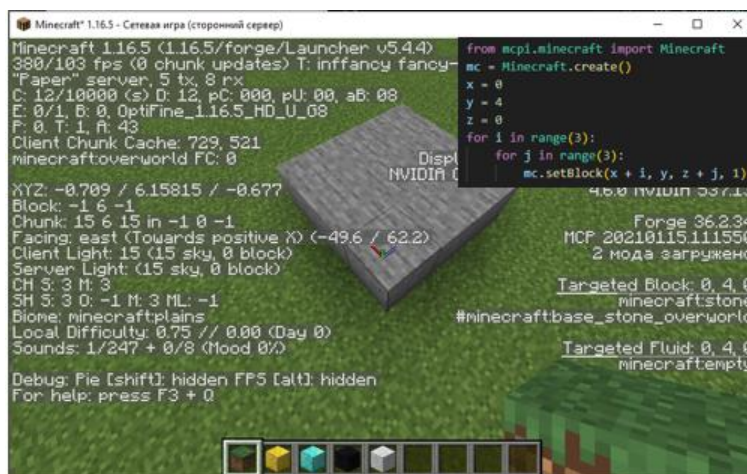


Рис. 4. Квадрат 3x3 из камня, построенный с помощью цикла

После демонстрации учителю построенной фигуры, вручную сломайте все установленные блоки.



В задании 1 и задании 2 для очистки поля каждый блок фигуры предлагалось сломать вручную. Квадрат 3x3 состоит всего из 9 блоков. Если же потребуется очистить поле от квадрата 8x8, то придется ломать уже целых 64 блока! Таким образом, уже сейчас требуется отладочный инструмент, позволяющий очистить плоскость заданной величины от всех блоков.

Измени тип блока, используемого в цикле, на воздух (0). В результате

выполнения цикла, увидим, что поле очистилось (Рис. 5).

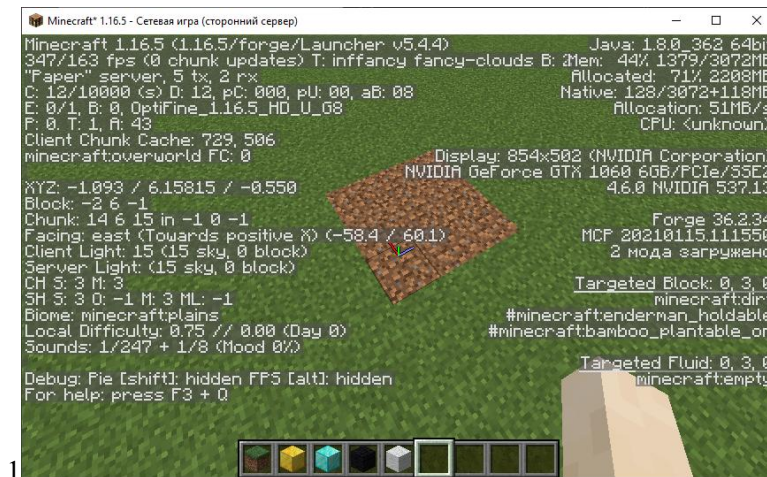


Рис. 5. Квадрат 3x3 из воздуха

В будущем предлагается использовать текущий код в качестве инструмента очистки игрового поля. Для этого создадим еще одну переменную «boardsize» и сохраним текущий код в качестве функции def clear(), где def – ключевое слово, которое определяет функцию с именем «clear», что значит «очистка».

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()
x = 0
y = 4
z = 0
razmerdoski=3

def clear():
    x = 0
    y = 4
    z = 0
    for i in range(razmerdoski):
        for j in range(razmerdoski):
            mc.setBlock(x + i, y, z + j, 0)

    for i in range(razmerdoski):
        for j in range(razmerdoski):
            mc.setBlock(x + i, y, z + j, 1)
```

Рис. 6. Определении функции «clear»

Дополните ваш код недостающими фрагментами и попробуйте его запустить.



Функция - это блок кода, который выполняет определенную задачу и может быть вызван из других частей программы. Функции позволяют структурировать код, делают его более читаемым и упрощают повторное использование.

Заметим, что после выполнения кода (Рис. 6) вновь получим квадрат 3x3 из камня. Это связано с тем, что функция «clear()» не была вызвана в коде, соответственно алгоритм не работает.

Чтоб выполнить код, содержащийся внутри функции «clear()», необходимо добавить в конец кода строку:



clear() - вызов функции

Проверьте работу функции clear(). После успешной проверки закомментируйте строку, используя символ «#».



#clear() - комментарий

Закомментированные строки кода не выполняются при запуске программы!

Таким образом, была создана функцию, которая по заданному размеру доски (boardsize), начиная с «начального блока», заменяет все блоки на воздух, а также научились вызывать эту функцию в коде программы. В дальнейшем будем использовать функцию clear() для очистки игрового поля от блоков.



Перейдем к самой задаче создания шахматной доски. В качестве белых клеток шахматного поля предлагается использовать блок кварца (тип 155), а в качестве черных клеток предлагается использовать блок обсидиана (тип 49).

Создайте квадрат размером 8x8 из блоков кварца.

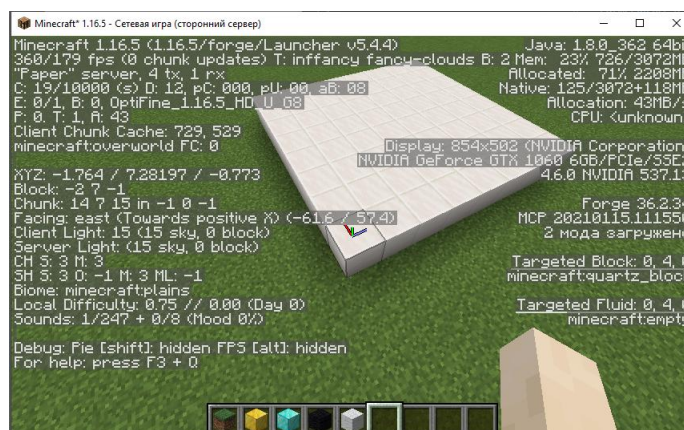


Рис. 7. Квадрат 8x8 из кварца

Вооружившись блоком обсидиана, постройте первые два ряда шахматной доски, как показано на рисунке 8.

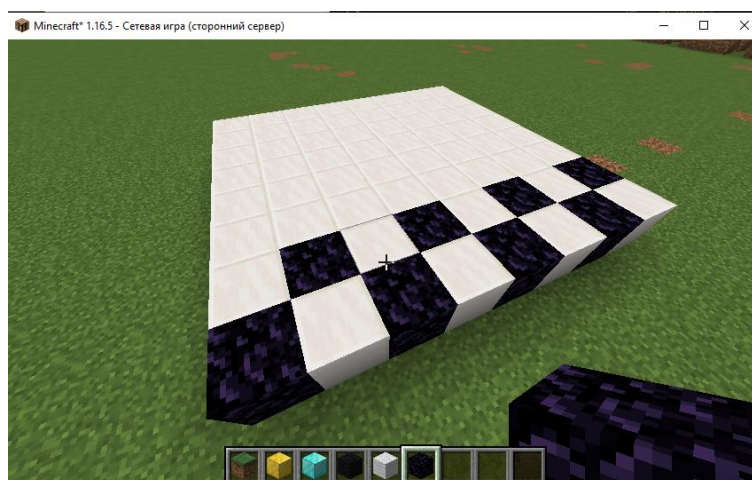


Рис. 8. Первые два ряда шахматной доски

Запишите в тетрадь координаты всех черных блоков первого ряда и второго ряда.

Заметим, что сумма координат $x+z$ для каждого черного блока обязательно является числом четным. Значит, при создании шахматной доски средствами Python, требуется уметь определять четность/нечетность заданного числа.



Из математики знаем, что число является четным в том случае, если оно без остатка делится на 2. В Python оператор для нахождения остатка от деления - это «%». Он возвращает остаток от деления одного числа на другое.

```
result = 10%3  
print(result)
```

При выполнении данного кода, консоль выведет 1.



Будем определять цвет блока относительно «начального блока». Цикл будет проходить по рядам и столбцам шахматной доски, определяя для каждой игровой клетки цвет по сумме индексов i и j . Если сумма $i+j$ четная, клетка черная, иначе белая.

Дополните код, с учетом вышесказанного. И запустите программу.

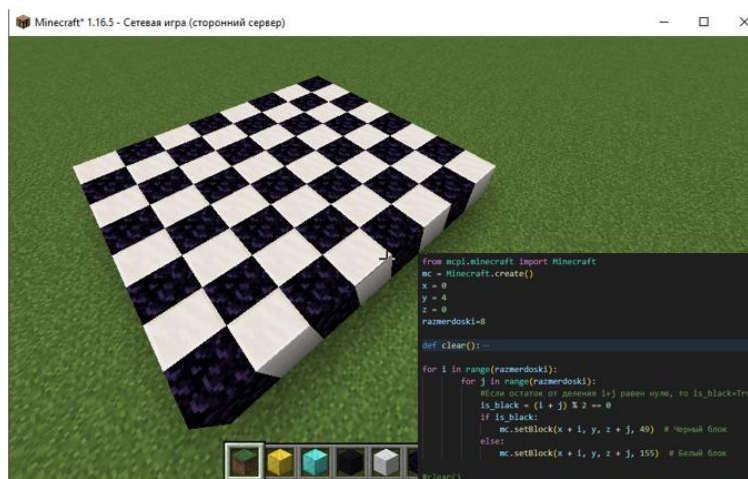


Рис. 9. Алгоритм построения шахматной доски 8x8



Задание 3. Используя текущий код, постройте конфигурации «шахматных полей» представленные на рисунке 10. Все конфигурации должны быть созданы в игровом мире одновременно.

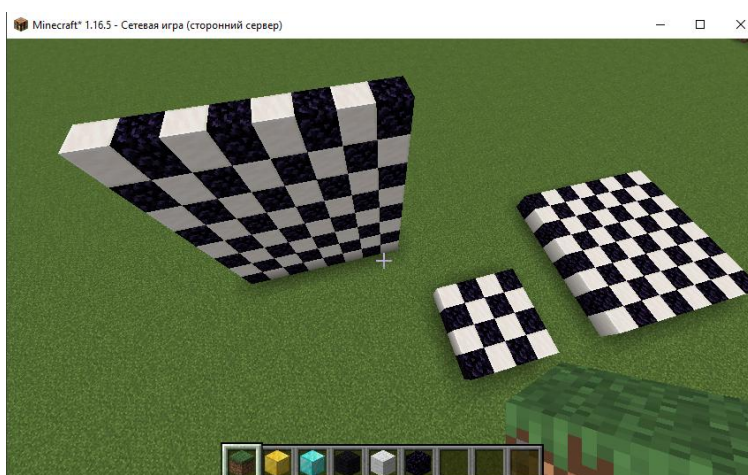


Рис. 10. Конфигурации шахматных полей



Задание 4. Очистите игровой мир, изменив и вызвав функцию «Clear».

После демонстрации учителю работы измененной функции «clear», используя сочетание клавиш Ctrl+Z, верните код к исходному состоянию (Рис. 9).



Теперь, когда была успешно построено шахматное поле размером 8x8, каждая клетка которого представлена одним блоком, важно задуматься о визуальной и функциональной составляющей.

В реальных шахматных досках клетки значительно крупнее, и они позволяют свободно разместить фигуры так, чтобы они имели достаточно места для перемещения. В Minecraft из-за особенностей графики и стиля

игры, клетки, состоящие из одного блока, не позволят перейти от шашек к шахматам, так же в текущем масштабе невозможно подписать столбцы и ряды шахматной доски (Рис. 1) и перейти от трехмерной системы координат к двумерной.

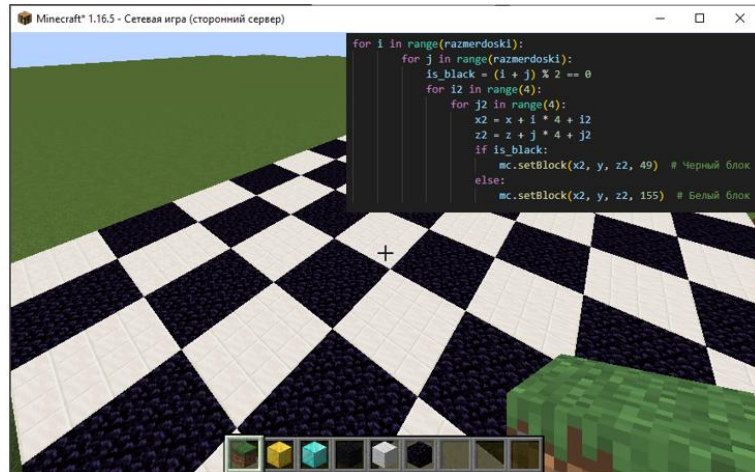


Рис. 11. Шахматная доска 8x8 с размером игровой клетки 4x4 блока

Продолжим создание шахматной доски, где каждая игровая клетка будет представлять собой область размером 4x4 блока. Для этого добавим внутренние циклы с переменными $i2$ и $j2$, которые будут использоваться для прохода по каждой клетке внутри 4x4 блока. Переменные $x2$ и $z2$ будут вычислять координаты каждой клетки внутри блока игровой доски.

Дополните ваш код (Рис. 11) и запустите.



Задание 5. Используя любой яркий блок, постройте буквы и цифры для шахматной доски для каждого ряда и столбца соответственно (Рис. 1)

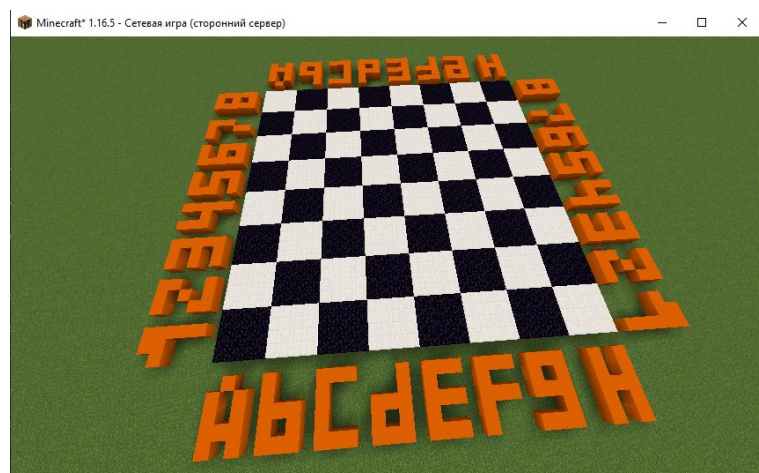


Рис. 12. Шахматная доска с подписанными рядами и столбцами



Задание 6. Определите функцию: `def create_chessboard(x,y,z,`

boardsize). Сохраните основной код в созданную функцию. Продемонстрируйте последовательный вызов функций:



```
clear()
create_chessboard(x, y, z, boardsize)
```

Заключение: в третьей лабораторной работе мы познакомились с вложенными циклами и функциями в Python, разработали функции `def clear()` и `def create_chessboard(x, y, z, boardsize)`, позволяющие соответственно очищать заданную область от блоков и строить шахматную доску с заданными параметрами.

Лабораторная работа №6

Реализация управления игровым процессом через игровой чат

Цель: разработать цикл для обработки игровых событий, реализовать управление игровым процессом через чат игры, познакомиться с обработкой исключений в Python и функцией `split()`.

Создайте файл `lab6.py` в своей папке. Откройте его с помощью VSC.

Импортируйте в него все функции, разработанные в предыдущей лабораторной работе.



