

Колледж космического машиностроения и технологий

КУРСОВОЙ ПРОЕКТ

По МДК.01.02 «Прикладное программирование»

Тема: «Разработка игры «Симулятор Программиста» на Python»

Выполнил студент

Зайцев Никита Евгеньевич

Группа П2-18

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(Подпись)

\_\_\_\_\_\_\_\_\_\_\_\_ (Дата сдачи работы)

Проверил преподаватель

Гусятинер Леонид Борисович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(Подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(Оценка)

Королёв 2020 г.

Оглавление

[**Введение** 3](#_Toc75478005)

[**1.** **Теоретическая часть** 4](#_Toc75478006)

[**1.1.** **Описание предметной области** 4](#_Toc75478007)

[**1.2.** **Описание существующих разработок** 5](#_Toc75478008)

[**1.2.1.** **PyGame.** 5](#_Toc75478009)

[**1.2.2.** **PyQt.** 6](#_Toc75478010)

[**1.2.3.** **Unreal Engine.** 7](#_Toc75478011)

[**1.2.4.** **Unity** 8](#_Toc75478012)

[**2.** **Проектная часть** 10](#_Toc75478013)

[**2.1.** **Диаграмма прецедентов** 10](#_Toc75478014)

[**2.2.** **Выбор инструментов** 11](#_Toc75478015)

[**2.3.** **Проектирование сценария** 12](#_Toc75478016)

[**2.4.** **Диаграмма классов** 13](#_Toc75478017)

[**2.5.** **Описание главного модуля** 14](#_Toc75478018)

[**2.6.** **Описание спецификаций к модулям** 26](#_Toc75478019)

[**2.7.** **Описание модулей** 27](#_Toc75478020)

[**2.8.** **Описание тестовых наборов модулей** 28](#_Toc75478021)

[**2.9.** **Описание применения средств отладки** 31](#_Toc75478022)

[**3.** **Эксплуатационная часть** 32](#_Toc75478023)

[**3.1** **Руководство оператора** 32](#_Toc75478024)

[**Заключение** 40](#_Toc75478025)

[**Список литературы и интернет-источников** 41](#_Toc75478026)

[**Приложение 1. Код главного модуля game.py.** 42](#_Toc75478027)

[**Приложение 2. Код модуля базы данных settings.py.** 52](#_Toc75478028)

# **Введение**

Целью данного курсового проекта является показ возможностей библиотеки “Tkinter”. Эта тема является актуальной для студентов, которые хотят попробовать создавать инкрементальные игры, но в более простой среде. Данный курсовой проект позволит понять основные принципы разработки своей игры. Так же в данном проекте будет простой для понимания интерфейс и небольшой порог вхождения.

**В первой части** будет рассмотрена предметная область данной темы, а также несколько продуктов по данной теме.

**Во второй части** будут рассмотрены инструменты и модули, которые были разработаны, структура программной части и листинги ключевых частей программных модулей.

**В третьей части** будет рассмотрено руководство для пользователей.

**В заключительной части** будет приведен To-do лист с планами по доработки программы, а также сделаны общие выводы о получившемся проекте.

# **Теоретическая часть**

## **Описание предметной области**

Tkinter — кросс-платформенная событийно-ориентированная графическая библиотека на основе средств Tk, данная библиотека позволяет создавать специальные окна, взаимодействовать и располагать внутри окон элементы интерфейса, картинки. Входит в стандартную библиотеку Python.

Помимо библиотеки Tkinter есть и другие среды для разработки игр, но большинство устанавливаются в виде отдельного приложения для компьютера, а данная библиотека позволяет напрямую писать весь код необходимый для комфортной и многофункциональной разработки своей инкрементальной игры.

Инкрементальная игра (Кликер) - это компьютерная игра, игровой процесс в которой состоит из выполнения простых действий, таких как многократное нажатие на экран. Нажатия, как правило, выполняются, чтобы заработать игровую валюту. В некоторых играх постоянно нажимать на экран и вовсе не нужно (при условии покупки различных улучшений), так валюта зарабатывается сама, в том числе и в отсутствие игрока.

В инкрементальной игре игрок нажимает на кнопку и получает определённое количество валюты. Игрок может потратить заработанные «деньги» на покупку игровых предметов или навыков, которые позволят игроку зарабатывать валюту быстрее или автоматически, даже без необходимости нажатия кнопок. Часто в таких играх есть источники дохода, которые зарабатывают некоторое количество монет за определённое время. Например, это могут быть здания, фабрики или фермы. Каждый из этих источников увеличивает количество зарабатываемой валюты в секунду. Но стоимость зданий тоже растёт экспоненциально, поэтому переход, скажем, с здания третьего уровня на здание четвёртого занимает столько же времени, сколько переход с первого здания на второе (иногда — даже больше).

Инкрементальные игры набрали популярность в 2013 году.

## **Описание существующих разработок**

В этом разделе рассмотрены некоторые уже имеющиеся программы и библиотеки Python для создания игры, а также их цели и требования.

Интерфейс таких программ чаще всего простой и понятный для обычных пользователей.

Цели использования библиотеки **Tkinter**:

1. Создания своих проектов;
2. Изменение заднего фона на цвет или картинку;
3. Возможность добавлять практически любой тип интерфейса в исполняемое окно;
4. Редактирование объектов: формы и цвета;
5. Добавление текста в исполняемое окно;
6. Изменение цвета и формы текста;
7. Назначение координат для объектов;
8. Добавление картинок;
9. Привязка коллизии для объектов на поле;
10. Отслеживание нажатий клавиш клавиатуры и мыши;
11. Простота в изучении и использовании библиотеки
12. Предустановлена в Python

Основные требования:

1. Запись и редактирование кода.
2. Сохранение кода после выхода.

### **PyGame.**

**Pygame** — отдельная библиотека языка программирования Python, созданная для разработки 2D игр, которая позволяет на игровом поле: визуализировать простые фигуры и картинки, передвигать их в режиме реального времени, создание простых коллизий и т.д.

Базовые возможности:

1. Создания своих проектов;
2. Изменение заднего фона на цвет или картинку;
3. Добавление на поле объектов;
4. Редактирование объектов: формы и цвета;
5. Добавление текста на поле;
6. Изменение цвета и формы текста;
7. Назначение координат для объектов;
8. Передвижение объектов по координатам;
9. Добавление картинок;
10. Добавление коллизии;
11. Привязка коллизии для объектов на поле;
12. Отслеживание нажатий клавиш клавиатуры и мыши;
13. Отслеживание положения мыши на поле;

### **PyQt.**

PyQt — это библиотека, которая позволяет использовать фреймворк Qt GUI (GUI — это графический интерфейс пользователя) в Python. Сам Qt, как известно, написан на C++. Используя его в Python, вы можете создавать приложения намного быстрее, не жертвуя при этом значительной частью производительности C++.

PyQt5 это самая последняя, пятая версия Qt. Еще можно найти в интернете случайное упоминание PyQt4, но эта версия устарела и больше не поддерживается.

Базовые возможности:

1. Создания своих проектов;
2. Изменение заднего фона на цвет или картинку;
3. Добавление на поле объектов;
4. Редактирование объектов: формы и цвета;
5. Добавление текста на поле;
6. Изменение цвета и формы текста;
7. Назначение координат для объектов;
8. Передвижение объектов по координатам;
9. Добавление картинок;
10. Отслеживание нажатий клавиш клавиатуры и мыши;
11. Отдельное приложение для дизайна интрефейса

### **Unreal Engine.**

Приложение для создания проектов и последующая разработка игры внутри этого проекта.

Данный движок создан на языке программирование C++. Для написания кода можно использовать другие языки программирования, например Python, который будет использоваться в движке. Для этого нужно использовать приложение Visual Studio для которого нужно установить расширение нужного языка.

Базовые возможности:

1. Создание проекта.
2. Добавление карт.
3. Изменение масштаба и формы карт.
4. Добавление на карту объектов с коллизией и без нее.
5. Создание объекта.
6. Изменение уже созданного объекта.
7. Изменение параметров объекта.
8. Присвоение объекту своей текстурной развертки.
9. Возможность рисования объектом на карте.
10. Добавление сложных моделей на карту.
11. Создание анимации для объектов.
12. Создание визуальных эффектов.
13. Добавление эффектов на карту.
14. Создание источников освещения.
15. Изменение параметров для освещения.

Возможности при установке плагинов:

1. Более продвинуты и реалистичные эффекты.
2. Добавление новых функций для объектов.
3. Изменение визуального восприятия объектов.

### **Unity**

**Unity** — еще одни движок для разработки игр. Его отличие от Unreal Engine состоит в том, что Unity подходит для создания игр с мульташкой графикой.

