**олимпиада школьников «Шаг в будущее»  
По профилю «Инженерное дело»**

*регистрационный номер*

Секция: \_\_\_\_\_\_\_\_\_\_\_\_\_\_Компьютерные системы (ИУ3, ИУ6)\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*название секции*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_ПРОГРАММНЫЙ ПРОДУКТ “КОМПЬЮТЕРНАЯ ИГРА ТАНКИ90”\_\_\_\_

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

*название работы*

Автор: *\_\_\_\_\_*Акимов Владислав Евгеньевич*\_\_\_\_*

*фамилия, имя, отчество*

**\_\_\_\_\_\_\_\_** **ГБОУ Школа № 1811\_\_\_\_\_\_\_**

*наименование учебного заведения, класс*

**Научный руководитель: \_\_\_\_Пономарев Андрей Дмитриевич\_\_\_\_**

*фамилия, имя, отчество*

**\_\_\_\_\_\_\_МГТУ им. Н.Э. Баумана\_\_\_\_\_\_\_\_**

*место работы*

**\_\_\_старший преподаватель каф. ИУ6\_\_\_**

*звание, должность*

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

*подпись научного руководителя*

**Москва – 2024**

#### АННОТАЦИЯ

Представленная работа демонстрирует процесс разработки на языке Python компьютерной игры с возможностью одновременного участия двух человек. За основу разработки взята настоящая классика игровой индустрии Tanks1990 для платформы Dendy.

Главная идея проекта – создать законченный продукт на языке Python с целью демонстрации навыков программирования и навыков анализа и обучения новым подходам. Именно для этих целей было решено при написании кода опираться на распространенные подходы в коммерческой разработке ПО.

В работе уделяется внимание контролю версионности с помощью GitHub, организации стандартизированного написания кода с использованием стилей и рекомендаций PEP8 и выбора общей среды для разработки PyCharm.

Дополнительно, перед началом реализации проекта была проведена необходимая подготовка и разработаны диаграмма классов, схема компоновки и схема взаимодействия модулей для контроля этапов разработки программного продукта и понимания взаимодействия модулей в процессе написания кода.

В результате получился готовый программный продукт с возможностью запуска под Windows из собранного .exe файла, который позволяет двум игрокам управлять танками на предварительно сгенерированном поле из разных блоков (с учетом того, что генерация всегда позволит завершить игру). Дополнительно к проекту прилагается носитель с проектом для ознакомления с модулями и стилем написания кода.

#### СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 5](#_Toc160565491)

[1. Исследование существующих технологий 6](#_Toc160565492)

[1.1 История и описание игрового проекта 6](#_Toc160565493)

[1.1.1 Зарождение 6](#_Toc160565494)

[1.1.2 Заключение 7](#_Toc160565495)

[1.2 Технологии и инструменты разработки 7](#_Toc160565496)

[1.2.1 Python 7](#_Toc160565497)

[1.2.2 Библиотека Pygame 9](#_Toc160565498)

[1.2.3 Среда разработки PyCharm 10](#_Toc160565499)

[1.2.4 Стандарты написания кода 11](#_Toc160565500)

[1.2.4.1 Отступы 12](#_Toc160565501)

[1.2.4.2 Максимальная длина строки кода 12](#_Toc160565502)

[1.2.4.3 Пробелы 13](#_Toc160565503)

[1.2.4.4 Наименования 14](#_Toc160565504)

[1.2.4.5 Комментарии 15](#_Toc160565505)

[1.2.5 Заключение 16](#_Toc160565506)

[2. Разработка игры и написание кода 17](#_Toc160565507)

[2.1 Анализ 17](#_Toc160565508)

[2.1.1 Выбор типов данных 17](#_Toc160565509)

[2.1.2 Структура приложения 17](#_Toc160565510)

[2.1.3 Разработка диаграммы вариантов использования 18](#_Toc160565511)

[2.1.4 Разработка диаграммы классов 18](#_Toc160565512)

[2.1.5 Разработка схемы взаимодействия с папками файлов 20](#_Toc160565513)

[2.1.6 Заключение 22](#_Toc160565514)

[2.2 Разработка графического интерфейса 22](#_Toc160565515)

[2.2.1 Разработка интерфейса заставочного экрана 22](#_Toc160565516)

[2.2.2 Разработка интерфейса дополнительного экрана 23](#_Toc160565517)

[2.2.3 Разработка интерфейса игрового процесса 23](#_Toc160565518)

[2.2.4 Разработка интерфейса паузы и победы 24](#_Toc160565519)

[2.2.5 Заключение 25](#_Toc160565520)

[2.3 Тестирование 25](#_Toc160565521)

[2.3.1 Разработка технологии тестирования 26](#_Toc160565522)

[2.3.2 Разработка тестов и результаты тестирования 28](#_Toc160565523)

[ЗАКЛЮЧЕНИЕ 29](#_Toc160565524)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 31](#_Toc160565525)

[ПРИЛОЖЕНИЕ А 32](#_Toc160565526)

[ПРИЛОЖЕНИЕ Б 33](#_Toc160565527)

[ПРИЛОЖЕНИЕ В 34](#_Toc160565528)

[ПРИЛОЖЕНИЕ Г 39](#_Toc160565529)

[Функциональное и оценочное тестирование 39](#_Toc160565530)

[Оценочное тестирование 41](#_Toc160565534)

### ВВЕДЕНИЕ

В современном мире компьютерные игры играют значительную роль в повседневной жизни людей, предоставляя им возможность отдохнуть, расслабиться, и даже развивать навыки. Одной из наиболее популярных и любимых жанров являются игры, в которых игроки управляют различными боевыми машинами, такими, как танки. Классическая игра "Танки" имеет свою уникальную историю, и до сих пор остается популярной по всему миру. Однако, с развитием технологий, графических и звуковых возможностей, а также с появлением новых механик геймплея, возникает потребность в модернизированных версиях классических игр. В этой работе представляется концепция и реализация программного продукта в виде компьютерной игры "Танки90", что представляет собой обновленную и улучшенную версию классической игры “Танки”.

Целью данного проекта является создание улучшенного игрового процесса, а также более современного, адаптированного под современные платформы продукта с различными возможностями для игроков. Данный проект также направлен на усовершенствование механики игры, графического и аудиовизуального оформления, а также возможности игры в многопользовательском режиме.

Целью работы является не только создание программного продукта, но и изучение и применение современных технологий и методик разработки приложений. В этой связи, данная работа предполагает анализ уже существующих информационных ресурсов, статей, книг и специализированных журналов по теме разработки игр, компьютерной графики, а также методов программирования многопользовательских приложений.

### 1. Исследование существующих технологий

### 1.1 История и описание игрового проекта

История классической игры “Танки” насчитывает десятилетия популярности и влияния на игровую индустрию. Эта игра, изначально представленная в виде аркады, получила признание благодаря своей простой, но захватывающей механике игры, аркадному геймплею и возможности мультиплеера. Давайте взглянем на историю классической игры "Танки" с момента ее зарождения и до современности.

### 1.1.1 Зарождение

В начале 1990-х годов компания Dendy выпустила легендарную серию военных игр под названием "Танчики". Они были вдохновлены оригинальной японской аркадной игрой под названием "Battle City", выпущенной в 1985 году компанией Namco. "Battle City" была популярной игрой в Японии и затем была адаптирована для платформы Dendy.

Итак, зарождение танчиков Dendy началось с аркадной классики. Эта игра достаточно быстро завоевала популярность благодаря захватывающему игровому процессу и увлекательным сражениям между танками. Она привлекла внимание как детей, так и взрослых благодаря своей простой, но захватывающей механике игры.

Компания Dendy заметила успех "Battle City" и решила создать собственную версию игры для своей платформы. Так появилась серия игр "Танчики", которая стала одной из самых популярных серий игр для Dendy.

В "Танчиках" игрок управлял танком, стреляя по вражеским танкам и защищая свою базу. Со временем игры серии "Танчики" развились и стали более сложными, добавляя новые уровни, врагов и возможности для игроков.

Итак, именно благодаря успеху "Battle City" и желанию Dendy привнести что-то подобное на свою платформу, родилась легендарная серия игр "Танчики". Эти игры оставили незабываемые воспоминания и до сих пор вызывают ностальгию у любителей классических игр. Кроме того, эволюция "Танчиков" также предполагала развитие многопользовательского режима. Позднее игры в серии "Танчики" стали предлагать дополнительные режимы совместной игры, что разнообразило возможности для игроков наслаждаться игрой вместе.

