**Министерство образования и науки Астраханской области**

Государственное бюджетное общеобразовательное учреждение

Астраханской области

**«Астраханский технический лицей»**

**(ГБОУ АО «АТЛ»)**

**Создание игры “Понг”**

**Творческий проект**

Выполнила:

ученица 10В класса

Гвоздарева Екатерина Сергеевна

Руководитель проекта:

учитель информатики

Мамцев Виталий Сергеевич.

**г. Астрахань**

**2023**

Оглавление

[Введение 3](#_Toc133859362)

[Язык Python 4](#_Toc133859363)

[Библиотека PyGame 6](#_Toc133859364)

[Суть игры 6](#_Toc133859365)

[Этапы создания игры 7](#_Toc133859366)

[Заключение 15](#_Toc133859367)

[Список литературы 16](#_Toc133859368)

[Приложения 17](#_Toc133859369)

# Введение

В век технологий у нас существует множество развлечений и способов скоротать время, поэтому в настоящее время создание компьютерных игр является актуальным вопросом. Сейчас это сложный процесс, задействующий множество разработчиков и даже целые команды. Для создания игры нужно иметь большое количество профессиональных знаний, которые можно получить при создании менее сложных игр.

Аркадные игры – один из самых популярных жанров, ведь они характеризуются коротким временем, но интенсивным игровым процессом, что привлекает многих игроков.

Понг является ярким представителем этого вида игр. Главной задачей этой игры является не только скоротать время, но и развить реакцию, навык быстрого мышления.

Для реализации кода программы в качестве языка программирования был выбран язык Python. Данный язык программирования в совокупности с библиотекой PyGame подходит для небольших, простых игр, а также синтаксис языка простой и понятный, что сильно упрощает задачу.

**Цель:**

Разработка компьютерной игры «Понг» на основе использования библиотеки PyGame из языка программирования Python.

**Задачи:**

1. Изучить литературу по заданной теме;
2. Написать код программы;
3. Создать документацию проектной работы;
4. Оформить проектную работу в виде презентации;
5. Презентовать готовую работу.

# Язык Python

Python – это высокоуровневый объектно-ориентированный язык программирования с динамической строгой типизацией и автоматическим управлением памятью. Разработчики используют Python, потому что он эффективен, прост в изучении и чтении и работает на разных платформах.

История

Разработка его реализации началась в 1989 году сотрудником голландского института CWI Гвидо ван Россумом. Для распределённой операционной системы Amoeba требовался расширяемый скриптовый язык, и Гвидо начал разрабатывать Python на досуге, позаимствовав некоторые наработки для языка ABC. В феврале 1991 года Гвидо опубликовал исходный текст в группе новостей alt.sources. С самого начала Python проектировался как объектно-ориентированный язык.

Гвидо ван Россум назвал язык в честь популярного британского комедийного телешоу 1970-х «Летающий цирк Монти Пайтона», поскольку автор был поклонником этого телешоу, как и многие другие разработчики того времени, а в самом шоу прослеживалась некая параллель с миром компьютерной техники.

За 30 с лишним лет Python превратился из домашнего прототипа в мировой язык программирования. А самое главное — с 2019 года он полностью независим от основателя и развивается силами сообщества, что означает стабильность на протяжении долгого будущего.

Преимущества

Специалисты выделяют массу преимуществ Python — остановимся на ключевых из них.

* **Простота синтаксиса.**

Разработчики могут легко читать и понимать программы на Python, поскольку язык имеет базовый синтаксис, похожий на синтаксис английского.

* **Расширяемость и гибкость.**

Python можно легко расширить для взаимодействия с другими программными системами или встроить в программы в качестве компонента. Он очень и очень гибкий. Это даёт больше возможностей для использования языка в разных сферах.

* **Интерпретируемость и кроссплатформенность.**

Python можно переносить на различные операционные системы: Windows, macOS, Linux и Unix.

* **Стандартизированность.**

У Python есть единый стандарт для написания кода — Python Enhancement Proposal или PEP, благодаря чему язык остаётся читабельным даже при переходе от одного программиста к другому.

* **Open Source.**

У интерпретатора Python открытый код, то есть любой, кто заинтересован в развитии языка, может поучаствовать в его разработке и улучшении.

* **Широта применения.**

Наиболее широко Python используется в web-разработке, работе с данными, автоматизации бизнес-процессов и GameDev.