На Unity написаны тысячи игр, приложений, визуализации математических моделей, которые охватывают множество платформ и жанров. При этом Unity используется как крупными разработчиками, так и независимыми студиями.

Редактор Unity имеет простой Drag&Drop интерфейс, а также установкой плагинов KALI который легко настраивать, состоящий из различных окон, благодаря чему можно производить отладку игры прямо в редакторе. Движок использует для написания скриптов C#. Ранее поддерживались также Boo и модификация JavaScript, известная как UnityScript.

Базовые возможности:

1. Создание проекта.
2. Добавление карт.
3. Изменение масштаба и формы карт.
4. Добавление на карту объектов с коллизией и без нее.
5. Создание объекта.
6. Изменение уже созданного объекта.
7. Изменение параметров объекта.
8. Присвоение объекту своей текстурной развертки.
9. Возможность рисования объектом на карте.
10. Добавление сложных моделей на карту.
11. Создание анимации для объектов.
12. Создание визуальных эффектов.
13. Добавление эффектов на карту.
14. Создание источников освещения.
15. Изменение параметров для освещения.

Возможности при установке плагинов:

1. Более продвинуты и реалистичные эффекты.
2. Добавление новых функций для объектов.
3. Изменение визуального восприятия объектов.

# **Проектная часть**

## **Диаграмма прецедентов**

В этом разделе представлена диаграмма прецедентов. На диаграмме показаны все возможные функциональные и поведенческие отношения.

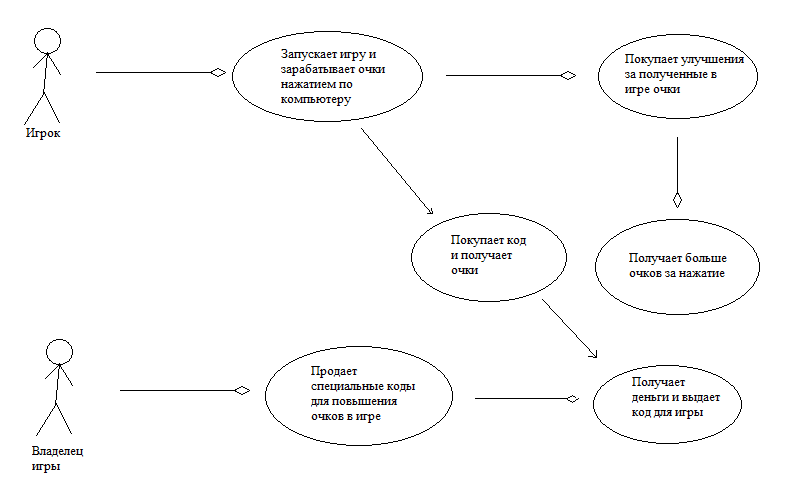


Рисунок 2. Диаграмма прецедентов

## **Выбор инструментов**

При выборе инструментов было проведено сравнение по критериям, представленных в таблице 1.

Степень важности критерия выбиралась из: низкая, ниже средней, средняя, ниже высокой, высокая.

Таблица 1. Критерии выбора инструмента.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Критерий | Участие в корпоративном проекте | Простота сопровождения | Наличие библиотек | Наличие документации на русском языке | Скорость разработки |
| Важность критерия | Высокая | Средняя | Высокая | Ниже средней | Ниже высокой |

Исходя из этих критериев, я сравнил 3 языка программирования от 0 до 10 баллов за критерий.

Таблица 2. Оценка языков программирования.

|  |  |  |  |
| --- | --- | --- | --- |
|  | C++ | Python | Object Pascal |
| Участие в корпоративном проекте | 10 | 8 | 4 |
| Простота сопровождения | 7 | 10 | 3 |
| Наличие библиотек | 6 | 10 | 4 |
| Наличие документации на русском языке | 8 | 6 | 6 |
| Скорость разработки | 6 | 10 | 3 |
| Итого баллов | 37 | 44 | 20 |

По результатам сравнения был выбран язык программирования Python.

## **Проектирование сценария**

В данном разделе приведен сценарий использования игры.

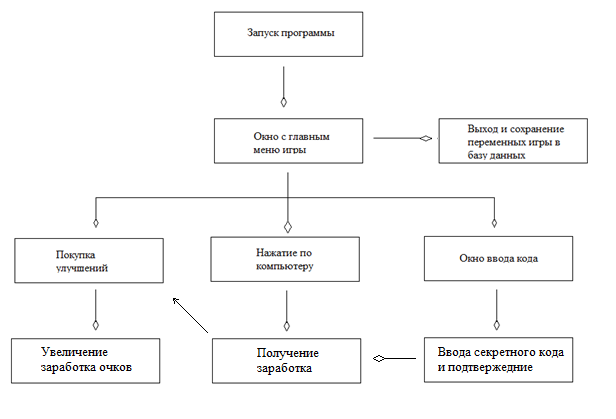


Рисунок 3. Сценарий использования

Пользователь после запуска программы может выполнить 4 действия: Нажать по компьютеру, купить улучшение, выйти из программы или запустить окно для ввода специального кода.

При выборе выхода, игра сохраняет данные в базу данных и заканчивает свою работу, при выборе ввода специального кода, запускается отдельное окно для ввода и при правильном вводе кода, выдаются игровые бонусы.

## **Диаграмма классов**

В данном разделе представлены все классы, использующиеся в проекте, а также их отношения между собой.

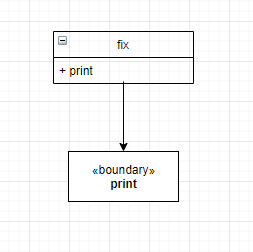


Рисунок 4. Диаграмма классов для проекта

Класс «fix» содержит вывод уведомление об успешной загрузке в консоль.

## **Описание главного модуля**

В главный модуль также импортируется еще 1 модуль, который отвечает за загрузку переменных из базы данных и за создание самой базы данных.

Полный код главного модуля находится в приложении 1.

**Листинг 1. Создание интерфейса игры.**

# Создание главного окна игры

mainwindow = Tk()

mainwindow.geometry("580x360+700+300")

mainwindow.title("Симулятор программиста")

ico = Image.open('img\Icon.ico')

photo = ImageTk.PhotoImage(ico)

mainwindow.wm\_iconphoto(False,photo)

mainwindow["bg"] = "gray22"

mainwindow.protocol('WM\_DELETE\_WINDOW', save)

mainmenu = Menu(mainwindow)

mainwindow.config(menu=mainmenu)

# Создание меню с информацией

helpmenu = Menu(mainmenu, tearoff=0)

helpmenu2 = Menu(helpmenu, tearoff=0)

mainmenu.add\_cascade(label="О игре",menu=helpmenu)

helpmenu.add\_cascade(label="Информация",menu=helpmenu2)

helpmenu2.add\_command(label="Версия v1.1",)

helpmenu2.add\_separator()

helpmenu2.add\_command(label="Секретный код - P2-18\_ZN",)

helpmenu.add\_separator()

helpmenu.add\_command(label="Выход",)

# Переменные для обновления интерфейса игры

MoneyText = StringVar()

MoneyText.set("{}".format(int(money)))

UpgradeText = StringVar()

UpgradeText.set("{}".format(int(upgrade)))

PriceRAMset = StringVar()

PriceRAMset.set("{}".format(int(priceRAM)))

MoneyRAMset = StringVar()

MoneyRAMset.set("{}".format(int(moneyRAM)))

KolVoRAMset = StringVar()

KolVoRAMset.set("{}".format(int(KolVoRAM)))

PriceCPUset = StringVar()

PriceCPUset.set("{}".format(int(priceCPU)))

MoneyCPUset = StringVar()

MoneyCPUset.set("{}".format(int(moneyCPU)))

KolVoCPUset = StringVar()

KolVoCPUset.set("{}".format(int(KolVoCPU)))

PriceCARDset = StringVar()

PriceCARDset.set("{}".format(int(priceCARD)))

MoneyCARDset = StringVar()

MoneyCARDset.set("{}".format(int(moneyCARD)))

PriceCARDset = StringVar()

PriceCARDset.set("{}".format(int(priceCARD)))

MoneyCARDset = StringVar()

MoneyCARDset.set("{}".format(int(moneyCARD)))

KolVoCARDset = StringVar()

KolVoCARDset.set("{}".format(int(KolVoCARD)))

