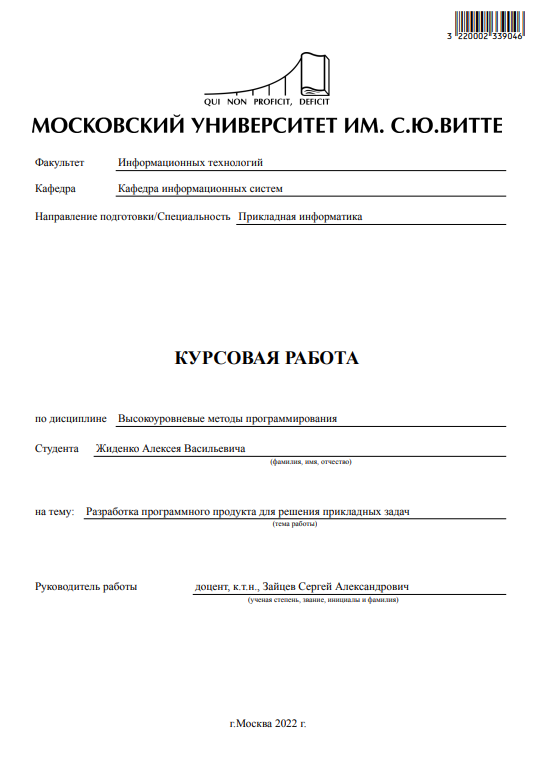
**ВВЕДЕНИЕ**

В условиях быстро развивающейся современной высокотехнологичной экономики актуальной задачей является разработка различных автоматизированных систем. Они позволяют наиболее эффективно использовать различные современные средства информационной вычислительной техники и решать прикладные задачи. Для успешного решения таких задач нужно уметь использовать высокоуровневые средства разработки и декомпозировать задачу.

В данной работе решаются задачи высокоуровневого программирования, согласно индивидуальному заданию.

Актуальность курсовой работы связана с наличием большого количества прикладных задач, которые необходимо эффективно решать с помощью инструментов высокоуровневых методов программирования.

*Проект выполняется совместно с Шустовым С.А.*

Оглавление

[**1.** **ПОСТАНОВКА ЗАДАЧИ КУРСОВОЙ РАБОТЫ** 3](#_Toc102594040)

[**1.1.** **Исходные данные к задаче курсовой работы.** 3](#_Toc102594041)

[1.1.1. Язык программирования python 3](#_Toc102594042)

[1.1.2. Основные библиотеки pygame, time, os, numpy, sys, math 3](#_Toc102594043)

[1.1.3. Игра сделана в формате 2D. Игра имеет функциональное меню с кнопками 'Start', 'Continue', 'Options', 'About', 'Quit', из которых все функциональны и стабильно работают. Функция 'Options' включает в себя настройку полноэкранного режима и возможность вкл/выкл 'Показать ФПС'. Кнопки Start, Continue, Quit - работают соответственно. Сама игра сделана в стиле известной игры 'Mario' : ГГ(Главный герой) может передвигатьсяь влево/вправо, прыгать/падать. Задача игрока добраться до финиша на определённом уровне. Когда игрок пройдёт все уровни - он выиграет. 3](#_Toc102594044)

[1.2. Анализ исходных данных курсовой работы 4](#_Toc102594045)

[1.2.1. Грамотно распределить время работы в команде для более координированной работы с кодом. 4](#_Toc102594046)

[1.2.2. Для более удобной работы игры необходимо создать главное меню игры: присваиваем кажой кнопке определённый индекс, добавляем обновление кнопки при нажатии(анимация),добавляем проверку события для нажатой и отпущенной кнопки, добавляем задний фон и анимацию фона и прочее. 4](#_Toc102594047)

[1.2.3. Для удобства пользователя необходимо добавить функцию полного экрана и возможность убрать/показать FPS в меню 'Options' 4](#_Toc102594048)

[1.2.4. При написании кода понадобится импортировать библиотеки: Pygame, sys, os, time, numpy(Некоторые будут убраны после оптимизации кода) 4](#_Toc102594049)

[1.2.5. Будет добавлен файл data/engine.py - в котором будет прописаны: базовая физика и механика игры, прорисовка и отображение игрового мира, подгрузка файлов. А так же это файл отвечает за полную настройку игры. - Ядро игры. 4](#_Toc102594050)