* **Стандартная библиотека.**

Python имеет большую стандартную библиотеку, содержащую многократно используемые коды практически для любой задачи. В результате разработчикам не требуется писать код с нуля.

* **Популярность.**

В интернете доступно множество полезных ресурсов для изучения Python. Например, вы можете легко найти видеоролики, учебные пособия, документацию и руководства для разработчиков.

Недостатки

* **Низкая производительность.**

Хотя и по количеству писать кода меньше, но Python требует высоких вычислительных мощностей серверов и компьютеров, что делает его не таким быстрым, как хотелось бы. Он отстаёт от других языков по части производительности. В условиях сильного развития мощностей это не так заметно, как раньше, но всё равно даёт о себе знать.

* **Динамическая типизация.**

Python относится к языкам с динамической типизацией, что потребляет много ресурсов и имеет низкую скорость выполнения программы.

Использование

Python используется во многих областях, но из основных сфер использования можно выделить:

* Веб-разработка
* Data Science и Machine Learning
* Написание скриптов

Помимо этого, Python применяется для написания и разработки программных и мобильных приложений, во встроенных системах, системном администрировании, а также в GameDev.

# Библиотека PyGame

PyGame - набор модулей (библиотек) языка программирования Python, предназначенный для написания компьютерных игр и мультимедиа-приложений. Она помогает разработчику не только описать геймплей, но и работать с клавиатурой, мышью, акселерометром, звуком и видео.

Она дает необходимый минимальный инструментарий для:

* рисования графических объектов
* отслеживания различных событий (клавиатуры, мыши, таймера и т.п)
* отслеживания и изменения состояний объектов (создание анимации, контроль столкновений)
* быстрой отрисовки изменений на экране устройства пользователя
* работы со звуковыми эффектами

# Суть игры

Понг — это сама простота: движущаяся точка, две вертикальные линии и квадратные цифры, отображающие текущий счет. Игра имитирует настольный теннис, когда два человека соревнуются друг с другом, пытаясь отправить мяч мимо ракетки соперника. Игроки могут перемещать ракетку по вертикали, чтобы отбить мяч. Если ты промахиваешься, соперник зарабатывает очко. Первый игрок, набравший 11 очков, побеждает.

История разработки прототипа

Фактически Pong был разработан молодым инженером по имени Аллан Алкорн как учебный проект, тогда он только-только окунулся в мир видеоигр и не знал, с чего начать. Нолан Бушнелл и Самуэль «Тед» Дабни как раз основали компанию Atari и наняли Алкорна третьим штатным сотрудником. На тот момент видеоигры можно было встретить разве что в научно-исследовательских лабораториях и университетах. Под прикрытием контракта для General Electric Бушнелл попросил Алкорна разработать игру, похожую на пинг-понг, но в которую можно было бы играть на телевизоре.

Бушнелл разбирался в экономике пинбольных автоматов и других аркад, поэтому принял решение добавить к аркадной версии Pong монетоприемник. Если верить слухам, люди выстраивались в очередь, чтобы закинуть в автомат свой четвертак и поиграть. В конце концов, монетоприемник переполнился и автомат перестал работать.

Atari продала около 35 000 аркадных машин барам и ресторанам по всей стране, но настоящий перелом случился в 1975 году, когда Atari представила Home Pong, в который можно было играть дома на телевизоре. В рекламе говорилось, что Home Pong работает с любым телевизором, но на борту у консоли только одна игра. Сотни тысяч комплектов Pong удалось продать через сеть универмагов Sears. Pong, кроме того, входил в базовый комплект Atari 2600, видеоигровой системы, представленной в 1977 году и умевшей воспроизводить игры со сменных носителей, картриджей.

В 2015 году Strong National Museum of Play включил Pong в свой Всемирный зал славы видеоигр, приняв во внимание его популярность и влияние на зарождение индустрии видеоигр в целом.

Pong не была ни первой видеоигрой, ни первой видеоигрой в настольный теннис, ни первой домашней видеоигрой, и всё же именно она познакомила миллионы детей и их родителей с идеей видеоигр.

