

УТВЕРЖДАЮ

Руководитель практики от Университета

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

*(должность, ФИО)*

МП \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(Подпись)*

Дата « \_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 202\_ год

**ОТЧЕТ**

**ПО УЧЕБНОЙ ПРАКТИКЕ**

ПМ.05 Проектирование и разработка информационных систем

|  |  |
| --- | --- |
| Воробьев Алексей Владимирович | |
| *(Ф.И.О. обучающегося)* | |
| 09.02.07 Информационные системы и программирование | |
| *(специальность)* | |
|  | |
| Учебная группа | ИСПк-204-52-00 |
|  |  |
| Место прохождения практики | ФГБОУ ВО «Вятский государственный университет», |
| Колледж ВятГУ | *(наименование организации, структурного подразделения организации)* |
|  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Итоговая оценка: |  | | | | |
| Руководитель  практики от университета |  |  |  |  |  |
|  | *(дата)* |  | *(подпись)* |  | *(Ф.И.О.)* |

Киров, 2023 г.



ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ НА ПРАКТИКУ

|  |  |
| --- | --- |
| Ф.И.О. обучающегося | Воробьев Алексей Владимирович |
| Специальность | 09.02.07 Информационные системы и программирование |
| Учебная группа | ИСПк-204-52-00 |
| Вид практики | учебная практика |

Сроки прохождения практики с 09.01.2023 по 10.06.2023

Место прохождения практики

ФГБОУ ВО «Вятский государственный университет», Колледж ВятГУ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | (наименование организации, структурного подразделения организации) | | |
| № | Виды работ, выполняемых обучающимися во время практики | | Объем работ (час) | Формируемые компетенции | |
| 1 | Пройти инструктаж по ознакомлению с правилами внутреннего трудового распорядка, охраны труда, техники безопасности, противопожарной безопасности, санитарно-эпидемиологическими правилами и гигиеническими нормативами, а также вводный инструктаж и инструктаж на рабочем месте | | 2 | - | |
| 2 | Постановка задачи | | 8 | ОК2, ОК6, ОК7, ОК9 | |
| 3 | Настройка рабочего окружения | | 8 | ОК1, ОК8, ОК9 | |
| 4 | Анализ предметной области, постановка расширенного технического задания | | 10 | ПК5.1, ОК3, ОК4 | |
| 5 | Проектирование | | 35 | ПК5.2, ПК5.3, | |
| 6 | Реализация программного кода | | 35 | ПК5.4, ПК5.8, | |
| 7 | Тестирование и отладка полученного кода | | 22 | ПК5.5, ПК6.4, ОК2, ОК9 | |
| 8 | Подготовка отчетной документации, включая детальное описание разработанных и использованных алгоритмов | | 10 | ПК5.6, ПК5.7, ОК9, ОК10, ОК11 | |
| 9 | Подготовка к сдаче и сдача промежуточной аттестации | | 2 | ОК4, ОК5 | |

|  |  |  |  |
| --- | --- | --- | --- |
| Индивидуальное задание на практику разработано в соответствии с рабочей программой практики. | | | |
| Руководитель  практики от университета |  |  |  | |  |  |
|  | (дата) |  | (подпись) | |  | (Ф.И.О.) |
| С индивидуальным заданием ознакомлен(а) |  |
|  | (дата, подпись обучающегося) |

**ОТЗЫВ РУКОВОДИТЕЛЯ ПРАКТИКИ**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Ф.И.О. обучающегося | | Воробьев Алексей Владимирович | | | |
| Специальность | | 09.02.07 Информационные системы и программирование | | | |
| Учебная группа | | ИСПк-204-52-00 | | | |
| Вид практики | | учебная практика | | | |
| Сроки прохождения практики с | | | 09.01.2023 | по | 10.06.2023 |
| Место прохождения практики | | | ФГБОУ ВО «Вятский государственный университет», Колледж ВятГУ | | |
|  | (наименование организации, структурного подразделения организации) | | | | |

ВИДЫ И КАЧЕСТВО ВЫПОЛНЕНИЯ РАБОТ

|  |  |  |  |
| --- | --- | --- | --- |
| Вид работ | Критерий выполнения работ | | |
| Выполнены полностью самостоятельно | Выполнены с незначительной помощью наставника | Выполнены с помощью наставника |
| Постановка задачи | V |  |  |
| Настройка рабочего окружения | V |  |  |
| Анализ предметной области, постановка расширенного технического задания | V |  |  |
| Проектирование | V |  |  |
| Реализация программного кода | V |  |  |
| Тестирование и отладка полученного кода | V |  |  |
| Подготовка отчетной документации, включая детальное описание разработанных и использованных алгоритмов | V |  |  |
| Подготовка к сдаче и сдача промежуточной аттестации | V |  |  |

Обучающийся ознакомлен с правилами внутреннего трудового распорядка, охраны труда, техники безопасности, противопожарной безопасности, санитарно-эпидемиологическими правилами и гигиеническими нормативами, а также прошел вводный инструктаж и инструктаж на рабочем месте.

Во время прохождения учебной практики обучающимся освоены следующие профессиональные и общие компетенции:

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование компетенции | Показатели оценки | Оценка | |
| Освоена | Не освоена |
| ПК 5.1. Собирать исходные данные для разработки проектной документации на информационную систему. | Способен выполнять анализ предметной области, выявляя существенные элементы, оказывающие влияние на проектируемую систему | V |  |
| ПК 5.2. Разрабатывать проектную документацию на разработку информационной системы в соответствии с требованиями заказчика. | Способен выполнять декомпозицию сущностей с целью получения наиболее полной картины о целесообразной структуре разработки | V |  |
| ПК 5.3. Разрабатывать подсистемы безопасной информационной системы в соответствии с техническим заданием. | Способен прогнозировать потенциально некорректные действия пользователя и предусматривать соответствующие реакции со стороны системы | V |  |
| ПК 5.4. Производить разработку модулей информационной системы в соответствии с техническим заданием. | Способен создавать программный код, отвечающий предъявляемым требованиям | V |  |
| ПК 5.5. Осуществлять тестирование информационной системы на этапе опытной эксплуатации с фиксацией выявленных ошибок кодирования в разрабатываемых модулях информационной системы. | Способен выполнять оценку корректности функционирования системы | V |  |
| ПК 5.6. Разрабатывать техническую документацию на эксплуатацию информационной системы. | Способен разрабатывать проектную, техническую и пользовательскую документации | V |  |
| ПК 5.7. Производить оценку информационной системы для выявления возможности ее модернизации. | Способен оценивать перспективы дальнейшего развития программной системы | V |  |
| ПК 5.8. Осуществлять интеграцию информационной системы с другими программными продуктами. | Способен повторно использовать готовые шаблонные решения при разработке программного продукта | V |  |
| ОК 01. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам. | Способен анализировать специализированную информацию и находить оптимальные пути решения задач профессиональной деятельности | V |  |
| ОК 02. Осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности. | Способен анализировать специализированную информацию и находить оптимальные пути решения задач профессиональной деятельности | V |  |
| ОК 03. Планировать и реализовывать собственное профессиональное и личностное развитие. | Способен находить пути улучшения имеющихся решений, позволяющих повысить их общий качественный уровень | V |  |
| ОК 04. Работать в коллективе и команде, эффективно взаимодействовать с коллегами, руководством, клиентами. | Способен конструктивно обмениваться информацией с коллегами, грамотно формулировать запросы в целях получения разъясняющей информации | V |  |
| ОК 05. Осуществлять устную и письменную коммуникацию на государственном языке с учетом особенностей социального и культурного контекста. | Способен конструктивно обмениваться информацией с коллегами, грамотно формулировать запросы в целях получения разъясняющей информации | V |  |
| ОК 06. Проявлять гражданско-патриотическую позицию, демонстрировать осознанное поведение на основе традиционных общечеловеческих ценностей. | Способен конструктивно обмениваться информацией с коллегами, грамотно формулировать запросы в целях получения разъясняющей информации | V |  |
| ОК 07. Содействовать сохранению окружающей среды, ресурсосбережению, эффективно действовать в чрезвычайных ситуациях. | Способен эффективно использовать компьютерное время, а также материальные ресурсы, необходимые для решения поставленных задач | V |  |
| ОК 08. Использовать средства физической культуры для сохранения и укрепления здоровья в процессе профессиональной деятельности и поддержания необходимого уровня физической подготовленности. | Способен соблюдать требования внутреннего трудового распорядка организации, охраны труда и техники безопасности в целях сохранения собственного здоровья | V |  |
| ОК 09. Использовать информационные технологии в профессиональной деятельности. | Способен применять стек современных средств разработки ПО для решения задач профессиональной деятельности в заданном контексте | V |  |
| ОК 10. Пользоваться профессиональной документацией на государственном и иностранном языке. | Способен использовать в своей работе специализированную документацию | V |  |
| ОК 11. Планировать предпринимательскую деятельность в профессиональной сфере. | Способен проектировать алгоритмические решения, принимая во внимание имеющиеся ресурсные ограничения | V |  |