[1.2.6. Необходимо нарисовать все необходимые изображения и включить их в код, а именно: кнопки(в нажатом и отпущенном виде), игрок(в статичном положении и в движении), фон игры(в игре и в состоянии паузы), эффекты частиц и прочего, игровые объекты(платформы, цепи, стены) 5](#_Toc102594051)

[1.3. Выбор и обоснование необходимых библиотек и среды разработки. 5](#_Toc102594052)

[1.3.1. Поскольку в методических указаниях к выполнению курсовой работы указано использование языка программирования Python, среды разработки PyCharm, а также стандартных библиотек и модулей стандартной библиотеки, то для решения заданий курсовой работы были выбраны следующие средства: Язык программирования Python3 – согласно рекомендациям; Интегрированная среда разработки PyCharm – согласно рекомендациям; 5](#_Toc102594053)

[1.3.2. Pygame— библиотека Python для упрощения написания игрового кода на движке Python 5](#_Toc102594054)

[1.3.3. Sys— библиотека Python для прямого взаимодействия с интерпритатором. 5](#_Toc102594055)

[1.3.4. Os— библиотека Python для получения наглядной информации о работоспособности кода 5](#_Toc102594056)

[1.3.5. Time— библиотека Python для более удобной работы с секундами игры, FPS, выводом его на экран и прочее. 6](#_Toc102594057)

[1.3.6. Math— библиотека Python для более удобной работы с математическими функциями кода, и для составления более сложных вычислений. 6](#_Toc102594058)

[1.3.7. Numpy— библиотека Python необходима для составления высокоуровневых математических функций, составления массивов и пр. 6](#_Toc102594059)

[**1.4.** **Выводы по 1 главе.** 6](#_Toc102594060)

[1.4.1. Ссылка на проект Github: https://github.com/Shustovson/game-project 6](#_Toc102594061)

[**2.** **РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА** 6](#_Toc102594062)

[2.1. **Создание главного меню игры.** 6](#_Toc102594063)

[2.1.1. Для этого создаём файл menu.py, в который мы поместили классы: MenuScreen, OptionsMenu, VideoMenu - которые отрисовывают меню и проверяют события стартового экрана игры.(нажатие кнопок) В файл main.py мы поместили класс MainMenu - который является связующим классов выше, который инициализирует, проверяет на события и обнавляет экран главного меню. 6](#_Toc102594064)

[2.3. Отрисовка главного меню как написано выше осуществляется через классы: 10](#_Toc102594065)

[2.3.1. MainMenu: 10](#_Toc102594066)

[2.3.2. MenuScreen: 12](#_Toc102594067)

[2.3.3. OptionsMenu: 13](#_Toc102594068)

[2.3.4. VideoMenu: 15](#_Toc102594069)

[3.1. Запуск Меню - производится от запуска файла Game.py в PyCharm 17](#_Toc102594070)

[3.2. Кнопка Continue - запускает продолжение игры с последней контрольной точки 18](#_Toc102594071)

[3.3. Кнопка Options - задаёт настройки экрана и функцию вкл/ откл FPS, а так же смену кнопок управления(не реализованно) 18](#_Toc102594072)

[3.3.1. Меню Video 19](#_Toc102594073)

[3.4. Кнопка Start - начинает запуск новой игры. 19](#_Toc102594074)

1. **ПОСТАНОВКА ЗАДАЧИ КУРСОВОЙ РАБОТЫ**
   1. **Исходные данные к задаче курсовой работы.**

Формулировка: Создание 2D игры на языке программирования Python:

* + 1. Язык программирования python
    2. Основные библиотеки pygame, time, os, numpy, sys, math
    3. Игра сделана в формате 2D. Игра имеет функциональное меню с кнопками 'Start', 'Continue', 'Options', 'About', 'Quit', из которых все функциональны и стабильно работают. Функция 'Options' включает в себя настройку полноэкранного режима и возможность вкл/выкл 'Показать ФПС'. Кнопки Start, Continue, Quit - работают соответственно. Сама игра сделана в стиле известной игры 'Mario' : ГГ(Главный герой) может передвигатьсяь влево/вправо, прыгать/падать. Задача игрока добраться до финиша на определённом уровне. Когда игрок пройдёт все уровни - он выиграет.
  1. Анализ исходных данных курсовой работы