# Переменные с изображениями

img = ImageTk.PhotoImage(Image.open("img\comp.png"))

img2 = ImageTk.PhotoImage(Image.open("img\cart.png"))

img3 = ImageTk.PhotoImage(Image.open("img\money.png"))

img4 = ImageTk.PhotoImage(Image.open("img\shop\cpu.png"))

img5 = ImageTk.PhotoImage(Image.open("img\shop.png"))

img6 = ImageTk.PhotoImage(Image.open("img\logo.png"))

img7 = ImageTk.PhotoImage(Image.open("img\shop\card.png"))

img8 = ImageTk.PhotoImage(Image.open("img\shop\oram.png"))

img9 = ImageTk.PhotoImage(Image.open("img\logoshop.png"))

img10 = ImageTk.PhotoImage(Image.open("img\error.png"))

img11 = ImageTk.PhotoImage(Image.open("img\codebg.png"))

# Показ лого и создание фонового оформления игры

Logo = Label(mainwindow, background="gray22", image=img6)

Logo.place(x=28, y=0)

LogoShop = Label(mainwindow, background="#414141", image=img9)

LogoShop.place(x=10, y=85)

# Вкладка 'Магазин'

lb3 = Label(mainwindow, text="Магазин", background="#414141", font = ('Arial' , 15), foreground="#ccc")

lb3.place(x=53, y=89)

b4 = Label(mainwindow, background="#414141", image=img2)

b4.place(x=18, y=87)

# Создание окна с информацией о счете в игре

MoneyEntry = Label(mainwindow, image=img3, background="gray22")

MoneyEntry.place(x=220, y=12)

ShopEntry = Label(mainwindow, image=img5, background="#414141")

ShopEntry.place(x=10, y=123)

lb1 = Label(mainwindow, textvariable=MoneyText, font = ('Arial' , 40), background="#414141", foreground="#ccc")

lb1.place(x=230, y=14)

lb2 = Label(mainwindow, text="Заработок за нажатие:", background="#414141", foreground="#ccc")

lb2.place(x=230, y=70)

lb1 = Label(mainwindow, textvariable=UpgradeText, background="#414141", foreground="#ccc")

lb1.place(x=359, y=70)

# Создание товара для покупки в магазине

b2 = Button(mainwindow, background="#414141", relief = 'flat', image=img8, activebackground="#333333", command=upgradeRAM)

b2.place(x=37, y=130)

lb7 = Label(mainwindow, text="RAM", background="#414141", foreground="#ccc")

lb7.place(x=49, y=190)

lb8 = Label(mainwindow, text="Цена:", background="#414141", foreground="#ccc")

lb8.place(x=36, y=210)

lb13 = Label(mainwindow, textvariable=PriceRAMset, background="#414141", foreground="#ccc")

lb13.place(x=70, y=210)

lb9 = Label(mainwindow, text="Заработок:", background="#414141", foreground="#ccc")

lb9.place(x=26, y=230)

lb14 = Label(mainwindow, textvariable=MoneyRAMset, background="#414141", foreground="#ccc")

lb14.place(x=90, y=230)

lb25 = Label(mainwindow, text="Уровень:", background="#414141", foreground="#ccc")

lb25.place(x=33, y=250)

lb26 = Label(mainwindow, textvariable=KolVoRAMset, background="#414141", foreground="#ccc")

lb26.place(x=87, y=250)

# Создание окна с информацией о счете в игре

b5 = Button(mainwindow, background="#414141", relief = 'flat', image=img4, activebackground="#333333", command=upgradeCPU)

b5.place(x=130, y=130)

lb4 = Label(mainwindow, text="CPU", background="#414141", foreground="#ccc")

lb4.place(x=143, y=190)

lb5 = Label(mainwindow, text="Цена:", background="#414141", foreground="#ccc")

lb5.place(x=126, y=210)

lb15 = Label(mainwindow, textvariable=PriceCPUset, background="#414141", foreground="#ccc")

lb15.place(x=160, y=210)

lb6 = Label(mainwindow, text="Заработок:", background="#414141", foreground="#ccc")

lb6.place(x=116, y=230)

lb16 = Label(mainwindow, textvariable=MoneyCPUset, background="#414141", foreground="#ccc")

lb16.place(x=180, y=230)

lb23 = Label(mainwindow, text="Уровень:", background="#414141", foreground="#ccc")

lb23.place(x=126, y=250)

lb24 = Label(mainwindow, textvariable=KolVoCPUset, background="#414141", foreground="#ccc")

lb24.place(x=180, y=250)

# Создание окна с информацией о счете в игре

b6 = Button(mainwindow, background="#414141", relief = 'flat', image=img7, activebackground="#333333", command=upgradeCARD)

b6.place(x=223, y=139)

lb10 = Label(mainwindow, text="Videocard", background="#414141", foreground="#ccc")

lb10.place(x=222, y=190)

lb11 = Label(mainwindow, text="Цена:", background="#414141", foreground="#ccc")

lb11.place(x=218, y=210)

lb17 = Label(mainwindow, textvariable=PriceCARDset, background="#414141", foreground="#ccc")

lb17.place(x=251, y=210)

lb12 = Label(mainwindow, text="Заработок:", background="#414141", foreground="#ccc")

lb12.place(x=211, y=230)

lb17 = Label(mainwindow, textvariable=MoneyCARDset, background="#414141", foreground="#ccc")

lb17.place(x=275, y=230)

lb22 = Label(mainwindow, text="Уровень:", background="#414141", foreground="#ccc")

lb22.place(x=221, y=250)

lb25 = Label(mainwindow, textvariable=KolVoCARDset, background="#414141", foreground="#ccc")

lb25.place(x=275, y=250)

# Кнопка для открытия дочернего окна

b3 = Button(mainwindow, text="Ввести код", relief = 'flat', background="#555", foreground="#ccc", activebackground="#333333", width=15, height=1, command=code)

b3.place(x=103, y=290)

# Кнопка для запуска функции которая считывает нажатия

clickbtn = Button(mainwindow, background="gray22", relief = 'flat', image=img, activebackground="#333333", command=click)

clickbtn.place(x=320, y=100)

# Запуск цикла обработки событий

mainloop()

Данный метод создает специальное окно, в котором расположен интерфейс игры

**Листинг 2. Функции для покупки улучшений.**

def upgradeRAM():

global money

global upgrade

global priceRAM

global moneyRAM

global KolVoRAM

if money >= priceRAM:

money -= priceRAM

upgrade += moneyRAM

KolVoRAM += 1

priceRAM = priceRAM\*1.07\*\*KolVoRAM

moneyRAM += 1

# Обновление переменных в интерфейсе игры

UpgradeText.set("{}".format(int(upgrade)))

MoneyText.set("{}".format(int(money)))

PriceRAMset.set("{}".format(int(priceRAM)))

MoneyRAMset.set("{}".format(int(moneyRAM)))

KolVoRAMset.set("{}".format(int(KolVoRAM)))

'''Функция для покупки процессора'''

def upgradeCPU():

global money

global upgrade

global priceCPU

global moneyCPU

global KolVoCPU

if money >= priceCPU:

money -= priceCPU

upgrade += moneyCPU

KolVoCPU += 1

priceCPU = priceCPU\*1.07\*\*KolVoCPU

moneyCPU += 2

# Обновление переменных в интерфейсе игры

UpgradeText.set("{}".format(int(upgrade)))

MoneyText.set("{}".format(int(money)))

PriceCPUset.set("{}".format(int(priceCPU)))

MoneyCPUset.set("{}".format(int(moneyCPU)))

KolVoCPUset.set("{}".format(int(KolVoCPU)))

'''Функция для покупки видеокарты'''

def upgradeCARD():

global money

global upgrade

global priceCARD

global moneyCARD

global KolVoCARD

if money >= priceCARD:

money -= priceCARD

upgrade += moneyCARD

KolVoCARD += 1

priceCARD = priceCARD\*1.07\*\*KolVoCARD

moneyCARD += 5

# Обновление переменных в интерфейсе игры

UpgradeText.set("{}".format(int(upgrade)))

MoneyText.set("{}".format(int(money)))

PriceCARDset.set("{}".format(int(priceCARD)))

MoneyCARDset.set("{}".format(int(moneyCARD)))

KolVoCARDset.set("{}".format(int(KolVoCARD)))

Данный метод срабатывает и улучшает заработок очков игроку при нажатии по определенной кнопки интерфейса игры и при наличии нужного количества очков.

**Листинг 3. Функция заработка очков.**

def click():

global money

global upgrade

money += upgrade

# Обновление переменных в интерфейсе игры

MoneyText.set("{}".format(int(money)))

При нажатии по определенной кнопки интерфейса игры, срабатывает данный метод и выдает полученный заработок на баланс игрока.