### 1.1.2 Заключение

Подводя итог краткому историческому экскурсу в историю развития игрового направления, можно сказать, что история классической игры “Танки” является неотъемлемой частью культуры видеоигр, символизируя захватывающую механику, увлекательный геймплей и влияние на развитие игровой индустрии; поэтому целью работы является внести свой вклад в развитие этой сферы, создав собственную версию “Танков”.

### 1.2 Технологии и инструменты разработки

### 1.2.1 Python

Python – высокоуровневый интерпретируемый язык программирования, при это Python был создан в конце 1980-х, а первая версия, Python 0.9.0, была выпущена в феврале 1991 года Гвидо ван Россумом, голландским программистом. Гвидо ван Россум хотел создать язык программирования, который был бы прост в использовании, но при этом мощный и с элегантным синтаксисом. Название языка Python было вдохновлено популярным на тот момент британским комедийным телешоу "Летающий цирк Монти Пайтона". Python характеризуется своей простотой в изучении, читаемостью кода и разнообразием применений.

Одной из выдающихся особенностей Python является его простота в написании и читаемость кода. Python использует отступы для обозначения блоков кода, что делает его более читаемым и удобным для новичков и опытных разработчиков.

Еще одно важное преимущество языка Python – это тот факт, что он универсальный язык программирования, который нашел широкое применение в самых разных областях. Рассмотрим некоторые из них:

* **Веб-разработка**

Python широко используется для разработки веб-приложений. Фреймворки, такие как Django, Flask, Pyramid предоставляют разработчикам мощные инструменты для создания веб-серверов, сайтов, веб-приложений и API. Python также используется для веб-скрапинга, автоматизации тестирования веб-приложений и многих других целей.

* **Анализ данных и научные вычисления**

Python обрел большую популярность в области научных вычислений и анализа данных благодаря библиотекам как NumPy, SciPy, Pandas, Matplotlib, которые предоставляют мощные инструменты для работы с данными, включая чтение, обработку, визуализацию и анализ.

* **Машинное обучение и искусственный интеллект**

Библиотеки, такие как TensorFlow, Keras, PyTorch, scikit-learn, делают Python одним из основных языков для разработки и прототипирования моделей машинного обучения и искусственного интеллекта. Python предлагает удобный интерфейс для обучения моделей и работы с большими объемами данных.

* **Автоматизация и скриптинг**

Python используется для автоматизации рутины, создания скриптов и инструментов для управления файловой системой, сетевыми операциями, администрированием операционных систем и конфигурацией программ.

* **Разработка игр**

Python популярен среди разработчиков игр, особенно для создания прототипов, разработки геймплея и использования в игровых скриптах.

* **Видео и аудио обработка**

Python используется для обработки видео и аудио данных, редактирования медиа-контента, создания графики и других операций, связанных с мультимедиа.

Далее стоит отметить, что Python имеет огромное сообщество разработчиков по всему миру, которые активно создают новые библиотеки, фреймворки и инструменты для различных целей. Это помогает Python оставаться в числе самых популярных языков программирования. Python поддерживает различные платформы, такие как Windows, macOS и Linux, что позволяет использовать его для разработки программного обеспечения на различных устройствах и операционных системах.

В заключение можно сказать, что Python – это мощный и гибкий язык программирования, который подходит как для начинающих, так и для опытных разработчиков. Его разнообразие применений, понятный синтаксис и обширная экосистема делают его одним из самых популярных языков программирования в мире. Именно тот факт, что знание языка Python открывает огромное количество направлений развития – от компьютерных игр до специализированного научного ПО и работы с искусственным интеллектом – и стал определяющим фактором при выборе мной среды разработки.

### 1.2.2 Библиотека Pygame

Pygame представляет собой бесплатную и открытую библиотеку Python, предназначенную для создания компьютерных игр и мультимедийных приложений. Она предоставляет разработчикам инструменты для создания игр и мультимедийных приложений с использованием Python.

Ниже представлены основные возможности рассматриваемой библиотеки:

* **Работа с графикой и звуком**: Pygame предоставляет функции для рисования 2D-графики, включая спрайты, фоновые изображения, анимацию и многое другое. Кроме того, библиотека обеспечивает возможности для работы с звуком, воспроизведения аудио и управления звуковыми эффектами.
* **Управление вводом**: Pygame предоставляет возможности для обработки пользовательского ввода, такие как обработка клавиш, мыши и геймпадов, что делает ее идеальной для создания игровых механик.
* **Кроссплатформенность**: Pygame совместима с различными платформами, что позволяет разрабатывать игры для различных операционных систем.
* **Простота использования**: Pygame разработана так, чтобы облегчить создание игровых приложений, особенно для начинающих разработчиков. Библиотека предоставляет удобные и понятные средства для работы с графикой и звуком.

Таким образом можно сделать вывод, что Pygame представляет собой мощный инструмент для создания 2D-игр и мультимедийных приложений с использованием Python. Благодаря его простоте использования и обширным возможностям работы с графикой и звуком, Pygame стал популярным выбором для начинающих и опытных разработчиков. И из вышеперечисленного следует, что Pygame прекрасно подходит для моего проекта.

### 1.2.3 Среда разработки PyCharm

PyCharm является одним из лучших выборов для разработчиков Python, особенно из-за следующих причин:

* **Специализация на Python**: PyCharm предназначен специально для разработки на Python, что делает его идеальным выбором для тех, кто работает исключительно с этим языком программирования. Имеет мощные инструменты для анализа кода, автодополнения и отладки, специфически нацеленные на Python.
* **Интегрированная среда разработки**: PyCharm предлагает полностью интегрированную среду разработки с широким спектром инструментов, включая редактор кода, отладчик, терминал и систему контроля версий, что делает процесс разработки более эффективным.
* **Поддержка виртуальных окружений**: PyCharm облегчает работу с виртуальными окружениями Python, позволяя изолировать зависимости проекта и управлять их установкой и обновлением.
* **Широкий набор возможностей**: PyCharm предоставляет мощные инструменты для анализа кода, рефакторинга, работу с базами данных и веб-технологиями, что делает его полезным даже для более широкого спектра разработчиков.
* **Активное сообщество и поддержка**: PyCharm имеет активное сообщество пользователей и разработчиков, а также широкий набор обучающих материалов и документации, что облегчает обучение и использование этой среды разработки.

Но когда речь идет о сравнении PyCharm с другими средами разработки, важно учитывать различные факторы, такие как функциональные возможности, производительность, поддержка языков программирования и общее удобство использования

Выбор между PyCharm и другими средами разработки зависит от индивидуальных потребностей, предпочтений и особенностей проекта. PyCharm обычно предпочтителен для разработки на Python, особенно для крупных и сложных проектов, где важна интегрированная отладка, анализ кода и другие инструменты, предоставляемые данной IDE. Поэтому, в целом, выбор PyCharm оправдан для разработчиков, которые хотят эффективно работать с Python, получая доступ к множеству инструментов и ресурсов, которые облегчают разработку качественного кода и приложений.

### 1.2.4 Стандарты написания кода

В крупных компаниях, на проектах, которые пишутся сразу несколькими командами разработчиков очень важно следовать общим стандартам именования и форматирования кода. Именно для этого и существуют специализированные стандарты написания кода, про которые пойдет речь ниже. В качестве основы для стандарта написания кода на Python используется PEP 8 (PEP - Python Enhancement Proposal) - официальное руководство по написанию кода на Python. PEP 8 фокусируется на вопросах читаемости кода и обеспечивает общие правила форматирования, структурирования и документирования кода на Python. Вот общий план написания кода в соответствии с PEP 8:

### 1.2.4.1 Отступы

Рекомендуется использовать 4 пробела для каждого уровня отступа вместо табуляции - это обеспечивает унификацию и читаемость кода на различных устройствах и в различных редакторах.

Вот пример использования отступов в Python в соответствии со стандартом PEP8:

Листинг 1

# Плохой код: использование табуляции для отступов

def my\_function():

print("Hello, world!")

# Хороший код: использование 4 пробелов для отступов

def my\_function():

print("Hello, world!")

### 1.2.4.2 Максимальная длина строки кода

Рекомендуется соблюдать ограничение в 79 символов на строку кода, чтобы обеспечить удобство чтения на широком спектре устройств и редакторов, за исключением комментариев и докстрингов, которые могут быть длиннее.