**Краткая характеристика работы обучающегося**

|  |
| --- |
| Программа практики выполнена в полном объеме. Все виды работ выполнялись в срок, |
| без существенных замечаний. В достаточной степени была проявлена самостоятельность |
| и умение грамотно пользоваться источниками информации, находящимися в свободном |
| доступе. |

|  |  |  |
| --- | --- | --- |
|  |  | Руководитель практики  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Подпись ФИО  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (должность)  Дата «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 202\_\_ год |

**СОДЕРЖАНИЕ**

[**ВВЕДЕНИЕ** 3](#_Toc136681340)

[**1. СВЕДЕНИЯ О РАБОТЕ, ВЫПОЛНЕННОЙ В ПЕРИОД ПРОХОЖДЕНИЯ УЧЕБНОЙ ПРАКТИКИ** 4](#_Toc136681341)

[**2. НАСТРОЙКА РАБОЧЕГО ОКРУЖЕНИЯ** 5](#_Toc136681342)

[**3. ФОРМУЛИРОВКА ИНДИВИДУАЛЬНОГО ЗАДАНИЯ** 6](#_Toc136681343)

[**4. ОПИСАНИЕ ВЫПОЛНЕНИЯ ИНДИВИДУАЛЬНОГО ЗАДАНИЯ** 7](#_Toc136681344)

[**ЗАКЛЮЧЕНИЕ** 16](#_Toc136681345)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ** 17](#_Toc136681346)

[**ПРИЛОЖЕНИЕ А** 18](#_Toc136681347)

[**ПРИЛОЖЕНИЕ Б** 32](#_Toc136681348)

[**ПРИЛОЖЕНИЕ В** 33](#_Toc136681349)

[**ПРИЛОЖЕНИЕ Г** 34](#_Toc136681350)

# **ВВЕДЕНИЕ**

Учебная практика ПМ.05 проходила на базе Колледжа ФГБОУ ВО «Вятский государственный университет» в период с 09.01.2023 г. по 14.06.2023 г. по субботам.

Цель практики: сформировать у обучающихся представление о процессе разработки прикладного программного обеспечения.

Задачи практики:

– дать представление о различных методах проектирования прикладного программного обеспечения;

– сформировать навыки написания документации, требуемой при разработке прикладного программного обеспечения;

– дать представление о различных инструментах, используемых при разработке прикладного программного обеспечения.

# **1. СВЕДЕНИЯ О РАБОТЕ, ВЫПОЛНЕННОЙ В ПЕРИОД ПРОХОЖДЕНИЯ УЧЕБНОЙ ПРАКТИКИ**

В период с 09.01.2023 по 14.06.2023 при прохождении учебной практики ПМ.05 на базе Колледжа ФГОБУ ВО «Вятский государственный университет» был выполнен следующий перечень работ, представленный в таблице 1.

Таблица 1 – Сведения о работе, выполненной в период практики

| Дата | Краткое содержание выполненных работ |
| --- | --- |
| 14.01.2023 | Ознакомление с правилами внутреннего трудового распорядка, охраны труда, техники безопасности, противопожарной безопасности, санитарно-эпидемиологическими правилами и гигиеническими нормативами, а также вводный инструктаж и инструктаж на рабочем месте |
| 15.01.2023-28.01.2023 | Постановка задачи |
| 29.01.2023-  12.02.2023 | Настройка рабочего окружения |
| 13.02.2023-26.03.2023 | Анализ предметной области, постановка расширенного технического задания |
| 27.03.2023-27.03.2023 | Проектирование |
| 28.03.2023-30.04.2023 | Реализация программного кода |
| 01.05.2023-21.05.2023 | Тестирование и отладка полученного кода |
| 22.05.2023-08.06.2023 | Подготовка отчетной документации, включая детальное описание разработанных и использованных алгоритмов |
| 09.06.2023-14.06.2023 | Подготовка к сдаче и сдача промежуточной аттестации |

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

(дата) (подпись)

# **2. НАСТРОЙКА РАБОЧЕГО ОКРУЖЕНИЯ**

Python - универсальный язык, широко используется во всем мире для самых разных целей - базы данных и обработка текстов, встраивание интерпретатора в игры, программирование GUI и быстрое создание прототипов (RAD).

PyCharm - это кроссплатформенная интегрированная среда разработки для языка программирования Python, разработанная компанией JetBrains на основе IntelliJ IDEA. Предоставляет пользователю комплекс средств для графического отладчика и работы с кодом.

PyQt5 - это набор инструментов для разработки графического интерфейса пользователя (GUI) на языке Python с использованием библиотеки Qt. Qt является кросс-платформенным фреймворком для разработки приложений, который предоставляет мощные средства для создания графических интерфейсов, обработки событий, управления виджетами и многого другого.

Интерфейс IDE PyCharm представлен на рисунке 1

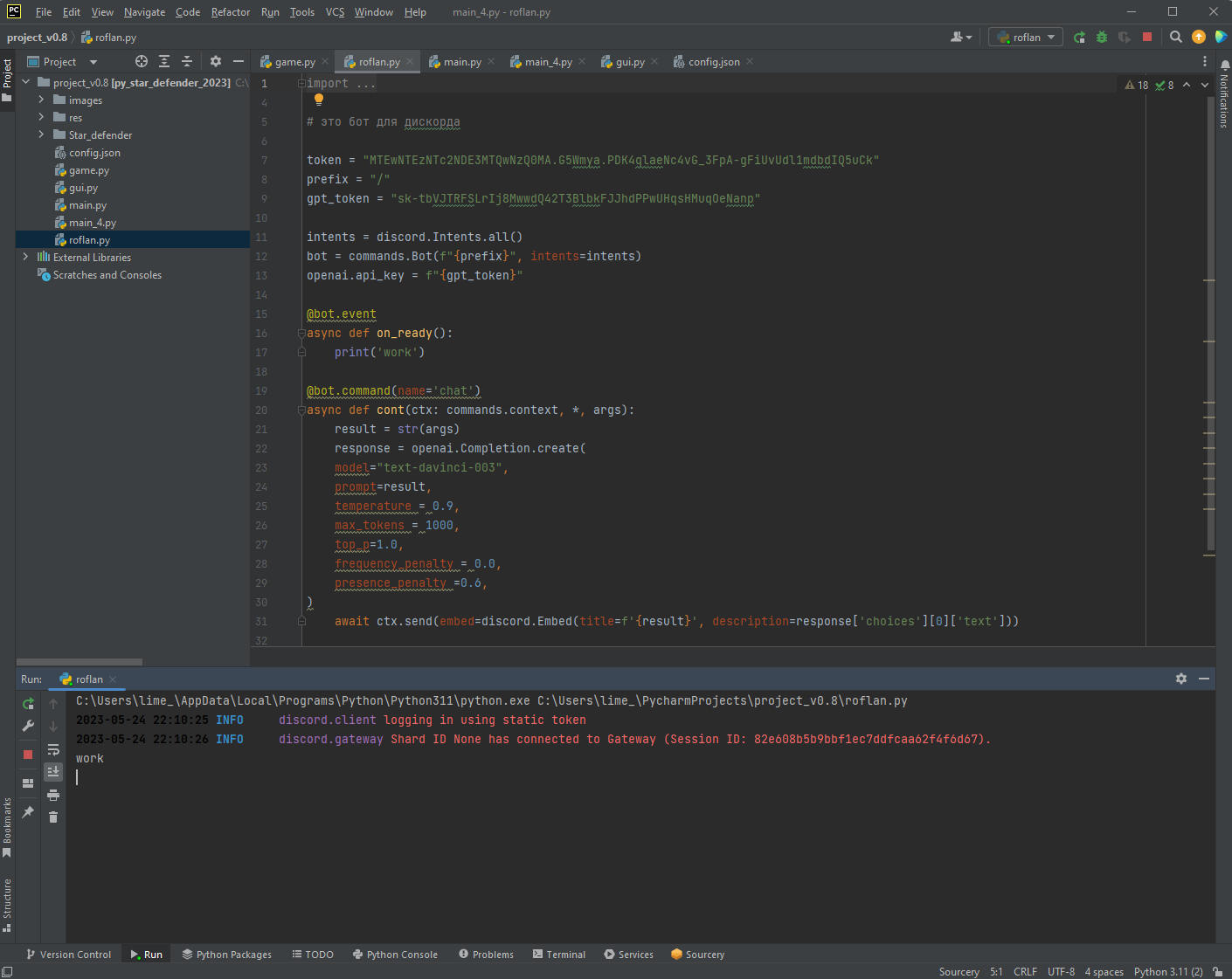


Рисунок 1 – IDE PyCharm

# **3. ФОРМУЛИРОВКА ИНДИВИДУАЛЬНОГО ЗАДАНИЯ**

В соответствии с индивидуальным заданием на учебную практику необходимо:

Создать Cайд-скроллер шутер “Star defender”, который должен иметь игровой магазин и улучшения в нем. Cайд-скроллер шутер будет бесплатным с русским интерфейсом.