```
from lab5 import *
```

Прежде чем приступить к реализации основного цикла для обработки игровых событий, необходимо вернуться к содержимому файлов `lab3-lab5` и проверить состояние исходного кода. Данные файлы должны содержать либо строки создания переменных, либо функции, разработанные в Лаб. 3 – Лаб. 5



Задание 1. Убедитесь, что в файлах `lab3-lab5` нет исполняемых инструкций. Продемонстрируйте учителю вызов функций `clear()` и `start_game()`



При запуске игры, необходимо получить в игровом чате приветственное сообщение, которое будет являться маркером, что программа работает корректно, а также небольшую инструкцию по управлению игровым процессом.



```
mc.postToChat("Игра началась!")
mc.postToChat("Введите ход (например, 'a3-b4')")
```

Затем приведем доску к стартовой конфигурации для начала игры в шашки.



```
clear()
start_game()
```

Для реализации управления игровым процессом через чат игры, необходимо научить программу «прослушивать» сообщения в игровом чате. Создадим метод **get_chat_message()**, который будет на протяжении всей игры отслеживать поступающие в чат сообщения.



```
def get_chat_message():
    while True:
        chats = mc.events.pollChatPosts()
        if chats:
            return chats[0].message
```

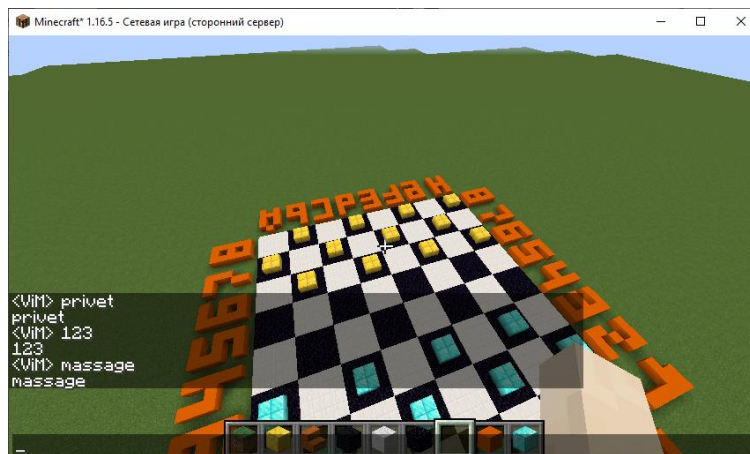
В теле цикла при работы программы создается переменная `chats`, которой присваивается значение функции, отслеживающей события в игровом чате **mc.events.pollChatPosts()**. Если значение данной функции ненулевое (не пустое), то функция **get_chat_message()** возвращает данное сообщение. Таким образом, функция будет всегда возвращать пойманный текст из игрового чата.



Задание 2. Продемонстрируйте учителю работу функции **get_chat_message()**. Отправьте в игровой чат 2-3 пойманных сообщений. Фрагмент кода для демонстрации и результат работы приведены на картинке ниже.



```
while True:
    mc.postToChat(get_chat_message())
```



Для реализации игрового цикла будем использовать цикл `while`, так как точное число повторений тела цикла неизвестно.



```
while True:
    call = get_chat_message()
    if call == "exit":
        mc.postToChat("Игра завершена!")
        break
```

Создадим в теле цикла переменную `call`, которая будет получать из функции `get_chat_message()` пойманные в ходе прослушки сообщения из игрового чата. Если полученным сообщением является «exit», то цикл прерывается. Результат работы программы приведен на рисунке 1. Таким образом, можем реализовать команды, управляющие игровыми событиями.

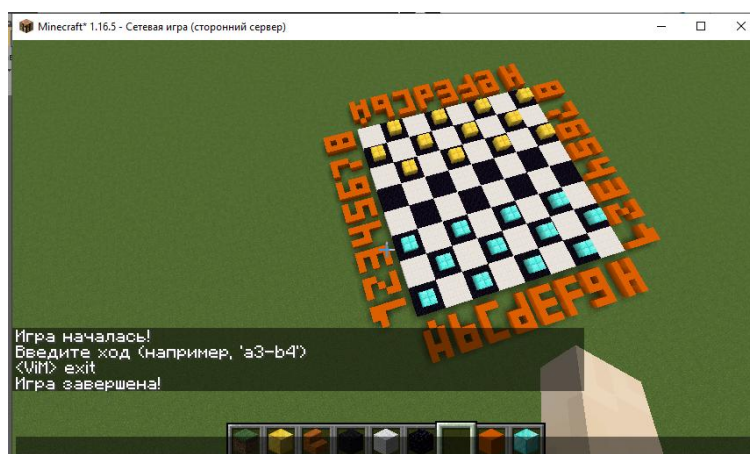
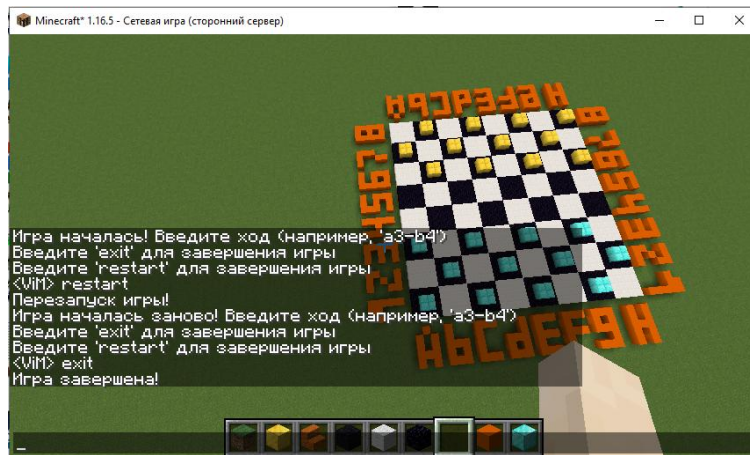


Рис. 1. Результат работы основного цикла игры.



Задание 3. Используя оператор ветвления `elif`, самостоятельно реализуйте команду перезапуска игры «restart». Дополните справку по управлению ходом игры добавленными командами. Результат выполнения задания 3 приведен на рисунке ниже.



Приведем ветвление полученных сообщений из игрового чата для игрового цикла:

```
if полученное сообщение == "команда 1":
    инструкции команды 1
elif полученное сообщение == "команда 2":
    инструкции команды 2
else:
    ход шашкой
```

Если полученное из игрового чата сообщение не является управляющей командой, значит оно задает ход шашкой. Функция **move(start, end)** принимает два аргумента в текстовом формате, например **move("a3", "b4")**. Встает задача получить из текстового сообщения ('a3'-b4') аргументы для функции **move(start, end)**.



```
start, end = call.split('-')
move(start, end)
```

Воспользуемся функцией **split()**.



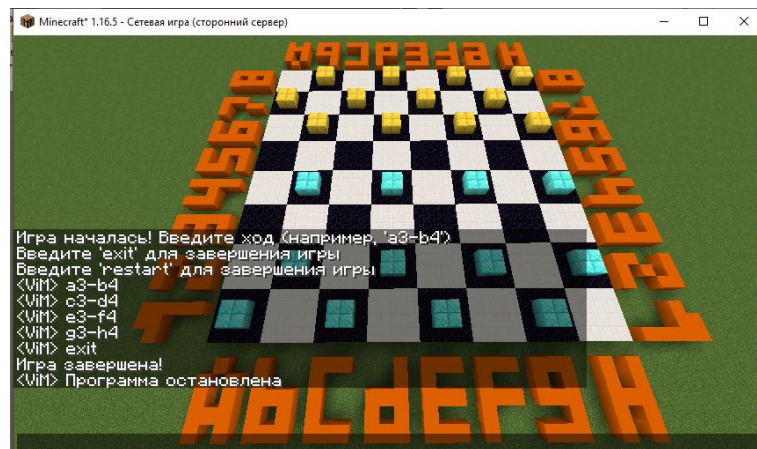
Функция **split()** используется в Python для разделения строки на части по разделителю. Если использовать **split("-")**, то строка будет разделена на две части по символу дефиса. Создадим внутри блока ветвления **else** переменные **start** и **end** и присвоим им соответственно координаты, задающие ход шашкой. Далее передаем полученные текстовые переменные в качестве аргументов метода **move(start, end)**.



Задание 4. Прдемонстрируйте учителю возможности по управлению игровым процессом через игровой чат. Введите в игровой чат последовательно сообщения:

1. a3-b4
2. c3-d4
3. e3-f4
4. g3-h4
5. exit
6. Программа остановлена

На рисунке ниже приведена конфигурация игровой доски, полученная в результате ввода сообщений в игровой чат



Запустите программу. Попробуйте ввести в игровой чат сообщение произвольного формата, например, 123.

```
Traceback (most recent call last):
  File "f:\MyProject\lab6.py", line 31, in <module>
    start, end = call.split('-')
ValueError: not enough values to unpack (expected 2, got 1)
```

Программа остановилась с ошибкой, которая вызвана тем, что введённая строка не содержит дефиса, который является разделителем для функции `split()`.

Для сообщения «123-321» получим уже вторую ошибку.

```
Traceback (most recent call last):
  File "f:\MyProject\lab6.py", line 32, in <module>
    move(start, end)
  File "f:\MyProject\lab5.py", line 7, in move
    block_type = mc.getBlock(MineToChess[start])
```

В данном примере ошибка вызвана тем, что словарь `MineToChess` не содержит записей с ключами 123-321.

Следовательно, необходимо научиться обрабатывать исключения. Для целей данной лабораторной работы будет достаточно отбрасывать все непредусмотренные в программе сообщения.



В Python для обработки исключений используется конструкция `try-except`. Она позволяет "ловить" и обрабатывать ошибки, которые возникают во время выполнения программы, чтобы предотвратить её экстренное завершение.



Задание 5. Дополните блок ветвления `else` предложенной ниже синтаксической конструкцией. Для блока 1 используйте уже имеющиеся инструкции, блок 2 должен выводить сообщение в игровой чат «Неверный ввод. Повторите ваш ход».