# Этапы создания игры

В качестве основы был взят код программы для игры “Арканоид”. Из программы игры «Арканоид» взяты механика движения мяча и платформы, их прорисовка и создание окна:

Для создания кода программы нам понадобится импорт дополнительных библиотек:

import pygame  
import random

Запуск библиотеки PyGame для последующего использования её команд:

pygame.init()

С помощью таблицы RGB выбираются нужные значения и объявляются цвета:

black = (0, 0, 0)  
white = (255, 255, 255)

Создание игрового окна с помощью команды display.set\_mode() для последующей реализации игры:

screen\_width = 800  
screen\_height = 500  
screen = pygame.display.set\_mode((screen\_width, screen\_height))

Для создания границ прямоугольной платформы и мяча была использована команда Rect():

block\_width = 15  
block\_height = 100

block\_speed = 10  
block\_rect = pygame.Rect(screen\_width / 2 - block\_width / 2, screen\_height - block\_height \* 2, block\_width, block\_height)

ball\_radius = 15  
ball\_speed = 7

ball\_first\_collide = False  
ball\_x\_speed = 0  
ball\_y\_speed = ball\_speed  
ball\_rect = pygame.Rect(screen\_width / 2 - ball\_radius, screen\_height / 2 - ball\_radius, ball\_radius \* 2, ball\_radius \* 2)

Создание переменной clock:

clock = pygame.time.Clock()

Инициализация переменных для механики цикла:

game\_over = False  
game = True

Запуск цикла игры:

while game:

Проверка нажатия клавиш, выход из игры:

for event in pygame.event.get():

if event.type == pygame.QUIT:   
 game = False  
 continue

elif event.type == pygame.KEYDOWN:  
 if event.key == pygame.K\_ESCAPE:  
 game = False  
 continue

Заливка экрана цветом с помощью команды fill():

screen.fill(black)

Проверка на завершение игровой партии:

if not game\_over:

Проверка нажатия клавиш движения платформы (для движения влево используется левая клавиша-стрелка управления курсором, а для движения вправо - правая):

keys = pygame.key.get\_pressed()

if keys[pygame.K\_LEFT]:  
 block\_rect.x -= block\_speed  
elif keys[pygame.K\_RIGHT]:  
 block\_rect.x += block\_speed

Вырисовывание платформы с помощью команды draw.rect():

pygame.draw.rect(screen, white, block\_rect)

Так как мяч будет двигаться по экрану, то нам необходимо выполнить проверку на столкновение мяча, которое регулируется командами движения мяча:

if block\_rect.colliderect(ball\_rect):  
 if not ball\_first\_collide:  
 if random.choice(['right', 'left']) == 'right':  
 ball\_x\_speed = ball\_speed  
 else:  
 ball\_x\_speed = -ball\_speed  
 ball\_first\_collide = True  
 ball\_y\_speed = -ball\_speed

Движение шарика при помощи добавления значения скорости к изначальному положению:

ball\_rect.x += ball\_x\_speed  
ball\_rect.y += ball\_y\_speed

Отскок шарика от границ экрана:

if ball\_rect.bottom >= screen\_height:  
 game\_over = True  
 ball\_y\_speed = -ball\_speed  
elif ball\_rect.top <= 0:  
 ball\_y\_speed = ball\_speed  
elif ball\_rect.left <= 0:  
 ball\_x\_speed = ball\_speed  
elif ball\_rect.right >= screen\_width:  
 ball\_x\_speed = -ball\_speed

Вырисовывание шарика с помощью команды draw.circle():

pygame.draw.circle(screen, yellow, ball\_rect.center, ball\_radius)

По причине того, что мяч двигается с очень большой скоростью, для комфортной игры нам надо использовать замедление скорости смены кадров с помощью команды tick():

clock.tick(60)

Для отображения игрового процесса добавляем обновление экрана:

pygame.display.flip()

Выход из игры с помощью команды quit(), завершение работы кода:

pygame.quit()

После этого была развернута платформа, добавлена вторая, счетчик очков, наименование игроков, изменены цвета на более приятные, созданы старт, рестарт и меню:

Для этого подключаем библиотеки, описанные ранее.

Далее идёт объявление цветов:

blue = (100, 112, 219)  
green = (0, 200, 60)  
yellow = (225, 225, 0)  
white = (255, 255, 255)

Добавление музыки для комфортной игры:

pygame.mixer.music.load("back.mp3")  
pygame.mixer.music.play(-1)  
bumb\_sound = pygame.mixer.Sound("bum.mp3")

Создание игрового окна с помощью команды display.set\_mode() и названия с помощью команды display.set\_caption():

title = 'Понг'  
screen\_width = 800  
screen\_height = 500  
screen = pygame.display.set\_mode((screen\_width, screen\_height))   
pygame.display.set\_caption(title)

