

Колледж космического машиностроения и технологий

КУРСОВОЙ ПРОЕКТ

По МДК.01.02 «Прикладное программирование»

Тема: «Разработка веб-приложения «текстовый редактор» на JavaScript»

Выполнил студент

Яковлев Прокопий Максимович

Группа П2-18

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

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

Проверил преподаватель

Гусятинер Леонид Борисович

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

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

Королёв 2021 г.

Оглавление

[**Введение** 3](#_Toc75563342)

[**1.** **Теоретическая часть** 5](#_Toc75563343)

[**1.1.** **Описание предметной области** 5](#_Toc75563344)

[**1.2.** **Описание существующих разработок** 6](#_Toc75563345)

[**1.2.1.** **Unreal Engine.** 6](#_Toc75563346)

[**1.2.2.** **Unity.** 8](#_Toc75563347)

[**2.** **Проектная часть** 10](#_Toc75563348)

[**2.1.** **Диаграмма прецедентов** 10](#_Toc75563349)

[**2.2.** **Выбор инструментов** 11](#_Toc75563350)

[**2.3.** **Диаграмма классов** 13](#_Toc75563351)

[**2.4.** **Описание главного модуля** 14](#_Toc75563352)

[**2.5.** **Описание спецификаций к модулям** 18](#_Toc75563353)

[**2.6.** **Описание модулей** 19](#_Toc75563354)

[**2.7.** **Описание тестовых наборов модулей** 20](#_Toc75563355)

[**2.8.** **Описание применения средств отладки** 22](#_Toc75563356)

[**3. Эксплуатационная часть** 23](#_Toc75563357)

[**3.1.** **Руководство оператора** 23](#_Toc75563358)

[**Заключение** 25](#_Toc75563359)

[**Список литературы и интернет-источников** 25](#_Toc75563360)

# **Введение**

Целью данного курсового проекта является написание одной из первых аркадных игр “Pong”

С начала 80-х годов началось широкое внедрение ЭВМ в деятельность человека. ЭВМ позволяет повсеместно автоматизировать человеческую деятельность и высвободить интеллектуальные ресурсы для творческого труда. В связи с широким внедрением ЭВМ возникает необходимость в своеобразной минимализации затрат времени на освоение этой техники максимально широким кругом пользователей.

Слово «компьютер» означает «вычислить», т.е. устройство для вычислений. Это связано с тем, что первые компьютеры создавались как устройства для вычислений.

Хотя компьютеры создавались для численных расчетов, скоро оказалось, что они могут обрабатывать и другие виды информации - ведь практически все они могут быть представлены в числовой форме. Сейчас с помощью компьютеров не только проводятся числовые расчеты, но и подготавливаются к печати книги, создаются рисунки, кинофильмы, музыка, осуществляется управление заводами и космическими кораблями и т.д. Компьютеры превратились в универсальные средства для обработки всех видов информации, используемых человеком.

Большинство (более 90%) современных компьютеров являются IBM PC - совместимыми персональными компьютерами. Полная программная совместимость этих компьютеров привела к появлению сотен тысяч рассчитанных для них программ, охватывающих практически все сферы человеческой деятельности.

Относительно высокие возможности IBM PC - совместимых компьютеров по переработке информации позволили использовать их (а не более мощные компьютеры) как для решения подавляющего большинства задач в бизнесе, так и для почти всех личных нужд пользователя таких, как развлечение и компъютерные игры.

Компьютерные игры так же существуют для того , чтобы превратить досуг или свободную минутку человека в развлекательное, а иногда даже поучительное время препровождение. Многие компъютерные игры имеют не только развлекательную функцию, но и так же способные развивать реакцию, скорость мышления. Данная игра пинг-понг с компъютером направлена на развитие быстроты реакции. Ведь с каждым новым уровнем скорость передвижения мяча по игровому полю становится значительно быстрее, что потребует от игрока молниеносную реакцию и способность думать на шаг вперед.

Написание игры происходило при использовании библиотеки “Pygame”.

Вышеназванная библиотека является актуальной для людей, которые только хотят начать создавать игры, но не готовы к сложной среде.

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.