**Листинг 4. Функция для открытия дочернего окна.**

def code():

'''Функция для проверки секретного кода и получение бонусов за правильный ввод'''

def secret() :

global money

global upgrade

global SecretCodeActive

# Проверка введеного кода

if SecretCodeEN.get() == 'P2-18\_ZN':

# Проверка не введен ли код дважды

if SecretCodeActive == 0:

SecretCodeActive = 1

# Получение бонусов

money = 10000

upgrade += 10

# Обновление переменных в интерфейсе игры

UpgradeText.set("{}".format(int(upgrade)))

MoneyText.set("{}".format(int(money)))

# Уведомление о активации кода

codebg2 = Label(code, background="gray22", image=img10)

codebg2.place(x=1,y=155)

lb21 = Label(code, text="Код активирован", background="gray22", foreground="#ccc")

lb21.place(x=53, y=155)

else:

# Уведомление о повторной активации кода

codebg2 = Label(code, background="gray22", image=img10)

codebg2.place(x=1,y=155)

lb21 = Label(code, text="Код уже был активирован", background="gray22", foreground="#ccc")

lb21.place(x=28, y=155)

else:

# Уведомление о ошибки активации кода

codebg2 = Label(code, background="gray22", image=img10)

codebg2.place(x=1,y=155)

lb21 = Label(code, text="Ошибка активации кода", background="gray22", foreground="#ccc")

lb21.place(x=33, y=155)

Данный метод открывает окно для ввода специального кода.

**Листинг 5. Функция сохранение переменных в базу данных.**

'''Функция для сохранение при выходе из игры'''

def save():

global money

global upgrade

global priceRAM

global moneyRAM

global KolVoRAM

global priceCPU

global moneyCPU

global KolVoCPU

global priceCARD

global moneyCARD

global KolVoCARD

# Создание списка для записи в базу данных

save = [('1','money', money),

('2','upgrade ', upgrade),

('3','priceRAM', priceRAM),

('4','moneyRAM', moneyRAM),

('5','KolVoRAM', KolVoRAM),

('6','priceCPU', priceCPU),

('7','moneyCPU', moneyCPU),

('8','KolVoCPU', KolVoCPU),

('9','priceCARD', priceCARD),

('10','moneyCARD', moneyCARD),

('11','KolVoCARD', KolVoCARD),

('12', 'SecretCodeActive', SecretCodeActive)]

# Удаление старой таблицы

curs.execute('DELETE FROM Save')

# Запись списка в таблицу

curs.executemany('INSERT INTO Save VALUES (?, ?, ?)' , save)

# Сохранение базы данных

conn.commit()

# Создание окна предупреждения перед выходом из игры

if messagebox.askokcancel('Выход', ' Прогресс сохранен\nВы действительно хотите закрыть игру?'):

mainwindow.destroy()

Данный метод сохраняет данные из переменных в базу данных.

**Листинг 6. Создание дочернего окна.**

# Создание дочернего окна

code = Toplevel(mainwindow)

code.title("Ввод кода")

code.geometry("200x175+480+380")

code["bg"] = "gray22"

ico = Image.open('img\Icon.ico')

photo = ImageTk.PhotoImage(ico)

code.wm\_iconphoto(False,photo)

# Переменная для ввода кода

SecretCodeEN = StringVar()

# Интерфейс дочернего окна

codebg = Label(code, background="#414141", image=img11)

codebg.place(x=1,y=20)

b18 = Button(code, text="Подтвердить", relief = 'flat', background="#555", font = ('Arial' , 13), foreground="#ccc", activebackground="#333333", width=15, height=1, command=secret)

b18.place(x=30, y=110)

en1 = Entry(code, width=15, textvariable=SecretCodeEN, background="gray22", relief = 'flat', font = ('Arial' , 13), foreground="#ccc")

en1.place(x=33, y=74)

lb19 = Label(code, text="Введите код", background="#414141", font = ('Arial' , 20), foreground="#ccc")

lb19.place(x=19, y=25)

Данный метод создает окно ввода кода и делает расстановку элементов интерфейса.

**Листинг 7. Создание базы данных и загрузка данных в переменные.**

# Создание базы данных и курсора.

conn = sqlite3.connect('save\SaveFile.db')

curs = conn.cursor()

# Проверка удалена ли таблица в файле 'SaveFile.db'.

try:

save = [('1','money', 0),

('2','upgrade ', 1),

('3','priceRAM', 50),

('4','moneyRAM', 5),

('5','KolVoRAM', 0),

('6','priceCPU', 5000),

('7','moneyCPU', 20),

('8','KolVoCPU', 0),

('9','priceCARD', 25000),

('10','moneyCARD', 60),

('11','KolVoCARD', 0),

('12', 'SecretCodeActive', 0)]

curs.execute("""CREATE TABLE Save(id TEXT,

Name TEXT,

Save INTENGER)""")

curs.executemany('INSERT INTO Save VALUES (?, ?, ?)' , save)

conn.commit()

except sqlite3.OperationalError:

None

# Загрузка переменных из базы данных в игру.

\*save, = curs.execute('SELECT \* FROM Save')

# Переменные денег, цены, улучшения.

money = save[0][2]

upgrade = save[1][2]

priceRAM = save[2][2]

moneyRAM = save[3][2]

KolVoRAM = save[4][2]

priceCPU = save[5][2]

moneyCPU = save[6][2]

KolVoCPU = save[7][2]

priceCARD = save[8][2]

moneyCARD = save[9][2]

KolVoCARD = save[10][2]

SecretCodeActive = save[11][2]

Данный метод создает базу данных и загружает сохраненные данные в переменные.

**Листинг 8. Вывод об успешной загрузке.**

# Класс для вывода принта в консоль

class fix:

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

None

def print():

print('Загрузка прошла успешно')

# Запуск класса

fix.print()

Данный метод выводит в консоль уведомление об успешной загрузке игры.

## **Описание спецификаций к модулям**

В данном разделе описан публичный член модулей курсового проекта.

В модуле (settings.py) содержится публичный метод класса fix. Так как все эти методы были описаны в разделе 2.4, в данном разделе описание данных методов опущено. В данном модуле также есть один объект:

1. Print – объект, который реализует вывод уведомления в консоль.

## **Описание модулей**

Кроме главного модуля программа содержит еще один модуль который реализует загрузку базы данных (settings.py).

Полный код модуля приведен в приложении 2. В этом разделе приведена блок схема класса fix из модуля settings.py.

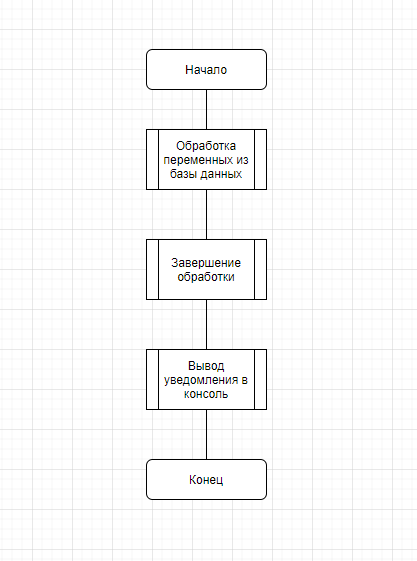


Рисунок 5. Блок-схема класса fix.

## **Описание тестовых наборов модулей**

В этом разделе будет визуализирован процесс работы приложения в виде тестов:

Тест 1. Запуск приложения.

Действия: Запустить exe файл.

Ожидаемый результат: Успешный запуск окна.

Результат теста:

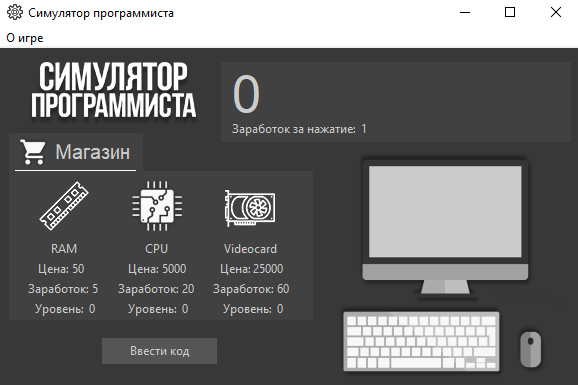


Рисунок 6. Тест 1

Тест 2. Заработок при нажатии

Действия: Нажать на компьютер.

Ожидаемый результат: Получение 1 очка на баланс игрока.

Результат теста:

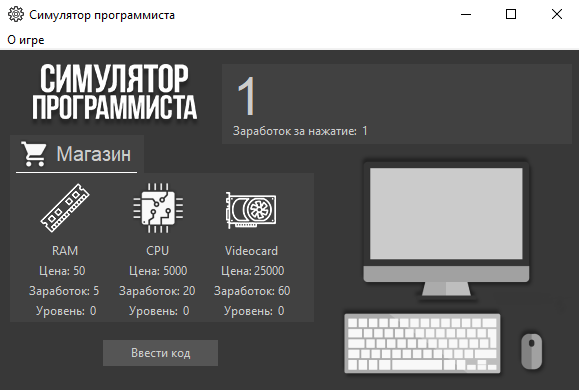


Рисунок 7. Тест 2

Тест 3. Покупка улучшения

Действия: Накопить 50 очков для покупки улучшения “RAM” в магазине, а после нажать на иконку оперативной памяти.