Для создания границ ракеток, сетки и мяча использована команду Rect():

block\_speed\_1 = 10  
block\_width\_1 = 15  
block\_height\_1 = 100  
block\_rect\_1 = pygame.Rect(778, 200, block\_width\_1, block\_height\_1)  
  
block\_speed\_2 = 10  
block\_width\_2 = 15  
block\_height\_2 = 100  
block\_rect\_2 = pygame.Rect(7, 200, block\_width\_2, block\_height\_2)

line = pygame.Rect(397, 0, 3, 500)

ball\_radius = 15  
ball\_speed = 7  
ball\_x\_speed = random.choice([ball\_speed, -ball\_speed])  
ball\_y\_speed = 0  
ball\_rect = pygame.Rect(screen\_width / 2 - ball\_radius, screen\_height / 2 - ball\_radius, ball\_radius \* 2, ball\_radius \* 2)

Объявление имён игроков, количества очков:

player\_1 = 'Player 1'  
player\_2 = 'Player 2'  
points = 11  
score\_1 = 0  
score\_2 = 0

Создание шрифтов, при помощи команд font.match\_font и font.Font:

arial\_font = pygame.font.match\_font('arial')  
arial\_font\_30 = pygame.font.Font(arial\_font, 30)  
arial\_font\_36 = pygame.font.Font(arial\_font, 36)

Создание переменной clock:

clock = pygame.time.Clock()

Объявление булевых переменных для последующего использования в цикле:

game\_over = False  
game = True  
menu\_start = True

Функция для перезапуска игры (возвращение мяча и ракеток на свои позиции, обнуление переменных):

def restart():  
 global game\_over  
 global ball\_x\_speed  
 global ball\_y\_speed  
 global score\_1  
 global score\_2  
 global menu\_start  
  
 game\_over = False  
 menu\_start = False  
  
 block\_rect\_1.centerx = 777  
 block\_rect\_1.bottom = 300  
 block\_rect\_2.centerx = 23  
 block\_rect\_2.bottom = 300  
  
 ball\_rect.center = [screen\_width / 2, screen\_height / 2]  
 ball\_x\_speed = random.choice([ball\_speed, -ball\_speed])  
 ball\_y\_speed = 0  
  
 score\_1 = 0  
 score\_2 = 0

Запуск игрового цикла:

while game:

Заливка экрана цветом с помощью команды fill():

screen.fill(blue)

Проверка нажатия клавиш, выход из игры:

for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 game = False  
 continue  
 elif event.type == pygame.KEYDOWN:  
 if event.key == pygame.K\_ESCAPE:  
 game = False  
 continue

Проверка нажатия клавиши Space и вызов функции для запуска/перезапуска:

elif event.key == pygame.K\_SPACE:  
 restart()

Проверка нажатия клавиши M и запуск меню:

elif event.key == pygame.K\_m:  
 menu\_start = True

Если игра не окончена и пользователь не в меню:

if not game\_over and not menu\_start:

Проверка кнопок движения платформ и их движение (ракетка первого игрока двигается клавишами W и S, а второго – стрелками вверх и вниз):

keys = pygame.key.get\_pressed()  
if keys[pygame.K\_DOWN]:  
 block\_rect\_1.y += block\_speed\_1  
elif keys[pygame.K\_UP]:  
 block\_rect\_1.y -= block\_speed\_1  
if keys[pygame.K\_s]:  
 block\_rect\_2.y += block\_speed\_2  
elif keys[pygame.K\_w]:  
 block\_rect\_2.y -= block\_speed\_2

Вырисовывание ракеток и сетки с помощью команды draw.rect():

pygame.draw.rect(screen, white, line)   
pygame.draw.rect(screen, green, block\_rect\_1)   
pygame.draw.rect(screen, green, block\_rect\_2)

Проверка столкновение ракетки с мячом, реализация скачка и подключение звука удара:

if block\_rect\_1.colliderect(ball\_rect):  
 bumb\_sound.play()  
 if random.choice(['up', 'down']) == 'up':  
 ball\_y\_speed = ball\_speed  
 else:  
 ball\_y\_speed = -ball\_speed  
 ball\_x\_speed = -ball\_speed  
elif block\_rect\_2.colliderect(ball\_rect):  
 bumb\_sound.play()  
 if random.choice(['up', 'down']) == 'up':  
 ball\_y\_speed = ball\_speed  
 else:  
 ball\_y\_speed = -ball\_speed  
 ball\_x\_speed = ball\_speed