# **4. ОПИСАНИЕ ВЫПОЛНЕНИЯ ИНДИВИДУАЛЬНОГО ЗАДАНИЯ**

**4.1 Анализ предметной области и обзор аналогов**

***Предметная область***

Для кого: для детей, подростков и взрослых.

Скролл-шутер подвид компьютерных игр в жанре шутер. В этих играх игровой персонаж продвигается вперёд автоматически и часто представляет собой летающее транспортное средство типа воздушного судна или космического аппарата, которое расстреливает множество врагов, одновременно уворачиваясь от препятствий.

Скролл-шутеры обычно разделяют на два игровых стиля — так называемый «маниакальный», где во главе угла рефлексы и реакция, и «методический», где успех определяется запоминанием последовательностей и безошибочными действиями. Скролл-шутер подвид компьютерных игр в жанре шутер.

Данная игра развивает реакцию и рефлексы и применяется для развлечения.

На рисунке 2 представлена игра  Jamestown; это игра в жанрах экшен и аркада разработанная инди-компаниями Final Form Games и Final Form LLC. Она была выпущена в 2011.



Рисунок 2 – игровой процесс JAMESTOWN

Скролл-шутер с мультиплеером до четырех игроков, повествующий о битве, развернувшейся в 17-м веке на Британском Колониальном Марсе.

В отличие от классических представителей жанра, у каждого игрока тут свой набор очков здоровья, из-за чего игра будет продолжаться до тех пор, пока жив хотя бы один участник сессии.

Преимуществом данной игры является мультиплеер на 4-х человек.

Недостатком данной игры является необходимости платить за неё.

На рисунке 3 представлена игра Aces of the Luftwaffe; разработчик и издатель: HandyGames

это игра в жанре экшен, с поджанрами shoot 'em up и cкролл-шутер. Дата выхода: 6 мая 2015г



Рисунок 3 – игровой процесс Aces of the Luftwaffe

Горизонтальный скролл-шутер на тематику Второй мировой. Игрок играет за одного из трех пилотов. Каждый пилот обладает своим самолетом, на котором есть разные типы орудия и спецприемы. Изначально доступен только один не прокаченный пилот, а остальных надо открывать за очки, которые падают с убитых врагов. Так же новые карты тоже открываются за очки.

К преимуществом данной игры можно отнести разные типы орудия, спецприемы и бесплатную модель распространения.

К недостатком можно отнести много рекламы, и вынужденное перепрохождение уровней для открытия нового контента.

 На рисунке 4 представлена игра Star Defender 2. компьютерная казуальная игра в жанре аркады, разработанная компанией Awem Studio. Дата выхода: 2005 г.



Рисунок 4 - игровой процесс Star Defender 2

Star Defender 2 (рус. «Звездный защитник 2») - компьютерная казуальная игра в жанре скролл-шутер, разработанная компанией Awem Studio.

Игроку предстоит победить всех инопланетян на протяжении многих уровней, в конце каждой миссии нужно сразится с боссом. Корабль, которым управляет игрок, способен улучшать свои характеристики прямо во время боя, собирая оружие и бонусы.

К плюсам данной игры можно отнеси красивую графику

К недостаткам относятся необходимости платить за неё и отсутствие постоянных улучшений

**4.2 Разработка структуры приложения и алгоритмов функционирования**

Сайд-скроллер шутер начинается с меню где находятся 4 кнопки «Играть», «Магазин», «Правила» и «Выход». При нажатии на кнопку играть открывается окно с игрой где происходит игровой процесс, где можно заработать очки, которые можно потратить в магазине. Выйти из игры можно закрыть на кнопку “Escape”.

На кнопку “Магазин” открывается игровой магазин где тратятся игровые очки и покупаются улучшения для игрока. Если очков для покупки улучшения недостаточно, то вместо улучшения появится надпись “недостаточно очков”. Так же есть кнопка “Выход” которая возвращает в главное меню.

На кнопку “Правила” открываются правила, и кнопка “Назад” которая возвращает в главное меню.

И на кнопку “Выход” Сайд-скроллер шутер закрывается.

Схема алгоритма программы указана на рисунке 5.