```
try:
    блок 1
except Exception as e:
    блок 2
```

Если задание выполнено корректно, то в игровом чате, при вводе непредусмотренных сообщений, будем всегда получать предупреждение.

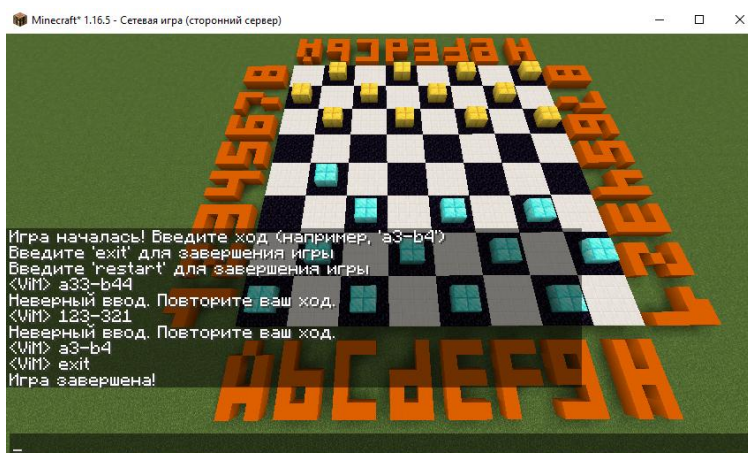


Рис. 2. Обработка исключений

Заключение: в шестой лабораторной работе мы познакомились с обработкой исключений и функцией `split()` в Python, разработали цикл для обработки игровых событий, реализовать управление игровым процессом через чат игры.

Проанализировав структуру лабораторных работ и их содержание, можно составить методические рекомендации использования разработанных

лабораторных работ для обучения школьников программированию на языке Python в компьютерной игре Minecraft.

Разработанные лабораторные работы лучше всего использовать для работы с учащимися 10-11, так как предполагается, что они имеют базовые знания и представления о синтаксисе Python, способны самостоятельно решать несложные задачи. В ходе выполнения лабораторных работ формируются умения работы с функциями, сама структура кода не является привычной процедурной, поэтому нужно уметь ориентироваться в нем.

Разработанные лабораторные работы можно использовать во внеурочной деятельности. Возможно применение работ и на уроках, если это не мешает выполнению учебного плана. Внеурочная деятельность позволяет изучить сложный материал простым и понятным языком в игровой форме. Кроме того, данную тему можно использовать в рамках проектной деятельности школьников.

В качестве формы организации деятельности обучающихся можно выбрать индивидуальную, фронтальную или групповую. Учащиеся могут выполнять задания самостоятельно или под руководством преподавателя. Лабораторные работы можно проводить в группах по 2-3 человека.

Индивидуальная форма работы подходит для обучающихся, которые нуждаются в индивидуальном подходе и которым сложно работать в группе. Она позволяет учителю придать индивидуальную направленность образовательному процессу, а также детально изучить знания и навыки каждого ученика.

Фронтальная форма работы позволяет учителю работать со всем классом, а обучающиеся могут проявить свои знания и умения перед всем классом.

Групповая форма работы предполагает сотрудничество обучающихся в небольших группах для выполнения заданий. Её преимущество заключается в том, что она способствует коллективному обмену знаниями и умениями, а также позволяет учителю работать с группами различной подготовки.

В начале урока учитель проводит инструктаж, в котором говорит о цели занятия и порядок выполнения лабораторной работы. Во время проведения урока учитель наблюдает за обучающимися, оказывает им помощь, корректирует их деятельность, контролирует правильность выполнения работы.

После выполнения каждого самостоятельного задания учащиеся подзывают учителя для проверки. Преподаватель проверяет правильность и помогает в редактировании программы если это необходимо.

Особое внимание стоит уделить окончанию урока. За несколько минут до завершения работы следует предупредить обучающихся, что время, отведённое на выполнение работы, заканчивается. Необходимо завершить оформление основных фрагментов кода, сохранить рабочий файл и привести в порядок рабочее место. Обязательно следует обсудить выполнение работы, сделать выводы.

Оценивание результата выполненных обучающимися лабораторных работ может производиться дифференцированно по пятибалльной шкале. Оценка «5» ставится, если учащийся выполняет работу в полном объёме с соблюдением необходимых требований. Оценка «4» ставится, если в лабораторной работе не выполнено одно задание. Оценка «3» ставится, если в лабораторной работе выполнено 1-2 задания.

Итоговый контроль реализуется в форме защиты итоговых проектов. Защита итогового проекта предполагает публичное выступление учащегося с демонстрацией результатов работы программы и презентации, содержащей аргументы в пользу практической ценности работы, анализ полученных результатов, оценку достигнутого прогресса в знаниях. Также обязательной частью отчёта являются ответы на вопросы одноклассников. В завершении учащиеся могут сыграть партию в шашки в разработанной программе.

Таким образом, разработанные лабораторные работы по программированию на языке Python в компьютерной игре Minecraft позволяют организовать учебный процесс получения знаний обучающимися

более интегрированным и современным. Лабораторные работы способствуют развитию алгоритмического мышления, учат мыслить логически, развивают воображение, креативность.

2.2. АПРОБАЦИЯ РАЗРАБОТАННЫХ МАТЕРИАЛОВ МЕТОДОМ ЭКСПЕРТНЫХ ОЦЕНОК.

Апробация разработанных лабораторных работ проводилась методом экспертных оценок. Экспертами выступали студенты Уральского государственного педагогического университета Института математики, информатики и информационных технологий, в количестве 15 человек. Экспертам были представлены лабораторные работы. Оценить разработанные материалы эксперты могли, отвечая на заранее подготовленные вопросы.

Целью апробации было выявление целесообразности применения лабораторных для обучения школьников программированию на языке Python в компьютерной игре Minecraft. Экспертам были заданы следующие вопросы:

1. Соответствуют ли разработанные лабораторные работы выбранному разделу «Алгоритмизация и программирование»?
 - а. полностью соответствуют;
 - б. частично соответствуют;
 - с. не соответствуют.
2. Доступно ли изложен материал в разработанные лабораторных работах?
 - а. да;
 - б. частично;
 - с. нет.
3. Насколько интересны разработанные лабораторные работы?
 - а. очень интересны;

- b. интересны;
 - c. не интересны.
- 4. Насколько эффективны разработанные лабораторные работы для внедрения в образовательный процесс?
 - a. эффективны;
 - b. частично эффективны;
 - c. не эффективны.
- 5. Соответствуют ли разработанные лабораторные работы требованиям ФГОС к формированию УУД?
 - a. полностью соответствуют;
 - b. частично соответствуют;
 - c. не соответствуют.
- 6. Как Вы считаете, способны ли учащиеся самостоятельно выполнить все задания?
 - a. да, способны;
 - b. не уверен;
 - c. нет, не способны.
- 7. Стали бы вы использовать разработанные лабораторные работы в своей педагогической деятельности?
 - a. использовал(а) бы полностью;
 - b. использовал(а) бы частично;
 - c. не использовал(а) бы.

Результаты апробации представлены в виде диаграмм, сформированным по результатам обработки экспертных оценок.

1. Соответствуют ли разработанные лабораторные работы выбранному разделу «Алгоритмизация и программирование»?

15 ответов

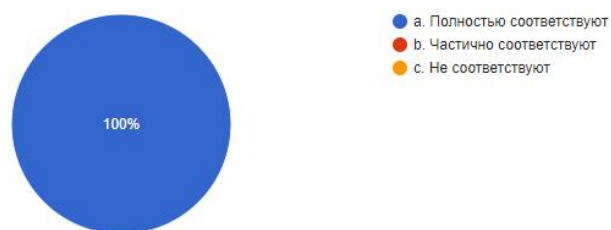


Рис. 2. Вопрос 1

2. Доступно ли изложен материал в разработанные лабораторных работах?

15 ответов

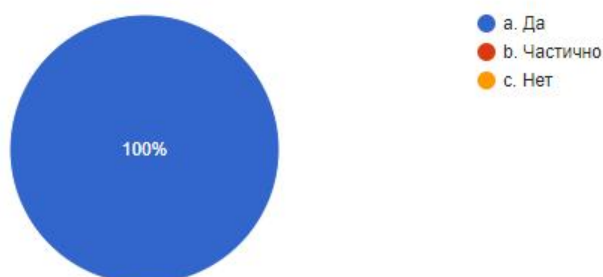


Рис. 3. Вопрос 2

3. Насколько интересны разработанные лабораторные работы?

15 ответов

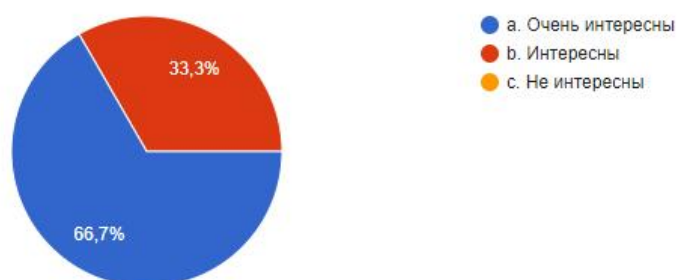


Рис. 4. Вопрос 3

4. Насколько эффективны разработанные лабораторные работы для внедрения в образовательный процесс?

15 ответов

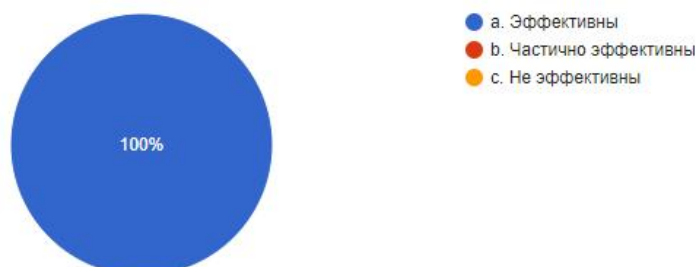


Рис. 5. Вопрос 4

5. Соответствуют ли разработанные лабораторные работы требованиям ФГОС к формированию УУД

15 ответов

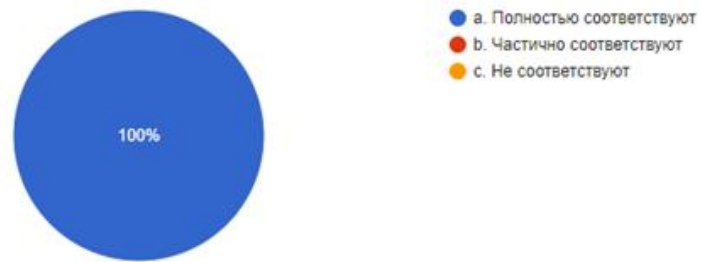


Рис. 6. Вопрос 5

6. Как Вы считаете, способны ли учащиеся самостоятельно выполнить все задания?

15 ответов

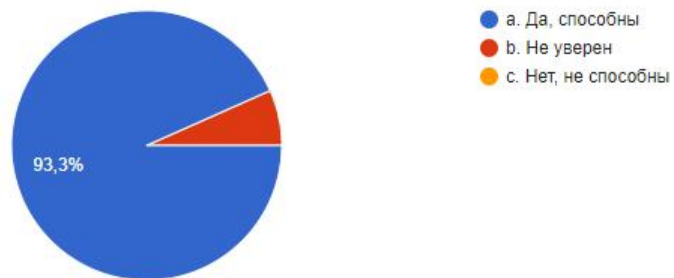


Рис. 7. Вопрос 6

7. Стали бы вы использовать разработанные лабораторные работы в своей педагогической деятельности?

15 ответов

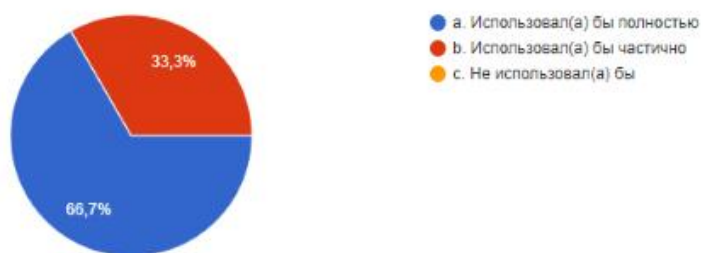


Рис. 8. Вопрос 7

Таким образом, апробация подтвердила возможность и целесообразность использования разработанных лабораторных работ при обучении школьников программированию на языке Python в компьютерной игре Minecraft.

ЗАКЛЮЧЕНИЕ

Подводя итоги проведённого исследования можно сделать следующие выводы:

1. В ходе анализа литературных источников были выявлены причины, по которым детям нравятся компьютерные игры, рассмотрены примеры компьютерных игр, которые применяются при обучении школьников программированию.

2. Были рассмотрены различные компьютерные игры для внедрения обучения программированию на языке Python, выявлены преимущества Python как языка программирования, а также обоснована возможность и целесообразность использования игры Minecraft для обучения программированию на Python в процессе изучения информатики в современной школе.

3. В ходе анализа литературных источников было выделено понятие лабораторная работа, рассмотрены различные основания для их классификации, проанализирован опыт использования лабораторных работ при обучении школьников информатике.

4. Были разработаны 6 лабораторных работ. Представлены особенности их проведения, цель, структура и содержание каждой.

5. Сформулированы методические рекомендации по применению разработанных лабораторных работ в учебном процессе, проведена апробация разработанных материалов.

Таким образом, можно считать, что задачи исследования выполнены, цель достигнута.

ЛИТЕРАТУРА

1. CodeCombat – Игра для изучения Python и JavaScript / CodeCombat.
– URL: <https://codecombat.com> (дата обращения: 29.03.2024).
2. Papert S. MINDSTORMS: Children, Computers, and Powerful Ideas. ISBN: 0-465-04627-4 изд. New York: Basic Books, Inc., Publishers, 1980. 230 с.
3. Андрусенко Евгений Юрьевич Формирование учебно-познавательных компетенций учащихся на лабораторных работах по информатике и ИКТ // ИСОМ. 2015. №1. URL: <https://cyberleninka.ru/article/n/formirovanie-uchebno-poznavatelnyh-kompetentsiy-uchaschihsya-na-laboratornyh-rabotah-po-informatike-i-ikt> (дата обращения: 11.06.2024).
4. Бабенко В.В., Гольчевский Ю.В. Выбор языков программирования и средств проектирования для обучения специалистов по направлению «Прикладная информатика» // Прикладная информатика. – 2013. – №4 (46). – С. 1-5.
5. Бадмаева Н. Ц. Мотивационные механизмы развития общих умственных способностей. // Вестник Бурятского государственного университета. Серия «Психология», 2005. - С.126-131.
6. Босова, Л. Л. Информатика. 7-9 классы. Компьютерный практикум / Л. Л. Босова, А. Ю. Босова, Н. А. Аквилянов – М.: БИНОМ. Лаборатория знаний, 2021. - 192 с.
7. Герасимова Е.К., Гавриловская Н.В., Соболева Е.В., Суворова Т.Н. исследование условий включения элементов геймификации в изучение основ алгоритмизации и программирования для повышения качества образовательных результатов обучающихся // ПНиО. 2021. №4 (52). URL: <https://cyberleninka.ru/article/n/issledovanie-usloviy-vklyucheniya-elementov->

[geymifikatsii-v-izuchenie-osnov-algoritmizatsii-i-programmirovaniya-dlya-povysheniya](#) (дата обращения: 28.03.2024).

8. Глухов, В. А. Элементы геймификации в инструментарии современного преподавателя информатики для развития интереса обучающихся к изучению программирования / В. А. Глухов, Ю. В. Шилова // Актуальные вопросы развития профессионализма педагогов в современных условиях : сборник материалов Международной электронной научно-практической конференции : в 4 т., Донецк, 01–31 октября 2018 года / ГОУ ДПО «Донецкий республиканский институт дополнительного педагогического образования». Том 2. Часть 1. – Донецк: Истоки, 2018. – С. 48-58.

9. Городняя Л. В., Боярских А.С. О языке начального обучения параллельному программированию // Вестник НГУ. Серия: Информационные технологии. 2012. №2. URL: <https://cyberleninka.ru/article/n/o-yazyke-nachalnogo-obucheniya-parallelnomu-programmirovaniyu> (дата обращения: 23.03.2024).

10. Еркина С.Л. Современные образовательные технологии. URL: http://portal.tpu.ru/SHARED/k/KAV47/education/Tab2/pt_v_t.pdf (дата обращения 29. 03. 2024).

11. Игра Check iO URL: <https://py.checkio.org/> (дата: 23.03.2024).

12. Игра CodeMonkey URL: <https://www.codemonkey.com/ru/courses/> (дата обращения: 23.03.2024).

13. Игра Minecraft | Образовательное издание Minecraft / Официальный сайт Minecraft URL: <https://education.minecraft.net/enus/homepage> (дата обращения: 27.05.2024).

14. Игра Roblox URL: <https://create.roblox.com/docs/tutorials/first-experience> (дата обращения: 27.05.2024).

15. Индекс ТIOBE за май 2024 г. // ТIOBE URL: <https://www.tiobe.com/tiobe-index/> (дата обращения: 23.03.2024).

16. Климекова Е. Является ли Python подходящим языком программирования для преподавания программирования в средних школах? // Международный журнал информационных и коммуникационных технологий в образовании. – 2017. – № 4 (2). – С. 5-14.

17. Лабораторные работы по языку программирования Python // Инфоурок URL: <https://infourok.ru/laboratornye-raboty-po-yazykuprogramirovaniya-python-4575742.html> (дата обращения: 23.03.2024).

18. Методическая разработка "Курс лабораторных работ по программированию" // Инфоурок URL: <https://infourok.ru/metodicheskaya-razrabotka-kurs-laboratornih-rabot-poprogramirovaniyu-2815064.html> (дата обращения: 21.03.2024).

19. Никитин П. В., Горохова Р. И., Зайков А. С. Применение компьютерных игр как фактор повышения качества обучения информатике // ОТО. 2015. №3. URL: <https://cyberleninka.ru/article/n/primenenie-kompyuternyh-igr-kak-faktor-povysheniya-kachestva-obucheniya-informatike> (дата обращения: 23.03.2024).

20. Никонова, А. Э., Газейкина, А. И. Выпускная квалификационная работа Обучение школьников программированию на языке Python в компьютерной игре Minecraft // Электронная библиотека УРГПУ URL: <http://elar.uspu.ru/handle/uspu/18202> (дата обращения: 23.03.2024)

21. Педагогический энциклопедический словарь / главный редактор: Б. М. Бим-Бад. – 3-е издание, стереотипное. – Москва: Большая Российская Энциклопедия, 2009. – 527 с.

22. Приказ Министерства просвещения Российской Федерации "Об утверждении федерального государственного образовательного стандарта

основного общего образования" от 31.05.2021 № 287. - URL: <https://clck.ru/34cNWV> (дата обращения: 23.03.2024).

23. Программирование в среде ROBLOX // Сайт Государственного бюджетного общеобразовательного учреждения города Москвы «Школа № 1517» URL: <https://1517.mskobr.ru/edu-news/7612> (дата обращения: 23.03.2024).

24. Программирование на языке Python для школьников: Учебное пособие по изучению языка программирования Python / Л. Самыкбаева, А. Беяев, А. Палитаев, И. Ташиев, С. Маматов. – Фонд Сорос-Кыргызстан, 2019 – 84 с

25. Соболева Е. В., Караваев Н. Л., Перевозчикова М. С. Совершенствование содержания подготовки учителей к разработке и применению компьютерных игр в обучении // Вестник Новосибирского государственного педагогического университета. 2017. Т. 7. № 6. С. 54-70. URL: <http://doi.org/10.15293/2226-3365.1706.04>. (дата обращения: 28.03.2024).

26. Чадина, Е. Г. Использование игровых интернет-сервисов для изучения основ программирования в школе / Е. Г. Чадина // Электронное обучение в непрерывном образовании. – 2016. – № 1. – С. 1272-1276

27. Широкова, Е. А. Лабораторная работа как средство понимающего усвоения старшеклассниками понятий математического анализа / Е. А. Широкова – Текст : электронный // Известия Российского государственного педагогического университета им. А.И. Герцена. – 2008. – № 69. – С. 508-513. – URL: <https://elibrary.ru/item.asp?id=11838067> (дата обращения: 23.04.2024)

28. Ширшова, Т. А. Лабораторные работы как средство мотивации и активизации учебной деятельности учащихся / Т. А. Ширшова, Т. А. Полякова – Текст : электронный // Омский научный вестник. – 2015. – № 4(141). – С. 188-190. – URL: <https://elibrary.ru/item.asp?id=24254743> (дата

обращения: 23.04.2024).

29. Школа программирования для детей Coddy URL:
https://coddyschool.com/courses/python_i_javascript_igrovoe_programmirovanie_v_codecombat/ (дата обращения: 29.03.2024).

30. Школа программирования для детей PIXEL URL:
<https://clubpixel.ru/> (дата обращения: 29.03.2024).

ПРИЛОЖЕНИЯ

Лабораторная работа №1

Запуск и настройка программного обеспечения

Цель: Научиться запускать необходимое программное обеспечение. Знакомство с первой функцией библиотеки Minecraft Pi Edition API, mc.postToChat()

Перед началом обучения программированию на python в компьютерной игре Minecraft, необходимо установить следующее программное обеспечение:

- интерпретатор Python для вашей операционной системы;

<https://www.python.org/>

- среда разработки Visual Studio (или любая другая);

<https://code.visualstudio.com/>

- Java;

<https://www.java.com/ru/download/>

- ProgMine Launcher.

<https://progmine.ru/files/ProgMine-1.0.2.exe>

1. Запускаем на рабочем столе «ProgMine Server»



Рис. 1. Ярлык ProgMine Server

2. После запуска программы ожидайте создание локального сервера для Minecraft. Дождитесь сообщения «Timings Reset» в одной из консолей

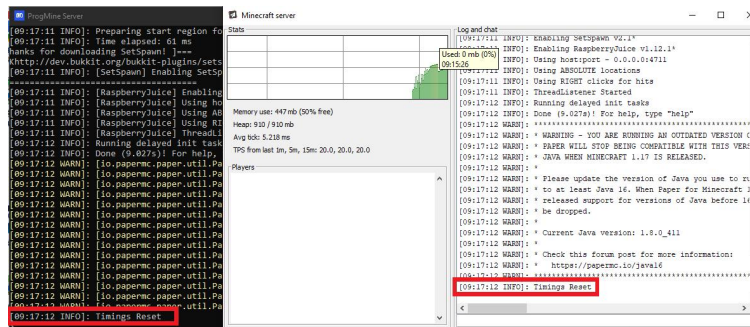


Рис. 2. Запуск локального сервера Minecraft

3. Далее нужно запустить «ProgMine Launcher»



Рис. 3. Ярлык ProgMine Launcher

4. В появившемся окне, последовательно пройдите процедуру авторизации: «Начать игру» → Укажите логин (любой) → «Выберите клиент» «Сборка ProgMine 1.16.5» → «Зайти в игру».

Дождитесь запуска игры Minecraft

5. Игра встретит нас привычным меню. Нажимаем «Сетевая игра» → «Local Server»

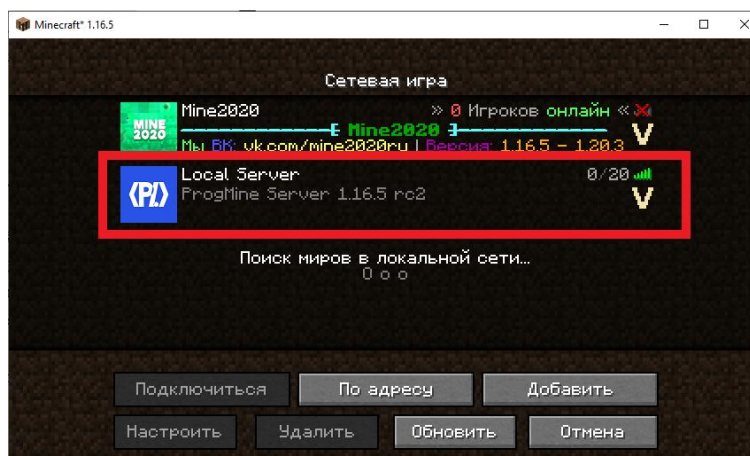


Рис. 4. Вход на локальный сервер ProgMine

Добро пожаловать в мир Minecraft!

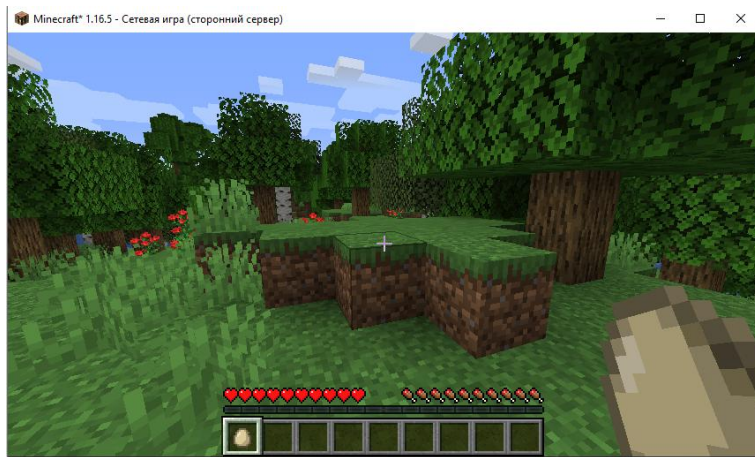


Рис. 5. Новый мир Minecraft

6. Запустите среду разработки Visual studio code



Рис. 6. Ярлык VSC

7. Интерфейс Visual studio code приветствует нас пустым файлом main.py, который является файлом исходного кода на языке программирования Python.
8. Напишите первые строки кода, которые позволят взаимодействовать с миром Minecraft:

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()
```