Счет очков игроков:

if ball\_rect.left == 0:  
 score\_2 += 1  
elif ball\_rect.right == screen\_width:  
 score\_1 += 1

Ограничение движения ракеток размерами экрана с помощью команды clamp\_ip:

block\_rect\_1.clamp\_ip(screen.get\_rect())  
block\_rect\_2.clamp\_ip(screen.get\_rect())

Движение шарика при помощи добавления значения скорости к изначальному положению:

ball\_rect.x += ball\_x\_speed  
ball\_rect.y += ball\_y\_speed

Отскок мяча от границ экрана:

if ball\_rect.bottom >= screen\_height:  
 ball\_y\_speed = -ball\_speed  
elif ball\_rect.top <= 0:  
 ball\_y\_speed = ball\_speed  
elif ball\_rect.left <= 0:  
 ball\_x\_speed = ball\_speed  
elif ball\_rect.right >= screen\_width:  
 ball\_x\_speed = -ball\_speed

Проверка хватает ли очков игрокам для победы:

if score\_1 == points:  
 winner = player\_1  
 game\_over = True  
elif score\_2 == points:  
 winner = player\_2  
 game\_over = True

Если не в меню:

if not menu\_start:

Вырисовывание мяча для фона:

pygame.draw.circle(screen, yellow, ball\_rect.center, ball\_radius)

Создание текста:

score\_print\_1 = arial\_font\_30.render('Score: ' + str(score\_1), True, white)  
score\_print\_2 = arial\_font\_30.render('Score: ' + str(score\_2), True, white)  
player1\_text = arial\_font\_30.render(player\_1, True, white)  
player2\_text = arial\_font\_30.render(player\_2, True, white)

Если игра не окончена, то печатаем названия игроков и их очки:

if not game\_over:  
 screen.blit(score\_print\_1, [50, 50])  
 screen.blit(score\_print\_2, [650, 50])  
 screen.blit(player1\_text, [50, 15])  
 screen.blit(player2\_text, [650, 15])

Если игра окончена, то выводятся названия игроков, их очки, победитель и название кнопки для начала новой игры:

else:  
 retry\_surface = arial\_font\_36.render('press SPACE to restart', True, white)  
 win\_text = arial\_font\_30.render('{} wins'.format(winner), True, white)  
 screen.blit(score\_print\_1, [50, 50])  
 screen.blit(score\_print\_2, [650, 50])  
 screen.blit(player1\_text, [50, 15])  
 screen.blit(player2\_text, [650, 15])  
 screen.blit(retry\_surface, [250, 250])  
 screen.blit(win\_text, [335, 150])

Если пользователь находится в меню, то печатаем названия игроков, нужное количество очков, названия кнопок для выхода и старта:

else:   
 player1\_text = arial\_font\_30.render(player\_1, True, white)  
 player2\_text = arial\_font\_30.render(player\_2, True, white)  
 retry\_surface = arial\_font\_36.render('press SPACE to start', True, white)  
 esc\_text = arial\_font\_36.render('press Esc to exit', True, white)  
 points\_text = arial\_font\_36.render('Required points: {}'.format(points), True, white)  
 screen.blit(player1\_text, [50, 15])  
 screen.blit(player2\_text, [650, 15])  
 screen.blit(retry\_surface, [260, 200])  
 screen.blit(esc\_text, [290, 250])  
 screen.blit(points\_text, [270, 15])

Изменяем скорость смены кадров с помощью команды tick():

clock.tick(60)

Для отображения игрового процесса добавляем обновление экрана:

pygame.display.flip()

Выход из игры при помощи команды quit(), завершение работы кода:

pygame.quit()

# Заключение

В ходе работы над данным проектом, изучив все необходимые элементы библиотеки PyGame языка Python, всю нужную литературу и пройдя все этапы разработки, была разработана игра Понг.

Игра создана для двух игроков, управляющих своими ракетками в разных частях игрового поля, отбивая прыгающий мячик. Игра идет до тех пор, пока один из игроков не получит 11 очков, как и в стандартном Pong.

При выполнении данной работы:

* получен практический опыт создания несложных игр
* приобретены и закреплены знания об основных функциях библиотеки PyGame и языка Python

Результат работы имеет практическое применение: благодаря данной игре человек может скоротать время, развить реакцию и навык быстрого мышления.