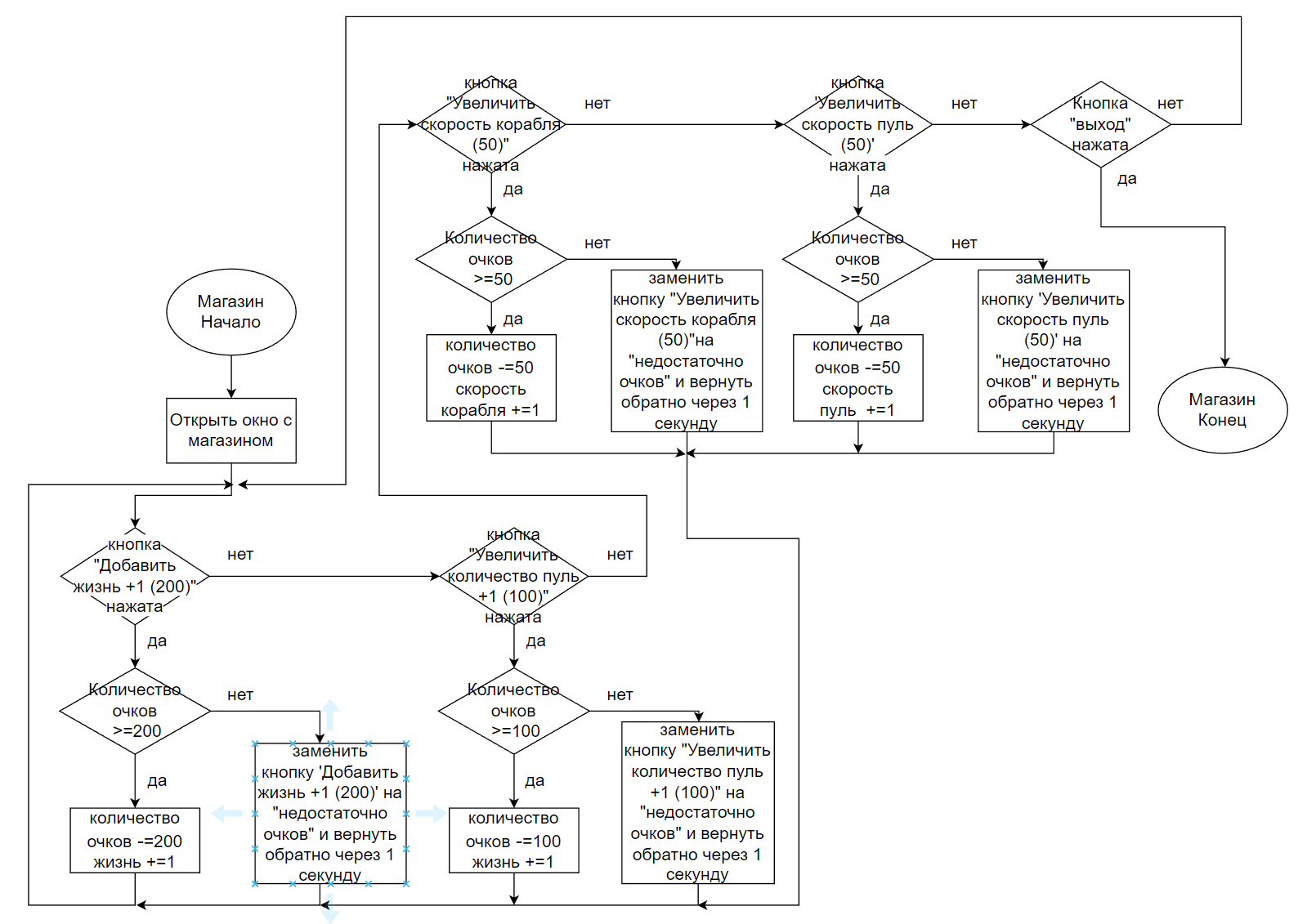


Рисунок 5 – Схема алгоритма Магазин

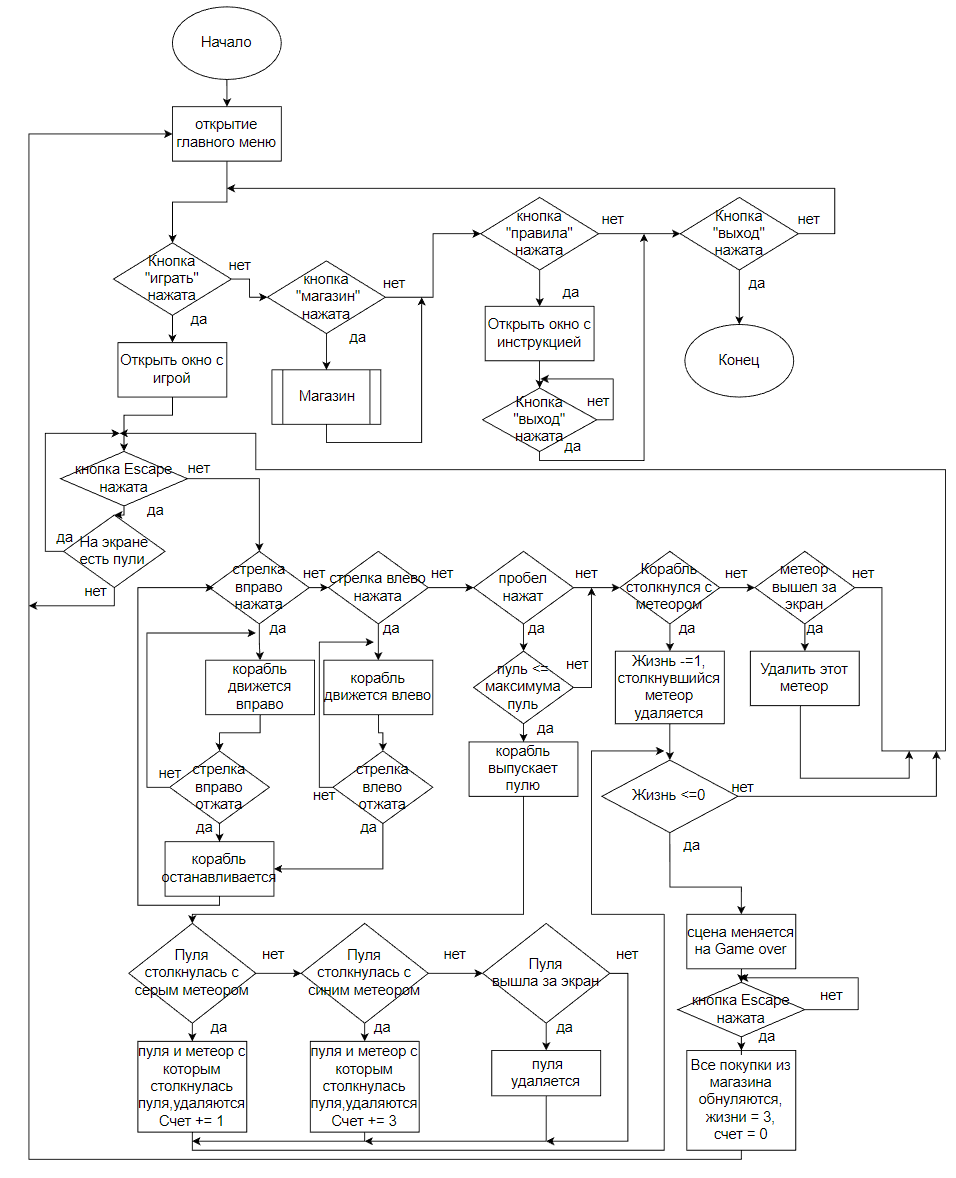


Рисунок 6 – Схема алгоритма сайд-скроллер шутер «Star defender»

**4.3 Проектирование программного обеспечения**

Прототип главного меню на рисунке 6.

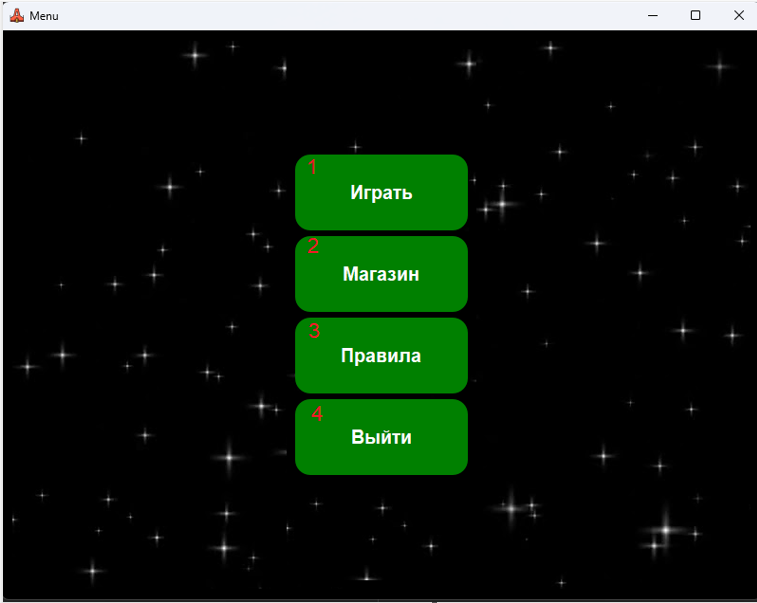


Рисунок 6 – Прототип главного меню

1 – Кнопка Играть открывает игровое окно.

2 – Кнопка Магазин открывает игровой магазин.

3 – Кнопка Правила открывает правила.

4 – Кнопка Выйти закрывает сайд-скроллер шутер.

Прототип окна инструкции на рисунке 7.

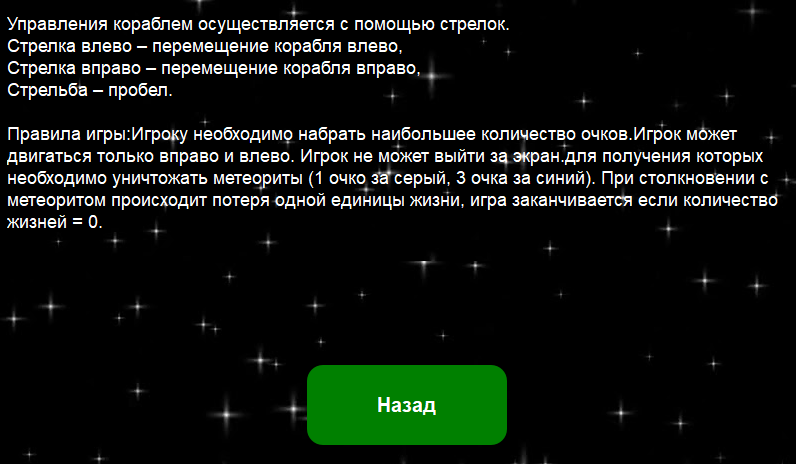


Рисунок 7 – Прототип окна инструкции

Это окно подробно описывает как нужно играть в игру.

Прототип окна игрового магазина на рисунке 8.



Рисунок 8 – Прототип окна игрового магазина

В этом окне покупаются улучшения для игрока

Прототип окна инструкции на рисунке 9.

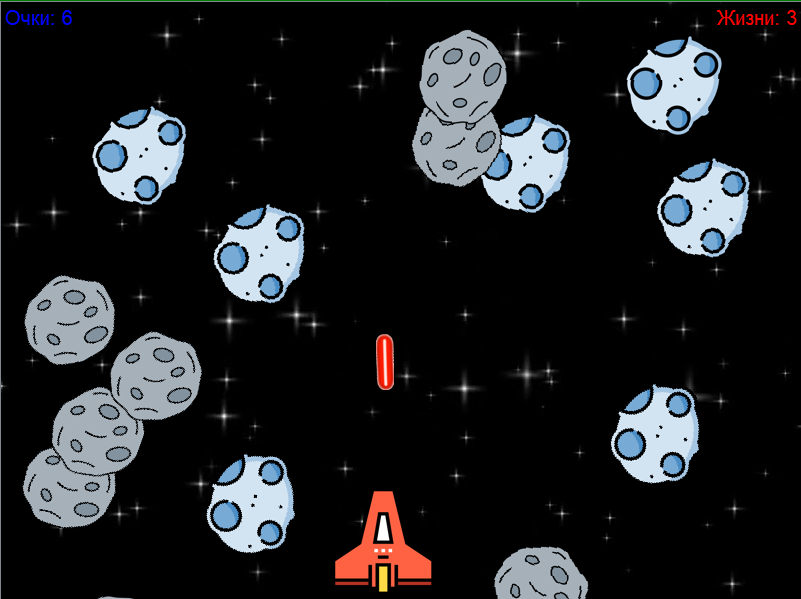


Рисунок 9 – Прототип игрового процесса

В этом окне изображен процесс вылета пули и появление синих и серых метеоров.