Для выполнения задания нужно проделать следующие этапы:

* + 1. Грамотно распределить время работы в команде для более координированной работы с кодом.
    2. Для более удобной работы игры необходимо создать главное меню игры: присваиваем кажой кнопке определённый индекс, добавляем обновление кнопки при нажатии(анимация),добавляем проверку события для нажатой и отпущенной кнопки, добавляем задний фон и анимацию фона и прочее.
    3. Для удобства пользователя необходимо добавить функцию полного экрана и возможность убрать/показать FPS в меню 'Options'
    4. При написании кода понадобится импортировать библиотеки: Pygame, sys, os, time, numpy(Некоторые будут убраны после оптимизации кода)
    5. Будет добавлен файл data/engine.py - в котором будет прописаны: базовая физика и механика игры, прорисовка и отображение игрового мира, подгрузка файлов. А так же это файл отвечает за полную настройку игры. - Ядро игры.
    6. Необходимо нарисовать все необходимые изображения и включить их в код, а именно: кнопки(в нажатом и отпущенном виде), игрок(в статичном положении и в движении), фон игры(в игре и в состоянии паузы), эффекты частиц и прочего, игровые объекты(платформы, цепи, стены)
  1. Выбор и обоснование необходимых библиотек и среды разработки.

* + 1. Поскольку в методических указаниях к выполнению курсовой работы указано использование языка программирования Python, среды разработки PyCharm, а также стандартных библиотек и модулей стандартной библиотеки, то для решения заданий курсовой работы были выбраны следующие средства: Язык программирования Python3 – согласно рекомендациям; Интегрированная среда разработки PyCharm – согласно рекомендациям;
    2. Pygame— библиотека Python для упрощения написания игрового кода на движке Python
    3. Sys— библиотека Python для прямого взаимодействия с интерпритатором.
    4. Os— библиотека Python для получения наглядной информации о работоспособности кода
    5. Time— библиотека Python для более удобной работы с секундами игры, FPS, выводом его на экран и прочее.
    6. Math— библиотека Python для более удобной работы с математическими функциями кода, и для составления более сложных вычислений.
    7. Numpy— библиотека Python необходима для составления высокоуровневых математических функций, составления массивов и пр.
  1. **Выводы по 1 главе.**

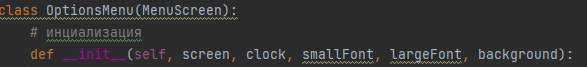
Изучив полученный материал из главы 1, можно сделать вывод, что задание включает наиболее часто используемые прикладные задачи высокоуровневого программирования – это построение приложений с графическим пользовательским интерфейсом, использование алгоритмов, работа в наиболее популярной среде разработки, ставшей де-факто стандартом для разработчиков на языке программирования Python.

Успешное выполнения задания будет свидетельствовать о полноте усвоенного материала в рамках дисциплины «Высокоуровневые методы программирования» и умение решать задачи низкой и средней сложности.

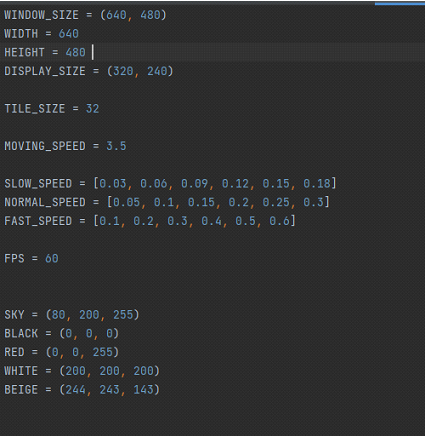
* + 1. Ссылка на проект Github: https://github.com/Shustovson/game-project

1. **РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА**
   1. **Создание главного меню игры.** 
      1. Для этого создаём файл menu.py, в который мы поместили классы: MenuScreen, OptionsMenu, VideoMenu - которые отрисовывают меню и проверяют события стартового экрана игры.(нажатие кнопок) В файл main.py мы поместили класс MainMenu - который является связующим классов выше, который инициализирует, проверяет на события и обнавляет экран главного меню.