9. Далее вызовем простейшую функцию, которая позволит отправить сообщение в игровой чат:

```
mc.postToChat("Нас ждет большой успех!")
```

10. Запустите программу, нажав на стрелочку в верхнем правом углу экрана.

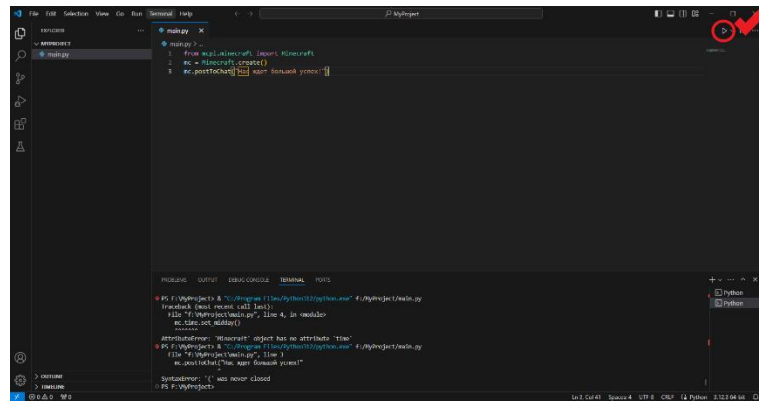


Рис. 7. Запуск программы Python

11. Проверьте результат выполнения программы

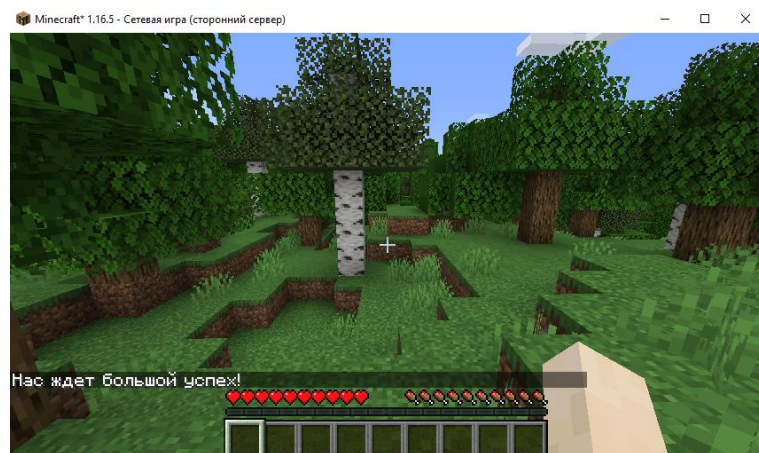


Рис. 8. Результат выполнения программы

Заключение: в рамках данной лабораторной работы был изучен процесс запуска необходимого программного обеспечения, а также рассмотрен пример простой команды, которая выводит текст в игровой чат.

Лабораторная работа №2

Система координат. Установка блоков

Цель: познакомиться с системой координат в Minecraft и научиться устанавливать блоки в определенных координатах с помощью языка программирования Python.



В Minecraft каждый блок имеет свои координаты, которые определяют его местоположение в игровом мире. Система координат Minecraft состоит из трех осей: X, Y и Z.

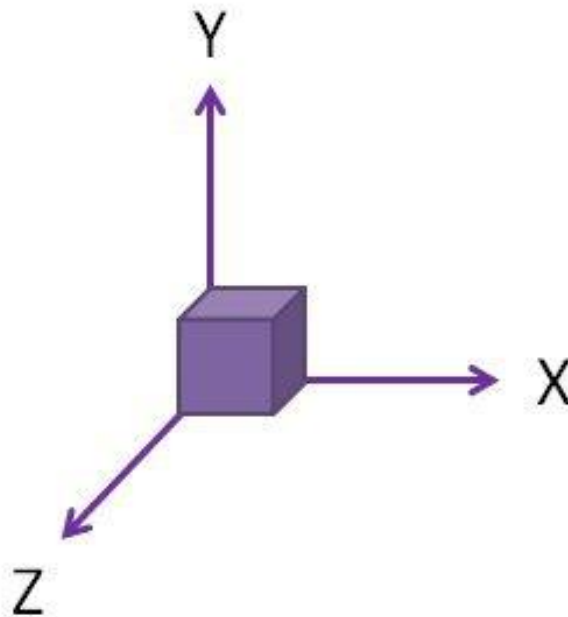


Рис. 1. Трехмерная система координат

Ось X идет вправо и влево, от центра игрока. Ось Y определяет высоту блока. Ось Z идет вперед и назад, от центра игрока.

Например, если игрок находится в центре игрового мира, т.е. его координаты равны (0, 0, 0), то блок с координатами (10, 5, 20) будет находиться на расстоянии 10 блоков вправо от центра игрока, на высоте 5 блоков над землей и на расстоянии 20 блоков вперед от центра игрока.

5. Запустите последовательно ProgMine Server → ProgMine launcher → Visual Code Studuo (см. Лаб.1)



Узнать свои координаты в игре Minecraft можно, нажав клавишу «F3»

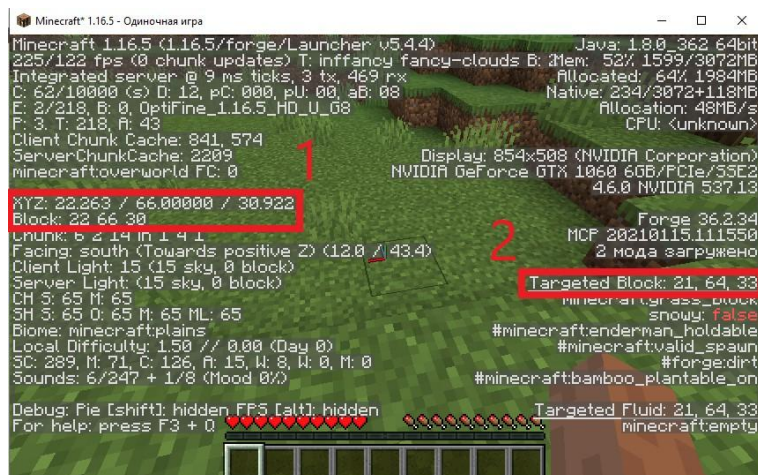


Рис. 2. Просмотр координат в Minecraft

На скриншоте красным контуром выделены координаты, которые предлагается использовать при взаимодействии с миром Minecraft через Python.