Длинные строки кода следует разбивать на подходящие по смыслу блоки.

Вот пример использования ограничения длины строки кода на Python в соответствии со стандартом PEP 8:

Листинг 2

# Плохой код: превышение максимальной длины строки

my\_long\_string = "This is a very long string that exceeds the recommended line length and makes the code less readable."

# Хороший код: разбиение длинной строки на подходящие блоки

my\_long\_string = ("This is a very long string that does not exceed the recommended line length "

"and is split into smaller, more readable blocks.")

В примере длинная строка была разбита на две строки, чтобы соблюсти ограничение в 79 символов. Соблюдение ограничения длины строки на коде на Python способствует читаемости и пониманию кода.

### 1.2.4.3 Пробелы

Следует избегать пробелов в конце строк и вставлять 2 пустые строки между функциями и классами для улучшения читаемости.

Рассмотрим примеры использования пробелов в коде на Python, согласно стандарту PEP 8:

Избегание пробелов в конце строк:

Листинг 3

# Плохой код: пробелы в конце строки

result = some\_function()\_

# Хороший код: отсутствие пробелов в конце строки

result = some\_function()

Вставка пустой строки между функциями и классами:

Листинг 4

# Плохой код: отсутствие пустых строк между функциями

def function1():

pass

def function2():

pass

# Хороший код: наличие пустых строк между функциями

def function1():

pass

def function2():

pass

В первом примере приводится ситуация, когда пробелы в конце строки могут непреднамеренно добавиться, что следует избегать согласно стандарту PEP 8.

Во втором примере показывается, что использование пустых строк между функциями способствует улучшению читаемости кода, делая его более структурированным и понятным.

Использование пробелов в соответствии с рекомендациями PEP 8 обеспечивает создание чистого, читаемого и структурированного кода на Python, что в свою очередь способствует его удобству поддержки и последующих изменений.

### 1.2.4.4 Наименования

Правильное наименование переменных, функций, классов и модулей является важной составляющей чистого и понятного кода.

Вот несколько ключевых рекомендаций по наименованию, описанных в PEP 8:

* **Имена переменных**: в переменных следует использовать маленькие буквы, а слова разделять символами подчеркивания ("snake\_case"). Например: my\_variable, user\_name, и т.д.
* **Имена функций и методов**: аналогично переменным, для функций и методов также используется стиль "snake\_case". Например: calculate\_area(), get\_user\_info(), и т.д.
* **Имена классов**: для имен классов следует использовать стиль "CapWords", где каждое слово в имени пишется с заглавной буквы без пробелов. Например: MyClass, CarModel, и т.д.
* **Имена констант**: в качестве константных значений рекомендуется использовать заглавные буквы и символ подчеркивания для разделения слов. Например: MAX\_HOURS = 24, PI = 3.14, и т.д.

Примеры:

Листинг 5

# Переменные

my\_variable = 42

user\_name = "Alice"

# Функции

def calculate\_area(width, height):

return width \* height

def get\_user\_info(user\_id):

# Логика получения информации о пользователе

pass

# Классы

class MyClass:

def \_\_init\_\_(self):

# Конструктор класса

pass

class CarModel:

# Описание класса модели автомобиля

pass

# Константы

MAX\_HOURS = 24

PI = 3.14

Эти рекомендации помогают создавать понятный, последовательный и читаемый код на Python, что в свою очередь облегчает его понимание и поддержку, особенно при долгосрочной разработке проектов.

### 1.2.4.5 Комментарии

Основное внимание уделяется созданию информативных комментариев, обеспечивающих понимание происходящего внутри кода.

Согласно PEP 8, комментарии должны быть полезными, ясными и сжатыми. Они не должны быть очевидными или дублировать информацию, явно выраженную в коде. Вместо этого комментарии должны объяснять неочевидные аспекты кода, его алгоритмы и решения, а не то, что само по себе очевидно.

Пример комментария, соответствующего рекомендациям PEP 8:

Листинг 6

# Вычисляем среднее значение списка

total = sum(values)

count = len(values)

average = total / count # Комментарий объясняющий расчет среднего

В этом примере комментарий явно описывает, что происходит в коде, в то время как сам код уже довольно понятен. Непосредственно перед сложным алгоритмом или решением, комментарии могут сделать код более доступным.

Однако, следует помнить, что избыточное использование комментариев, особенно если они не несут дополнительной информации или банальны, также может усложнить чтение кода. Поэтому ключевым является баланс между информативностью комментариев и их избыточностью.

Таким образом, комментарии в коде, соответствующие рекомендациям PEP 8, помогают другим разработчикам понимать намерения автора кода, а также делают его более доступным для поддержания и развития.

### 1.2.5 Заключение

Подводя итог первой, исследовательской, части, для реализации программного продукта, было решено использовать язык программирования Python, полностью соблюдая стандарты написания кода PEP8. Это позволит унифицировать стиль написания кода в случае участия в разработке большого количества людей. В качестве IDE был выбран PyCharm, который изначально создавался как среда профессиональной разработки под язык Python в крупных компаниях. Следующим шагом была выбрана библиотека Pygame, содержащая в себе необходимый набор функционала для решения поставленной задачи и содержащая подробную документацию для решения непредвиденных проблем.

### 2. Разработка игры и написание кода

### 2.1 Анализ

Согласно проведенному исследованию необходимо разработать приложение-игру.

### 2.1.1 Выбор типов данных

Необходимо выбрать для реализации проекта файлы с такими расширениями, чтобы большинство платформ могли поддерживать необходимые файлы, поэтому ниже будут рассмотрены самые популярные и кросс-платформленные:

* **.mp3** - звуковой файл. Файлы с расширением .mp3 являются форматом сжатого аудио, который позволяет хранить звуковые данные в сжатой форме.
* **.wav** - аудиофайл без сжатия. Файлы с расширением .wav представляют собой аудиофайлы без какой-либо потери качества.
* **.png** и **.jpeg** (или **.jpg**) - форматы изображений. Файлы типов .png и .jpeg/.jpg являются самыми распространенными.

Важно отметить, что все эти типы файлов широко поддерживаются на различных платформах, потому что стандарты для них хорошо установлены и поддерживаются в различных операционных системах и языках программирования, включая Python.

### 2.1.2 Структура приложения

Сама по себе структура довольно простая. Она состоит из графического интерфейса, который постоянно обновляется. Серверной же частью является обработка действий пользователя и создание, перемещение и взаимодействие различных объектов, например, таких как блоки кирпичей и снарядом. Так же серверной частью является запуск различных аудиофайлов, например, музыки на заставочном экране и уникальных звуков попадания по различных объектам, изображённых на экране. Серверная часть на вход получает массивы экземпляром классов, каждый из которых имеет те или иные уникальные свойства, в том числе данные о том, какие звуки относятся именно к этим объектам. Кроме того, поступающие команды действий (перемещение синего танка вперёд, назад, поворот налево, направо, выстрел. Всё то же самое и у красного танка. Действия на заставочном экране), так же поступают в виде массива ключей клавиш клавиатуры функциям, и от них – классам.

### 2.1.3 Разработка диаграммы вариантов использования

Приложение поддерживает следующие роли пользователей с соответствующими привилегиями – Игрок 1 и Игрок 2. Опишем их подробнее.

Роль “Игрок” – сюда относятся все возможные пользователи. Роль становится доступна всем пользователям, которые зашли в приложение. Деление игроков на “1” и “2” указывает на то, какие клавиши будет использовать тот или иной игрок во время игры. Оба пользователя начинают с заставочного экрана

На рисунке 1 приведена диаграмма вариантов использования программной системы. Соединяющие линии означают, что с обоих объектов между которыми находится линия можно свободно перемещаться в обоих направлениях.

На диаграмме вариантов использования приложения наглядно показано, каким образов пользователи могут взаимодействовать с приложением.

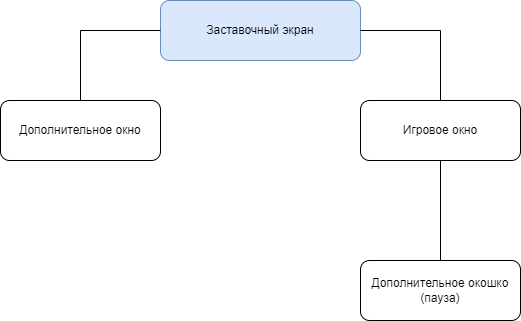


Рисунок 1 – Диаграмма вариантов использования приложения

### 2.1.4 Разработка диаграммы классов

Разрабатываемое приложение состоит из 5 классов: “UI”, “Tank”, “Blocks”, “Bullet”, “Bang”, 4 функций: “draw\_blocks”, “images”, “draw\_intro”, “game” и одного файла глобальных переменных – “Global”. Диаграмма классов и файлов представлена на рисунке 2.