Ожидаемый результат: Удаление 50 очков с баланса и получение значения “6” в графе “Заработок за нажатие”

Результат теста:

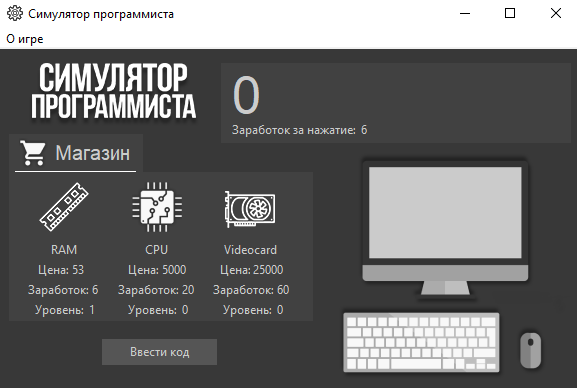


Рисунок 8. Тест 3

Тест 4. Сохранение данных

Действия: Закрыть программу

Ожидаемый результат: Значения из переменных игры запишутся в базу данных.

Результат теста:

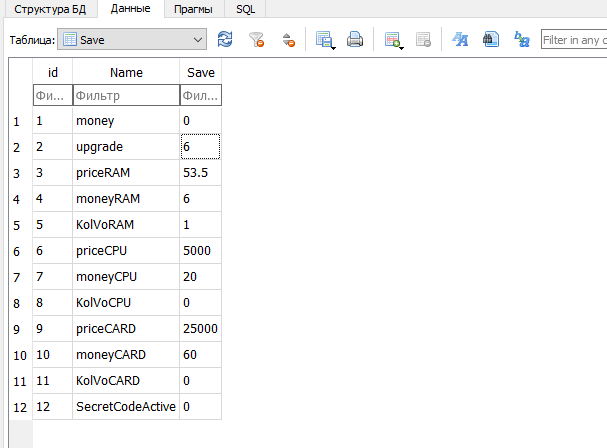


Рисунок 9. Тест 4

## **Описание применения средств отладки**

В этом разделе показано умение применять средства отладки.

В ходе написания курсового проекта при попытке запустить скрипт было получено данное сообщение:

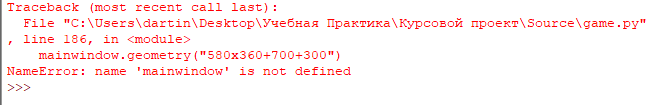


Рисунок 10. До применения средств отладки

После получения данного сообщения были просмотрены 185 и 186 строки модуля game.py и была обнаружена ошибка, которая впоследствии была устранена, а после попытки запуска скрипта получено данное сообщение:



Рисунок 11. После применения средств отладки

Это означает что ошибка была устранена и скрипт запустился.

# **Эксплуатационная часть**

## **Руководство оператора**

**АННОТАЦИЯ**

В данном программном документе приведено руководство оператора по применению и эксплуатации программы «Симулятор программиста», предназначенной для облегчения работы отелей.

В данном программном документе, в разделе «Назначение программы» указаны сведения о назначении программы и информация, достаточная для понимания функций программы и ее эксплуатации.

В разделе «Условия выполнения программы» указаны условия, необходимые для выполнения программы (минимальный состав аппаратных и программных средств и т.п.).

В данном программном документе, в разделе «Выполнение программы» указана последовательность действий оператора, обеспечивающих загрузку, запуск, выполнение и завершение программы, приведено описание функций, формата и возможных вариантов команд, с помощью которых оператор осуществляет загрузку и управляет выполнением программы, а также ответы программы на эти команды.

Оформление программного документа «Руководство оператора» произведено по требованиям ЕСПД (ГОСТ 19.101-77 [[1]](#footnote-1)1), ГОСТ 19.103-77 [[2]](#footnote-2)2), ГОСТ 19.104-78\* [[3]](#footnote-3)3), ГОСТ 19.105-78\* [[4]](#footnote-4)4), ГОСТ 19.106-78\* [[5]](#footnote-5)5), ГОСТ 19.505-79\* [[6]](#footnote-6)6), ГОСТ 19.604-78\* [[7]](#footnote-7)7)).

1. **Назначение программы**
   1. **Функциональное назначение программы.**

Специальное программное обеспечение «Симулятор программиста» используется для понимания работы с библиотекой.

* 1. **Эксплуатационное назначение программы.**

Специальное программное обеспечение «Симулятор программиста» может эксплуатироваться любым пользователем, который хочет изучить библиотеку.

* 1. **Состав функций**
     1. **Функция заработка при нажатии.**

Эта функция позволяет получать очки за нажатие.

* + 1. **Функция покупки улучшения.**

Эта функция позволяет покупать улучшения для повышения заработка.

* + 1. **Функция ввода кода.**

Эта функция позволяет ввести секретный код и получить за это игровые бонусы.

* + 1. **Функция сохранения данных игры в базу данных.**

Эта функция позволяет сохранять прогресс в игре.

1. **Условия выполнения программы**
   1. **Минимальный состав аппаратных средств**

ОС: Windows 10

Процессор: Как минимум 1 ГГц или SoC.

ОЗУ: 1 ГБ (для 32-разрядных систем) или 2 ГБ (для 64-разрядных систем).

Место на жестком диске: 16 ГБ (для 32-разрядных систем) или 20 ГБ (для 64-разрядных систем).

Видеоадаптер: DirectX версии не ниже 9 с драйвером WDDM 1.0.

Дисплей: 800 x 600.

* 1. **Минимальный состав программных средств**

Дополнительные программные средства не требуются.

* 1. **Требование к персоналу (пользователю)**

Конечный пользователь должен уметь пользоваться клавиатурой и уметь запускать приложения.

1. **Выполнение программы**
   1. **Загрузка и запуск программы**

Перед запуском программы вам нужно распаковать файлы прилагаемые на DVD носителе и запустить exe файл.

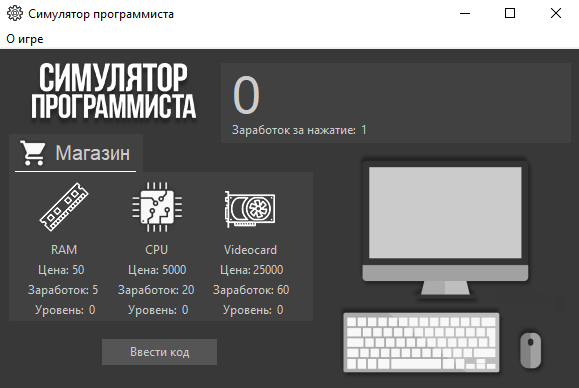


Рисунок 12. Окно игры

Ввести код – открывает окно для ввода специального кода.

* 1. **Выполнение программы**
     1. **Функция заработка при нажатии.**

Для заработка очков, нажмите по компьютеру и посмотрите на баланс.

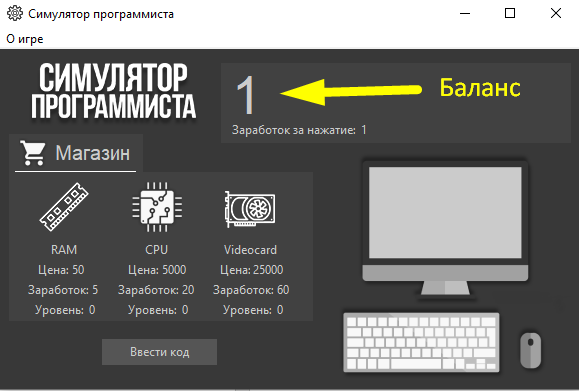


Рисунок 13. Нажатие по компьютеру

* + 1. **Функция покупки улучшения.**

Для покупки улучшений, вам нужно накопить 50 очков при помощи нажатия по компьютеру, при накоплении нужно количества очков, нужно нажать на иконку оперативной памяти, после покупки ваш заработок за нажатие увеличиться на 5.