- «XYZ:» – Текущие координаты игрока в игровом мире. Точные координаты персонажа равны (22.263, 66, 30.922)
- «Block:» - Также показывают координаты игрока, но с привязкой к блоку, на котором игрок стоит. (22, 66, 30)
- «Targeted block:» - Координаты блока, на который смотрит игрок. Игрок смотрит на блок с координатами (21, 64, 33)



Обратите внимание, что координаты «XYZ:» и «Block:» практически равны. Это значит, что за координаты игрока в игровом мире можно использовать значение «Block:», то есть **координаты игрока в игровом мире (22, 66, 30), а координаты блока под игроком (22, 65, 30).**



Установить блок в мире Minecraft поможет команда

```
mc.setBlock(X, Y, Z, Type)
```

X,Y,Z – это координаты по которым размещаем блок. Type – тип блока, который размещаем. Всего в игре Minecraft существует 255 различных блоков, включая камень, золото, древесину (9 деревьев существует в игре).



Полный перечень блоков и их id вы можете найти на сайте <https://minecraft-ids.grahamedgecombe.com/>



Задание 1. Найдите ровную поверхность 5x5 блоков. Установите на землю обсидиановый блок (Type=49) недалеко от себя.

Вместо координат X Y Z укажите значения координат, куда хотите установить блок. Используйте значения координат «Targeted block». Чтоб установить обсидиановый блок сверху над «Targeted block», необходимо прибавить к Y единицу.



Рис. 3. Пример установки блока



Задание 2. Используя поставленный блок, постройте стену из обсидиана шириной 4 блока и высотой 5 блоков, как показано на рисунке.

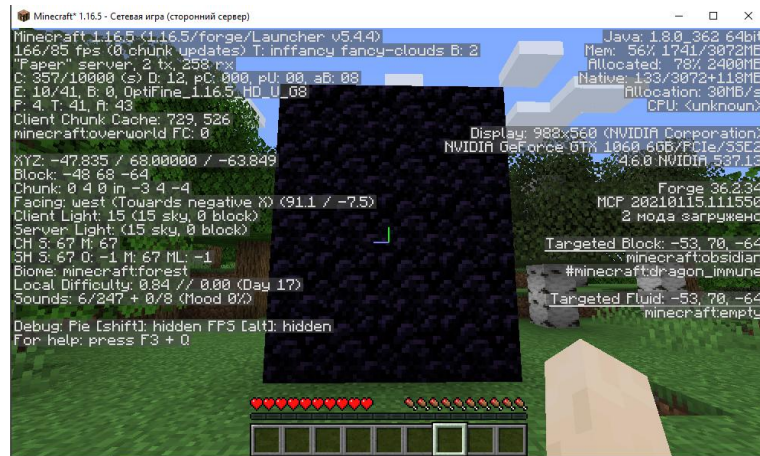


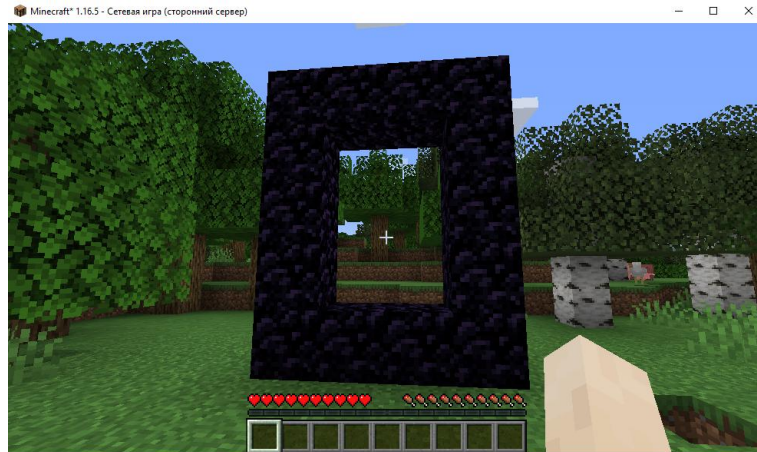
Рис. 4. Стена из обсидиана



Чтоб удалить блок из мира Minecraft, нужно просто задать ему Type=0, что означает воздух



Задание 3. Удалите лишние блоки обсидиана, чтоб у вас образовалась рамка.



Задание 4. На любой блок внутри рамки установите блок огня Type=51

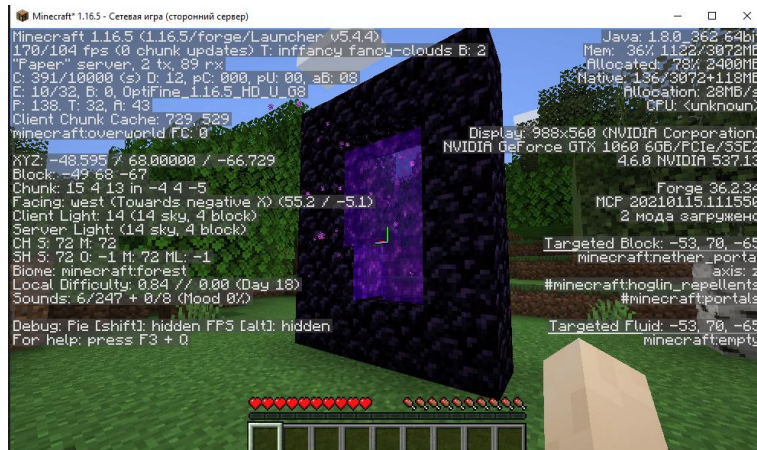


Рис. 5. Портал

Если все выполнено верно, то у вас получился отличный портал!

Познакомимся со следующей командой

mc.setBlocks(X1, Y1, Z1, X2, Y2, Z2, Type)

Данная команда устанавливает блоки одного типа, где X1, Y1, Z1 координаты начала, а X2, Y2, Z2 координаты конца диапазона, в который будут размещены блоки.



Задание 5. Постройте портал снова, используя команду **mc.setBlocks()**

Заключение: во второй лабораторной работе мы познакомились с системой координат в Minecraft, изучили основные функции взаимодействия с блоками: `mc.setBlock()` и `mc.setBlocks()`, научились не только устанавливать,

но и удалять блоки, а также построили первый портал в мире Minecraft с помощью программирования на Python.

Лабораторная работа №3

Создание шахматной доски

Цель: Научиться создавать сложные структуры из блоков в игровом мире Minecraft, познакомиться с вложенными циклами и функциями в Python.

Создайте файл lab3.py в своей папке. Откройте его с помощью VSC.



Шахматная доска - это классическое игровое поле для шахмат/шашек, состоящее из квадратных клеток, укладываемых в 8 рядов и 8 столбцов. Каждая клетка доски имеет уникальные координаты, которые обычно обозначаются буквами от "a" до "h" для столбцов и цифрами от 1 до 8 для рядов. Например, клетка в нижнем левом углу доски обозначается как "a1", а клетка в верхнем правом углу - как "h8".

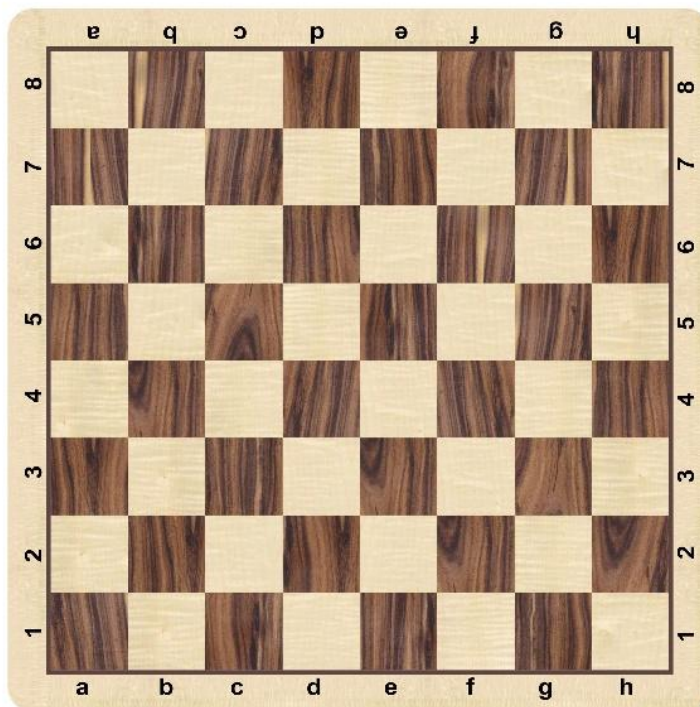


Рис. 1. Шахматная доска

Для того, чтобы научиться создавать шахматную доску в игре Minecraft с помощью Python, необходимо:

4. Уметь создавать плоские фигуры из заданных блоков;
5. уметь использовать циклы и ветвление для автоматизации задач;
6. уметь определять четность/нечетность заданного числа.

Система координат Minecraft состоит из трех осей: X, Y и Z (см Лаб.2).



определим за точку отсчета или «начальный блок» для построения шахматной доски блок (0;3;0).



```
mc.player.setPos(0, 4, 0)
```

Для начала, телепортируемся в заданные координаты.



Не забудь, что если необходимо оказаться над желаемым блоком, то необходимо добавить единицу к координате Y!

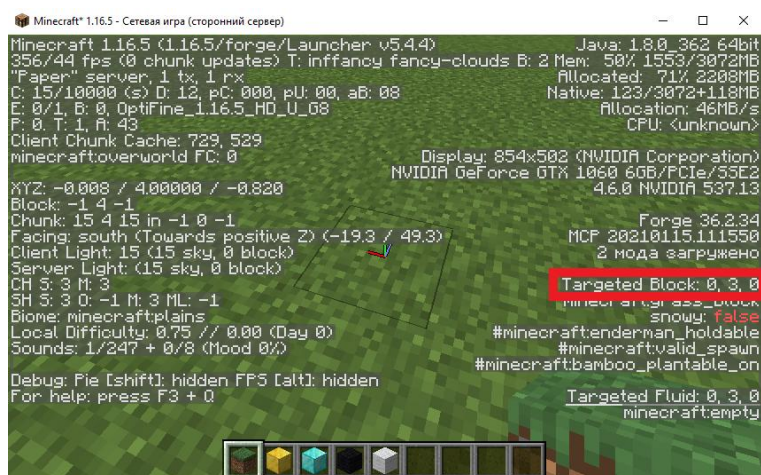


Рис. 2. Точка отсчета



Задание 1. Используя `mc.setBlock()`, постройте квадрат 3 на 3, как показано на рисунке 3.

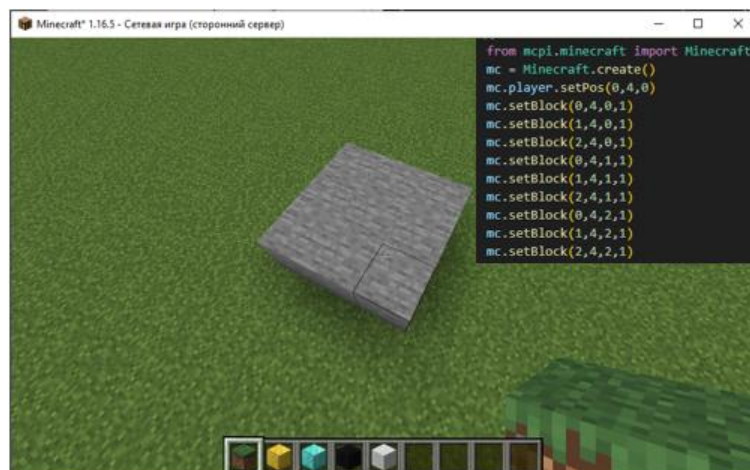


Рис. 3. Квадрат 3x3 из камня

После демонстрации учителю построенной фигуры, вручную сломайте все установленные блоки.



В данном примере каждая строка кода `mc.setBlock()` используется для установки блока в определенных координатах. Однако эти координаты увеличиваются на 1 для каждого блока.

Вместо того чтобы вручную указывать каждый набор координат, необходимо использовать циклы для автоматического увеличения значений координат.

Обратим внимание, что меняются только координаты X и Z, значит потребуется два счетчика.



```
for i in range(3):  
    for j in range(3):  
        mc.setBlock(x + i, y, z + j, 1)
```



Цикл, находящийся внутри другого цикла, называется вложенным!

Цикл «for i in range(3)» и «for j in range(3)» выполнится ровно три раза, при этом счетчики i и j будут принимать значения от 0 до 2!



Задание 2. Объявите 3 переменные «x», «y», «z» и задайте им значение координат «начального блока». Постройте фигуру из задания 1, используя цикл, как показано на рисунке.

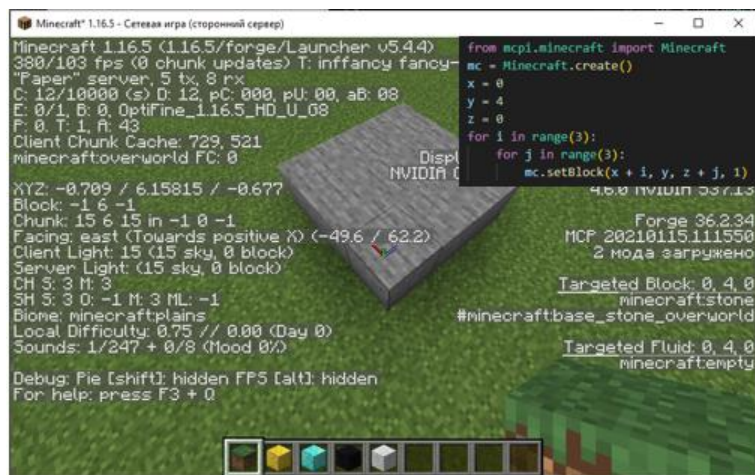


Рис. 4. Квадрат 3x3 из камня, построенный с помощью цикла

После демонстрации учителю построенной фигуры, вручную сломайте все установленные блоки.