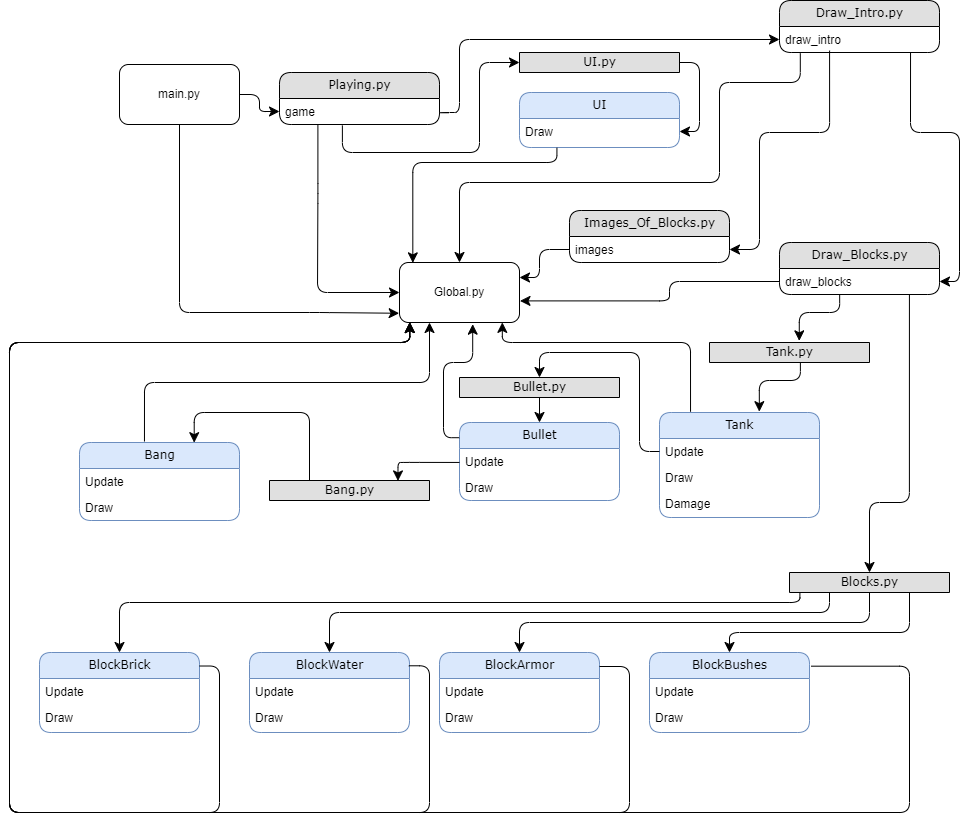


Рисунок 2 – Диаграмма классов и файлов приложения

Таким образом, например, класс “UI” используется при начале и окончании “боя” и необходим он для обработки побед и начиная нового матча, а класс “Tank” используется для работы с танками.

Подробнее со всеми классами, полями, методами и функциями можно ознакомиться в Приложении В.

Ниже продемонстрированы алгоритмы создания блоков воды и кирпичейz так, чтобы место положение каждого было уникально и ни с кем не пересекалось. Ниже приведены примеры кода создания 20 блоков воды и 50 блоков кирпичей:



Рисунок 3 – Код-алгоритм расстановки блоков воды (слева) и

код-алгоритм расстановки блоков кирпичей (справа)

### 2.1.5 Разработка схемы взаимодействия с папками файлов

На рисунке 4 представлена разработанная схема взаимодействия файлов Python с папками звуков, песен и картинок, согласно представленной ранее диаграмме классов системы.

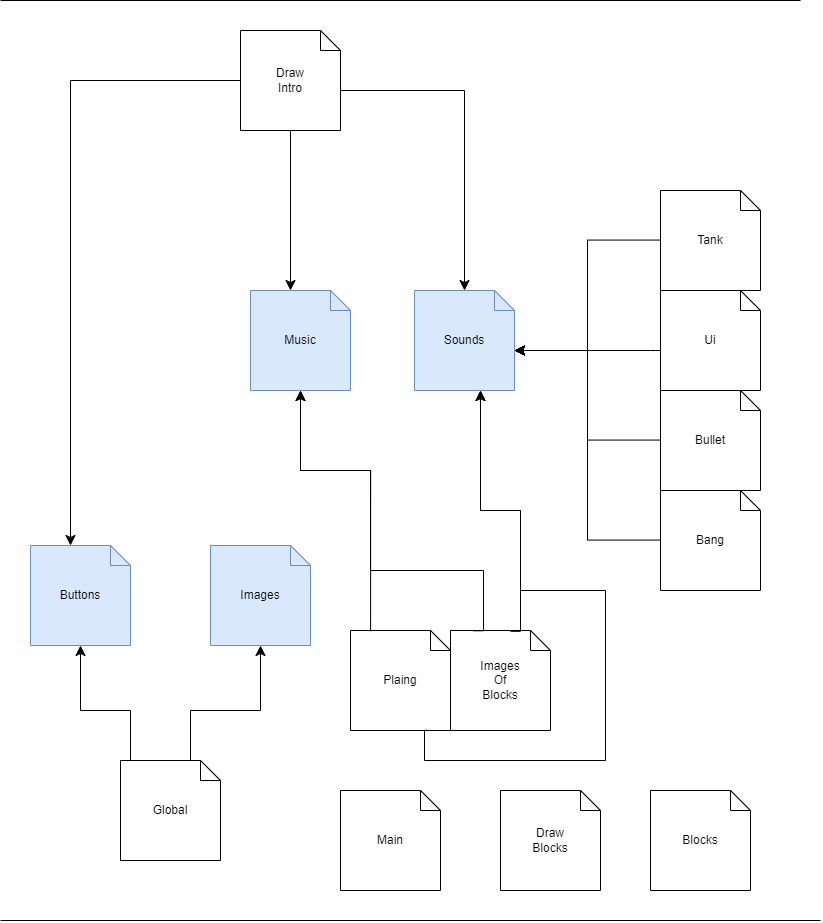


Рисунок 4 – Схема взаимодействия файлов

Доступ к папке “sounds”, содержащей файлы со звуками, в основном, для “боёв” и расширениями wav, для вызова различных звуков имеют файлы “Playing”, “Images\_Of\_Blocks” и “Draw\_Intro” и классы “Tank”, “UI”, “Bullet”, “Bang”.

Доступ к папке “music”, содержащей песни и мелодии для заставочного экрана с расширениями mp3, для вызова различных мелодий имеют файлы “Playing”, “Images\_Of\_Blocks” и “Draw\_Intro”.

Доступ к папке “buttons”, содержащей картинки клавиш в форматах jpeg и png, для использования различных картинок имеют файлы “Global” и “Draw\_Intro”.

Доступ к папке “images”, содержащей картинки блоков, танков и взрывов в форматах jpeg и png, для использование различных картинок имеет файл “Global”.

Такие файлы как “Main”, “Draw\_Blocks”, “Blocks” не используют данные папки ресурсов.

### 2.1.6 Заключение

После создания всех необходимых файлов и папок, папки необходимо заполнить соответствующими файлами, например, в папку со звуками загрузить звуки выстрелов, взрывов, разрушений и т.п., и так со всеми папками. Для поиска таких ресурсов я воспользовался интернет-источниками тематических ресурсов и нашёл подходящие файлы с необходимыми расширениями. Подробнее о расположении ресурсов и файлов игры можно узнать в Приложении А.

### 2.2 Разработка графического интерфейса

Следующим шагом перейдем к разработке графического интерфейса всего приложения, а точнее его составляющим, таким как заставочный экран, карта боя, дополнительное окно, окна победы и паузы.

### 2.2.1 Разработка интерфейса заставочного экрана

Для начала был разработан дизайн заставочного экрана. При его создании необходимо сделать так, чтобы на нём находились различные анимации, которые работали бы без какого-либо вмешательства. Кроме того, дизайн должен быть знакомым и простым, он должен выглядеть привлекательным и не перегружать глаза чрезмерным количеством разноразмерных или разноцветных объектов.