# Список литературы

* Proproprogs [Электронный ресурс] – сайт для изучения языков программирования [https://proproprogs.ru](https://proproprogs.ru/modules/dobavlyaem-zvuk-v-igrovoy-process-moduli-mixer-i-music)
* Pythonru [Электронный ресурс]– учебная платформа для изучения Python [https://pythonru.com/uroki](https://pythonru.com/uroki/biblioteka-pygame-chast-1-vvedenie)
* Standalone Coder [Электронный ресурс]– ютуб канал <https://www.youtube.com/watch?v=tSPF1sjwhho>
* Руководство по PyGame от waksoft.susu [Электронный ресурс] <https://waksoft.susu.ru/2019/04/24/pygame-shpargalka-dlja-ispolzovanija/>

# Приложения

Первоначальный код:

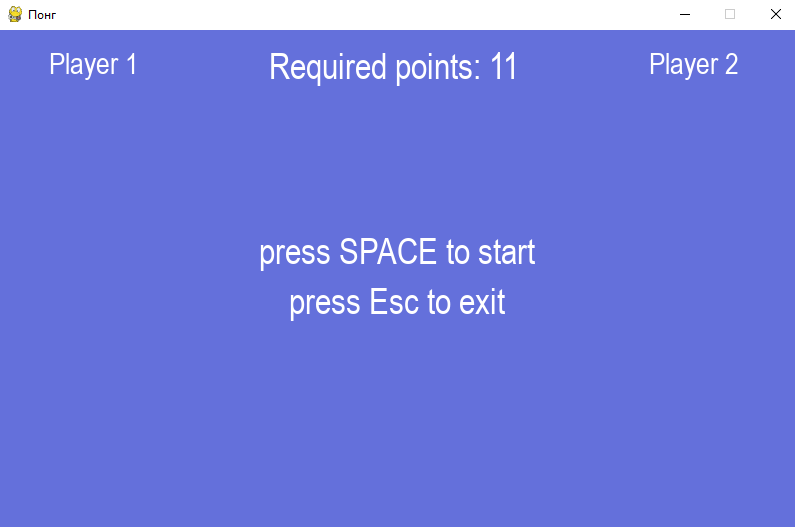
import pygame  
import random  
  
pygame.init()  
black = (0, 0, 0)  
white = (255, 255, 255)  
  
screen\_width = 800  
screen\_height = 500  
screen = pygame.display.set\_mode((screen\_width, screen\_height))   
  
block\_width = 100  
block\_height = 15  
block\_speed = 10  
block\_rect = pygame.rect.Rect(screen\_width / 2 - block\_width / 2, screen\_height - block\_height \* 2, block\_width,  
 block\_height)  
  
ball\_radius = 15  
ball\_speed = 7  
ball\_first\_collide = False  
ball\_x\_speed = 0  
ball\_y\_speed = ball\_speed  
ball\_rect = pygame.rect.Rect(screen\_width / 2 - ball\_radius, screen\_height / 2 - ball\_radius, ball\_radius \* 2,  
 ball\_radius \* 2)  
  
clock = pygame.time.Clock()  
  
game\_over = False  
game = True  
while game:  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:   
 game = False  
 continue  
 elif event.type == pygame.KEYDOWN:  
 if event.key == pygame.K\_ESCAPE:  
 game = False  
 continue  
  
 screen.fill(black)  
 if not game\_over:  
 keys = pygame.key.get\_pressed()  
 if keys[pygame.K\_LEFT]:  
 block\_rect.x -= block\_speed  
 elif keys[pygame.K\_RIGHT]:  
 block\_rect.x += block\_speed  
 pygame.draw.rect(screen, white, block\_rect)   
 if block\_rect.colliderect(ball\_rect):   
 if not ball\_first\_collide:  
 if random.choice(['right', 'left']) == 'right':  
 ball\_x\_speed = ball\_speed  
 else:  
 ball\_x\_speed = -ball\_speed  
 ball\_first\_collide = True  
 ball\_y\_speed = -ball\_speed  
  
 ball\_rect.x += ball\_x\_speed  
 ball\_rect.y += ball\_y\_speed  
  
 if ball\_rect.bottom >= screen\_height:  
 game\_over = True  
 ball\_y\_speed = -ball\_speed  
 elif ball\_rect.top <= 0:  
 ball\_y\_speed = ball\_speed  
 elif ball\_rect.left <= 0:  
 ball\_x\_speed = ball\_speed  
 elif ball\_rect.right >= screen\_width:  
 ball\_x\_speed = -ball\_speed  
  
 pygame.draw.circle(screen, white, ball\_rect.center, ball\_radius)   
  
 clock.tick(60)   
 pygame.display.flip()   
pygame.quit()