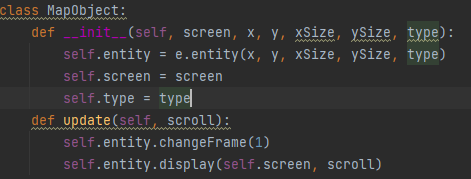
* + 1. Добавляем файл Settings.py - в который мы добавили такие функции, как длинна и высота окна, размер окна, обозначения цветов, скорость движения игрока - для поддержания работоспособности и читабельности кода.



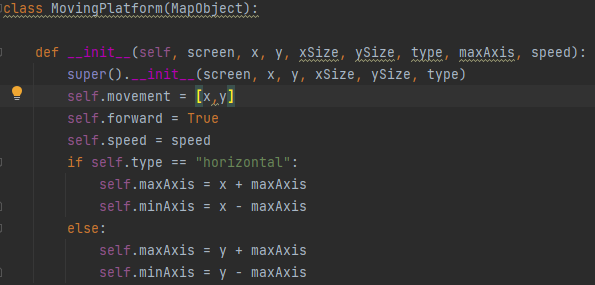
* 1. Создание игры.

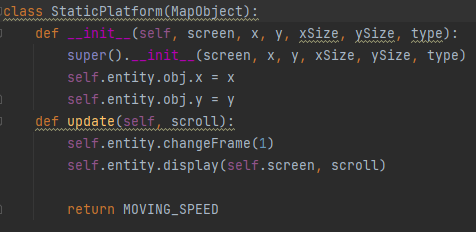
Для начала мы добавили файл objects.py, в который поместили класс MapLevel - в котором происходит загрузка изображений для игры (карты, объекты игрового окружения), задаётся размер объектов(изображений), отрисовка игровых плит, стен, шипов и т.д и MapObject в котором будет инициализироваться карта.



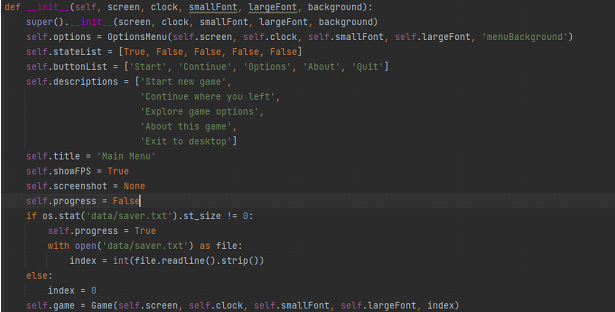


* + 1. Кроме того, в файле objects.py добавляются классы StaticPlatform и MovingPlatform в которых задаются платформы и соответственно прописывается то, где находится платформа, движется она или нет и как движется (Если это предусмотрено)





* 1. Отрисовка главного меню как написано выше осуществляется через классы:
     1. MainMenu:

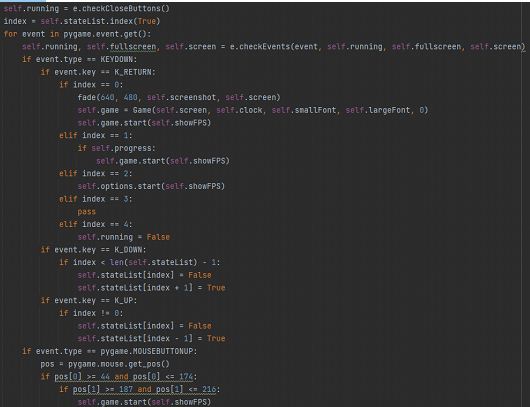


Где по средствам ‘\_\_init\_\_’ задаются параметры класса(экран, время, и фон)

В этой инициализации задаются Наименования кнопок в главном меню, комментарии(описание) кнопок, устанавливается файл для сохранения игры, задаётся заголовок экрана.