Дополнительно, как и в любой качественной игре, должна прилагаться инструкция, как играть, на какие клавиши нажимать. Поэтому было принято решение добавить эту инструкцию в дизайн и создать анимации подчёркивания и изменения цвета картинок клавиш управления при определённых действиях, и так же создать условие, чтобы, только пройдя это “обучение”, пользователи могли начать бой.

И чтобы понизить вероятность того, что пользователь заскучает, не успев провести хотя бы один бой, было принято решение добавить на главный экран фоновую музыку, которая, кроме прямого назначения, создавала бы настроение и погружала в мир игры.

### 2.2.2 Разработка интерфейса дополнительного экрана

Кроме того, почти у всех разработчиков игр в каждой игре есть какое-либо “пасхальное яйцо”, таким образом я решил добавить свою пасхалку, заключающуюся в том, что с заставочного экрана можно будет перейти в дополнительное окно, кликая на анимацию взрывов, на котором изображены картинки всех объектов в структурированном виде, которые используются в приложении.

### 2.2.3 Разработка интерфейса игрового процесса

После создания заставочного с дополнительном окнами, пришло время создать окно игрового процесса, на котором будет происходить бой.

Так как танки денди игра с пиксельными объектами, которым не нужен большой экран, было принято решение за основу окна взять размеры 800 на 600 пикселей, так как этот размер окна будет максимально подходить под стилистику игры.

Кроме того, в оригинальной версии игры цвет заднего фона приложения – чёрный, я решил его не менять, но изменить визуальную составляющую блоков и танков игры. Я постарался внести в графическую часть более грубые цвета, демонстрирующие большие спокойствие и уверенность, чтобы глазу было приятно смотреть на предстающую перед ним картинку, и картинка выглядела более реалистично.

Из основных механик, в игре танк должен разрушать кирпичи выстрелами по ним, использовать блоки брони как укрытие, использовать воду как объект, через который можно стрелять, но нельзя проезжать, и использовать кусты, как укрытие, в котором можно скрыться и из которого можно застать соперника врасплох неожиданной атакой. При разработке таких механик возникают две проблемы. Первая проблема – если снаряд улетает за границы карты, экрана приложения, то он не пропадает, а продолжает лететь, нагружая компьютер. И вторая проблема заключается в том, что при движении танков, из-за несвоевременной отрисовки объектов танк едет “над” кустами, в то время как должен находиться “под” ними.

Первая проблема была решена путём добавления условия, что если координаты снаряда равны границам, то снаряд удаляется, а для пользователей демонстрирует ту же анимацию и звук, что и при соприкосновении снаряда с блоком брони.

Вторая проблема была исправлена путём изменения последовательности добавления объектов в массив, в котором все они хранятся до момента обновления данных. То есть, если они изображались в прямой последовательности, которая состояла сначала из блоков и только потом танков,

Что рисовало танки поверх остальных блоков, было сделано так, что сначала рисуются танки и только после них различные блоки.

И, как и при разработке заставочного экрана было принято решение добавить фоновую музыку, создающую полноценную погружающую атмосферу.

### 2.2.4 Разработка интерфейса паузы и победы

У каждой игры такого типа есть демонстрация результата и возможность поставить игру на паузу. Поэтому для игры необходимо было создать дополнительные окошки, в которых будут продемонстрированы результаты матча и текущий счёт.

Следовательно, первым этапом нужно было создать окно, которое будет отображаться по окончанию боёв и которое будет показывать, кто победил в этом бою. Так как всё должно быть в одном стиле, цвет окна чередуется с цветами танков, но, по настройкам палитры, полностью совпадает со стилистикой. Таким образом цвет окошка победы совпадает с цветом победившего танка и демонстрирует текущее положение матча

Вторым этапом являлось создание окна паузы игры. Оно должно отображать текущую позицию, у кого сколько очков, и быть нейтрального цвета, но указывать на то, что игра стоит на паузе. Таким образом было решено, что окно паузы отображает текущую позицию матча и имеет серый цвет, который на фоне чёрного заднего фона будет выделяться, но не будет указывать на какое-либо превосходство той или иной стороны.

### 2.2.5 Заключение

По итогам проведенной работы, были разработаны основные интерфейсы для игры, которые включают в себя:

* заставочный экран игры с возможностью запуска боя, отображением краткой инструкции, возможностью найти пасхалку и функционалом выбора сторон игроками;
* интерфейс дополнительного экрана пасхалки, который запускается при срабатывании триггера при нажатии на анимацию взрыва;
* интерфейс паузы и победы;
* разработан концепт различных игровых блоков, каждый из которых имеет уникальную игровую механику.

Более подробно с внешним видом всех игровых элементов можно ознакомиться в Приложении Б.

### 2.3 Тестирование

Следующий важный этап – это тестирование. Независимо от опыта и внимательности разработчика, всегда присутствует вероятность наличия функциональной ошибки либо неудовлетворенности конечного пользователя выбранным подходом к реализации UI/UX. Для уменьшения этих вероятностей существуют функциональное тестирование и оценочное тестирование программного продукта.

### 2.3.1 Разработка технологии тестирования

Из возможных видов тестов было принято решение провести функциональное и оценочное тестирования. Необходимо провести функциональное тестирование для основных функций системы, как для администратора, так и для пользователя, а именно для:

1. Отображение анимаций.
2. Отображение дополнительного заставочного экрана
3. Воспроизведение всех звуков и мелодий
4. Реагирование на перемещения и клики мышкой
5. Реагирование на использование клавиатуры
6. Отображение “карты боя” и всех составляющих
7. Изменение текстовых составляющих приложения

Для тестирования приложения была спроектирована таблица 1 для занесения результатов тестирования.

Таблица 1 – Шаблон таблицы результатов тестирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Действие | Состояние | Ожидаемый результат | Полученный результат |
| Порядковый номер тестирования | Совершаемое пользователем действие | Состояние, в котором находится пользователь | Ожидаемый результат в результате совершения действия | Полученный результат в результате совершения действия |

Было решено провести оценочное тестирование удобства эксплуатации системы и удобства использования. Для тестирования удобства эксплуатации приложение было дано 5 абитуриентам ГБОУ школы № 1811 для оценки по 5 критериям:

1) стабильность;

2) производительность;

3) простота использования;

4) понятность действий;

5) наличие вспомогательное информации.

Стабильность – состояние, когда приложение не зависает, не останавливает работу. Хорошая стабильность – замечаний не выявлено или при тестировании были выявлены незначительные замечания. Плохая стабильность – в результате тестирование были замечены существенные ошибки.

Производительность – приложение быстро отвечает, отсутствует ожидание между запросом от пользователя и ответом от приложения. Хорошая производительность – игра работает быстро. Плохая производительность – имеется существенная разница между нажатием клавиши и откликом приложения.

Простота использования – приложение удобное, кнопки, элементы и вспомогательная информация продуманы логично.

Просто использовать – удобное расположение кнопок, понятная логика взаимодействия с кнопками, иначе необходимо поставить плохую оценку.

Понятность действий – пользователь легко понимает, как использовать основные функции. Оценка “да” - пользователю легко понять, как взаимодействовать с ботом, в обратном случае – “нет”.

Наличие вспомогательное информации – пользователь может найти вспомогательную информацию по работе основных функций. Оценка “да” – пользователь находит ответы, на возникшие у него вопросы, в противном случае – оценка “нет”. Для результатов была спроектирована таблица 2.

Таблица 2 – Шаблон таблицы результатов тестирования удобства эксплуатации

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Критерий | Варианты оценки | 1 человек | … | N человек | Итого |
| Наименование критерия | Возможные варианты оценки | Оценка первого человека, выбранная из вариантов оценки | … | Оценка n-ого человека, выбранная из вариантов оценки | Процент положительных ответ |

### **2.3.2 Разработка тестов и результаты тестирования**

Прежде всего необходимо проверить систему на наличие ошибок, а затем оценить соответствие требованиям и выявить недочеты. Функциональное тестирование поможет обнаружить ошибки. Будут проверены входные данные и процесс их обработки, работа отдельных функций и системы в целом, без углубления в код. Оценочное тестирование позволит оценить систему на соответствие требованиям и выявить недочеты, связанные с неудобством для пользователя.

Более подробно подход к функциональному и оценочному тестированию приведен в Приложении Г. Здесь же сразу перейдем к результатам, которые были получены при проведении тестирования.