Данная игра является неотъемлемой частью эпохи аркадных игр, которые положили фундамент развития игровой индустрии.

И мне тоже захотелось написать нечто подобное этому.

# **Теоретическая часть**

## **Описание предметной области**

“*Pong*”

9 ноября 1972 года, компания Atari, основанная 27-летним Ноланом Бушнеллом в июне того же года, объявила о выпуске первой компьютерной игры, снискавшей широкую популярность, – Pong. Первая версия этого будущего хита воплотилась в жизнь в громоздком теле аркадных игровых автоматов.

*Pong* является простым [спортивным симулятором](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%BE%D1%80%D1%82%D0%B8%D0%B2%D0%BD%D1%8B%D0%B9_%D1%81%D0%B8%D0%BC%D1%83%D0%BB%D1%8F%D1%82%D0%BE%D1%80) [настольного тенниса](https://ru.wikipedia.org/wiki/%D0%9D%D0%B0%D1%81%D1%82%D0%BE%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D1%82%D0%B5%D0%BD%D0%BD%D0%B8%D1%81). Небольшой квадратик, заменяющий пинг-понговый мячик, двигается по экрану по линейной [траектории](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B0%D0%B5%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D1%8F_%D0%BC%D0%B0%D1%82%D0%B5%D1%80%D0%B8%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B9_%D1%82%D0%BE%D1%87%D0%BA%D0%B8). Если он ударяется о периметр игрового поля, то его траектория изменяется в зависимости от [угла столкновения](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5_(%D1%84%D0%B8%D0%B7%D0%B8%D0%BA%D0%B0)). Если шарик отбивается ракеткой игрока, то его движение дополнительно зависит от скорости и направления движения ракетки[[1]](https://ru.wikipedia.org/wiki/Pong_(%D0%B8%D0%B3%D1%80%D0%B0)#cite_note-_a4fb47da0a2cce5a-1). Управление ракетками в *Pong* осуществляется с помощью [paddle](https://ru.wikipedia.org/wiki/Paddle)-контроллера, управляющего посредством поворотом ручки вокруг своей оси. Периметр игрового поля обозначен краями экрана, а мячик не может покинуть поле через верхний или нижний край. В верхней части поля отображаются очки игроков, у каждого на своей половине экрана.

Игровой процесс состоит в том, что игроки передвигают свои ракетки вертикально для защиты своих [ворот](https://ru.wikipedia.org/wiki/%D0%92%D0%BE%D1%80%D0%BE%D1%82%D0%B0_(%D1%81%D0%BF%D0%BE%D1%80%D1%82)). В начале каждого раунда мячик подаётся одному из игроков, и раунд продолжается до тех пор, пока один из игроков не заработает очко. Это происходит тогда, когда его противник не может отбить мячик. Со временем игры скорость движения мячика постепенно увеличивается, и так игра усложняется. Особенностью игры является то, что ракетки не могут дойти до самого верха экрана и отбить мячик, если он туда попадает — в этом случае противнику засчитывается очко. Вся игра в *Pong* длится не более трёх-четырёх минут[[2]](https://ru.wikipedia.org/wiki/Pong_(%D0%B8%D0%B3%D1%80%D0%B0)#cite_note-AlAlcornInterview-2).

Первоначальная версия *Pong* разработана для двух игроков, когда каждый управляет своей ракеткой с помощью своего контроллера. В более поздних версиях игры стал доступен однопользовательский режим, когда одна из ракеток управляется [компьютерным игроком](https://ru.wikipedia.org/wiki/%D0%98%D0%B3%D1%80%D0%BE%D0%B2%D0%BE%D0%B9_%D0%B8%D1%81%D0%BA%D1%83%D1%81%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D0%BB%D0%BB%D0%B5%D0%BA%D1%82).

Ниже представленная версия игры разработана для игры двух человек.

## **Описание существующих разработок**

В этом разделе рассмотрены некоторые уже имеющиеся аналоги Игры “Pong”.

Интерфейс таких программ для чаще всего простой, красивый, не обладает ярким контрастом. Обычно в подобных играх используются тёмные или же “нейтральные цвета”, к примеру тёмно-зелёный или же светло-синий.