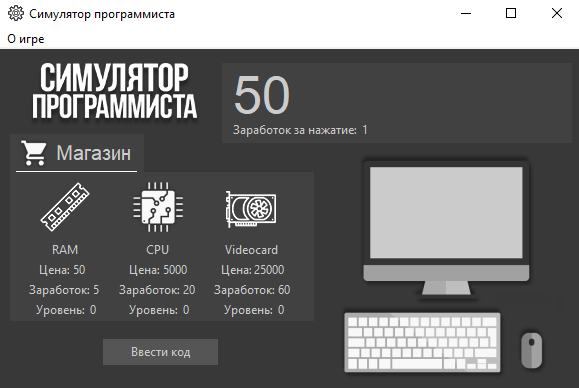


Рисунок 14. Накопление 50 очков

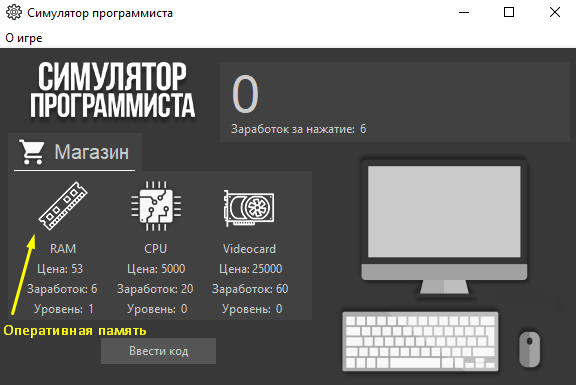


Рисунок 15. Покупка улучшения

* + 1. **Функция ввода кода.**

Нажмите на кнопку “Ввести код”, откроется окно с вводом кода, введите в поле “P2-18\_ZN”, если код был введен верно, то на игровой баланс выдается 10000 очков и заработок 10 очков за нажатие по компьютеру.

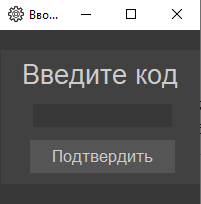


Рисунок 16. Окно ввода кода

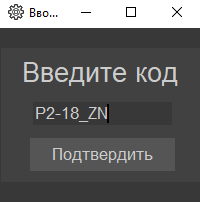


Рисунок 17. Специальный код

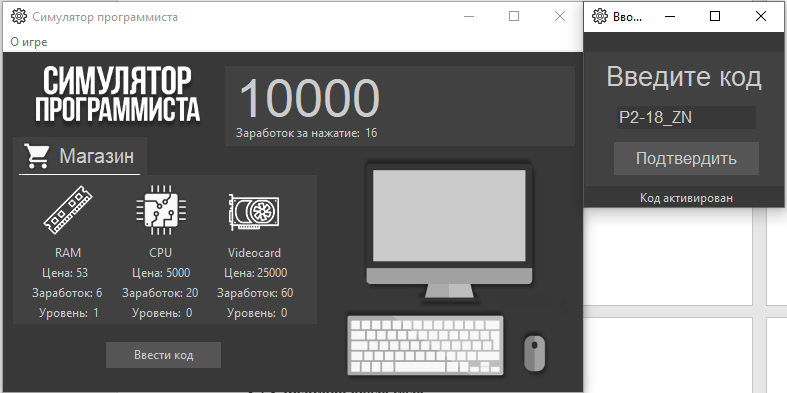


Рисунок 18. Успешный ввод кода и выдача бонусов

* + 1. **Функция сохранения данных игры в базу данных.**

Наведите курсор на кнопку закрытия программы, после нажатия игра сохранит данные в базу данных, появиться окно предупреждения об выходе из программы, нажмите на кнопку “Ок”, программа завершит выполнение.

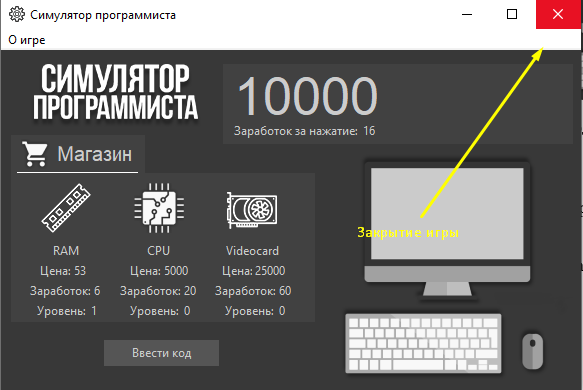


Рисунок 19. Кнопка закрытие игры

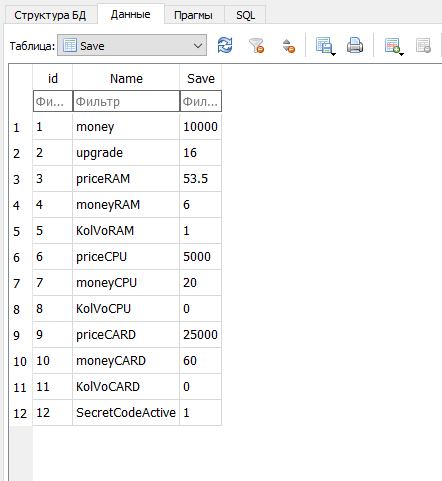


Рисунок 20. Сохраненные данные

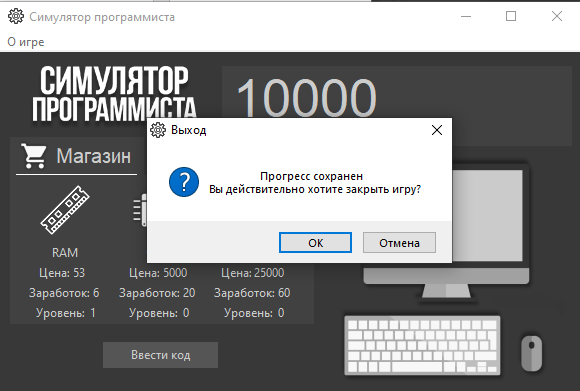


Рисунок 21. Окно предупреждения

* 1. **Завершение работы программы**

Чтобы завершить работу программы нажмите на кнопку «Закрыть» в верхнем правом углу окна.

# **Заключение**

В результате выполнения курсового проекта была написана программа «Симулятор программиста» для изучения библиотеки.

В ходе работы были проанализированы предметная область, существующие разработки, посвященные данному направлению, получены практические навыки по созданию игры на Tkinter.

Также планируется продолжать работу над данным проектом с целью расширения возможностей и удобства приложения для пользователей. Планы по доработкам представлены ниже.

To-do лист:

1. Расширение магазина.
2. Создание отдельного окна с магазином
3. Добавить возможность создавать свои коды и передавать их другим игрокам
4. Улучшение внешнего вида игры.

# **Список литературы и интернет-источников**

1. Онлайн-учебник по библиотеке Tkinter

https://pythonru.com/uroki/biblioteka-pygame-chast-1-vvedenie

1. Изображения для интерфейса игры без авторских прав

<https://icon-icons.com/ru/>

1. Онлайн-учебник по Python

<https://pythonworld.ru/samouchitel-python>

1. Как скомпилировать Python код в .exe

<https://www.make-info.com/kak-skompilirovat-python-kod-v-exe/>

1. ico иконки

<https://icons8.ru/icons/set/ico>

# **Приложение 1. Код главного модуля game.py.**

*from tkinter import \**

*from tkinter import messagebox*

*from PIL import ImageTk, Image*

*import sqlite3*

*from settings import \**

*'''Функция для покупки оперативной памяти'''*

*def upgradeRAM():*

*global money*

*global upgrade*

*global priceRAM*

*global moneyRAM*

*global KolVoRAM*

*if money >= priceRAM:*

*money -= priceRAM*

*upgrade += moneyRAM*

*KolVoRAM += 1*

*priceRAM = priceRAM\*1.07\*\*KolVoRAM*

*moneyRAM += 1*

*# Обновление переменных в интерфейсе игры*

*UpgradeText.set("{}".format(int(upgrade)))*

*MoneyText.set("{}".format(int(money)))*

*PriceRAMset.set("{}".format(int(priceRAM)))*

*MoneyRAMset.set("{}".format(int(moneyRAM)))*

*KolVoRAMset.set("{}".format(int(KolVoRAM)))*

*'''Функция для покупки процессора'''*

*def upgradeCPU():*

*global money*

*global upgrade*

*global priceCPU*

*global moneyCPU*

*global KolVoCPU*

*if money >= priceCPU:*

*money -= priceCPU*

*upgrade += moneyCPU*

*KolVoCPU += 1*

*priceCPU = priceCPU\*1.07\*\*KolVoCPU*

*moneyCPU += 2*

*# Обновление переменных в интерфейсе игры*

*UpgradeText.set("{}".format(int(upgrade)))*

*MoneyText.set("{}".format(int(money)))*

*PriceCPUset.set("{}".format(int(priceCPU)))*

*MoneyCPUset.set("{}".format(int(moneyCPU)))*

*KolVoCPUset.set("{}".format(int(KolVoCPU)))*

*'''Функция для покупки видеокарты'''*

*def upgradeCARD():*

*global money*

*global upgrade*

*global priceCARD*

*global moneyCARD*

*global KolVoCARD*

*if money >= priceCARD:*

*money -= priceCARD*

*upgrade += moneyCARD*

*KolVoCARD += 1*

*priceCARD = priceCARD\*1.07\*\*KolVoCARD*

*moneyCARD += 5*

*# Обновление переменных в интерфейсе игры*

*UpgradeText.set("{}".format(int(upgrade)))*

*MoneyText.set("{}".format(int(money)))*

*PriceCARDset.set("{}".format(int(priceCARD)))*

*MoneyCARDset.set("{}".format(int(moneyCARD)))*

*KolVoCARDset.set("{}".format(int(KolVoCARD)))*

*'''Функция для считывания нажатия по кнопке'''*

*def click():*

*global money*

*global upgrade*

*money += upgrade*

*# Обновление переменных в интерфейсе игры*

*MoneyText.set("{}".format(int(money)))*

*'''Функция для создания дочернего окна, в который пользователь будет вводить секретный код'''*