**4.4 Тестирование программного обеспечения**

|  |  |  |  |
| --- | --- | --- | --- |
| №  п.п. | Действия пользователя | Критерии приемки | Отметка о прохождении |
| 1 | Нажатие кнопки «Играть» | Открытие игры |  |
| 2 | Нажатие кнопки «Магазин» | Открытие игрового магазина |  |
| 3 | Нажатие кнопки «Добавить жизнь +1 (200)» | Добавление одной жизни кораблю |  |
| 4 | Нажатие кнопки «Увеличить количество пуль +1 (100)» | Добавление одной пули кораблю |  |
| 5 | Нажатие кнопки «Увеличить скорость корабля (50)» | Добавление скорости кораблю |  |
| 6 | Нажатие кнопки «Увеличить скорость пуль (50)» | Добавление скорости пуле |  |
| 7 | Нажатие кнопки «Выход» в магазине | Выход в главное меню |  |
| 8 | Нажатие кнопки «Правила» | Открытие окна с правилами и управлением |  |
| 9 | Нажатие кнопки «Назад» в окне «Правила» | Выход в главное меню |  |
| 10 | Передвижение ползунка громкости | Изменение громкости музыки |  |
| 11 | Нажатие кнопки «Выход» в главном меню | Закрытие программы |  |
| 12 | Нажатие стрелки влево в игровом окне | Движение корабля влево |  |
| 13 | Нажатие стрелки вправо в игровом окне | Движение корабля вправо |  |
| 14 | Нажатие «Space» в игровом окне | Выстрел |  |
| 15 | Нажатие «Escape» в игровом окне | Выход в главное меню и сохранение жизней и очков |  |

# **ЗАКЛЮЧЕНИЕ**

В результате прохождения учебной практике была разработан сайд-скроллер шутер “Star defender” в котором можно покупать улучшения и сбивать метеоры.

Таким образом, программа учебной практики ПМ.05 в период с 09.01.2023 по 10.06.2023 была выполнена мной в полном объеме.

При выполнении работ, предусмотренных заданием на практику, в организации ФГБОУ ВО «Вятский государственный университет» мной, Воробьевым Алексеем Владимировичем, соблюдались правила внутреннего трудового распорядка, охраны труда, техники безопасности, противопожарной безопасности, санитарно-эпидемиологические правила и гигиенические нормативы. Пройден вводный инструктаж и инструктаж на рабочем месте. Замечаний не имел.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. PyQt5 Tutorial [Электронный ресурс]. URL: <https://pythonspot.com/pyqt5/> (дата обращения 01.03.2023).
2. Официальная документация PyQt5 [Электронный ресурс]. URL: <https://www.riverbankcomputing.com/static/Docs/PyQt5/> (дата обращения 01.03.2023).
3. Введение в PyQt5: Создание графических интерфейсов с помощью Python [Электронный ресурс]. URL: <https://realpython.com/python-pyqt-gui-calculator/> (дата обращения 01.03.2023).
4. PyQt5 Tutorials [Электронный ресурс]. URL: <https://zetcode.com/gui/pyqt5/> (дата обращения 01.03.2023).
5. PyQt5 - от простого к сложному [Электронный ресурс]. URL: <https://www.youtube.com/playlist?list=PLQVvvaa0QuDdVpDFNq4xso6h0AlBNNvnC> (дата обращения 01.03.2023).

# **ПРИЛОЖЕНИЕ А**

**ИСХОДНЫЙ КОД**

Game.py

import sys  
  
import PyQt5.QtMultimedia as M  
from PyQt5.QtWidgets import QGraphicsPixmapItem  
from PyQt5.QtCore import Qt, QTimer, QUrl  
from PyQt5.QtGui import QPixmap  
import random  
import json  
import pyautogui  
  
  
class Player(QGraphicsPixmapItem):  
 speed = 3  
 def \_\_init\_\_(self, scene, score, health, parent=None):  
 super().\_\_init\_\_(parent=parent)  
 scene.addItem(self)  
 self.motion = 0  
 self.moveTimer = QTimer()  
 self.moveTimer.timeout.connect(self.move)  
 self.moveTimer.start(16)  
  
 self.score = score  
 self.health = health  
  
 self.timer = QTimer()  
 self.timer.timeout.connect(self.spawnEnemy)  
 self.timer.start(700)  
  
 url = QUrl.fromLocalFile("./res/sounds/bullet.mp3")  
 media = M.QMediaContent(url)  
 self.bulletSound = M.QMediaPlayer()  
 self.bulletSound.setMedia(media)  
 self.bulletSound.setVolume(10)  
  
 self.setFocus()  
  
 def move(self):  
 self.setPos(self.x() + self.speed \* self.motion, self.y())  
  
 # def setFocus(self):  
 # self.time = QTimer()  
 # self.time.timeout.connect(self.setFocus())  
 # self.time.start(1)  
  
 def keyPressEvent(self, e):  
  
 if e.key() == Qt.Key\_Left and self.pos().x() > 0:  
 self.motion = -1  
  
 if self.pos().x() < 0:  
 self.setPos(0, self.y())  
  
 if e.key() == Qt.Key\_Right and self.pos().x() < 800:  
 self.motion = 1  
  
 if self.pos().x() >= 685:  
 self.setPos(685, self.y())  
  
  
 if e.key() == Qt.Key\_Space and Bullet.bullets > 0:  
 Bullet.bullets -= 1  
  
 if self.bulletSound.state() == M.QMediaPlayer.PlayingState:  
 self.bulletSound.setPosition(0)  
 elif self.bulletSound.state() == M.QMediaPlayer.StoppedState:  
 self.bulletSound.play()  
  
 bullet = Bullet(self.score)  
 bullet.setPos(  
 self.x() + self.pixmap().width() / 2 -  
 bullet.pixmap().width() / 2, self.y())  
 self.scene().addItem(bullet)  
  
 def keyReleaseEvent(self, e):  
 if e.key() == Qt.Key\_Left and self.motion != 1:  
 self.motion = 0  
  
 if self.pos().x() < 0:  
 self.setPos(0, self.y())  
  
 if e.key() == Qt.Key\_Right and self.motion != -1:  
 self.motion = 0  
  
 if self.pos().x() >= 685:  
 self.setPos(685, self.y())  
  
 if e.key() == Qt.Key\_Q:  
 self.speed += 1  
  
 if e.key() == Qt.Key\_Escape:  
 self.timer.stop()  
 self.scene().removeItem(self)  
 pyautogui.press('esc')  
  
 def hide(self):  
 self.timer.stop()  
 self.scene().removeItem(self)  
  
 def spawnEnemy(self):  
 self.scene().addItem(Meteor(self.health))  
 self.scene().addItem(Meteor\_blue(self.health))  
 self.setFocus()  
  
  
class Bullet(QGraphicsPixmapItem):  
 bullets = 3  
 motion = 1  
 def \_\_init\_\_(self, score, parent=None):  
 super().\_\_init\_\_(parent)  
 # self.setRect(0, 0, 10, 30)  
 self.pmap = QPixmap("./res/images/bullet.png")  
 self.setPixmap(self.pmap)  
 self.timer = QTimer()  
 self.timer.timeout.connect(self.move)  
 self.timer.start(16)  
 self.score = score  
  
 def move(self):  
 # If the bullet collides with the enemy destroy both  
 collidingItems = self.collidingItems()  
  
 for item in collidingItems:  
 if isinstance(item, Meteor):  
 self.score.increase()  
 return self.delobj(item)  
  
 for item in collidingItems:  
 if isinstance(item, Meteor\_blue):  
 self.score.increase\_3()  
 return self.delobj(item)  
  
 self.setPos(self.x(), self.y() - 2 \* self.motion)  
 if self.pos().y() < 0:  
 Bullet.bullets += 1  
 self.scene().removeItem(self)  
  
 def delobj(self, item):  
 Bullet.bullets += 1  
 self.scene().removeItem(item)  
 self.scene().removeItem(self)  
 return  
  
  
class Meteor(QGraphicsPixmapItem):  
 def \_\_init\_\_(self, health, parent=None):  
 super().\_\_init\_\_(parent)  
 self.health = health  
 self.motion = 1  
 # set random position  
 random\_number = random.randint(0, 780)  
  
 self.ast = QPixmap("./res/images/asteroid.png")  
 self.setPixmap(self.ast)  
 self.setPos(random\_number, 0)  
  