Главное отличие игр серии “Pong” это стилистика:

1. Поле где проходит игра(футбольное поле, космос, простой фон и небо.
2. Варианты мяча, часто совпадает с тематикой поля игры.
3. Количество игроков, это очень редкое и “странное отличие”.

Но обычно игра представлена в двух вариантах:

1. Для одного игрока
2. Для двух игроков.

В первом случае игра происходит против оппонента-бота. Во втором случае против игрока, который был подран по сети либо твоего друга, игра происходит на одной клавиатуре.

### **Unreal Engine.**

Приложение для содание 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.

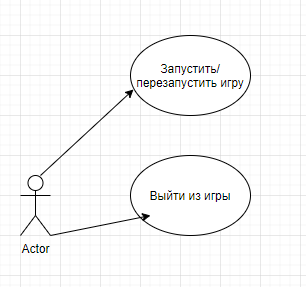


Рисунок 1. Диаграмма прецедентов работы пользователя

## **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.

Критерии выбора инструмента.

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

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

Оценка языков программирования.

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий\Язык 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 программы пользователем.

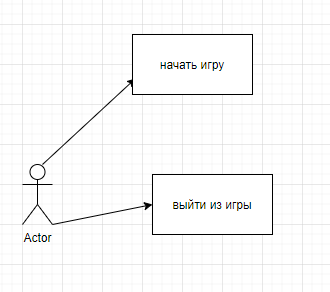


Рисунок 2. Сценарий использования

Пользователь после запуска программы может выполнить 2 действия: выйти из программы или начать игру.

При выборе выхода программа заканчивает свою работу.

## **Диаграмма классов**

В данном разделе представлен класс, использующийся в проекте.

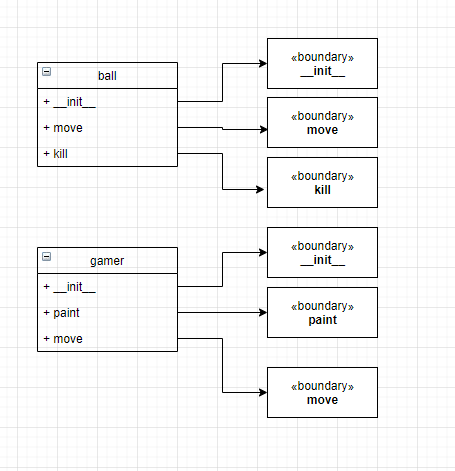


Рисунок 3. Диаграмма классов для проекта

Класс «Ball» является наследником метода move(), который позволяет передвигаться шарику.

## **Описание главного модуля**

В главный модуль входит класс ball который отвечает за функционал перемещения шарика и класс gamer, который отвечает за функционал игрока.

В классе ball реализованы следующие методы: \_\_init\_\_, move, kill

В классе gamer реализованы следующие методы: paint,\_\_init\_\_, move

**Листинг 1. Реализация движения шарика.**

class ball():

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

self.x = 0

self.y = 0

self.r = 10

self.vx = 0

self.vy = 0

self.goal = 0

self.id= canv.create\_oval(self.x-self.r,self.y-self.r,self.x+self.r,self.y+self.r, fill = 'white')

def move(self):

self.x += self.vx

self.y += self.vy

active\_wall = list(set(canv.find\_withtag('wall')) & set(canv.find\_overlapping(self.x - self.r\*0.7,self.y - self.r\*0.7,self.x + self.r\*0.7,self.y + self.r\*0.7)))

if active\_wall:

if 'x' in canv.gettags(active\_wall[0]):

self.vx = -self.vx

if 'y' in canv.gettags(active\_wall[0]):

self.vy = -self.vy

x1,y1,x2,y2 = canv.coords(active\_wall[0])

xc = (x1+x2)/2

w = abs(x1-x2)

self.vx += (self.x-xc)/w\*10

self.paint()

lines = canv.find\_overlapping(self.x - self.r\*0.7,self.y - self.r\*0.7,self.x + self.r\*0.7,self.y + self.r\*0.7)

if len(lines) > 1:

if "g1" in canv.gettags(lines[1]):

self.goal = 1

self.kill()

if "g2" in canv.gettags(lines[1]):

self.goal = 2

self.kill()

def kill(self):

global game

game = 0

self.x = 300

self.vx = 0

if self.goal == 2:

self.vy = -8

self.y = 100

if self.goal == 1:

self.y = 700

self.vy = 8

self.paint()

def paint(self):

canv.coords(self.id,self.x-self.r,self.y-self.r,self.x+self.r,self.y+self.r)

Данный метод реализует движение и отталкивание шарика.