*def code():*

*'''Функция для проверки секретного кода и получение бонусов за правильный ввод'''*

*def secret() :*

*global money*

*global upgrade*

*global SecretCodeActive*

*# Проверка введеного кода*

*if SecretCodeEN.get() == 'P2-18\_ZN':*

*# Проверка не введен ли код дважды*

*if SecretCodeActive == 0:*

*SecretCodeActive = 1*

*# Получение бонусов*

*money = 10000*

*upgrade += 10*

*# Обновление переменных в интерфейсе игры*

*UpgradeText.set("{}".format(int(upgrade)))*

*MoneyText.set("{}".format(int(money)))*

*# Уведомление о активации кода*

*codebg2 = Label(code, background="gray22", image=img10)*

*codebg2.place(x=1,y=155)*

*lb21 = Label(code, text="Код активирован", background="gray22", foreground="#ccc")*

*lb21.place(x=53, y=155)*

*else:*

*# Уведомление о повторной активации кода*

*codebg2 = Label(code, background="gray22", image=img10)*

*codebg2.place(x=1,y=155)*

*lb21 = Label(code, text="Код уже был активирован", background="gray22", foreground="#ccc")*

*lb21.place(x=28, y=155)*

*else:*

*# Уведомление о ошибки активации кода*

*codebg2 = Label(code, background="gray22", image=img10)*

*codebg2.place(x=1,y=155)*

*lb21 = Label(code, text="Ошибка активации кода", background="gray22", foreground="#ccc")*

*lb21.place(x=33, y=155)*

*# Создание дочернего окна*

*code = Toplevel(mainwindow)*

*code.title("Ввод кода")*

*code.geometry("200x175+480+380")*

*code["bg"] = "gray22"*

*ico = Image.open('img\Icon.ico')*

*photo = ImageTk.PhotoImage(ico)*

*code.wm\_iconphoto(False,photo)*

*# Переменная для ввода кода*

*SecretCodeEN = StringVar()*

*# Интерфейс дочернего окна*

*codebg = Label(code, background="#414141", image=img11)*

*codebg.place(x=1,y=20)*

*b18 = Button(code, text="Подтвердить", relief = 'flat', background="#555", font = ('Arial' , 13), foreground="#ccc", activebackground="#333333", width=15, height=1, command=secret)*

*b18.place(x=30, y=110)*

*en1 = Entry(code, width=15, textvariable=SecretCodeEN, background="gray22", relief = 'flat', font = ('Arial' , 13), foreground="#ccc")*

*en1.place(x=33, y=74)*

*lb19 = Label(code, text="Введите код", background="#414141", font = ('Arial' , 20), foreground="#ccc")*

*lb19.place(x=19, y=25)*

*'''Функция для сохранение при выходе из игры'''*

*def save():*

*global money*

*global upgrade*

*global priceRAM*

*global moneyRAM*

*global KolVoRAM*

*global priceCPU*

*global moneyCPU*

*global KolVoCPU*

*global priceCARD*

*global moneyCARD*

*global KolVoCARD*

*# Создание списка для записи в базу данных*

*save = [('1','money', money),*

*('2','upgrade ', upgrade),*

*('3','priceRAM', priceRAM),*

*('4','moneyRAM', moneyRAM),*

*('5','KolVoRAM', KolVoRAM),*

*('6','priceCPU', priceCPU),*

*('7','moneyCPU', moneyCPU),*

*('8','KolVoCPU', KolVoCPU),*

*('9','priceCARD', priceCARD),*

*('10','moneyCARD', moneyCARD),*

*('11','KolVoCARD', KolVoCARD),*

*('12', 'SecretCodeActive', SecretCodeActive)]*

*# Удаление старой таблицы*

*curs.execute('DELETE FROM Save')*

*# Запись списка в таблицу*

*curs.executemany('INSERT INTO Save VALUES (?, ?, ?)' , save)*

*# Сохранение базы данных*

*conn.commit()*

*# Создание окна предупреждения перед выходом из игры*

*if messagebox.askokcancel('Выход', ' Прогресс сохранен\nВы действительно хотите закрыть игру?'):*

*mainwindow.destroy()*

*# Создание главного окна игры*

*mainwindow = Tk()*

*mainwindow.geometry("580x360+700+300")*

*mainwindow.title("Симулятор программиста")*

*ico = Image.open('img\Icon.ico')*

*photo = ImageTk.PhotoImage(ico)*

*mainwindow.wm\_iconphoto(False,photo)*

*mainwindow["bg"] = "gray22"*

*mainwindow.protocol('WM\_DELETE\_WINDOW', save)*

*mainmenu = Menu(mainwindow)*

*mainwindow.config(menu=mainmenu)*

*# Создание меню с информацией*

*helpmenu = Menu(mainmenu, tearoff=0)*

*helpmenu2 = Menu(helpmenu, tearoff=0)*

*mainmenu.add\_cascade(label="О игре",menu=helpmenu)*

*helpmenu.add\_cascade(label="Информация",menu=helpmenu2)*

*helpmenu2.add\_command(label="Версия v1.1",)*

*helpmenu2.add\_separator()*

*helpmenu2.add\_command(label="Секретный код - P2-18\_ZN",)*

*helpmenu.add\_separator()*

*helpmenu.add\_command(label="Выход",)*

*# Переменные для обновления интерфейса игры*

*MoneyText = StringVar()*

*MoneyText.set("{}".format(int(money)))*

*UpgradeText = StringVar()*

*UpgradeText.set("{}".format(int(upgrade)))*

*PriceRAMset = StringVar()*

*PriceRAMset.set("{}".format(int(priceRAM)))*

*MoneyRAMset = StringVar()*

*MoneyRAMset.set("{}".format(int(moneyRAM)))*

*KolVoRAMset = StringVar()*

*KolVoRAMset.set("{}".format(int(KolVoRAM)))*

*PriceCPUset = StringVar()*

*PriceCPUset.set("{}".format(int(priceCPU)))*

*MoneyCPUset = StringVar()*

*MoneyCPUset.set("{}".format(int(moneyCPU)))*

*KolVoCPUset = StringVar()*

*KolVoCPUset.set("{}".format(int(KolVoCPU)))*

*PriceCARDset = StringVar()*

*PriceCARDset.set("{}".format(int(priceCARD)))*

*MoneyCARDset = StringVar()*

*MoneyCARDset.set("{}".format(int(moneyCARD)))*

*PriceCARDset = StringVar()*

*PriceCARDset.set("{}".format(int(priceCARD)))*

*MoneyCARDset = StringVar()*

*MoneyCARDset.set("{}".format(int(moneyCARD)))*

*KolVoCARDset = StringVar()*

*KolVoCARDset.set("{}".format(int(KolVoCARD)))*

*# Переменные с изображениями*

*img = ImageTk.PhotoImage(Image.open("img\comp.png"))*

*img2 = ImageTk.PhotoImage(Image.open("img\cart.png"))*

*img3 = ImageTk.PhotoImage(Image.open("img\money.png"))*

*img4 = ImageTk.PhotoImage(Image.open("img\shop\cpu.png"))*

*img5 = ImageTk.PhotoImage(Image.open("img\shop.png"))*

*img6 = ImageTk.PhotoImage(Image.open("img\logo.png"))*

*img7 = ImageTk.PhotoImage(Image.open("img\shop\card.png"))*

*img8 = ImageTk.PhotoImage(Image.open("img\shop\oram.png"))*

*img9 = ImageTk.PhotoImage(Image.open("img\logoshop.png"))*

*img10 = ImageTk.PhotoImage(Image.open("img\error.png"))*

*img11 = ImageTk.PhotoImage(Image.open("img\codebg.png"))*

*# Показ лого и создание фонового оформления игры*

*Logo = Label(mainwindow, background="gray22", image=img6)*

*Logo.place(x=28, y=0)*

*LogoShop = Label(mainwindow, background="#414141", image=img9)*

*LogoShop.place(x=10, y=85)*

*# Вкладка 'Магазин'*

*lb3 = Label(mainwindow, text="Магазин", background="#414141", font = ('Arial' , 15), foreground="#ccc")*