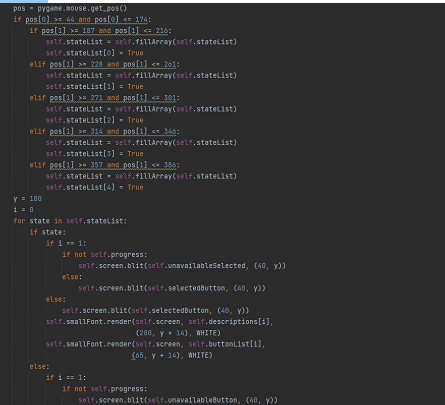
Кроме того, внутри класса задаются функции:

events:



Функция описывает действие программы при нажатии курсора на кнопки в главном меню

update:



Функция обновляет позицию кнопки и меняет её внешний вид (визуальный эффект наведения курсора или использования кнопок key\_arrow\_down, key\_arrow\_up)

* + 1. MenuScreen:



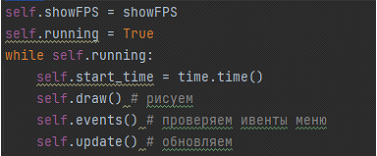
Родительский класс для всех классов связанных с меню

через \_\_init\_\_ здесь задаются такие базовые параметры как

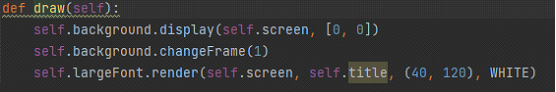
clock, screen, smallfont, largefont , showFPS , fullscreen а так же изображения кнопок: Активных(без выделения), Активных(с выделением), неактивных(без выделения), неактивных(с выделением)

Кроме того, внутри класса задаются функции:

start:

Запускаем отрисовку(draw) меню, показываем его и запускаем цикл running с проверкой на True

draw:

Функция выводит на дисплей задний фон и выводит на экран шрифт(largeFont) с установленными параметрами

И прочее...

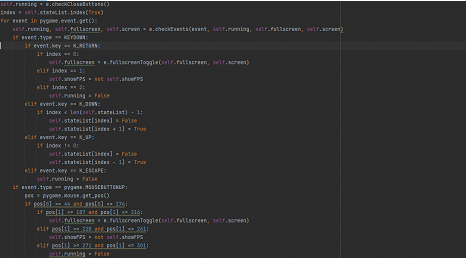
* + 1. OptionsMenu:

Класс для меню настроек, инициализируется и обращается к материнскому классу через super().\_\_init\_\_

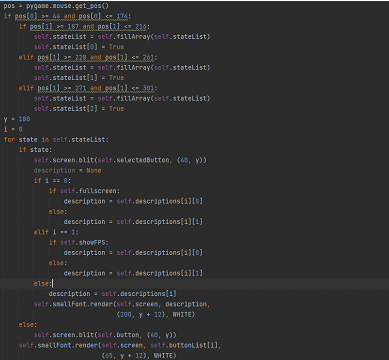
Задаёт лист активных\не активных кнопок, список названия кнопок и их описание, задаётся заголовок экрана и обращается к классу VideoMenu передавая параметры.

Кроме того, внутри класса задаются функции:

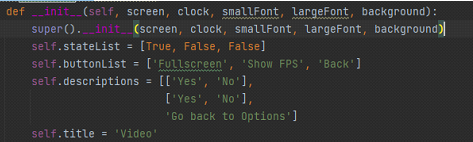
events:

Функция считывает событие нажатия на клавиши и выполняет привязанные к ним команды

update:

Функция считывает курсор и обновляет меню если пользователь куда-то нажал

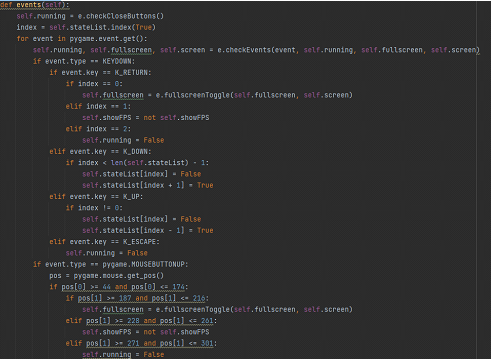
* + 1. VideoMenu:



Класс Меню видео где можно выбрать режим экрана и отображения FPS в игре, так же идёт с массивами кнопок, их активности/не активности, описанием и заголовком экрана.