 # connect  
 self.timer = QTimer()  
 self.timer.timeout.connect(self.move)  
 self.timer.start(16)  
  
 def move(self):  
  
 self.setRotation(self.rotation() + 0.2)  
  
 if self.pos().y() > (700 - self.pixmap().height()):  
 self.scene().removeItem(self)  
 return  
  
 collidingItems = self.collidingItems()  
 for item in collidingItems:  
 if isinstance(item, Player):  
 self.scene().removeItem(self)  
 self.health.decrease()  
 return  
  
 self.setPos(self.x(), self.y() + 1.3 \* self.motion)  
  
  
class Meteor\_blue(QGraphicsPixmapItem):  
 def \_\_init\_\_(self, health, parent=None):  
 super().\_\_init\_\_(parent)  
 self.health = health  
 self.motion = 1  
 # set random position  
 random\_number = random.randint(0, 780)  
  
 self.ast = QPixmap("./res/images/asteroid\_red.png")  
 self.setPixmap(self.ast)  
 self.setPos(random\_number, 0)  
  
 # connect  
 self.timer = QTimer()  
 self.timer.timeout.connect(self.move)  
 self.timer.start(16)  
  
 def move(self):  
  
 self.setRotation(self.rotation() - 0.1)  
  
 if self.pos().y() > (700 - self.pixmap().height()):  
 self.scene().removeItem(self)  
 return  
  
 collidingItems = self.collidingItems()  
 for item in collidingItems:  
 if isinstance(item, Player):  
 self.scene().removeItem(self)  
 self.health.decrease()  
 return  
  
 self.setPos(self.x(), self.y() + 1.7 \* self.motion)

main.py

import json  
  
from PyQt5.QtWidgets import (QApplication, QGraphicsScene,  
 QGraphicsView, QGraphicsItem)  
from PyQt5.QtCore import Qt, QUrl  
from PyQt5.QtOpenGL import QGLWidget  
from PyQt5.QtGui import QPixmap, QBrush, QImage,QIcon  
import PyQt5.QtMultimedia as M  
import game  
import gui  
import sys  
import functools  
  
scene = None  
  
with open("config.json", "r") as f:  
 file = json.load(f)  
file['health'] = 3  
with open("config.json", "w") as f:  
 json.dump(file, f)  
 print(f'main\_21\_line: {file}')  
  
with open("config.json", "r") as f:  
 file = json.load(f)  
file['score'] = 0  
with open("config.json", "w") as f:  
 json.dump(file, f)  
  
  
def gameOver(app, view, scene):  
 scene.clear()  
 gui.GameOver(scene)  
  
  
def gameStart():  
 app = QApplication(sys.argv)  
  
 # Создание сцены  
 global scene  
 scene = QGraphicsScene()  
  
 # Создание объекта, который добавляется на сцену  
 score = gui.Score(scene)  
 # health\_game = gui.Health(scene)  
 health\_game = gui.scene  
  
 player = game.Player(scene, score, health\_game)  
 player.setPixmap(QPixmap("./res/images/player.png"))  
 # player.setRect(0, 0, 100, 100)  
 # Объект должен быть в фокусе, чтобы видеть keyevents  
 player.setFlag(QGraphicsItem.ItemIsFocusable, True)  
 player.setFocus()  
  
 # Показать сцену  
 # Сначала виджет view получает событие, которое отправляет его на сцену  
 # Сцена отправляет событие объекту в фокусе  
 view = QGraphicsView(scene)  
 view.setBackgroundBrush(QBrush(QImage("./res/images/background.png")))  
 view.setAttribute(Qt.WA\_DeleteOnClose)  
 view.setViewport(QGLWidget())  
 view.setHorizontalScrollBarPolicy(Qt.ScrollBarAlwaysOff)  
 view.setVerticalScrollBarPolicy(Qt.ScrollBarAlwaysOff)  
 view.setFixedSize(800, 600)  
 scene.setSceneRect(0, 0, 800, 600)  
  
 # Установка условия окончания игры  
 health\_game.dead.connect(functools.partial(gameOver, app, view, scene))  
 # Установка позиции игрока  
 player.setPos(view.width() / 2, view.height() - player.pixmap().height())  
  
 # Воспроизведение фоновой музыки  
 url = QUrl.fromLocalFile("./res/sounds/background.wav")  
 media = M.QMediaContent(url)  
 playlist = M.QMediaPlaylist()  
 playlist.addMedia(media)  
 playlist.setPlaybackMode(M.QMediaPlaylist.Loop)  
 music = M.QMediaPlayer()  
 music.setPlaylist(playlist)  
 music.setVolume(10)  
 music.play()  
 view.show()  
 sys.exit(app.exec\_())  
  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 gameStart()

main\_4.py

import sys  
  
from PyQt5.QtWidgets import QDialog, QGraphicsScene, QGraphicsView, QGraphicsItem, \  
 QPushButton, QVBoxLayout, QWidget, QApplication, QLabel,QAction  
from PyQt5.QtGui import QPixmap, QImage, QBrush  
from PyQt5.QtCore import Qt, QUrl, QTimer  
import PyQt5.QtMultimedia as M  
import sys  
import json  
import gui  
from main import scene  
from gui import Health  
import game  
  
from PyQt5.QtCore import Qt  
from PyQt5.QtGui import QCursor,QIcon  
from PyQt5.QtWidgets import QApplication, QWidget, QHBoxLayout, QVBoxLayout, QPushButton, QLabel, QGridLayout, \  
 QSizePolicy, QSpacerItem  
  
main\_widget = None  
background = 'images/background.png'  
button\_style = 'color: #fff; ' \  
 'background: green; ' \  
 'width: 250px; ' \  
 'height: 80px;' \  
 'font: bold 20px Arial;' \  
 'border-radius: 16px'  
text\_style = 'color: #fff; ' \  
 'font: normal 18px Arial;'  
  
class Game\_yay\_Start(QDialog):  
 def close(self):  
 print("youlose")  
  
 def \_\_init\_\_(self, parent=None):  
 super().\_\_init\_\_(parent)  
 self.setWindowTitle('Star defender')  
 self.setSizePolicy(800, 600)  
 self.setStyleSheet("background-color: green;")  
  
 with open("config.json", 'r') as f:  
 file = json.load(f)  
 if file['health'] <= 0:  
 file['health'] = 3  
 with open("config.json", "w") as f:  
 json.dump(file, f)  
 print(f'main\_21\_line: {file}')  
  
 self.setWindowIcon(QIcon('icon.png'))  
 self.scene = QGraphicsScene()  
 self.view = QGraphicsView(self.scene)  
 self.view.setSceneRect(0, 0, 800, 600)  
 self.view.setHorizontalScrollBarPolicy(Qt.ScrollBarAlwaysOff)  
 self.view.setVerticalScrollBarPolicy(Qt.ScrollBarAlwaysOff)  
 self.view.setBackgroundBrush(QBrush(QImage("./res/images/background.png")))  
  
 self.score = gui.Score(self.scene)  
 self.health = gui.Health(self.scene)  
 self.player = game.Player(self.scene, self.score, self.health)  
 self.player.setPos(self.view.width() / 2, self.view.height() - self.player.pixmap().height())  
 self.player.setPixmap(QPixmap("./res/images/player.png"))  
 self.player.setFlag(QGraphicsItem.ItemIsFocusable, True)  
 self.player.setFocus()  
  
 self.health.dead.connect(self.gameOver)  
  
 layout = QVBoxLayout()  
 layout.addWidget(self.view)  
 self.setLayout(layout)  
  
 self.url = QUrl.fromLocalFile("/res/sounds/background.wav")  
 media = M.QMediaContent(self.url)  
 playlist = M.QMediaPlaylist()  
 playlist.addMedia(media)  
 playlist.setPlaybackMode(M.QMediaPlaylist.Loop)  
 self.music = M.QMediaPlayer()  
 self.music.setPlaylist(playlist)  
 self.music.setVolume(10)  
 self.music.play()  
  
 quit = QAction('Quit', self)  
 quit.triggered.connect(self.close)  
  