В задании 1 и задании 2 для очистки поля каждый блок фигуры предлагалось сломать вручную. Квадрат 3x3 состоит всего из 9 блоков. Если же потребуется очистить поле от квадрата 8x8, то придется ломать уже целых 64 блока! Таким образом, уже сейчас требуется отладочный инструмент, позволяющий очистить плоскость заданной величины от всех блоков.

Измени тип блока, используемого в цикле, на воздух (0). В результате выполнения цикла, увидим, что поле очистилось (Рис. 5).

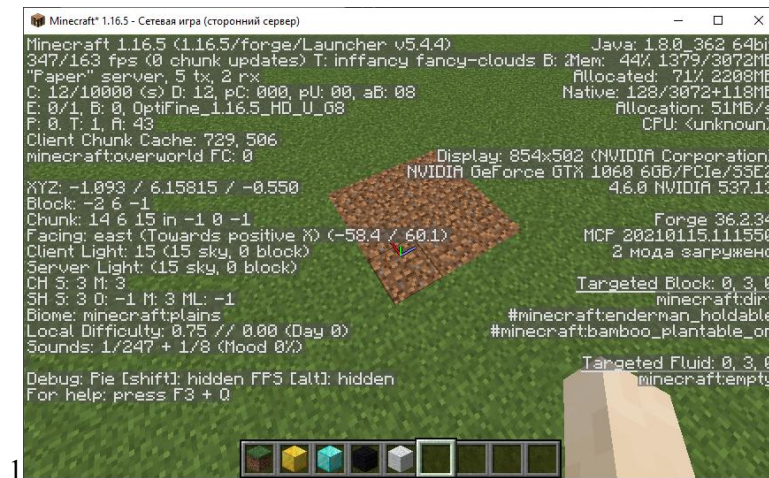


Рис. 5. Квадрат 3x3 из воздуха

В будущем предлагается использовать текущий код в качестве инструмента очистки игрового поля. Для этого создадим еще одну переменную «boardsize» и сохраним текущий код в качестве функции def clear(), где def – ключевое слово, которое определяет функцию с именем «clear», что значит «очистка».

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()
x = 0
y = 4
z = 0
razmerdoski=3

def clear():
    x = 0
    y = 4
    z = 0
    for i in range(razmerdoski):
        for j in range(razmerdoski):
            mc.setBlock(x + i, y, z + j, 0)

    for i in range(razmerdoski):
        for j in range(razmerdoski):
            mc.setBlock(x + i, y, z + j, 1)
```

Рис. 6. Определении функции «clear»

Дополните ваш код недостающими фрагментами и попробуйте его



запустить.

Функция - это блок кода, который выполняет определенную задачу и может быть вызван из других частей программы. Функции позволяют структурировать код, делают его более читаемым и упрощают повторное использование.

Заметим, что после выполнения кода (Рис. 6) вновь получим квадрат 3x3 из камня. Это связано с тем, что функция «clear()» не была вызвана в коде, соответственно алгоритм не работает.

Чтоб выполнить код, содержащийся внутри функции «clear()», необходимо добавить в конец кода строку:



```
clear() - вызов функции
```

Проверьте работу функции clear(). После успешной проверки закомментируйте строку, используя символ «#».



```
#clear() - комментарий
```

Закомментированные строки кода не выполняются при запуске программы!

Таким образом, была создана функцию, которая по заданному размеру доски (boardsize), начиная с «начального блока», заменяет все блоки на воздух, а также научились вызывать эту функцию в коде программы. В дальнейшем будем использовать функцию clear() для очистки игрового поля от блоков.



Перейдем к самой задаче создания шахматной доски. В качестве белых клеток шахматного поля предлагается использовать блок кварца (тип 155), а в качестве черных клеток предлагается использовать блок обсидиана (тип 49).

Создайте квадрат размером 8x8 из блоков кварца.

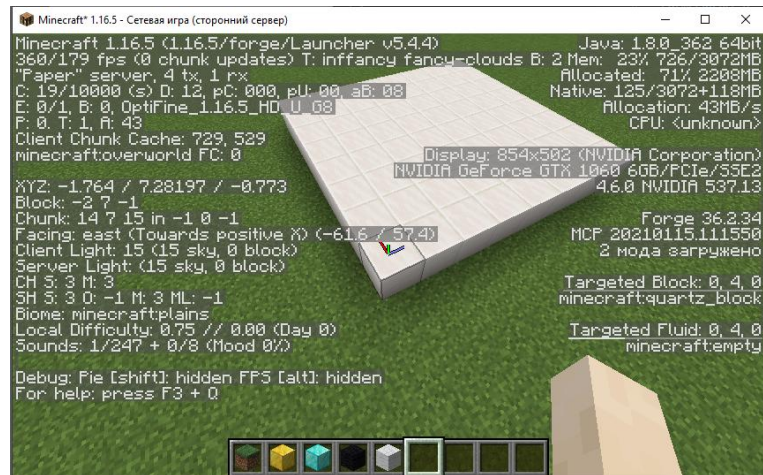


Рис. 7. Квадрат 8x8 из кварца

Вооружившись блоком обсидиана, постройте первые два ряда шахматной доски, как показано на рисунке 8.

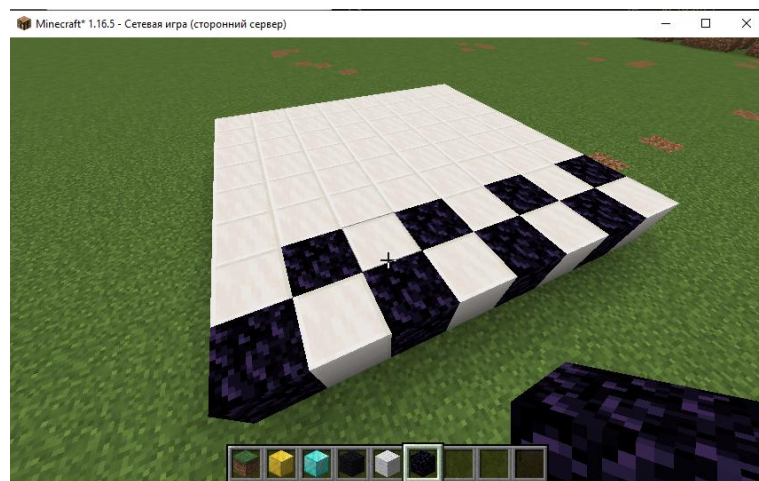


Рис. 8. Первые два ряда шахматной доски

Запишите в тетрадь координаты всех черных блоков первого ряда и второго ряда.

Заметим, что сумма координат $x+z$ для каждого черного блока обязательно является числом четным. Значит, при создании шахматной доски средствами Python, требуется уметь определять четность/нечетность заданного числа.



Из математики знаем, что число является четным в том случае, если оно без остатка делится на 2. В Python оператор для нахождения остатка от деления - это «%». Он возвращает остаток от деления одного числа на другое.

```
result = 10%3
print(result)
```

При выполнении данного кода, консоль выведет 1.



Будем определять цвет блока относительно «начального блока». Цикл будет проходить по рядам и столбцам шахматной доски, определяя для каждой игровой клетки цвет по сумме индексов i и j . Если сумма $i+j$ четная, клетка черная, иначе белая.

Дополните код, с учетом вышесказанного. И запустите программу.

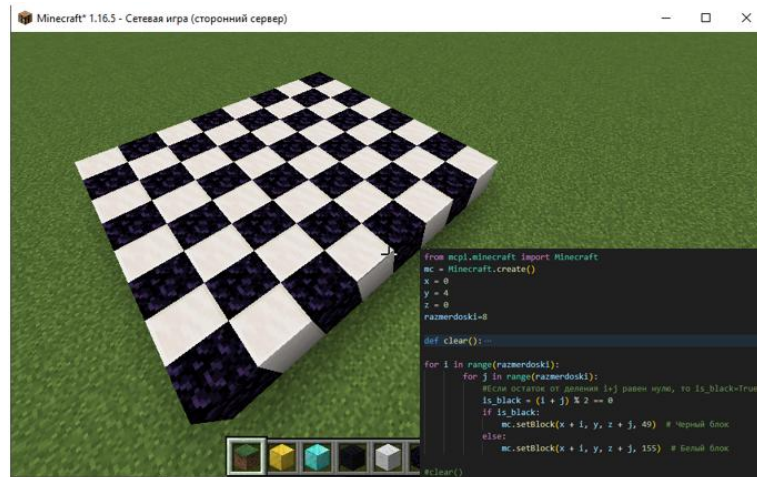


Рис. 9. Алгоритм построения шахматной доски 8x8



Задание 3. Используя текущий код, постройте конфигурации «шахматных полей» представленные на рисунке 10. Все конфигурации должны быть созданы в игровом мире одновременно.

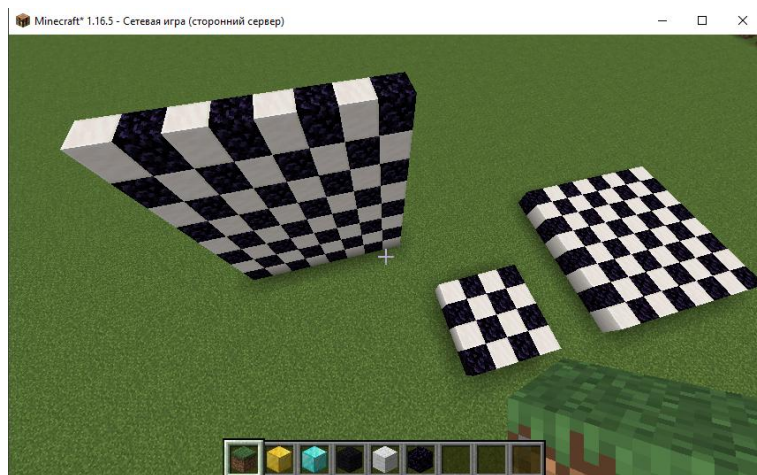


Рис. 10. Конфигурации шахматных полей



Задание 4. Очистите игровой мир, изменив и вызвав функцию «Clear». После демонстрации учителю работы измененной функции «clear», используя сочетание клавиш Ctrl+Z, верните код к исходному состоянию (Рис. 9).



Теперь, когда была успешно построено шахматное поле размером 8x8,

каждая клетка которого представлена одним блоком, важно задуматься о визуальной и функциональной составляющей.

В реальных шахматных досках клетки значительно крупнее, и они позволяют свободно разместить фигуры так, чтобы они имели достаточно места для перемещения. В Minecraft из-за особенностей графики и стиля игры, клетки, состоящие из одного блока, не позволяют перейти от шашек к шахматам, так же в текущем масштабе невозможно подписать столбцы и ряды шахматной доски (Рис. 1) и перейти от трехмерной системы координат к двумерной.

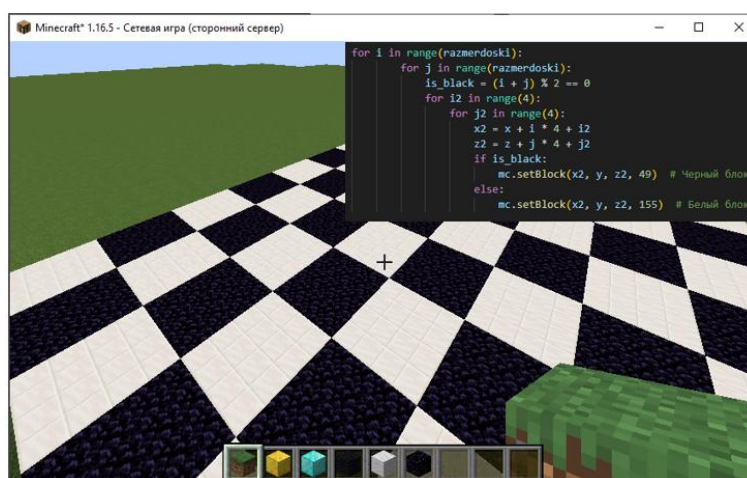


Рис. 11. Шахматная доска 8x8 с размером игровой клетки 4x4 блока

Продолжим создание шахматной доски, где каждая игровая клетка будет представлять собой область размером 4x4 блока. Для этого добавим внутренние циклы с переменными $i2$ и $j2$, которые будут использоваться для прохода по каждой клетке внутри 4x4 блока. Переменные $x2$ и $z2$ будут вычислять координаты каждой клетки внутри блока игровой доски.

Дополните ваш код (Рис. 11) и запустите.



Задание 5. Используя любой яркий блок, постройте буквы и цифры для шахматной доски для каждого ряда и столбца соответственно (Рис. 1)

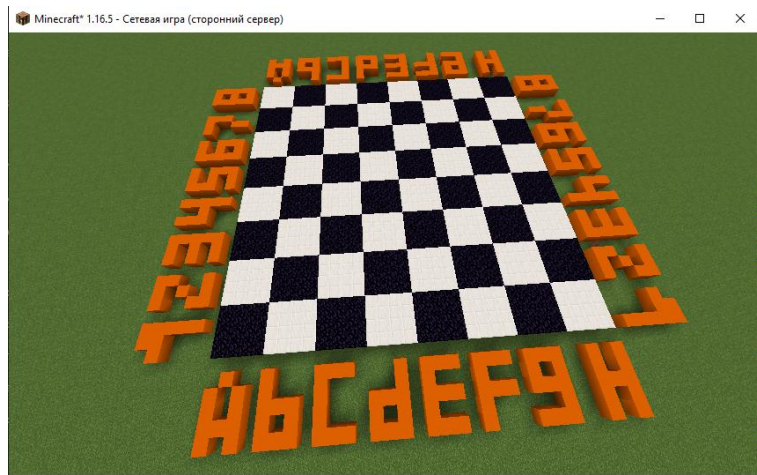


Рис. 12. Шахматная доска с подписанными рядами и столбцами



Задание 6. Определите функцию: `def create_chessboard(x,y,z, boardsize)`. Сохраните основной код в созданную функцию.

Продемонстрируйте последовательный вызов функций:



```
clear()
create_chessboard(x, y, z, boardsize)
```

Заключение: в третьей лабораторной работе мы познакомились с вложенными циклами и функциями в Python, разработали функции `def clear()` и `def create_chessboard(x, y, z, boardsize)`, позволяющие соответственно очищать заданную область от блоков и строить шахматную доску с заданными параметрами.

Лабораторная работа №4

Размещение шашек на доске

Цель: Научиться устанавливать шашки на игровой доске, разработать стартовую конфигурацию игровой доски для игры в шашки, познакомиться со словарями, кортежами и списками в Python.

Создайте файл lab4.py в своей папке. Откройте его с помощью VSC. Импортируйте в него все функции, разработанные в предыдущей лабораторной работе.



```
from lab3 import *
```



На стандартной доске для игры в шашки (Рис. 1) располагается 24 шашки: 12 белых и 12 черных. Они размещаются на черных клетках доски

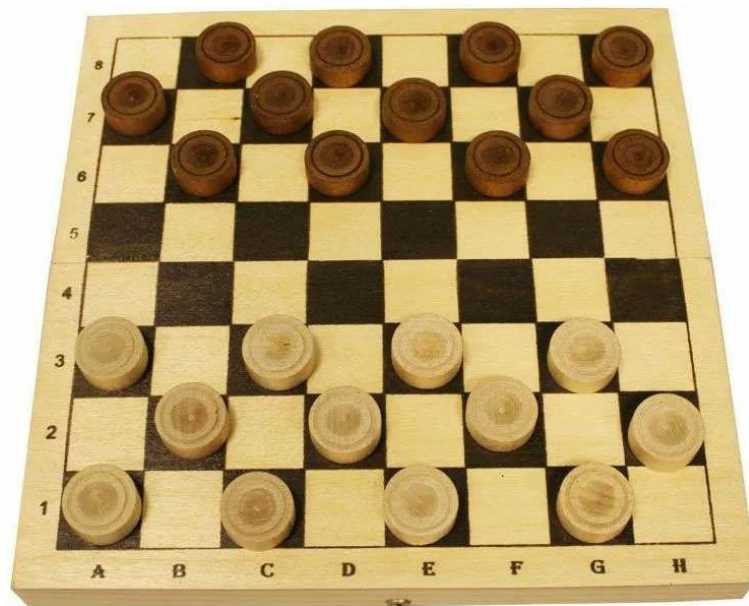


Рис. 1. Шашки

В качестве шашек предлагается использовать квадраты 2x2 из алмазных (тип 41) и золотых (тип 57) блоков.



```
for i in range(29, 29 + 2):  
    for j in range(25, 25 + 2):  
        mc.setBlock(i, 5, j, 41)
```

Создадим золотую шашку в игровой клетке b1. За «нулевой блок», от которого алгоритм начнет построение, возьмем координаты (29,5,25).

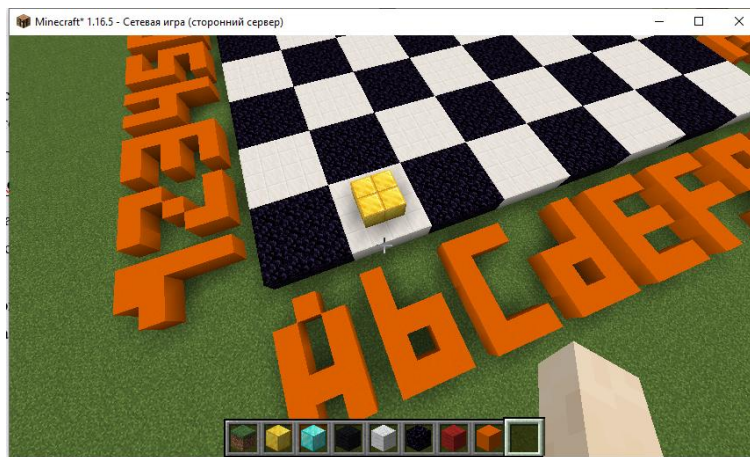


Рис. 2. Создание первой шашки



Задание 1. Используя цикл для создания шашки, расположите шашки как показано ниже.



После демонстрации выполненного задания, очистите доску функцией `clear()`, координата `y=5`.



Заметим, что все шашки будут располагаться в одной плоскости, значит использовать координату `Y` не потребуется. Следовательно, необходимо перейти от трехмерной системы координат Minecraft к двумерной системе координат, которая используется в шахках.

Создадим пустой словарь `MineToChess`.



```
MineToChess = {}
```



Словарь в Python - это структура данных, которая представляет собой коллекцию пар ключ-значение. Каждый элемент словаря состоит из ключа и соответствующего ему значения.

```
Имя словаря = {"ключ1": "Привет, словарь!"}
print(Имя словаря ["ключ1"])
```

В результате работы данного кода, консоль выведет «Привет, словарь!».



В качестве ключей для элементов словаря **MineToChess** будем использовать координаты игровых клеток доски (a1;b1 и т.д.), а в качестве значений будем передавать координаты Minecraft «нулевых блоков» для каждой шашки в виде кортежа.



Кортеж в Python — это неизменяемая упорядоченная коллекция объектов.



Добавим в словарь первый элемент.

```
MineToChess = { "a1": (29, 5, 29)}
```

Теперь можем получить координаты x y z для установки шашки, обратившись к словарю.

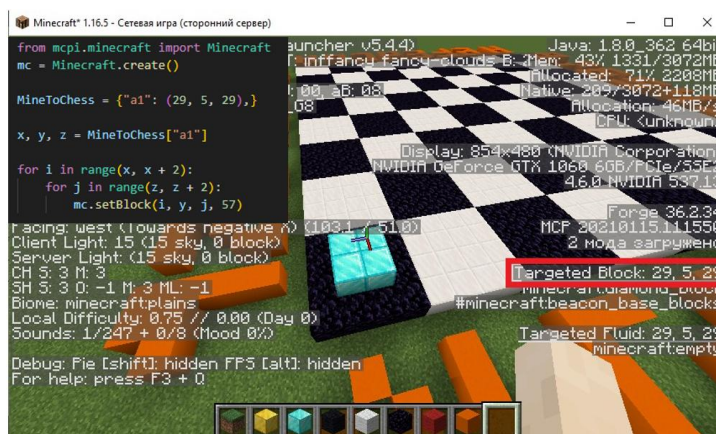


Рис. 2. Создание шашки с использованием словаря

Так, из словаря **MineToChess** по ключу «a1» передаем координаты x=29; y=5; z=29, и используем ранее приведенный цикл для создания шашки.



Задание 2. Заполните словарь **MineToChess** от ключа a1 до h8, как показано ниже.

```
MineToChess = {"a1": (29, 5, 29),
               "b1": (29, 5, 25),
               "c1": (29, 5, 21),
               "d1": (29, 5, 17),
               "e1": (29, 5, 13),
               "f1": (29, 5, 9),
               "g1": (29, 5, 5),
               "h1": (29, 5, 1),
               #и так далее
```

Рис. 3. Фрагмент словаря



Создадим функцию `def create_checker(key, block_type)`. В данную функцию будем передавать два аргумента: `key` и `block_type`.

```
1 from mcpi.minecraft import Minecraft
2 mc = Minecraft.create()
3 > MineToChess = {"a1": (29, 5, 29), ...
21
22 v def create_checker(key, block_type):
23     x, y, z = MineToChess[key]
24     for i in range(x, x + 2):
25         for j in range(z, z + 2):
26             mc.setBlock(i, y, j, block_type)
27
28 create_checker("a1", 57)
```

Рис. 4. Функция `create_checker`



Рассмотрим работу программы (Рис. 4):

1. Функция `create_checker` вызывается с аргументами "a1" и 57.
2. Из словаря `MineToChess` по ключу "a1" извлекаются координаты (29, 5, 29) и присваиваются переменным `x`, `y`, и `z` внутри функции.
3. По уже известному циклу, строится шашка из заданного типа блоков в указанных координатах.

Таким образом, успешно осуществили переход от трехмерной системы координат мира `Minecraft` к двумерной системе координат, используемой в шашках.



Задание 3. Используя функцию `create_checker`, расположите шашки как показано ниже



`DiamondPositions = []`



```
GoldPositions = []
```

Перейдем к созданию стартовой конфигурации. Создадим два списка **DiamondPositions** и **GoldPositions**.



Список в Python — это упорядоченная коллекция элементов, которая может содержать объекты любого типа данных, такие как числа, строки, другие списки и т. д. Списки очень похожи на кортежи, но их содержимое можно менять средствами Python прямо в процессе выполнения программы.



```
DiamondPositions = ["a1", "a3"]  
GoldPositions = ["a7", "b6"]
```

Наполним списки координатами стартовых игровых клеток для алмазных и золотых шашек. Добавим элементы в списки:



Задание 4. Заполните списки **DiamondPositions** и **GoldPositions** до конца.



Рассмотрим синтаксическую конструкцию:

```
for i in DiamondPositions:
```

В данном случае, вместо использования функции `range()` для генерации последовательности чисел, будем передавать сам список `DiamondPositions`. Каждый элемент списка будет последовательно присвоен переменной `i` для каждой итерации цикла.



Задание 5. Используя предложенную выше конструкцию, составьте цикл, устанавливающий алмазные шашки на стартовые клетки, как показано ниже



Задание 6. Создайте функцию `start_game()`. В результате вызовы

функции должны получить готовую для игры в шашки конфигурацию игрового поля. Продемонстрируйте вызов разработанной функции.

Заключение: в четвертой лабораторной работе мы познакомились со словарями, кортежами и списками в Python, научились устанавливать шашки на игровое поле. С помощью словаря **MineToChess** от трехмерной системы координат мира Minecraft перешли к двумерной системе координат, используемой в шашках и разработали функцию **start_game()**, задающую стартовую конфигурацию расположения шашек.

Лабораторная работа №5

Реализация основных игровых механик

Цель: разработать функции для реализации перемещения и захвата шашек, познакомиться с оператором `return`, константой `none` и функцией `time.sleep()` в Python

Создайте файл `lab5.py` в своей папке. Откройте его с помощью VSC.

Импортируйте в него все функции, разработанные в предыдущей лабораторной работе.



```
from lab4 import *
```



Приведем алгоритм функции, которая позволит переместить шашку из `a3` в `b4`:

1. Передать в функцию координаты начального и конечного положения шашки (`a3,b4`);
2. определить из какого типа блоков состоит шашка на `a3`;
3. создать шашку в `b4`, из блоков, полученных в п.2;
4. заменить все блоки в `a3` на воздух = создать шашку из воздуха.

Для реализации 3 и 4 шага алгоритма можем использовать ранее разработанную функцию `create_checker(key, block_type)` (см. Лаб. №4).

Чтоб определить какого цвета шашка на заданных координатах используем следующую конструкцию:



```
block_type = mc.getBlock(MineToChess["a3"])
```

Рассмотрим ее синтаксис:

1. Переменной `block_type` присваивается значение функции `mc.getBlock(x,y,z)`, которая позволяет получить тип блока в определенных координатах мира Minecraft;
2. Функция `mc.getBlock(x,y,z)` в качестве аргументов на вход получает набор координат `x,y,z` из словаря `MineToChess` по ключу `a3`



Задание 1. Создайте функцию `move()`, принимающую аргументы `start` и `end`. Используя вышеперечисленные идеи, напишите для нее инструкции,

позволяющие перемещать шашки на доске. Продемонстрируйте



преподавателю поочередный вызов функций:

```
clear()
start_game()
move ("a3", "b4")
```

Если функция **move()** работает корректно в результате выполнения задания 1 увидим следующее состояние игрового поля:



Рис. 1. Результат работы функции **move()**



Задание 2. Используя функцию **move()**, переместите 8 шашек как показано на рисунке ниже



Реализация механики захвата шашки уже не является задачей столь же тривиальной. Рассмотрим алгоритм захвата шашки:

1. Перед тем как делать ход, нужно проверить, есть ли возможность срубить шашку соперника. Это можно сделать с помощью функции,

которая будет проверять, есть ли шашка противника между начальной и конечной позициями хода.

2. Если шашка противника обнаружена, необходимо удалить ее с игрового поля.
3. После захвата шашки, необходимо обновить игровое положение выбранной шашки, переместив ее на новую позицию



Заметим, что для реализации шага 2 и 3 алгоритма можем использовать уже разработанные ранее функции **move()** и **create_checker()**.



В шашках при захвате пешки соперника зачастую приходится перепрыгивать через нее, и для этого нужно знать координаты клетки, через которую происходит прыжок. Нахождение средней точки требует арифметических операций с координатами, которые проще и естественнее выполнять в трехмерной системе координат.

В трехмерной системе координат x, y, z средняя точка ($middle_x$, $middle_y$, $middle_z$) между двумя точками ($start_x$, $start_y$, $start_z$) и (end_x , end_y , end_z) может быть найдена путем усреднения каждой координаты.



```
middle_x = (start_x + end_x) // 2
middle_y = (start_y + end_y) // 2
middle_z = (start_z + end_z) // 2
```

Начнем конструировать функцию **find_middle()**



```
def find_middle(start, end):
    start_x, start_y, start_z = MineToChess[start]
    end_x, end_y, end_z = MineToChess[end]
    middle_x = (start_x + end_x) // 2
    middle_y = (start_y + end_y) // 2
    middle_z = (start_z + end_z) // 2
```

Функция **find_middle()** принимает два аргумента **start** и **end**, представляющие собой строковые значения координат игрового поля. Далее извлекаем трехмерные координаты из словаря **MineToChess** для двух заданных точек на шахматной доске **start** и **end**, присваивая их переменным **start_x**, **start_y**, **start_z** и **end_x**, **end_y**, **end_z**. Вычисляем средние координаты (**middle_x**, **middle_y**, **middle_z**).



Заметим, если полученные трехмерные координаты средней точки содержатся в словаре **MineToChess**, то существует игровая клетка на доске, являющаяся средней клеткой относительно координат "start" и "end". Следовательно, функция **find_middle()** должна возвращать координаты игровой клетки.



```
def move(start, end):
    block_type = mc.getBlock(MineToChess[start])
    create_checker(end, block_type)
    create_checker(start, 0)
    if find_middle(start, end) != None:
        create_checker(find_middle(start, end), 0)
```



None — это специальная константа в Python, которая представляет отсутствие значения или нулевое значение.

В конечном счете, будет удобно возвращать результат работы функции **find_middle()** в качестве координаты игрового поля.



```
def find_middle(start, end):
    start_x, start_y, start_z = MineToChess[start]
    end_x, end_y, end_z = MineToChess[end]
    middle_x = (start_x + end_x) // 2
    middle_y = (start_y + end_y) // 2
    middle_z = (start_z + end_z) // 2

    for key, value in MineToChess.items():
        if value == (middle_x, middle_y, middle_z):
            return key
```

Дополним функцию циклом, который осуществляет сравнение полученных координат промежуточной клетки с каждым элементом словаря **MineToChess**. Для получения всех пар ключ-значение воспользуемся встроенной в Python функцией для работ со словарями **MineToChess.items()**. Если в телке цикла обнаружено совпадение, то функция **find_middle()** при вызове должна возвращать ключ/координату игровой клетки, для этого используем оператор **return key**.



Оператор **return** используется внутри функции для выхода из нее и возврата значения. Если не возвращаем значение явно, то автоматически возвращается **None**.



Задание 3. Добавьте приведенный выше код в файл программы.

Импортируйте модуль `time` и продемонстрируйте учителю поочередный вызов функций.



```
clear()
start_game()
move ('a3','b4')
time.sleep(1)
move ('d6','c5')
time.sleep(1)
move ('b4','d6')
time.sleep(1)
```

Если все фрагменты кода перенесены корректно, то в результате выполнения задания 3, в режиме реального времени будем наблюдать 3 хода с интервалом в одну секунду:

1. Ход 1: 'a3', 'b4'
2. Ход 2: 'd6', 'c5'
3. Рубим золотую шашку 3: 'b4', 'd6'

Конечное состояние игрового поля приведено на рисунке 2



Рис. 2. Результат работы выполнения задания 3

Заключение: в пятой лабораторной работе мы познакомились с оператором `return`, константой `None` и функцией `time.sleep()` в Python, разработали функции `move(start, end )` и `find_middle(start, end)`, реализующие перемещение и захват шашек на игровом поле.

Лабораторная работа №6

Реализация управления игровым процессом через игровой чат

Цель: разработать цикл для обработки игровых событий, реализовать управление игровым процессом через чат игры, познакомиться с обработкой исключений в Python и функцией **split()**.

Создайте файл lab6.py в своей папке. Откройте его с помощью VSC.

Импортируйте в него все функции, разработанные в предыдущей лабораторной работе.



```
from lab5 import *
```

Прежде чем приступить к реализации основного цикла для обработки игровых событий, необходимо вернуться к содержимому файлов lab3-lab5 и проверить состояние исходного кода. Данные файлы должны содержать либо строки создания переменных, либо функции, разработанные в Лаб. 3 – Лаб. 5



Задание 1. Убедитесь, что в файлах lab3-lab5 нет исполняемых инструкций. Продемонстрируйте учителю вызов функций **clear()** и **start_game()**



При запуске игры, необходимо получить в игровом чате приветственное сообщение, которое будет являться маркером, что программа работает корректно, а также небольшую инструкцию по управлению игровым процессом.



```
mc.postToChat("Игра началась!")  
mc.postToChat("Введите ход (например, 'a3-b4')")
```

Затем приведем доску к стартовой конфигурации для начала игры в шашки.



```
clear()  
start_game()
```

Для реализации управления игровым процессом через чат игры, необходимо научить программу «прослушивать» сообщения в игровом чате. Создадим метод **get_chat_message()**, который будет на протяжении всей игры отслеживать поступающие в чат сообщения.



```
def get_chat_message():
    while True:
        chats = mc.events.pollChatPosts()
        if chats:
            return chats[0].message
```

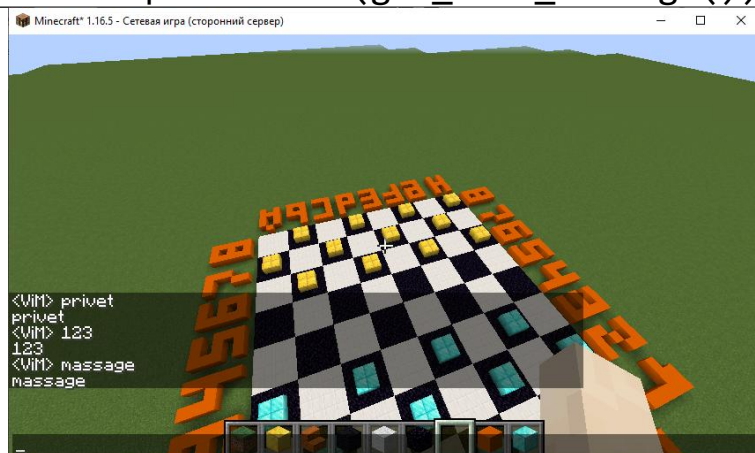
В теле цикла при работы программы создается переменная `chats`, которой присваивается значение функции, отслеживающей события в игровом чате `mc.events.pollChatPosts()`. Если значение данной функции ненулевое (не пустое), то функция `get_chat_message()` возвращает данное сообщение. Таким образом, функция будет всегда возвращать пойманный текст из игрового чата.



Задание 2. Продемонстрируйте учителю работу функции `get_chat_message()`. Отправьте в игровой чат 2-3 пойманных сообщений. Фрагмент кода для демонстрации и результат работы приведены на картинке ниже.



```
while True:
    mc.postToChat(get_chat_message())
```



Для реализации игрового цикла будем использовать цикл `while`, так как точное число повторений тела цикла неизвестно.



```
while True:
    call = get_chat_message()
    if call == "exit":
        mc.postToChat("Игра завершена!")
        break
```

Создадим в теле цикла переменную `call`, которая будет получать из функции `get_chat_message()` пойманные в ходе прослушки сообщения из

игрового чата. Если полученным сообщением является «exit», то цикл прерывается. Результат работы программы приведен на рисунке 1. Таким образом, можем реализовать команды, управляющие игровыми событиями.

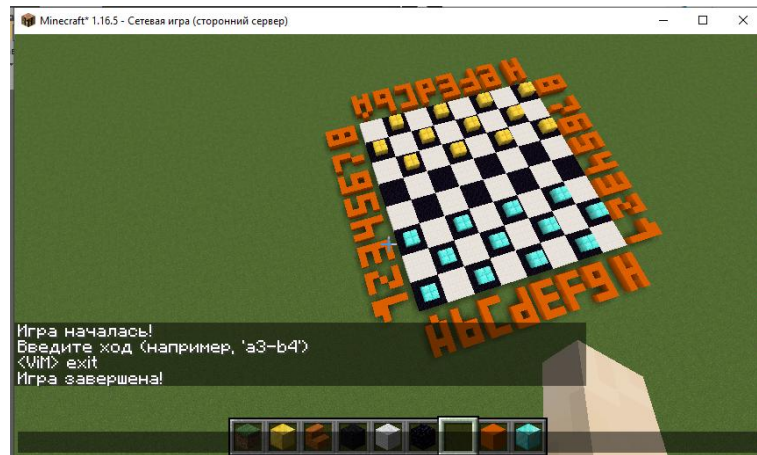
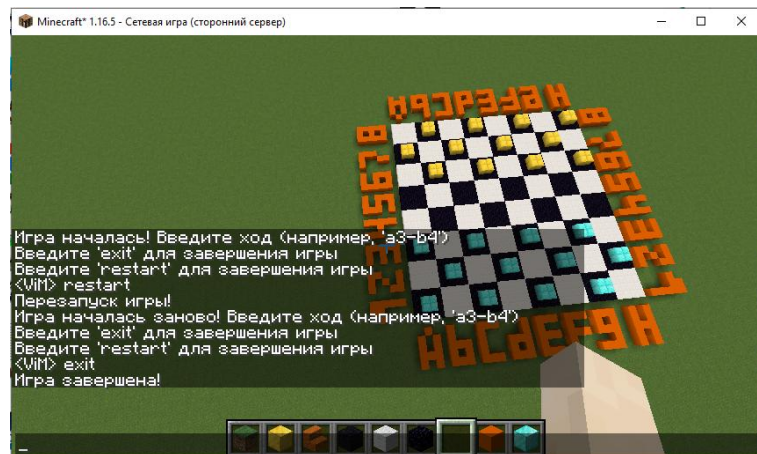


Рис. 1. Результат работы основного цикла игры.



Задание 3. Используя оператор ветвления **elif**, самостоятельно реализуйте команду перезапуска игры «restart». Дополните справку по управлению ходом игры добавленными командами. Результат выполнения задания 3 приведен на рисунке ниже.



Приведем ветвление полученных сообщений из игрового чата для игрового цикла:

```
if полученное сообщение == "команда 1":  
    инструкции команды 1  
elif полученное сообщение == "команда 2":  
    инструкции команды 2  
else:  
    ход шашкой
```

Если полученное из игрового чата сообщение не является

управляющей командой, значит оно задает ход шашкой. Функция **move(start, end)** принимает два аргумента в текстовом формате, например **move("a3", "b4")**. Встает задача получить из текстового сообщения ('a3'-b4') аргументы для функции **move(start, end)**.



```
start, end = call.split('-')  
move(start, end)
```

Воспользуемся функцией **split()**.



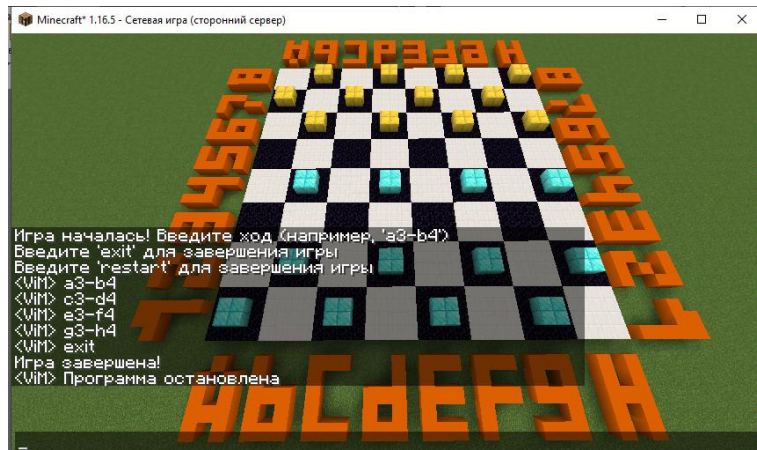
Функция **split()** используется в Python для разделения строки на части по разделителю. Если использовать **split("-")**, то строка будет разделена на две части по символу дефиса. Создадим внутри блока ветвления **else** переменные **start** и **end** и присвоим им соответственно координаты, задающие ход шашкой. Далее передаем полученные текстовые переменные в качестве аргументов метода **move(start, end)**.



Задание 4. Продемонстрируйте учителю возможности по управлению игровым процессом через игровой чат. Введите в игровой чат последовательно сообщения:

7. a3-b4
8. c3-d4
9. e3-f4
- 10.g3-h4
- 11.exit
- 12.Программа остановлена

На рисунке ниже приведена конфигурация игровой доски, полученная в результате ввода сообщений в игровой чат



Запустите программу. Попробуйте ввести в игровой чат сообщение произвольного формата, например, 123.

```
Traceback (most recent call last):
  File "f:\MyProject\lab6.py", line 31, in <module>
    start, end = call.split('-')
ValueError: not enough values to unpack (expected 2,
got 1)
```

Программа остановилась с ошибкой, которая вызвана тем, что введённая строка не содержит дефиса, который является разделителем для функции `split()`.

Для сообщения «123-321» получим уже вторую ошибку.

```
Traceback (most recent call last):
  File "f:\MyProject\lab6.py", line 32, in <module>
    move(start, end)
  File "f:\MyProject\lab5.py", line 7, in move
    block_type = mc.getBlock(MineToChess[start])
```

В данном примере ошибка вызвана тем, что словарь `MineToChess` не содержит записей с ключами 123-321.

Следовательно, необходимо научиться обрабатывать исключения. Для целей данной лабораторной работы будет достаточно отбрасывать все непредусмотренные в программе сообщения.



В Python для обработки исключений используется конструкция `try-except`. Она позволяет "ловить" и обрабатывать ошибки, которые возникают во время выполнения программы, чтобы предотвратить её экстренное завершение.



Задание 5. Дополните блок ветвления `else` предложенной ниже

синтаксической конструкцией. Для блока 1 используйте уже имеющиеся инструкции, блок 2 должен выводить сообщение в игровой чат «Неверный ввод. Повторите ваш ход».

```
try:
    блок 1
except Exception as e:
    блок 2
```

Если задание выполнено корректно, то в игровом чате, при вводе непредусмотренных сообщений, будем всегда получать предупреждение.

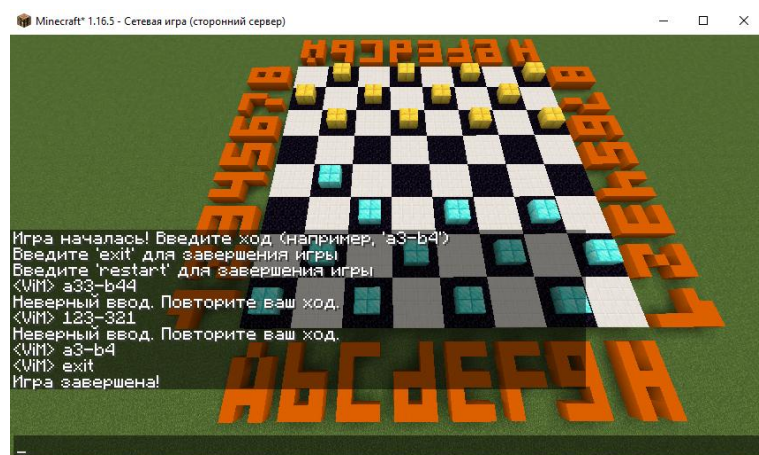


Рис. 2. Обработка исключений

Закключение: в шестой лабораторной работе мы познакомились с обработкой исключений и функцией **split()** в Python, разработали цикл для обработки игровых событий, реализовать управление игровым процессом через чат игры.