В результаты функционального тестирования не были обнаружены ошибки ввода и вывода подаваемых данных. Но было выявлено несколько несоответствий краёв подразумеваемых границ объектов относительно их изображения. Это влияет на реагирование приложения на позиции курсора мыши. Для исправление этой неточности были изменены некоторые местоположения как подразумеваемых границ объектов, так и изображаемых объектов.

Оценочное тестирование показало, что система удобна на 92%. По результатам тестирования были выявлены недочеты, связанные с отсутствием достаточной вспомогательной информации, указывающей на возможность поставить игру на паузу. Таким образом было принято решение исключительно во время матча в заголовке названия приложения указывать не только счёт игры, но и надпись указывающую, на какую клавишу необходимо нажать, чтобы игра остановилась или продолжилась, в зависимости от нахождения игры на паузе или в процессе обработки данных.

### **ЗАКЛЮЧЕНИ**Е

В ходе данной проектной работы был разработан программный продукт в виде компьютерной игры "Танки90", представляющий собой модернизированную версию классической игры "Танки". Целью проекта было создание улучшенного геймплея, современной графики и реализация возможности игры в многопользовательском режиме.

Так же, в процессе написания кода был использован язык программирования Python с библиотекой Pygame, что обеспечило широкие возможности по созданию графики, управлению игровым процессом и взаимодействию с пользователем. Встроенные возможности Python, такие как списки, циклы, функции и классы, были в полной мере задействованы для реализации игровой логики, пользовательского ввода, анимации, а также для управления поведением игровых объектов.

Кроме того, в процессе разработки игры был использован водопадный подход, в котором последовательно происходят фазы сбора и анализа требований, проектирования, реализации, тестирования интеграции и поддержки.

Игра "Танки90" представляет собой захватывающий мир танковых сражений, сочетающий в себе классическую ностальгию за оригинальной игрой "Танки" и современные технологии, чтобы предложить игрокам уникальный и захватывающий игровой опыт. Важно отметить, что чистота кода и стремление к соответствию стандартам играют важную роль при написании качественного и поддерживаемого программного обеспечения.

Данный проект не только продемонстрировал навыки разработки программного продукта, но и внес свой вклад в область развлекательного программирования, предлагая новый и увлекательный игровой опыт. Результаты разработки и исследования, проведенные в рамках данной проектной работы, могут служить основой для будущих разработок и расширения игры "Танки90", а также внесения своего вклада в игровую индустрию.

Например, в дальнейшем мне хотелось бы реализовать в игре возможности участвовать третьему игроку, управляя танком при помощи мышки, играть с разных устройств и создавать и редактировать карты и расширить возможности игры (создать большее количество декораций на уровнях, увеличить размер карт). Также, есть возможность разместить данную игру для общего пользования на таких сайтах, как **itch.io** и **gamejolt.com**

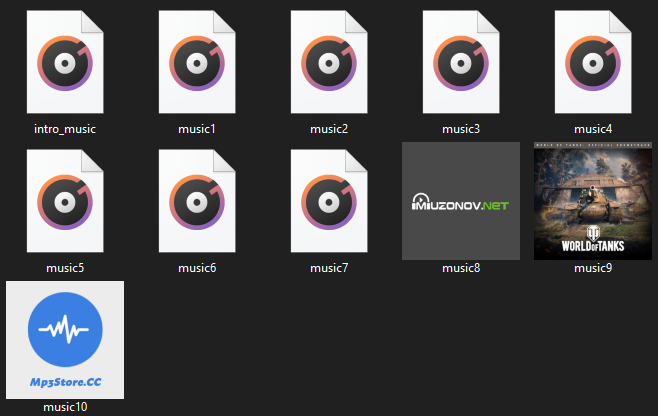
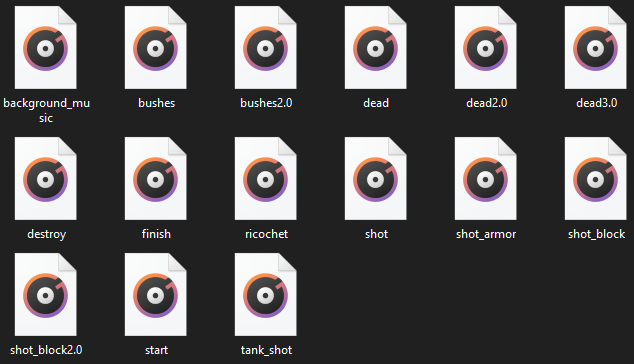
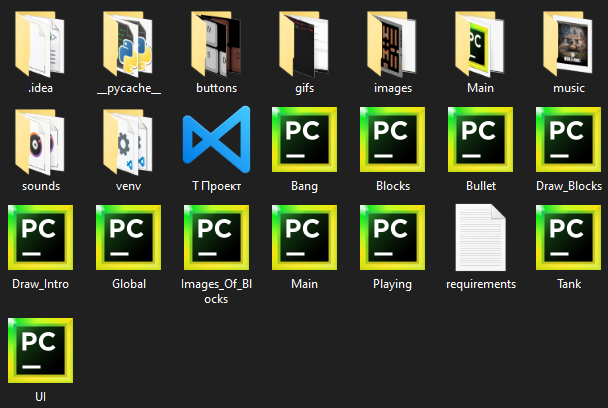
Таким образом, игра "Танки90" представляет собой успешный программный продукт, сочетающий в себе классические игровые принципы и передовые технологии, и является результатом целеустремленной и увлеченной работы.

Весь проект можно найти на GitHub по ссылке: https://github.com/Vlad-Akimov/Tanks1996

### СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. **Доусон, М.** Программируем на Python / Доусон, М. –Питер, 2020 г. –416с. – ISBN: 978-5-4461-1386-6
2. **Лутц, М.** Изучаем Python / Лутц М. – Москва, 2019 г. – 832 с. – ISBN: 978-5-907144-52-1
3. **McGugan, W.** Вeginning Game Development with Python and Pygame / McGugan, W. – England, 2007 г. – 340 c. – ISBN: 978-1590598726
4. [**horror\_x**](https://habr.com/ru/users/horror_x/) Ностальгия: роемся у «Танчиков» под капотом – URl: <https://habr.com/ru/articles/142126/> (дата обращения 5.3.2024). – Текст: электронный
5. **Guido van Rossum** PEP 8 – Style Guide for Python Code – URL: <https://peps.python.org/pep-0008/> (дата обращения 5.3.2024) ). – Текст: электронный
6. **Pete Shinners** Pygame Front Page – URL: <https://www.pygame.org/> (дата обращения 5.3.2024). Текст: электронный
7. **Junko Ozawa** Battle City – URL: <https://ru.wikipedia.org/wiki/Battle_City> (дата обращения 5.3.2024). Текст: электронный
8. [**borrego**](https://stackoverflow.com/users/2772/borrego) Questions tagged [pygame] – URL: <https://stackoverflow.com/questions/tagged/pygame> (дата обращения 5.3.2024). Текст: электронный
9. [**Jon Fincher**](https://realpython.com/pygame-a-primer/#author) PyGame: A Primer on Game Programming in Python – URL: <https://realpython.com/pygame-a-primer/> (дата обращения 5.3.2024) . Текст: электронный
10. **Asya Dyu** Самые популярные бесплатные движки для разработки игр – URL: <https://habr.com/ru/companies/timeweb/articles/659891/> (дата обращения 5.3.2024) . Текст: электронный