Конечный код:

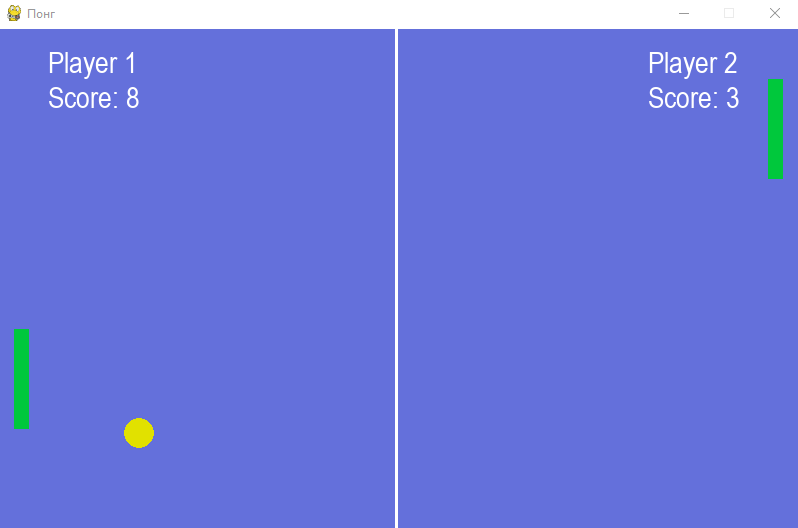
import pygame  
import random  
  
pygame.init()  
blue = (100, 112, 219)  
green = (0, 200, 60)  
yellow = (225, 225, 0)  
white = (255, 255, 255)  
  
pygame.mixer.music.load("back.mp3")  
pygame.mixer.music.play(-1)  
bumb\_sound = pygame.mixer.Sound("bum.mp3")  
  
title = 'Понг'  
screen\_width = 800  
screen\_height = 500  
screen = pygame.display.set\_mode((screen\_width, screen\_height))   
pygame.display.set\_caption(title)   
  
block\_speed\_1 = 10  
block\_width\_1 = 15  
block\_height\_1 = 100  
block\_rect\_1 = pygame.rect.Rect(778, 200, block\_width\_1, block\_height\_1)  
  
block\_speed\_2 = 10  
block\_width\_2 = 15  
block\_height\_2 = 100  
block\_rect\_2 = pygame.rect.Rect(7, 200, block\_width\_2, block\_height\_2)  
  
line = pygame.rect.Rect(397, 0, 3, 500)  
  
ball\_radius = 15  
ball\_speed = 7  
ball\_x\_speed = random.choice([ball\_speed, -ball\_speed])  
ball\_y\_speed = 0  
ball\_rect = pygame.rect.Rect(screen\_width / 2 - ball\_radius, screen\_height / 2 - ball\_radius, ball\_radius \* 2,  
 ball\_radius \* 2)  
  
player\_1 = 'Player 1'  
player\_2 = 'Player 2'  
points = 11  
score\_1 = 0  
score\_2 = 0  
  
arial\_font = pygame.font.match\_font('arial')  
arial\_font\_30 = pygame.font.Font(arial\_font, 30)  
arial\_font\_36 = pygame.font.Font(arial\_font, 36)  
  
clock = pygame.time.Clock()  
  
game\_over = False  
game = True  
menu\_start = True  
  
def restart():  
 global game\_over  
 global ball\_x\_speed  
 global ball\_y\_speed  
 global score\_1  
 global score\_2  
 global menu\_start  
  
 game\_over = False  
 menu\_start = False  
  
 block\_rect\_1.centerx = 777  
 block\_rect\_1.bottom = 300  
 block\_rect\_2.centerx = 23  
 block\_rect\_2.bottom = 300  
  
 ball\_rect.center = [screen\_width / 2, screen\_height / 2]  
 ball\_x\_speed = random.choice([ball\_speed, -ball\_speed])  
 ball\_y\_speed = 0  
  
 score\_1 = 0  
 score\_2 = 0  
  
while game:  
 screen.fill(blue)  
  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:   
 game = False  
 continue  
 elif event.type == pygame.KEYDOWN:  
 if event.key == pygame.K\_ESCAPE:  
 game = False  
 continue  
 elif event.key == pygame.K\_SPACE:  
 restart()  
 elif event.key == pygame.K\_m:  
 menu\_start = True  
  