Кроме того, внутри класса задаются функции:

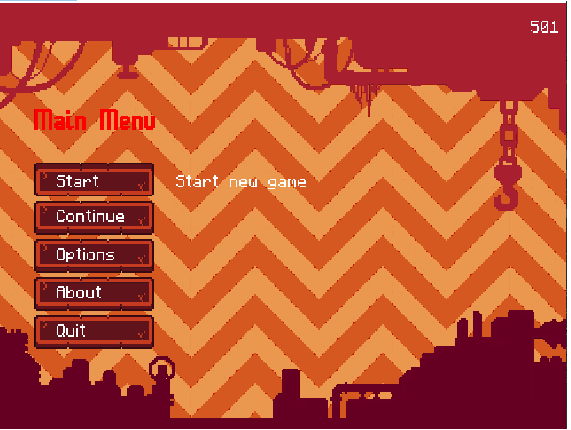
events:

Проверка события нажатия на клавиши

update:

Функция проверки нажатия кнопок.

1. **Тестирование главного меню игры**
   1. Запуск Меню - производится от запуска файла Game.py в PyCharm



* 1. Кнопка Continue - запускает продолжение игры с последней контрольной точки



* 1. Кнопка Options - задаёт настройки экрана и функцию вкл/ откл FPS, а так же смену кнопок управления(не реализованно)



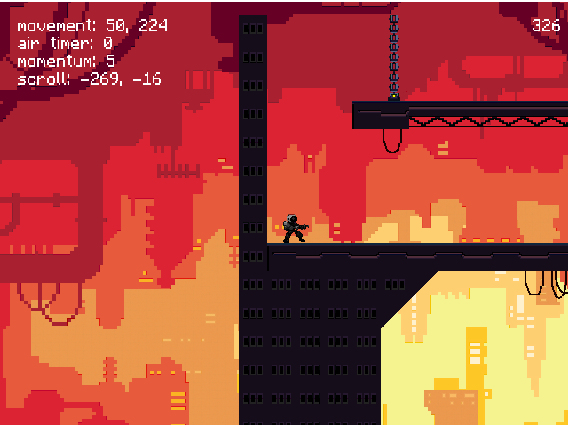
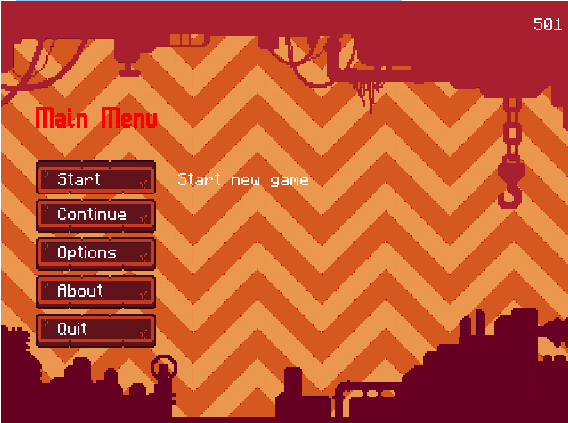
* + 1. Меню Video



Fullscreen -отвечает за вкл/откл функции полного экрана.

3Show FPS - отвечает за вкл/откл функции показа FPS

* 1. Кнопка Start - начинает запуск новой игры.



1. **Вывод**

Основной целью данной курсовой работы являлось написание функциональной игры. Осуществлена попытка решения предложенных задач, в ходе решения которых были углублены знания языка Python и его инфраструктуры, были реализованы алгоритмы и графические пользовательские интерфейсы программ, использованы некоторые дополнительные библиотеки. Помимо этого, было уделено внимание тестированию и отладке программного кода, что является неотъемлемой частью разработки программного обеспечения.

Получившиеся в результате разработки программы успешно выполняются интерпретатором языка Python.

1. **Список используемой Литературы**
   1. Любанович Б. Простой Python. Современный стиль программирования. – С-П.:Питер, - 2019
   2. Туториал по написанию кода на Python[Электронный ресурс], - <https://pythonru.com/uroki/biblioteka-pygame-chast-1-vvedenie>
   3. Пример написания игры на Pygame[Электронный ресурс], - <https://coderslegacy.com/python/pygame-rpg-building-the-base/>
   4. Содержательный туториал по библиотеке Pygame[Электронный ресурс], - <https://habr.com/ru/post/588605/>