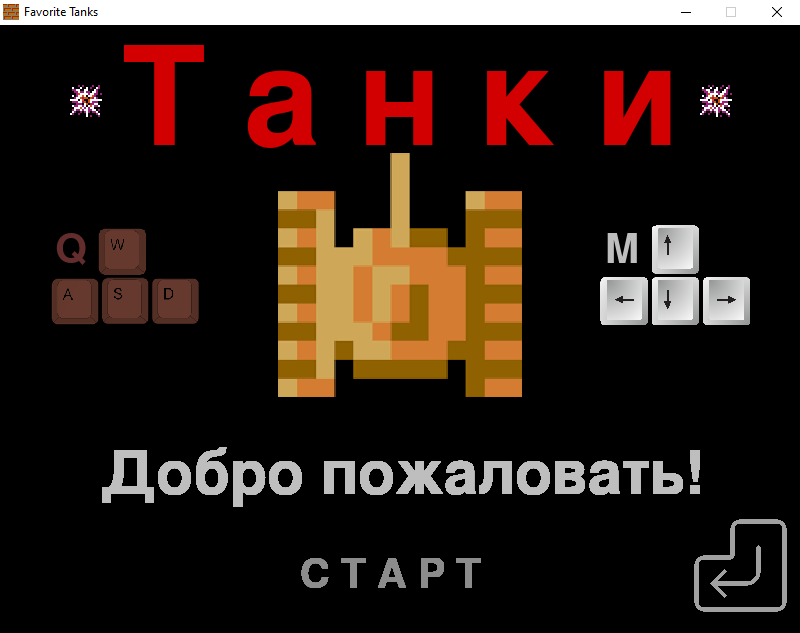
### ПРИЛОЖЕНИЕ А

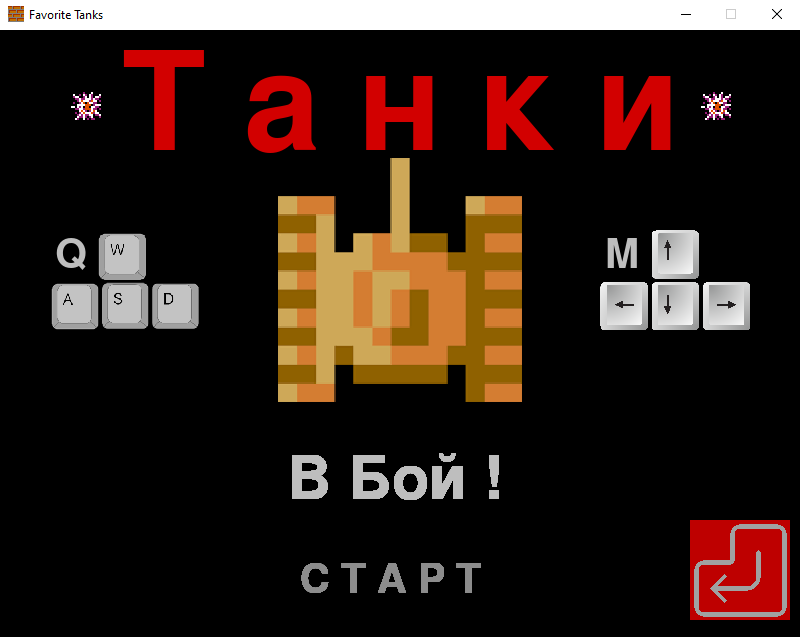
**Папки игровых ресурсов**

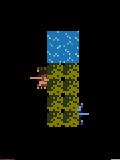
### ПРИЛОЖЕНИЕ Б

**Перечень игровых экранов**

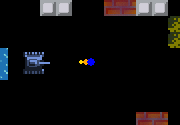
Дополнительный экран:

****Заставочный экран (первый запуск, была нажата клавиши “w”):

Заставочный экран (был сыгран один матч, курсор мыши находится над “enter”):

Танки “под” кустами:

Окно паузы:

Полёт снаряда:

### ПРИЛОЖЕНИЕ В

**Классы и файлы**

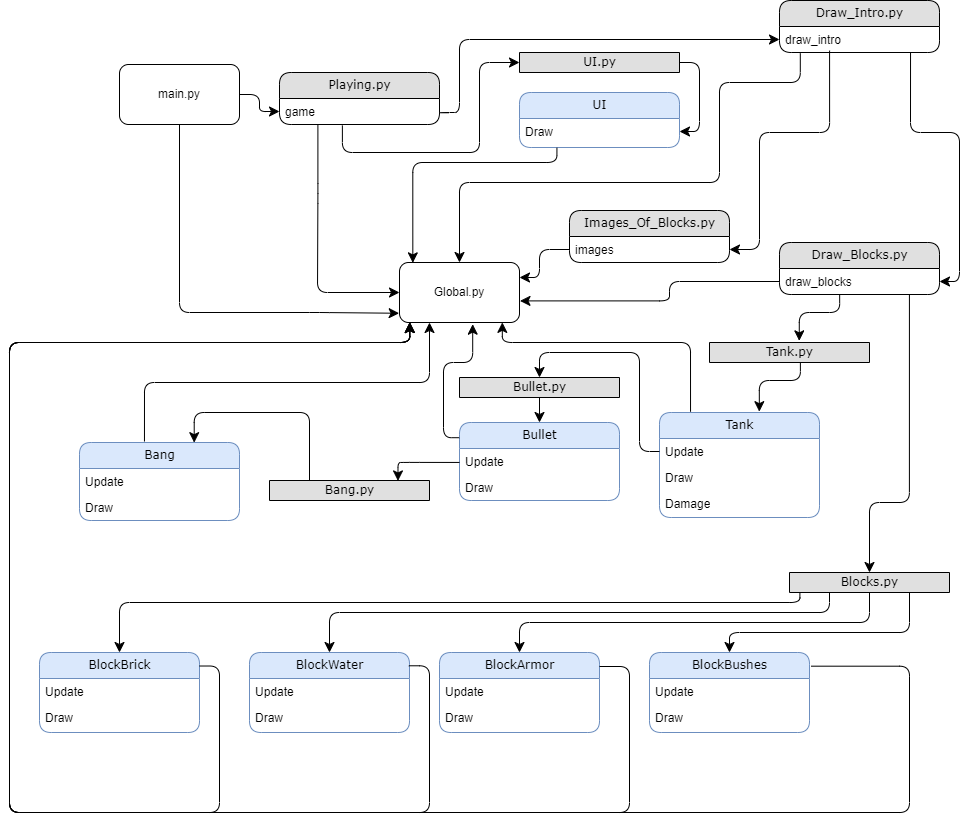


Диаграмма классов и файлов приложения

Таким образом класс “UI” используется при начале и окончании “боя”. Он необходим для обработки побед и начиная нового матча. Он имеет следующие методы:

* метод “init” – отвечает за начало матча;
* метод “draw” – отвечает за отрисовку победного окна.

Класс “Tank” используется для работы с танками. Он имеет следующие методы и основные свойства:

* поле “type” – имеет данные о типе объекта, для его уникальности, чтобы не смешивать танки с остальными объектами напрямую;
* поле “color” – имеет данные о цвете танка, для определения кому присваивать победу;
* поле “rect” – имеет данные о местоположении танка на “карте”, для корректного отбражения танка;
* метод “init” – отвечает за создание танка и его свойств;
* метод “update” – отвечает обновление данных танка, таких как перемещение, выстрел и т.п.;
* метод “draw” – отвечает за изображение танка на экране;
* метод “damage” – отвечает за попадание по танку “снарядом”.

Класс “Bullet” используется для работы со снарядами. Он имеет следующие методы и основные свойства:

* поле “parent\_x” – имеет данные о местоположении объекта по оси координат x;
* поле “parent\_y” – имеет данные о местоположении объекта по оси y;  
  поле “vector\_x” – имеет данные о направлении полёта снаряда по оси x;
* поле “vector\_y” – имеет данные о направлении полёта снаряда по оси y;
* поле “color” – имеет данные о цвете танка, для определения какому танку принадлежит снаряд;
* метод “init” – отвечает за создание снаряда и его свойств;
* метод “update” – отвечает обновление данных танка, таких как перемещение и попадание по объекту;
* метод “draw” – отвечает за изображение снаряда на экране.

Класс “Bang” используется для работы со взывами. Он имеет следующие методы и основные свойства:

* поле “other\_type” – имеет данные о типе объекта, с которым столкнулся снаряд. Это необходимо для определения дальнейших действий, таких как воспроизведение определённого звука и разрущение объекта;  
  поле “parent\_x” – имеет данные о местоположении столкновения снаряда с препятствием по оси x;
* поле “parent\_y” – имеет данные о местоположении столкновения снаряда с препятствием по оси y;
* поле “timer” – имеет данные о времени с момента столкновения снаряда с объектом, это необходимо для изменения изображения взрыва;
* метод “init” – отвечает за создание взрыва и его свойств;
* метод “update” – отвечает обновление данных взрыва, например, сколько время прошло, не пора ли изменять изображение взрыва;
* метод “draw” – отвечает за изображение взрывов на экране.

Класс “BlockBrick” используется для работы с кирпичами. Он имеет следующие методы и основные свойства:

* поле “rect” – имеет данные о местоположении и размере блока.
* поле “type” – имеет данные о типе объекта, это нужно корректной обработки взаимодействия снарядов и танков с блоком;
* метод “init” – отвечает за создание блока и его свойств;
* метод “update” – отвечает обновление данных блока, не сломали ли его и т.п.;
* метод “draw” – отвечает за изображение блока на экране.