 if not game\_over and not menu\_start:  
 keys = pygame.key.get\_pressed()  
 if keys[pygame.K\_DOWN]:  
 block\_rect\_1.y += block\_speed\_1  
 elif keys[pygame.K\_UP]:  
 block\_rect\_1.y -= block\_speed\_1  
 if keys[pygame.K\_s]:  
 block\_rect\_2.y += block\_speed\_2  
 elif keys[pygame.K\_w]:  
 block\_rect\_2.y -= block\_speed\_2  
  
 pygame.draw.rect(screen, white, line)   
 pygame.draw.rect(screen, green, block\_rect\_1)   
 pygame.draw.rect(screen, green, block\_rect\_2)   
  
 if block\_rect\_1.colliderect(ball\_rect):   
 bumb\_sound.play()  
 if random.choice(['up', 'down']) == 'up':  
 ball\_y\_speed = ball\_speed  
 else:  
 ball\_y\_speed = -ball\_speed  
 ball\_x\_speed = -ball\_speed  
 elif block\_rect\_2.colliderect(ball\_rect):  
 bumb\_sound.play()  
 if random.choice(['up', 'down']) == 'up':  
 ball\_y\_speed = ball\_speed  
 else:  
 ball\_y\_speed = -ball\_speed  
 ball\_x\_speed = ball\_speed  
 if ball\_rect.left == 0:  
 score\_2 += 1  
 elif ball\_rect.right == screen\_width:  
 score\_1 += 1  
  
 block\_rect\_1.clamp\_ip(screen.get\_rect())   
 block\_rect\_2.clamp\_ip(screen.get\_rect())  
  
 ball\_rect.x += ball\_x\_speed  
 ball\_rect.y += ball\_y\_speed  
  
 if ball\_rect.bottom >= screen\_height:  
 ball\_y\_speed = -ball\_speed  
 elif ball\_rect.top <= 0:  
 ball\_y\_speed = ball\_speed  
 elif ball\_rect.left <= 0:  
 ball\_x\_speed = ball\_speed  
 elif ball\_rect.right >= screen\_width:  
 ball\_x\_speed = -ball\_speed  
  
 if score\_1 == points:  
 winner = player\_1  
 game\_over = True  
 elif score\_2 == points:  
 winner = player\_2  
 game\_over = True  
  
 if not menu\_start:  
 pygame.draw.circle(screen, yellow, ball\_rect.center, ball\_radius)   
  
 score\_print\_1 = arial\_font\_30.render('Score: ' + str(score\_1), True, white)   
 score\_print\_2 = arial\_font\_30.render('Score: ' + str(score\_2), True, white)  
 player1\_text = arial\_font\_30.render(player\_1, True, white)  
 player2\_text = arial\_font\_30.render(player\_2, True, white)  
 if not game\_over:  
 screen.blit(score\_print\_1, [50, 50])   
 screen.blit(score\_print\_2, [650, 50])  
 screen.blit(player1\_text, [50, 15])  
 screen.blit(player2\_text, [650, 15])  
 else:  
 retry\_surface = arial\_font\_36.render('press SPACE to restart', True, white)  
 win\_text = arial\_font\_30.render('{} wins'.format(winner), True, white)  
 screen.blit(score\_print\_1, [50, 50])  
 screen.blit(score\_print\_2, [650, 50])  
 screen.blit(player1\_text, [50, 15])  
 screen.blit(player2\_text, [650, 15])  
 screen.blit(retry\_surface, [250, 250])  
 screen.blit(win\_text, [335, 150])  
 else:   
 player1\_text = arial\_font\_30.render(player\_1, True, white)  
 player2\_text = arial\_font\_30.render(player\_2, True, white)  
 retry\_surface = arial\_font\_36.render('press SPACE to start', True, white)  
 esc\_text = arial\_font\_36.render('press Esc to exit', True, white)  
 points\_text = arial\_font\_36.render('Required points: {}'.format(points), True, white)  
 screen.blit(player1\_text, [50, 15])  
 screen.blit(player2\_text, [650, 15])  
 screen.blit(retry\_surface, [260, 200])  
 screen.blit(esc\_text, [290, 250])  
 screen.blit(points\_text, [270, 15])  
  
 clock.tick(60)   
 pygame.display.flip()   
pygame.quit()

Графика игры:

Меню игры:



Процесс игры:



Конец партии:

