**Министерство образования и науки Российской Федерации**

Федеральное государственное автономное образовательное учреждение

высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Инженерная школа информационных технологий и робототехники

Направление: 09.03.02 «Информационные системы и технологии»

Отделение «Информационных технологий»

**Игровой бот с алгоритмическим ИИ**

Отчет по дисциплине

Учебно-исследовательская работа студентов

Выполнили студенты гр. 8И6А \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Голушков А.Н.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Видякин Г.И.

(Подпись) (Ф.И.О.)

Дата сдачи отчета преподавателю \_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г.

Руководитель Ассистент ОИТ Мыцко Е.А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Оценка руководителя) (Подпись)

\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_г.

(Дата проверки)

Учебно-исследовательскую работу студенты защитили с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

Члены комиссии: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_г.

(дата защиты)

Содержание

[Введение 3](#_Toc11878300)

[Цель работы 4](#_Toc11878301)

[Задачи 4](#_Toc11878302)

[1. Улучшенное информирование в отладке 4](#_Toc11878303)

[2. Питомцы 5](#_Toc11878304)

[3. Система торговли 8](#_Toc11878305)

[4. Новый этаж 8](#_Toc11878306)

[5. Новые предметы 11](#_Toc11878307)

[6. Алгоритмический ИИ 11](#_Toc11878308)

[7. Принятие сложных решений 13](#_Toc11878309)

[Заключение 17](#_Toc11878310)

[Список литературы 18](#_Toc11878311)

# Введение

Текстовый квест — это разновидность [компьютерной игр](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%B8%D0%B3%D1%80%D0%B0)ы, в которой взаимодействие с игроком осуществляется посредством текстовой информации. Существуют два вида интерфейса:

1. интерфейс с вводом текста с клавиатуры;
2. интерфейс в виде меню, где игрок выбирает действие из нескольких предложенных

Было решено сделать текстовую игру с помощью бота Telegram.

На первый взгляд, Telegram — это просто очередной мессенджер. Реклама гласит, что он быстр, защищен, в нем нет рекламы. Но самая важная для нас его функция — боты!

Бот — это не просто «автоответчик». Его правильнее считать автоматизированным помощником. Пользователи могут взаимодействовать с ботами при помощи сообщений, отправляемых через обычные или групповые чаты.

Telegram позволяет пользователям создавать своих ботов.

Логика бота контролируется при помощи HTTPS запросов к [API для ботов](https://tlgrm.ru/docs/bots/api).

В текстовой игре должен присутствовать алгоритмический искусственный интеллект.

В прошлом семестре был создан бот, в котором был реализован фундамент игры. Так каждый игрок имеет свои характеристики, как и другие персонажи. В игру были добавлены предметы и сюжетная линия, в которой необходимо сражаться с монстрами.

# Цель работы

Целью данного проекта является разработка игрового Telegram бота с алгоритмическим ИИ на языке программирования Python.

# Задачи

1. Улучшить информирование в отладке
2. Создание питомцев
3. Разделение сложности уровней
4. Улучшить систему торговли
5. Создание новых уровней
6. Создание новых предметов
7. Создание более сложной модели алгоритмического ИИ

# Улучшенное информирование в отладке

Первым делом для более удобной разработки приложения и отслеживания истории действий в приложения было добавлено логирование различных действий. Так выводится информация при отправлении сообщения, открытии комнаты, возникновении ошибки. Это значительно упростило дальнейшую отладку приложения

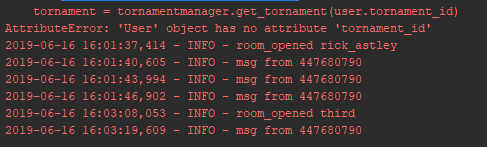


Рисунок – Логирование 1

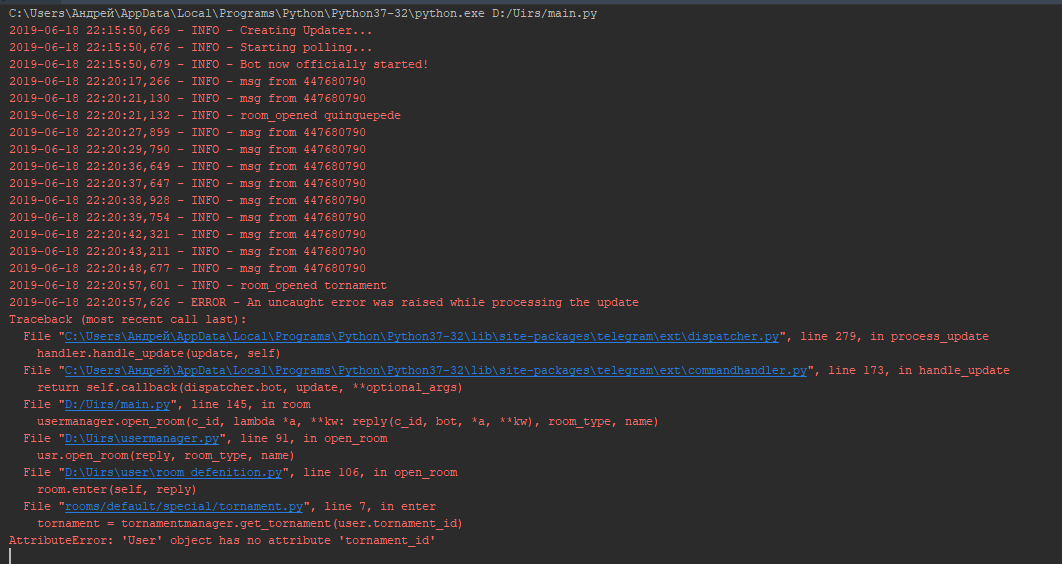


Рисунок – Логирование 2

Для отслеживания характеристик персонажа на стороне клиента, была добавлена команда /debug (рис. 3).

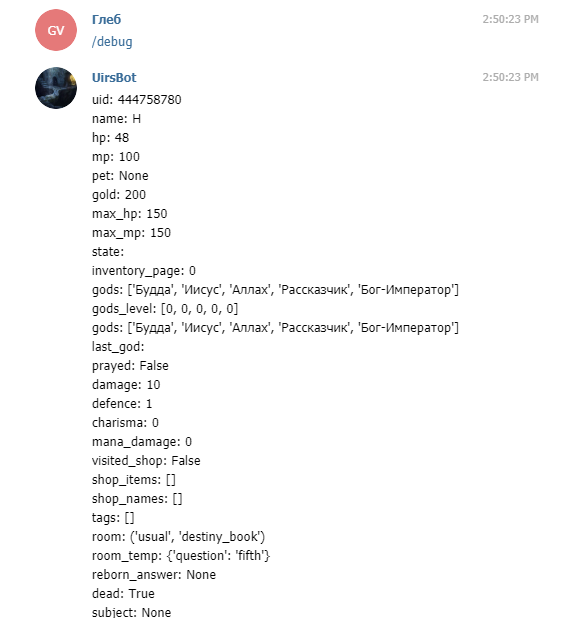


Рисунок – Вызов команды /debug

# Питомцы

В игру были добавлены питомцы. Питомец – спутник игрока, каждый питомец повышает какие-либо характеристики игрока, тем самым способствуя прохождению.

Всего 5 питомцев: медведь (рис. 4), собака (рис. 5), утка (рис. 6), лиса (рис. 7-8). Существенное отличие питомцев от предметов – их нельзя потратить, однако, добыть их можно только у торговца, у которого они появляются с очень маленьким шансом (0,05).

Медведь добавляет бонус к защите игрока.

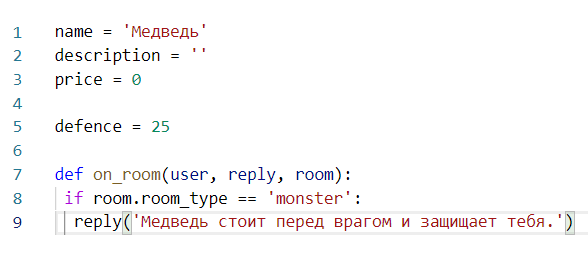


Рисунок – Медведь

Собака и утка добавляют бонусы к атаке игрока.

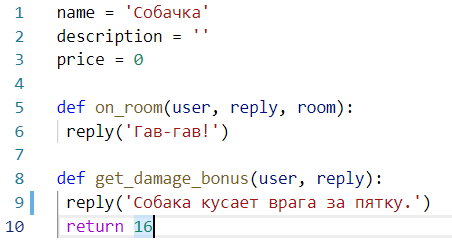


Рисунок – Собака

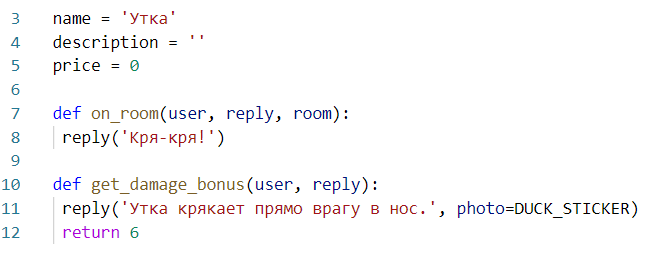


Рисунок – Утка

Лиса с вероятностью 0.15 или если магическая сила игрока больше 10, восстанавливает ⅓ от максимального здоровья игрока.

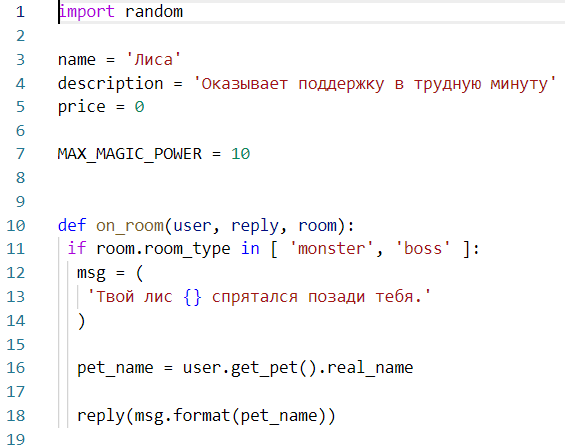


Рисунок – Лиса 1

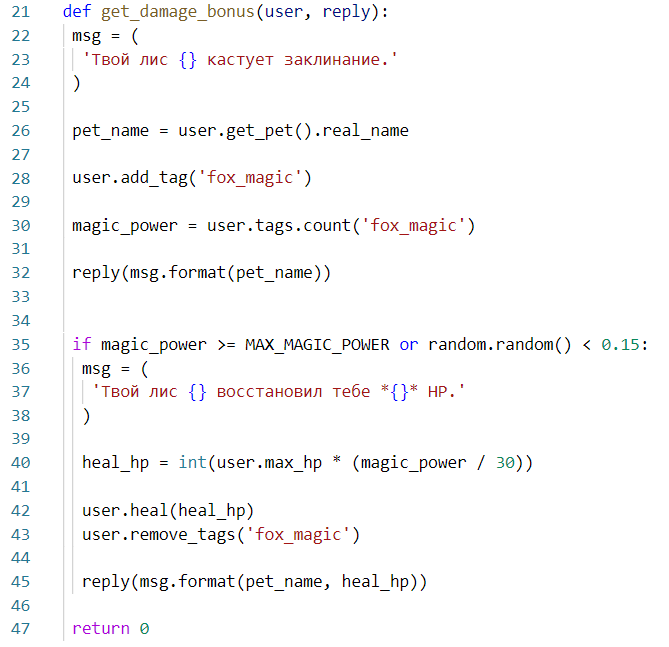


Рисунок – Лиса 2

# Система торговли

Так как в игре появляется большое количество предметов, необходимо было сделать и улучшенную систему торговли. Теперь каждый предмет можно продать у торговца за определенную сумму, причем на сумму влияют различные параметры:

* Разность между уровнем персонажа и уровнем вещи – цена обратно пропорциональна отношению уровня персонажа к уровню вещи.
* Уровень вещи (например, пистолет 3 уровня нанесет, 20 единиц урона, тогда как пистолет 1 уровня 10 единиц). Однако игроку не предоставляется информация об уровнях вещи. Это сделано для того, чтобы запутать игрока.

Но у торговца тяжело достать нужную вещь, поэтому был добавлен специальные персонаж – Лепрекон. В данном персонаже реализована функция аукциона. Игроки могут выставлять на аукцион вещи и давать им свою цену. Любой игрок, который вышел на лепрекона может купить вещь другого игрока, при этом лепрекон берет комиссию с игрока, который выставил вещь, в размере 10% от цены.

Таким образом люди сами будут формировать экономику игры и ценообразование не нужно будет регулировать.

# Новый этаж

Для разделения сложности уровней игровое пространство было решено разбить на 2 “этажа” – верхний и нижний. Каждый этаж имеет определенный набор монстров, который формируется в зависимости от сложности их прохождения (т.е. от уровня атаки и здоровья монстра), т.е. верхний этаж легче, нижний, соответственно, сложнее. При входе в подземелье игроку необходимо пройти несколько легких монстров, после этого в меню можно выбрать на каком этаже проходить игру. По умолчанию игрок находится на верхнем этаже.

При нажатии на кнопку “пойти вниз/вверх” бот присылает список монстров, которые встретились на целевом этаже (рис. 9-10). С помощью этого списка можно отследить состояние прохождения игры. После прохождения всех монстров на обоих этажах, в игру добавляется шанс попасть на босса (пример на рис. 11), у которого в разы больше атака и здоровье, чем у обычных монстров, однако, награда за них соответствующая. При этом сюжетная линия осталась прежней.

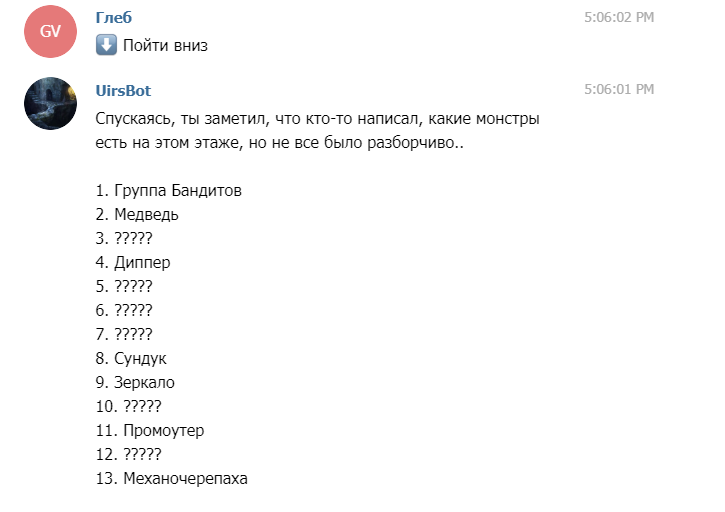


Рисунок – Нижний этаж

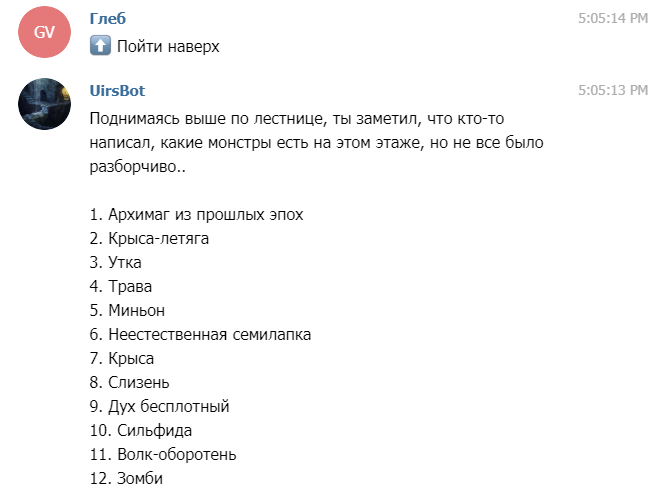


Рисунок – Верхний этаж



Рисунок – Босс

# Новые предметы

В игру были добавлены различные новые предметы для возможности улучшения своего персонажа. Например,

1. Новые виды оружие: Ak-47, М4, магический посох, лазерный пистолет, лазерная отвертка
2. Защитные вещи: щит, шапка из фольги, ботинки, латы, наколенники, механический панцирь, солдатский шлем, кольцо, мазь звездочка (рис. 11)
3. Вещи для выполнения различных заданий: конфетка, зуб василиска, мешочек со спойлерами, тыква, трезубец.

Данные предметы позволят игрокам по-разному развивать своих персонажей и в PvP и PvE боях получать различные преимущества.

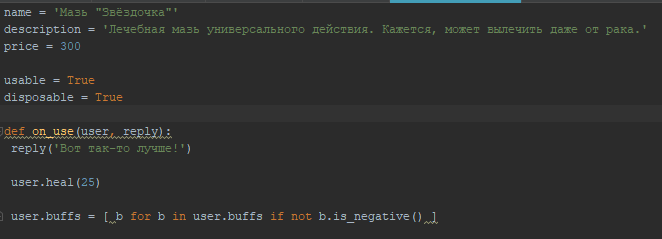


Рисунок – Мазь «Звездочка»

# Алгоритмический ИИ

Игровой ИИ сосредоточен на том, какие действия должен выполнять объект, исходя из условий, в которых находится. Обычно это называют управлением «интеллектуальными агентами», где агент является игровым персонажем, транспортным средством, ботом, а иногда и чем-то более абстрактным: целой группой сущностей или даже цивилизацией. В каждом случае это вещь, которая должна видеть свое окружение, принимать на его основе решения и действовать в соответствии с ними. Это называется циклом Sense/Think/Act (Чувствовать/Мыслить/Действовать):

Восприятие среды в текстовой игре – это ответы игрока и его характеристики, в зависимости от передаваемых алгоритму параметров он должен сделать цепочку решений и выдать ответ. Так на рисунке 10 указан перечень параметров, которые может оценивать игра. Это и предыдущая комната, и текущее HP игрока, и уровень доверия богов, и состояние инвентаря героя.

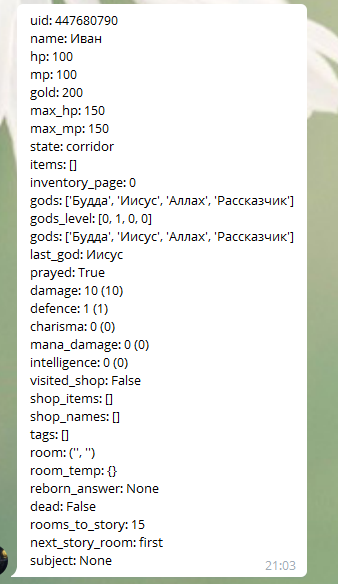


Рисунок 10 - параметры восприятия среды

# Принятие сложных решений

Хоть простые системы реакций очень действенны, бывает много ситуаций, когда их недостаточно. Иногда нужно принимать различные решения, основанные на том, что агент делает в настоящий момент, но представлять это за условие тяжело. Иногда существует слишком много условий, чтобы эффективно представить их в дереве решений или скрипте. Иногда нужно заранее оценивать, как изменится ситуация, прежде чем принимать решение о следующем шаге. Для решения этих проблем нужны более сложные подходы.

Для примера возьмем стража с такими состояниями:

* Патрулирующий (Patrolling).
* Атакующий (Attacking).
* Убегающий (Fleeing).

И такими условиями для изменения его состояния:

* Если страж видит противника, он атакует.
* Если страж атакует, но больше не видит противника, он возвращается к патрулированию.
* Если страж атакует, но сильно ранен, он убегает.

Также можно написать if-операторы с переменной-состоянием стража и различные проверки: есть ли поблизости враг, какой уровень здоровья NPC и т. д. Добавим еще несколько состояний:

* Бездействие (Idling) — между патрулями.
* Поиск (Searching) — когда замеченный враг скрылся.
* Просить о помощи (Finding Help) — когда враг замечен, но слишком силен, чтобы сражаться с ним в одиночку.

Выбор для каждого из них ограничен — например, страж не пойдет искать скрывшегося врага, если у него низкое здоровье.

В конце концов огромный список «если <x и y, но не z>, то <p>», может стать слишком громоздким, поэтому следует формализовать метод, который позволит нам держать в уме состояния и переходы между состояниями. Чтобы это сделать, примем во внимание все состояния, и под каждым состоянием запишем в список все переходы в другие состояния, вместе с необходимыми для них условиями (рис. 13).

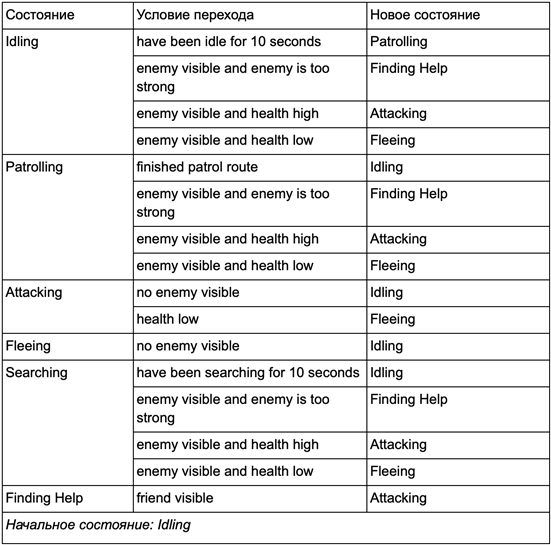


Рисунок – Список состояний

Нарисуем диаграмму и получим полный обзор того, как меняется поведение NPC (рис. 14).

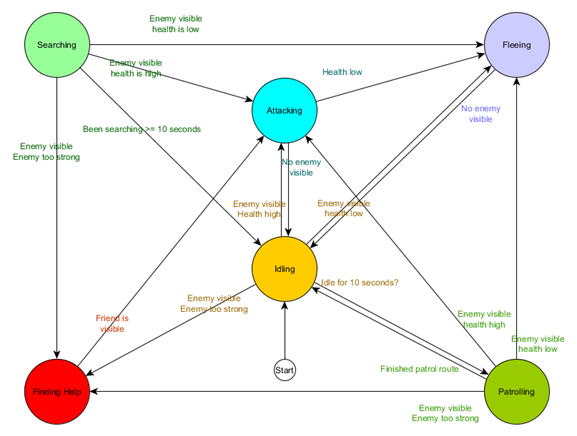


Рисунок – Диаграмма состояний

В нашем случае состояния были добавлены монстрам. Для примера возьмем монстра с такими состояниями.

* Бездействие (Idling)
* Атакующий (Attacking)
* Скрывающийся (Hiding) — Если монстр атакует, но сильно ранен, он убегает
* Ярость (Rage) -  Если монстр атакует, но сильно ранен, у него повышается атака

Каждому монстру был введен радиус обнаружения, т.е. вероятность, с которой игрок выйдет на определенного монстра (изначально вероятность для всех монстров была одинаковая, кроме боссов и сюжетной линии).

Монстр находится в состоянии бездействия, если на него не вышел игрок. При встрече с игроком монстр переходит в состояние атакующий. Для некоторых монстров было введено условие, что при низком уровне здоровья монстр может убежать (состояние скрывающийся), не дав себя убить. При этом радиус обнаружения этого монстра снижается за счет повышения радиуса у других монстров. Также снижается его показатель атаки. Аналогично, некоторые монстры при низком уровне здоровья могут переходить в состояние ярости, в котором увеличивается атака.

# Заключение

Результатом учебно-исследовательской работы является Telegram бот с применением алгоритмического игрового ИИ. Для достижения этой цели были изучены основы языка Python, принципы работы API, непосредственно сам Telegram API, устройство и система работы ботов, их регистрация и настройка, а также запуск собственного сервера. В ходе работы был изучен принцип построения игрового ИИ для принятия сложных решений.

В этом семестре мы улучшили функционал игры. Была переработана система торговли, расширен набор внутри игровых локаций, добавлены новые игровые предметы и питомцы, повышающие гибкость кастомизации игрового персонажа, доработан игровой ИИ. Также была начата разработка PvP режимов.

# Список литературы

1. [Музафаров](https://habr.com/users/M_Muzafarov/) М. Инструкция: Как создавать ботов в Telegram [Электронный ресурс]: [сайт]. – Инструкция – 2015. – Режим доступа: <https://habrahabr.ru/post/262247> свободный (дата обращения: 26.09.2018).
2. Пишем ботов для Telegram на языке Python [Электронный ресурс]: [сайт]. – Учебник – 2018. – Режим доступа: https://www.gitbook.com/book/groosha/telegram-bot-lessons/details свободный (дата обращения: 18.09.2018).
3. Документация Telegram [Электронный ресурс]: [сайт]. – Докуметация – 2018. – Режим доступа: https://tlgrm.ru/docs/bots/faq свободный (дата обращения: 15.09.2018).
4. Гук Н. Как создать игровой ИИ: гайд для начинающих [Электронный ресурс]: [сайт]. – Статья – 2018. – Режим доступа: <https://habr.com/company/pixonic/blog/428892/> свободный (дата обращения: 30.08.2018).
5. [Жислин](https://habr.com/users/M_Muzafarov/)а В. Создание искусственного интеллекта для игр — от проектирования до оптимизации [Электронный ресурс]: [сайт]. – Статья – 2015. – Режим доступа: <https://habr.com/company/intel/blog/265679/> свободный (дата обращения: 26.11.2018).
6. Бот для Telegram на Python: от первой строчки кода до запуска на Heroku [Электронный ресурс]: [сайт]. – Статья – 2017. – Режм доступа: <https://tproger.ru/translations/telegram-bot-create-and-deploy/> свободный (дата обращения: 26.11.2018).