**Листинг 2. Движение платформ.**

class gamer():

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

self.x = 0

self.y = 0

self.w = 60

self.v = 3

self.d = 4

self.mode = ''

self.score = 0

self.xy\_score = (0,0)

self.id= canv.create\_rectangle(self.x-self.w,self.y-self.d,self.x+self.w,self.y+self.d, fill = 'white', tags = ('wall','y'))

self.id\_score = canv.create\_text(0,0,text = '', font = 'Tahoma 24', fill = 'white')

def paint(self):

canv.coords(self.id, self.x-self.w,self.y-self.d,self.x+self.w,self.y+self.d)

canv.coords(self.id\_score, self.xy\_score[0],self.xy\_score[1])

canv.itemconfig(self.id\_score, text = self.score)

def move(self):

if self.mode == 'left' and self.x > self.w//2:

self.x -= self.v

elif self.mode == 'right' and self.x < (590-self.w//2):

self.x += self.v

self.paint()

b = ball()

b.x = 100

b.y = 100

b.vx = 4

b.vy = 4

canv.create\_line(10,10,10,790,width = 10, fill = 'white', tags = ('wall','x'))

canv.create\_line(590,10,590,790,width = 10, fill = 'white', tags = ('wall','x'))

canv.create\_line(10,790/2,590,790/2,width = 2, fill = 'white')

canv.create\_line(10,11,590,11,width = 2, fill = 'white', tag = 'g1')

canv.create\_line(10,789,590,789,width = 2, fill = 'white', tag = 'g2')

g1 = gamer()

g1.x = 300

g1.y = 20

g1.paint()

g1.xy\_score = (30,50)

g2 = gamer()

g2.x = 300

g2.y = 780

g2.paint()

g2.xy\_score = (30,450)

game = 1

def key\_press(event):

global game

if event.keycode == 37:

g2.mode = 'left'

elif event.keycode == 39:

g2.mode = 'right'

elif event.keycode == 65:

g1.mode = 'left'

elif event.keycode == 68:

g1.mode = 'right'

elif event.keycode == 32:

game = 1

def key\_release(event):

if event.keycode == 37 or event.keycode == 39:

g2.mode = ''

elif event.keycode == 65 or event.keycode == 68:

g1.mode = ''

root.bind('<Key>', key\_press)

root.bind('<KeyRelease>', key\_release)

while 1:

if game:

b.move()

g2.move()

g1.move()

if b.goal == 1:

g2.score += 1

b.goal = 0

elif b.goal == 2:

g1.score += 1

b.goal = 0

time.sleep(0.02)

canv.update()

Данный метод реализует управление платформами.

## **Описание спецификаций к модулям**

В данном разделе описан публичны член модулей курсового проекта.

В главном модуле (Курсовая) чсодержится публичный метод класса ball. Так как все эти методы были описаны в разделе 2.6, в данном разделе описание данных методов опущено. В данном модуле также есть объекты:

1. Move- отвечает за общее движение мяча.
2. Kill- отвечает за статус мяча(проигравший игрок).
3. Paint- отвечает за отрисовку мяча.

## **Описание модулей**

Кроме главного модуля программа не содержит более модулей.

Полный код модуля приведен в приложении 1. В этом разделе приведена блок-схема метода, который содержаться в классе \_\_init\_\_ из модуля ball.

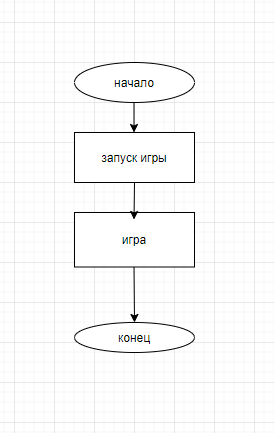


Рисунок 4. Блок-схема метода work

## **Описание тестовых наборов модулей**

В этом разделе будут продемонстрированы результаты тестирования «черного ящика».

Тест 1. Проигрыш (шарик покинул границы игрового поля), победа игрока “снизу”

Действия: Забить “гол” игроку “сверху”

Ожидаемый результат: +1 к очкам игрока “снизу”

Результат теста: пройден

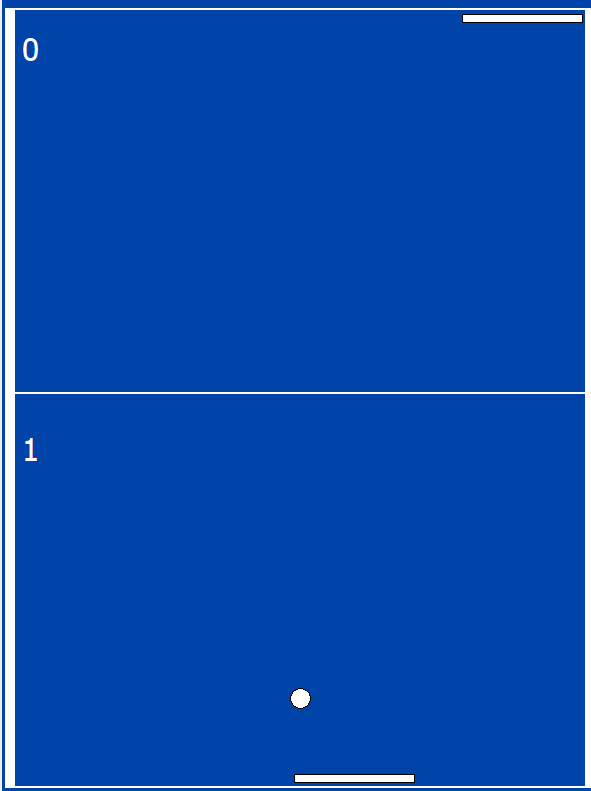


Рисунок 5. Тест 1

Тест 2. Символы юникода.

Проигрыш (шарик покинул границы игрового поля), победа игрока “сверху”

Действия: Забить “гол” игроку “снизу”

Ожидаемый результат: +1 к очкам игрока “сверху”

Результат теста: пройден

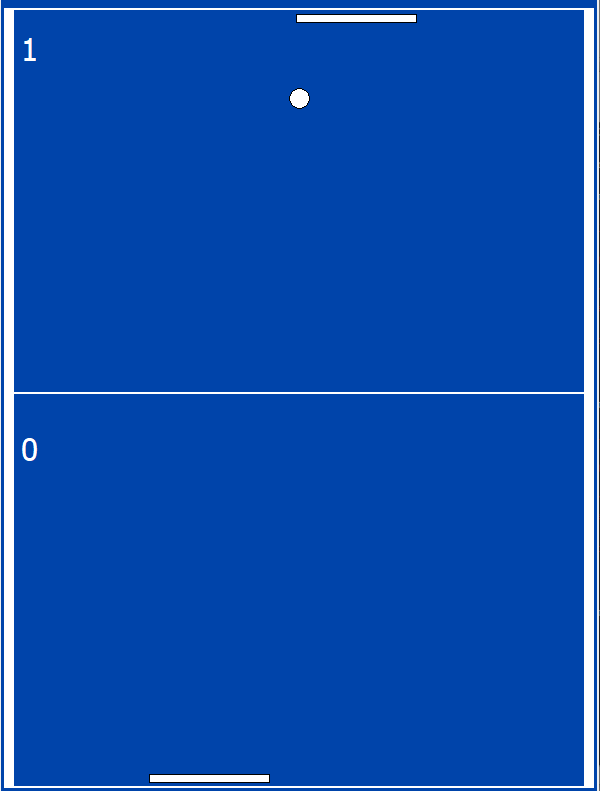


Рисунок 6. Тест 2

## **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:

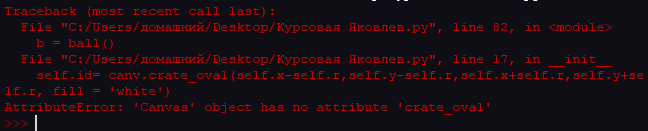


Рисунок 10. До применения средств отладки

После получения данного сообщения были просмотрены 17 и 75 строки модуля ball и была обнаружена ошибка, которая впоследствии была устранена, а после попытки запуска игры:

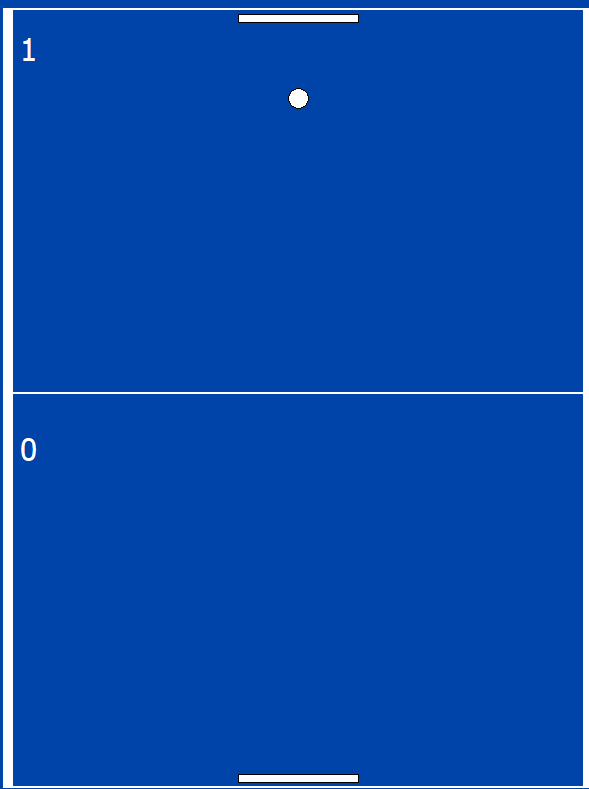


Рисунок 11. После применения средств отладки

## **3. Эксплуатационная часть**

## **Руководство оператора**

**АННОТАЦИЯ**

В данном программном документе приведено руководство оператора по применению и эксплуатации программы «Pong», предназначенной для досуга.

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**

Проведение досуга.

* 1. **Эксплуатационное назначение программы**

Специальное программное обеспечение «Pong» может эксплуатироваться любым пользователем, который хочет приятно провести время.

* 1. **Состав функций**
     1. **Функция движение мяча.**

Эта функция отвечает за движение мяча.

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 ничего не нужно создавать, просто запустить из любого удобного места игру и начать играть.

* 1. **Завершение работы программы**

Чтобы завершить работу программы нажмите на “крестик”.

# **Заключение**

В результате выполнения курсового проекта была написана программа «Pong» для времяпрепровождения.

В ходе работы были проанализированы предметная область, существующие разработки, посвященные данному направлению, получены практические навыки по созданию игр используя 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. Создание главного меню с реализацией функций игры по сети.
2. Несколькоуровневый режим одиночной игры.

# **Список литературы и интернет-источников**

1. Гагарина, Л. Г. Технология разработки программного обеспечения: учеб. пособие / Л.Г. Гагарина, Е.В. Кокорева, Б.Д. Сидорова-Виснадул; под ред. Л.Г. Гагариной. — Москва: ИД «ФОРУМ»: ИНФРА-М, 2019. — 400 с. — (Высшее образование: Бакалавриат). - ISBN 978-5-16-104071-3. - Текст: электронный. - URL: <https://znanium.com/catalog/product/1011120>
2. Скрапинг веб-сайтов с помощью Python Райан Митчелл
3. Create Simple GUI Applications, with Python Qt5 by Martin Fitzpatrick
4. В. Дронов, Н. Прохорёнок - Python 3 и PyQt 5. Разработка приложений. 2 изд.

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

**Листинг 4. Проект**

from tkinter import \*

from random import randrange as rnd

import time

root = Tk()

root.geometry('600x800')

canv = Canvas(root, bg = '#0044aa')

canv.pack(fill=BOTH,expand = 1)

class ball():

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

self.x = 0

self.y = 0

self.r = 10

self.vx = 0

self.vy = 0

self.goal = 0

self.id= canv.create\_oval(self.x-self.r,self.y-self.r,self.x+self.r,self.y+self.r, fill = 'white')

def move(self):

self.x += self.vx

self.y += self.vy

active\_wall = list(set(canv.find\_withtag('wall')) & set(canv.find\_overlapping(self.x - self.r\*0.7,self.y - self.r\*0.7,self.x + self.r\*0.7,self.y + self.r\*0.7)))

if active\_wall:

if 'x' in canv.gettags(active\_wall[0]):

self.vx = -self.vx

if 'y' in canv.gettags(active\_wall[0]):

self.vy = -self.vy

x1,y1,x2,y2 = canv.coords(active\_wall[0])

xc = (x1+x2)/2

w = abs(x1-x2)

self.vx += (self.x-xc)/w\*10

self.paint()

lines = canv.find\_overlapping(self.x - self.r\*0.7,self.y - self.r\*0.7,self.x + self.r\*0.7,self.y + self.r\*0.7)

if len(lines) > 1:

if "g1" in canv.gettags(lines[1]):

self.goal = 1

self.kill()

if "g2" in canv.gettags(lines[1]):

self.goal = 2

self.kill()

def kill(self):

global game

game = 0

self.x = 300

self.vx = 0

if self.goal == 2:

self.vy = -8

self.y = 100

if self.goal == 1:

self.y = 700

self.vy = 8

self.paint()

def paint(self):

canv.coords(self.id, self.x-self.r,self.y-self.r,self.x+self.r,self.y+self.r)

class gamer():

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

self.x = 0

self.y = 0

self.w = 60

self.v = 3

self.d = 4

self.mode = ''

self.score = 0

self.xy\_score = (0,0)

self.id= canv.create\_rectangle(self.x-self.w,self.y-self.d,self.x+self.w,self.y+self.d, fill = 'white', tags = ('wall','y'))

self.id\_score = canv.create\_text(0,0,text = '', font = 'Tahoma 24', fill = 'white')

def paint(self):

canv.coords(self.id, self.x-self.w,self.y-self.d,self.x+self.w,self.y+self.d)

canv.coords(self.id\_score, self.xy\_score[0],self.xy\_score[1])

canv.itemconfig(self.id\_score, text = self.score)

def move(self):

if self.mode == 'left' and self.x > self.w//2:

self.x -= self.v

elif self.mode == 'right' and self.x < (590-self.w//2):

self.x += self.v

self.paint()

b = ball()

b.x = 100

b.y = 100

b.vx = 4

b.vy = 4

canv.create\_line(10,10,10,790,width = 10, fill = 'white', tags = ('wall','x'))

canv.create\_line(590,10,590,790,width = 10, fill = 'white', tags = ('wall','x'))

canv.create\_line(10,790/2,590,790/2,width = 2, fill = 'white')

canv.create\_line(10,11,590,11,width = 2, fill = 'white', tag = 'g1')

canv.create\_line(10,789,590,789,width = 2, fill = 'white', tag = 'g2')

g1 = gamer()

g1.x = 300

g1.y = 20

g1.paint()

g1.xy\_score = (30,50)

g2 = gamer()

g2.x = 300

g2.y = 780

g2.paint()

g2.xy\_score = (30,450)

game = 1

def key\_press(event):

global game

if event.keycode == 37:

g2.mode = 'left'

elif event.keycode == 39:

g2.mode = 'right'

elif event.keycode == 65:

g1.mode = 'left'

elif event.keycode == 68:

g1.mode = 'right'

elif event.keycode == 32:

game = 1

def key\_release(event):

if event.keycode == 37 or event.keycode == 39:

g2.mode = ''

elif event.keycode == 65 or event.keycode == 68:

g1.mode = ''

root.bind('<Key>', key\_press)

root.bind('<KeyRelease>', key\_release)

while 1:

if game:

b.move()

g2.move()

g1.move()

if b.goal == 1:

g2.score += 1

b.goal = 0

elif b.goal == 2:

g1.score += 1

b.goal = 0

time.sleep(0.02)

canv.update()

mainloop()

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)