 # self.rejected.connect(self.handle\_rejected)  
  
 def handle\_rejected(self):  
 game.Player.hide(self)  
  
 def gameOver(self):  
 self.music.stop()  
 self.scene.clear()  
 gui.GameOver(self.scene)  
 self.close()  
  
  
class Main(QWidget):  
 def \_\_init\_\_(self, parent=None):  
 super().\_\_init\_\_(parent)  
  
 self.main\_widget = None  
 self.main\_layout = QHBoxLayout()  
  
 self.setWindowTitle('Menu')  
 self.setLayout(self.main\_layout)  
 self.setWindowIcon(QIcon('icon.png'))  
 self.setGeometry(800, 500, 800, 600)  
 self.setStyleSheet(f"background-image: url({background});background-color: black;")  
  
 self.open\_main\_widget('openMenu')  
  
 def open\_main\_widget(self, name\_widget):  
 if name\_widget == 'openMenu':  
 self.main\_widget = self.MenuWindow()  
 self.main\_layout.addWidget(self.main\_widget)  
 print('openMenu')  
 if name\_widget == 'openRules':  
 self.main\_widget = self.RulesWindow()  
 print('openRules')  
 if name\_widget == 'openShop':  
 self.main\_widget = self.ShopWindow()  
 print('openShop')  
 if name\_widget == 'openGame':  
 self.main\_widget.hide()  
 print('openGame')  
  
 item = self.main\_layout.itemAt(0)  
 self.main\_layout.removeItem(item)  
  
 self.main\_layout.addWidget(self.main\_widget)  
  
 def CloseMenuOpenShop(self):  
 self.open\_main\_widget('openShop')  
 self.btnPlay.deleteLater()  
 self.btnShop.deleteLater()  
 self.btnRulesAndControls.deleteLater()  
 self.btnExit.deleteLater()  
  
 def CloseMenuOpenRules(self):  
 self.open\_main\_widget('openRules')  
 self.btnPlay.deleteLater()  
 self.btnShop.deleteLater()  
 self.btnRulesAndControls.deleteLater()  
 self.btnExit.deleteLater()  
  
 def MenuWindow(self):  
 self.btnPlay = QPushButton('Играть', self)  
 self.btnPlay.setStyleSheet(button\_style)  
 self.btnPlay.setCursor(QCursor(Qt.PointingHandCursor))  
 self.btnPlay.clicked.connect(self.openGame)  
  
 self.btnShop = QPushButton('Магазин', self)  
 self.btnShop.setStyleSheet(button\_style)  
 self.btnShop.setCursor(QCursor(Qt.PointingHandCursor))  
 self.btnShop.clicked.connect(lambda: self.CloseMenuOpenShop())  
  
 self.btnRulesAndControls = QPushButton('Правила', self)  
 self.btnRulesAndControls.setStyleSheet(button\_style)  
 self.btnRulesAndControls.setCursor(QCursor(Qt.PointingHandCursor))  
 self.btnRulesAndControls.clicked.connect(lambda: self.CloseMenuOpenRules())  
  
 self.btnExit = QPushButton('Выйти', self)  
 self.btnExit.setStyleSheet(button\_style)  
 self.btnExit.setCursor(QCursor(Qt.PointingHandCursor))  
 self.btnExit.clicked.connect(lambda: exit())  
  
 self.menu\_layout = QVBoxLayout()  
 self.menu\_layout.addStretch()  
 self.menu\_layout.addWidget(self.btnPlay)  
 self.menu\_layout.addWidget(self.btnShop)  
 self.menu\_layout.addWidget(self.btnRulesAndControls)  
 self.menu\_layout.addWidget(self.btnExit)  
 self.menu\_layout.addStretch()  
  
 self.menu\_widget = QWidget()  
 self.menu\_widget.setFixedWidth(200)  
 self.menu\_widget.setLayout(self.menu\_layout)  
  
 self.window\_layout = QHBoxLayout()  
 self.window\_layout.addWidget(self.menu\_widget)  
  
 self.menu\_window = QWidget()  
 self.menu\_window.setLayout(self.window\_layout)  
 return self.menu\_window  
  
 def RulesClose(self):  
 self.open\_main\_widget('openMenu')  
 self.rules\_widget.deleteLater()  
 self.rules\_to\_menu.deleteLater()  
  
 def RulesWindow(self):  
 self.rules\_widget = QLabel()  
 self.rules\_widget.setFixedWidth(800)  
 self.rules\_widget.setStyleSheet(text\_style)  
 self.rules\_widget.setWordWrap(True)  
 self.rules\_widget.setText(  
 'Управления кораблем осуществляется с помощью стрелок.\n'   
 'Стрелка влево – перемещение корабля влево,\n'  
 'Стрелка вправо – перемещение корабля вправо,\n'  
 'Стрельба – пробел,\n'  
 'Escape во время игры - закрыть окно игры и сохранить прогресс.\n'  
 '\n'  
 'Правила игры:'  
 'Игроку необходимо набрать наибольшее количество очков.'  
 'Игрок может двигаться только вправо и влево. Игрок не может выйти за экран.'  
 'для получения которых необходимо уничтожать метеориты (1 очко за серый, 3 очка за синий). '  
 'При столкновении с метеоритом происходит потеря одной единицы жизни, '  
 'игра заканчивается если количество жизней = 0.'  
 )  
  
 self.rules\_to\_menu = QPushButton('Назад', self)  
 self.rules\_to\_menu.setFixedWidth(200)  
 self.rules\_to\_menu.setStyleSheet(button\_style)  
 self.rules\_to\_menu.setCursor(QCursor(Qt.PointingHandCursor))  
 self.rules\_to\_menu.clicked.connect(lambda: self.RulesClose())  
  
 self.window\_layout = QVBoxLayout()  
 self.window\_layout.addWidget(self.rules\_widget)  
 self.window\_layout.addWidget(self.rules\_to\_menu, alignment=Qt.AlignCenter)  
  
 self.rules\_window = QWidget()  
 self.rules\_window.setLayout(self.window\_layout)  
 return self.rules\_window  
  
 def Shopclose(self):  
  
 self.btnHP.deleteLater()  
 self.btnMaxBullet.deleteLater()  
 self.btnShipSpeed.deleteLater()  
 self.btnBulletSpeed.deleteLater()  
 self.btnExitToMenu.deleteLater()  
 self.open\_main\_widget('openMenu')  
  
 def message\_few\_score(self):  
 self.few\_label = QLabel("Недостаточно очков")  
 self.few\_label.setStyleSheet(f"{text\_style}")  
 timer = QTimer()  
 timer.singleShot(1000, self.few\_label.deleteLater)  
  
 def HPplus(self, score):  
 # health\_default = gui.health  
 # print(health\_default)  
 self.score\_label.setText(f"Score: {score}")  
 with open("config.json", "r") as f:  
 file = json.load(f)  
  
 # with open("config.json", "w") as f:  
 # json.dump(file, f)  
 if score >= 200:  
 # gui.Health(scene=scene).setPlainText(f"Health: {gui.health}")  
 file['score'] = score - 200  
 self.score -= 200  
 with open("config.json", "w") as f:  
 json.dump(file, f)  
 print(f'main4\_231\_line: {file}')  
 gui.increase()  
 else:  
 self.message\_few\_score()  
 self.window\_layout.addWidget(self.few\_label, 0, 0)  
 self.score\_label.setText(f"Очки: {score}")  
  
 def BulletsPlus(self, score):  
 with open("config.json", "r") as f:  
 file = json.load(f)  
 with open("config.json", "w") as f:  
 json.dump(file, f)  
 if score >= 100:  
 # gui.Health(scene=scene).setPlainText(f"Health: {gui.health}")  
 file['score'] = score - 100  
 self.score -= 100  
 with open("config.json", "w") as f:  
 json.dump(file, f)  
 print(f'main4\_231\_line: {file}')  
 self.score\_label.setText(f"Очки: {score}")  
 game.Bullet.bullets += 1  
 else:  
 self.message\_few\_score()  
 self.window\_layout.addWidget(self.few\_label, 0, 1)  
 self.score\_label.setText(f"Очки: {score}")  
  
 def ShipSpPlus(self, score):  
 with open("config.json", "r") as f:  
 file = json.load(f)  
 with open("config.json", "w") as f:  
 json.dump(file, f)  
 if score >= 50:  
 # gui.Health(scene=scene).setPlainText(f"Health: {gui.health}")  
 file['score'] = score - 50  
 self.score -= 50  
 with open("config.json", "w") as f:  
 json.dump(file, f)  
 print(f'main4\_231\_line: {file}')  
 game.Player.speed += 1  
 else:  
 self.message\_few\_score()  
 self.window\_layout.addWidget(self.few\_label, 1, 0)  
 self.score\_label.setText(f"Очки: {score}")  
  
 def BulSpPlus(self, score):  
 with open("config.json", "r") as f:  
 file = json.load(f)  
 with open("config.json", "w") as f:  
 json.dump(file, f)  
 if score >= 50:  
 # gui.Health(scene=scene).setPlainText(f"Health: {gui.health}")  
 file['score'] = score - 50  
 self.score -= 50  
 with open("config.json", "w") as f:  
 json.dump(file, f)  
 print(f'main4\_231\_line: {file}')  
 game.Bullet.motion += 1  
 else:  
 self.message\_few\_score()  
 self.window\_layout.addWidget(self.few\_label, 1, 1)  
 self.score\_label.setText(f"Очки: {score}")  
  
 def ShopWindow(self):  
 with open("config.json", "r") as f:  
 data = json.load(f)  
  
 score = data["score"]  
  