*lb3.place(x=53, y=89)*

*b4 = Label(mainwindow, background="#414141", image=img2)*

*b4.place(x=18, y=87)*

*# Создание окна с информацией о счете в игре*

*MoneyEntry = Label(mainwindow, image=img3, background="gray22")*

*MoneyEntry.place(x=220, y=12)*

*ShopEntry = Label(mainwindow, image=img5, background="#414141")*

*ShopEntry.place(x=10, y=123)*

*lb1 = Label(mainwindow, textvariable=MoneyText, font = ('Arial' , 40), background="#414141", foreground="#ccc")*

*lb1.place(x=230, y=14)*

*lb2 = Label(mainwindow, text="Заработок за нажатие:", background="#414141", foreground="#ccc")*

*lb2.place(x=230, y=70)*

*lb1 = Label(mainwindow, textvariable=UpgradeText, background="#414141", foreground="#ccc")*

*lb1.place(x=359, y=70)*

*# Создание товара для покупки в магазине*

*b2 = Button(mainwindow, background="#414141", relief = 'flat', image=img8, activebackground="#333333", command=upgradeRAM)*

*b2.place(x=37, y=130)*

*lb7 = Label(mainwindow, text="RAM", background="#414141", foreground="#ccc")*

*lb7.place(x=49, y=190)*

*lb8 = Label(mainwindow, text="Цена:", background="#414141", foreground="#ccc")*

*lb8.place(x=36, y=210)*

*lb13 = Label(mainwindow, textvariable=PriceRAMset, background="#414141", foreground="#ccc")*

*lb13.place(x=70, y=210)*

*lb9 = Label(mainwindow, text="Заработок:", background="#414141", foreground="#ccc")*

*lb9.place(x=26, y=230)*

*lb14 = Label(mainwindow, textvariable=MoneyRAMset, background="#414141", foreground="#ccc")*

*lb14.place(x=90, y=230)*

*lb25 = Label(mainwindow, text="Уровень:", background="#414141", foreground="#ccc")*

*lb25.place(x=33, y=250)*

*lb26 = Label(mainwindow, textvariable=KolVoRAMset, background="#414141", foreground="#ccc")*

*lb26.place(x=87, y=250)*

*# Создание окна с информацией о счете в игре*

*b5 = Button(mainwindow, background="#414141", relief = 'flat', image=img4, activebackground="#333333", command=upgradeCPU)*

*b5.place(x=130, y=130)*

*lb4 = Label(mainwindow, text="CPU", background="#414141", foreground="#ccc")*

*lb4.place(x=143, y=190)*

*lb5 = Label(mainwindow, text="Цена:", background="#414141", foreground="#ccc")*

*lb5.place(x=126, y=210)*

*lb15 = Label(mainwindow, textvariable=PriceCPUset, background="#414141", foreground="#ccc")*

*lb15.place(x=160, y=210)*

*lb6 = Label(mainwindow, text="Заработок:", background="#414141", foreground="#ccc")*

*lb6.place(x=116, y=230)*

*lb16 = Label(mainwindow, textvariable=MoneyCPUset, background="#414141", foreground="#ccc")*

*lb16.place(x=180, y=230)*

*lb23 = Label(mainwindow, text="Уровень:", background="#414141", foreground="#ccc")*

*lb23.place(x=126, y=250)*

*lb24 = Label(mainwindow, textvariable=KolVoCPUset, background="#414141", foreground="#ccc")*

*lb24.place(x=180, y=250)*

*# Создание окна с информацией о счете в игре*

*b6 = Button(mainwindow, background="#414141", relief = 'flat', image=img7, activebackground="#333333", command=upgradeCARD)*

*b6.place(x=223, y=139)*

*lb10 = Label(mainwindow, text="Videocard", background="#414141", foreground="#ccc")*

*lb10.place(x=222, y=190)*

*lb11 = Label(mainwindow, text="Цена:", background="#414141", foreground="#ccc")*

*lb11.place(x=218, y=210)*

*lb17 = Label(mainwindow, textvariable=PriceCARDset, background="#414141", foreground="#ccc")*

*lb17.place(x=251, y=210)*

*lb12 = Label(mainwindow, text="Заработок:", background="#414141", foreground="#ccc")*

*lb12.place(x=211, y=230)*

*lb17 = Label(mainwindow, textvariable=MoneyCARDset, background="#414141", foreground="#ccc")*

*lb17.place(x=275, y=230)*

*lb22 = Label(mainwindow, text="Уровень:", background="#414141", foreground="#ccc")*

*lb22.place(x=221, y=250)*

*lb25 = Label(mainwindow, textvariable=KolVoCARDset, background="#414141", foreground="#ccc")*

*lb25.place(x=275, y=250)*

*# Кнопка для открытия дочернего окна*

*b3 = Button(mainwindow, text="Ввести код", relief = 'flat', background="#555", foreground="#ccc", activebackground="#333333", width=15, height=1, command=code)*

*b3.place(x=103, y=290)*

*# Кнопка для запуска функции которая считывает нажатия*

*clickbtn = Button(mainwindow, background="gray22", relief = 'flat', image=img, activebackground="#333333", command=click)*

*clickbtn.place(x=320, y=100)*

*# Запуск цикла обработки событий*

*mainloop()*

# **Приложение 2. Код модуля базы данных settings.py.**

*from tkinter import \**

*from tkinter import messagebox*

*from PIL import ImageTk, Image*

*import sqlite3*

*# Класс для вывода принта в консоль*

*class fix:*

*def \_\_init\_\_():*

*None*

*def print():*

*print('Загрузка прошла успешно')*

*# Создание базы данных и курсора.*

*conn = sqlite3.connect('save\SaveFile.db')*

*curs = conn.cursor()*

*# Проверка удалена ли таблица в файле 'SaveFile.db'.*

*try:*

*save = [('1','money', 0),*

*('2','upgrade ', 1),*

*('3','priceRAM', 50),*

*('4','moneyRAM', 5),*

*('5','KolVoRAM', 0),*

*('6','priceCPU', 5000),*

*('7','moneyCPU', 20),*

*('8','KolVoCPU', 0),*

*('9','priceCARD', 25000),*

*('10','moneyCARD', 60),*

*('11','KolVoCARD', 0),*

*('12', 'SecretCodeActive', 0)]*

*curs.execute("""CREATE TABLE Save(id TEXT,*

*Name TEXT,*

*Save INTENGER)""")*

*curs.executemany('INSERT INTO Save VALUES (?, ?, ?)' , save)*

*conn.commit()*

*except sqlite3.OperationalError:*

*None*

*# Загрузка переменных из базы данных в игру.*

*\*save, = curs.execute('SELECT \* FROM Save')*

*# Переменные денег, цены, улучшения.*

*money = save[0][2]*

*upgrade = save[1][2]*

*priceRAM = save[2][2]*

*moneyRAM = save[3][2]*

*KolVoRAM = save[4][2]*

*priceCPU = save[5][2]*

*moneyCPU = save[6][2]*

*KolVoCPU = save[7][2]*

*priceCARD = save[8][2]*

*moneyCARD = save[9][2]*

*KolVoCARD = save[10][2]*

*SecretCodeActive = save[11][2]*

*# Запуск класса*

*fix.print()*

1. 1) ГОСТ 19.101-77 ЕСПД. Виды программ и программных документов [↑](#footnote-ref-1)
2. 2) ГОСТ 19.103-77 ЕСПД. Обозначение программ и программных документов [↑](#footnote-ref-2)
3. 3) ГОСТ 19.104-78\* ЕСПД. Основные надписи [↑](#footnote-ref-3)
4. 4) ГОСТ 19.105-78\* ЕСПД. Общие требования к программным документам [↑](#footnote-ref-4)
5. 5) ГОСТ 19.106-78\* ЕСПД. Общие требования к программным документам, выполненным печатным способом [↑](#footnote-ref-5)
6. 6) ГОСТ 19.505-79\* ЕСПД. Руководство оператора. Требования к содержанию и оформлению [↑](#footnote-ref-6)
7. 7) ГОСТ 19.604-78\* ЕСПД. Правила внесения изменений в программные документы, выполненные печатным способом [↑](#footnote-ref-7)