Класс “BlockWater” используется для работы с водой. Он имеет следующие методы и основные свойства:

* поле “rect” – имеет данные о местоположении и размере блока.
* поле “type” – имеет данные о типе объекта, это нужно корректной обработки взаимодействия снарядов и танков с блоком;
* метод “init” – отвечает за создание блока и его свойств;
* метод “update” – отвечает обновление данных блока, чтобы он отображался на экране;
* метод “draw” – отвечает за изображение блока на экране.

Класс “BlockArmor” используется для работы с блоком брони. Он имеет следующие методы и основные свойства:

* поле “rect” – имеет данные о местоположении и размере блока.
* поле “type” – имеет данные о типе объекта, это нужно корректной обработки взаимодействия снарядов и танков с блоком;
* метод “init” – отвечает за создание блока и его свойств;
* метод “update” – отвечает обновление данных блока, чтобы он отображался на экране;
* метод “draw” – отвечает за изображение блока на экране.

Класс “BlockBushes” используется для работы с кустами. Он имеет следующие методы и основные свойства:

* поле “rect” – имеет данные о местоположении и размере блока.
* поле “type” – имеет данные о типе объекта, это нужно корректной обработки взаимодействия снарядов и танков с блоком;
* метод “init” – отвечает за создание блока и его свойств;
* метод “update” – отвечает обновление данных блока, чтобы он отображался на экране;
* метод “draw” – отвечает за изображение блока на экране.

Функция “game” – основной цикл игр, именно здесь вызываются функции рисования заставочного экрана, матча и дополнительного окна.

Функция “draw\_intro” – функция, отвечающая за изображение заставочного экрана.

Функция “images” – функция, отвечающая за изображение дополнительного окошка, в котором изображаются картинки всех использующихся в приложении картинок.

Функция “draw\_blocks” – функция, отвечающая за создание автоматически генерируемых карт. Здесь ставятся танки на заранее известную позицию, и генерируются 20 блоков воды, 20 блоков брони, 30 блоков кустов и 50 блоков кирпичей. При этом каждый из блоков имеет своё собственное, уникальное, местоположение, которое не противоречит местоположению какого-либо другого блока или объекта на карте. Так же алгоритм генерации карт учитывает возможность патовой ситуации, когда нет возможности одному из игроков победить, из-за того, что его соперник был забаррикадирован блоком брони, поэтому алгоритм избегает таких ситуаций.

### ПРИЛОЖЕНИЕ Г

### Функциональное и оценочное тестирование

Чтобы удостоверится в правильности взаимодействия обычного пользователя с приложением, проведем функциональное тестирование основных функций системы. Результаты тестирования приведены в таблице 3.

Таблица 3 - Функциональное тестирование работы

|  |  |  |  |
| --- | --- | --- | --- |
| № | Действие | Ожидаемый результат | Полученный результат |
| 1 | Клик на танк заставочного экрана | Воспроизведение музыки | Воспроизведение музыки |
| 2 | Анимация “взрывов” на заставочном экране | Отображение анимации | Отображение анимации |
| 3 | Наведение курсора на “старт” | Изменение картинки курсора и цвета заднего фона объекта | Изменение картинки курсора и цвета заднего фона объекта |
| 4 | Наведение курсора на “enter” | Изменение картинки курсора и цвета заднего фона объекта | Изменение картинки курсора и цвета заднего фона объекта |
| 5 | Наведение курсора на танк заставочного экрана | Изменение картинки курсора | Изменение картинки курсора |
| 6 | Наведение курсора на “взрывы” | Изменение картинки курсора | Изменение картинки курсора |
| 7 | Наведение курсора на “клавиши управления танками” | Изменение картинки курсора | Изменение картинки курсора |
| 8 | Клик по “взрывам” | Переход в дополнительное окно | Переход в дополнительное окно |
| 9 | Нажимание “escape” в дополнительном окне | Переход на заставочный экран | Переход на заставочный экран |
| 10 | Нажимание клавиш управления танками | Изменение цвета картинки клавиш | Изменение цвета картинки клавиш |
| 11 | Клик по “старт” или “enter” или использование клавиши “enter” | Отображение анимации или начинание боя, воспроизведение мелодии | Отображение анимации или начинание боя, воспроизведение мелодии |
| 12 | Зажимание “клавиш управления танками” в “бою” | Перемещение танка на экране | Перемещение танка на экране |
| 13 | Нажимание “клавиши выстрела“ в “бою” | Анимация выстрела, создание снаряда с его последующим перемещением по заданной траектории | Анимация выстрела, создание снаряда с его последующим перемещением по заданной траектории |
| 14 | Нажимание “escape” в “бою” | Выход из “боя” и воспроизведение мелодии | Выход из “боя” и воспроизведение мелодии |
| 15 | Нажимание “enter” в “бою” | Пауза (“стоп игра”), отображение окна паузы | Пауза (“стоп игра”), отображение окна паузы |
| 16 | Попадание “снаряда” по блоку кирпича | Удаление блока кирпича, воспроизведение звука, создание “взрыва” | Удаление блока кирпича, воспроизведение звука, создание “взрыва” |
| 17 | Попадание “снаряда” по блоку брони | Воспроизведение звука, создание “взрыва” | Воспроизведение звука, создание “взрыва” |
| 18 | Попадание “снаряда” в край экрана | Воспроизведение звука, создание “взрыва” | Воспроизведение звука, создание “взрыва” |
| 19 | Попадание “снаряда” по танку | Воспроизведение звука и мелодии, создание “взрыва”, отображение окна победы, | Воспроизведение звука и мелодии, создание “взрыва”, отображение окна победы, |
| 20 | Выход из приложения | Воспроизведение мелодии, выход из приложения | Воспроизведение мелодии, выход из приложения |
| 21 | “Врезание” танка в блоки брони, кирпича, воды | Выполнение выбранного действия | Выполнение выбранного действия |
| 22 | “Прохождение” танком блоки кустов | Выполнение выбранного действия | Выполнение выбранного действия |
| 23 | Пропадание курсора на игровом экране с момента начала боя | Выполнение данного действия | Выполнение данного действия |
| 24 | Изменение курсора мыши в окне приложения | Выполнение данного действия | Выполнение данного действия |
| 25 | Изменение надписи “Добро пожаловать” на другие пять возможных | Выполнение данного действия | Выполнение данного действия |

В результаты функционального тестирования не были обнаружены ошибки ввода и вывода подаваемых данных. Но было выявлено несколько несоответствий краёв подразумеваемых границ объектов относительно их изображения. Это влияет на реагирование приложения на позиции курсора мыши. Для исправление этой неточности были изменены некоторые местоположения как подразумеваемых границ объектов, так и изображаемых объектов.



### Оценочное тестирование

Необходимо, чтобы система не только выполняла свои функции без ошибок, но и была удобной для пользователя. Для оценки удобства использования было решено провести оценочное тестирование. Для проведения тестирования удобства эксплуатации было решено дать протестировать систему 5 абитуриентам ГБОУ школы 1811 и оценить систему по следующим критериям:

1. стабильность;
2. производительность;
3. простота использования;
4. понятность действий;
5. наличие вспомогательное информации.

Результаты тестирования приведены в таблице 4.

Таблица 4 – Результаты тестирования удобства эксплуатации

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Критерий | Варианты оценки | 1 чело-век | 2 чело-век | 3 чело-век | 4 чело-век | 5 чело-век | Итого |
| Стабильность | Хорошая(Х)/ плохая(П) | Х | Х | Х | Х | Х | 100 |
| Производи-тельность | Хорошая(Х)/ плохая(П) | Х | Х | Х | Х | Х | 100 |
| Простота использова-ния | Хорошая(Х)/ плохая(П) | Х | Х | Х | Х | Х | 100 |
| Понятность действий | Да/Нет | Да | Да | Да | Нет | Да | 80 |
| Наличие вспомога-тельной информации | Да/Нет | Да | Да | Да | Нет | Да | 80 |

Оценочное тестирование показало, что система удобна на 92%. По результатам тестирования были выявлены недочеты, связанные с отсутствием достаточной вспомогательной информации, указывающей на возможность поставить игру на паузу. Таким образом было принято решение исключительно во время матча в заголовке названия приложения указывать не только счёт игры, но и надпись указывающую, на какую клавишу необходимо нажать, чтобы игра остановилась или продолжилась, в зависимости от нахождения игры на паузе или в процессе обработки данных.