 """  
 ПОПРАВИТЬ БАГ С НАЖАТИЕМ  
 """  
  
  
 self.score = score  
 self.score\_label = QLabel(f"Очки: {self.score}")  
 self.score\_label.setStyleSheet("font-size: 20px; color: red;")  
  
 self.btnHP = QPushButton('Добавить жизнь\n +1 (200)', self)  
 self.btnHP.setStyleSheet(button\_style)  
 self.btnHP.setCursor(QCursor(Qt.PointingHandCursor))  
 self.btnHP.clicked.connect(lambda: self.HPplus(self.score))  
  
 self.btnMaxBullet = QPushButton('Увеличить количество\n пуль +1 (100)', self)  
 self.btnMaxBullet.setStyleSheet(button\_style)  
 self.btnMaxBullet.setCursor(QCursor(Qt.PointingHandCursor))  
 self.btnMaxBullet.clicked.connect(lambda: self.BulletsPlus(self.score))  
  
 self.btnShipSpeed = QPushButton('Увеличить скорость\n корабля (50)', self)  
 self.btnShipSpeed.setStyleSheet(button\_style)  
 self.btnShipSpeed.setCursor(QCursor(Qt.PointingHandCursor))  
 self.btnShipSpeed.clicked.connect(lambda: self.ShipSpPlus(self.score))  
  
 self.btnBulletSpeed = QPushButton('Увеличить скорость\n пуль (50)', self)  
 self.btnBulletSpeed.setStyleSheet(button\_style)  
 self.btnBulletSpeed.setCursor(QCursor(Qt.PointingHandCursor))  
 self.btnBulletSpeed.clicked.connect(lambda: self.BulSpPlus(self.score))  
  
 self.btnExitToMenu = QPushButton('Выход', self)  
 self.btnExitToMenu.setStyleSheet(button\_style)  
 self.btnExitToMenu.setCursor(QCursor(Qt.PointingHandCursor))  
 self.btnExitToMenu.clicked.connect(lambda: self.Shopclose())  
  
 self.btnHP.setMinimumWidth(150)  
 self.btnMaxBullet.setMinimumWidth(150)  
 self.btnShipSpeed.setMinimumWidth(150)  
 self.btnBulletSpeed.setMinimumWidth(150)  
 self.btnExitToMenu.setMinimumWidth(150)  
  
 self.window\_layout = QGridLayout()  
 self.window\_layout.addWidget(self.btnHP, 0, 0)  
 self.window\_layout.addWidget(self.btnMaxBullet, 0, 1)  
 self.window\_layout.addWidget(self.btnShipSpeed, 1, 0)  
 self.window\_layout.addWidget(self.btnBulletSpeed, 1, 1)  
 self.window\_layout.addWidget(self.score\_label, 0, 3, 1, 2, alignment=Qt.AlignCenter)  
 self.window\_layout.addWidget(self.btnExitToMenu, 2, 0, 1, 2)  
  
 self.Shop\_widget = QWidget()  
 self.Shop\_widget.setFixedWidth(600)  
 self.Shop\_widget.setLayout(self.window\_layout)  
  
 return self.Shop\_widget  
  
 def openGame(self):  
 self.game\_dialog = Game\_yay\_Start()  
 self.game\_dialog.exec\_()  
  
 def keyPressEvent(self, event):  
 if event.key() == Qt.Key\_Escape:  
 self.close()  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app = QApplication(sys.argv)  
 window = Main()  
 # window.showMaximized()  
 window.show()  
 sys.exit(app.exec())

gui.py

from PyQt5.QtWidgets import QGraphicsTextItem  
from PyQt5.QtCore import Qt, QTimer  
from PyQt5.QtGui import QFont  
from PyQt5.QtCore import pyqtSignal  
import json  
import game  
  
with open("config.json", "r") as f:  
 file = json.load(f)  
health = file['health']  
print(f'gui\_11\_line: {health}')  
  
  
  
class Score(QGraphicsTextItem):  
  
 def \_\_init\_\_(self, scene, parent=None):  
 super().\_\_init\_\_(parent)  
 scene.addItem(self)  
 self.score = 0  
 self.setFont(QFont("Arial", 15))  
 self.setPlainText(f"Очки: {self.score}")  
 self.setDefaultTextColor(Qt.blue)  
  
 def increase(self):  
 self.score += 1  
 with open("config.json", "r") as f:  
 file = json.load(f)  
 self.setPlainText(f"Очки: {self.score}")  
 score\_dict = {"score": self.score, "health": file['health']}  
  
 with open("config.json", "w") as f:  
 json.dump(score\_dict, f)  
  
 def increase\_3(self):  
 self.score += 3  
 with open("config.json", "r") as f:  
 file = json.load(f)  
 self.setPlainText(f"Очки: {self.score}")  
 score\_dict = {"score": self.score, "health": file['health']}  
  
 with open("config.json", "w") as f:  
 json.dump(score\_dict, f)  
  
  
class Health(QGraphicsTextItem):  
 dead = pyqtSignal()  
  
 def \_\_init\_\_(self, scene, parent=None):  
 super().\_\_init\_\_(parent)  
 scene.addItem(self)  
 # print(scene)  
  
 with open("config.json", "r") as f:  
 file = json.load(f)  
 health2 = file['health']  
  
 self.setFont(QFont("Arial", 15))  
 self.setPlainText(f"Жизни: {health2}")  
 print(f'жизнь на поле(gui\_52\_line): {health2}')  
 self.setDefaultTextColor(Qt.red)  
 rect = self.boundingRect()  
 self.setPos((800) - rect.width(), self.y())  
  
 def decrease(self):  
 # global health  
 # health -= 1  
  
 with open("config.json", "r") as f:  
 file = json.load(f)  
  
 file['health'] -= 1  
 print(f"жизь: {file['health']}")  
 print(f'gui\_70\_line: {file}')  
 with open("config.json", "w") as f:  
 json.dump(file, f)  
 self.setPlainText(f"Жизни: {file['health']}")  
 if file['health'] <= 0:  
 self.dead.emit()  
  
  
def increase():  
 with open("config.json", "r") as f:  
 file = json.load(f)  
 file['health'] += 1  
 print(f"жизь: {file['health']}")  
 print(f'gui\_70\_line: {file}')  
 with open("config.json", "w") as f:  
 json.dump(file, f)  
 # Health(scene=scene).setPlainText(f"Health: {health}")  
  
  
class GameOver(QGraphicsTextItem):  
 def \_\_init\_\_(self, scene, parent=None):  
 super().\_\_init\_\_(parent)  
 scene.addItem(self)  
 game.Player.speed = 3  
 game.Bullet.bullets = 3  
 game.Bullet.motion = 1  
 with open("config.json", "r") as f:  
 file = json.load(f)  
 self.score = file['score']  
 self.setPlainText(f" Вы набрали очков:{self.score}\n Игра окончена\n нажмите на Escape чтобы выйти в главное меню")  
 self.setDefaultTextColor(Qt.red)  
 self.setFont(QFont("Arial", 20))  
 rect = self.boundingRect()  
 self.setPos((800 / 2) - rect.width() / 2, (600 / 2) - rect.height() / 2)  
 print(self.boundingRect(), rect.width())

# **ПРИЛОЖЕНИЕ Б**

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

# **ПРИЛОЖЕНИЕ В**

**ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ**

# **ПРИЛОЖЕНИЕ Г**

**РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**