Санкт-Петербургский политехнический университет Петра Великого

Физико-механический институт

**Высшая школа теоретической механики и математической физики**

**КУРСОВой проект**

**Создание игры «Четыре в ряд»**

**на языке программирования Python**

по дисциплине «Языки программирования»

Выполнили

студент гр. 5030103/10001 <*подпись>* Е. С. Сорокопудова

Преподаватель,

сотрудник ВШТМиМФ <*подпись*> А. А. Кедров

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2023г.

Санкт-Петербург

2023

**СОДЕРЖАНИЕ**

[Введение 3](#_Toc135492506)

[Основная часть 4](#_Toc135492507)

[Принцип работы игры 4](#_Toc135492508)

[Описание кода Python 6](#_Toc135492509)

[Заключение 11](#_Toc135492510)

[Список используемых источников 12](#_Toc135492511)

# **ВВЕДЕНИЕ**

Цель проекта:

Реализация и визуализация компьютерной игры «Четыре в ряд» с использованием полученных в ходе изучения языка программирования Python знаний.

Задачи:

1.Создание игрового поля

2.Создание падающих фишек

3.Создание разных исходов игры

Объект исследования:

Объектом исследования является язык программирования Python.

Python— высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. Язык является полностью объектно-ориентированным в том плане, что всё является объектами. Синтаксис ядра языка минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации. Сам же язык известен как интерпретируемый и используется в том числе для написания скриптов.

# **ОСНОВНАЯ ЧАСТЬ**

## ПРИНЦИП РАБОТЫ ИГРЫ

**«Четыре в ряд»** или **«Соедини четыре»** (*Connect four*) — игра для двоих, в которой игроки сначала выбирают цвет фишек (у нас красный и синий), а затем ходят по очереди, роняя фишки в ячейки вертикальной доски. Цель игры — расположить раньше противника подряд по горизонтали, вертикали или диагонали четыре фишки своего цвета.



При запуске игрока встречает кнопка «Начать игру»

При выборе фишкой места падения на доступную ячейку нужного столбца отбрасывается «тень». Это сделано для упрощения ориентирования на игровом поле.

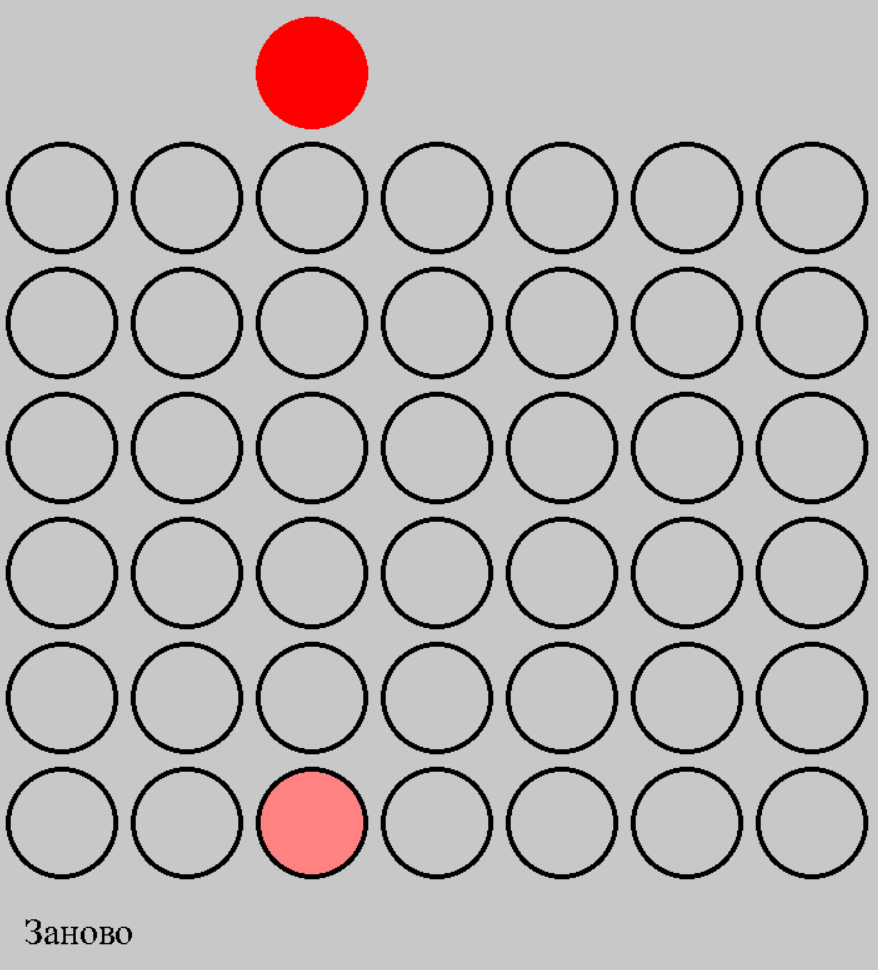
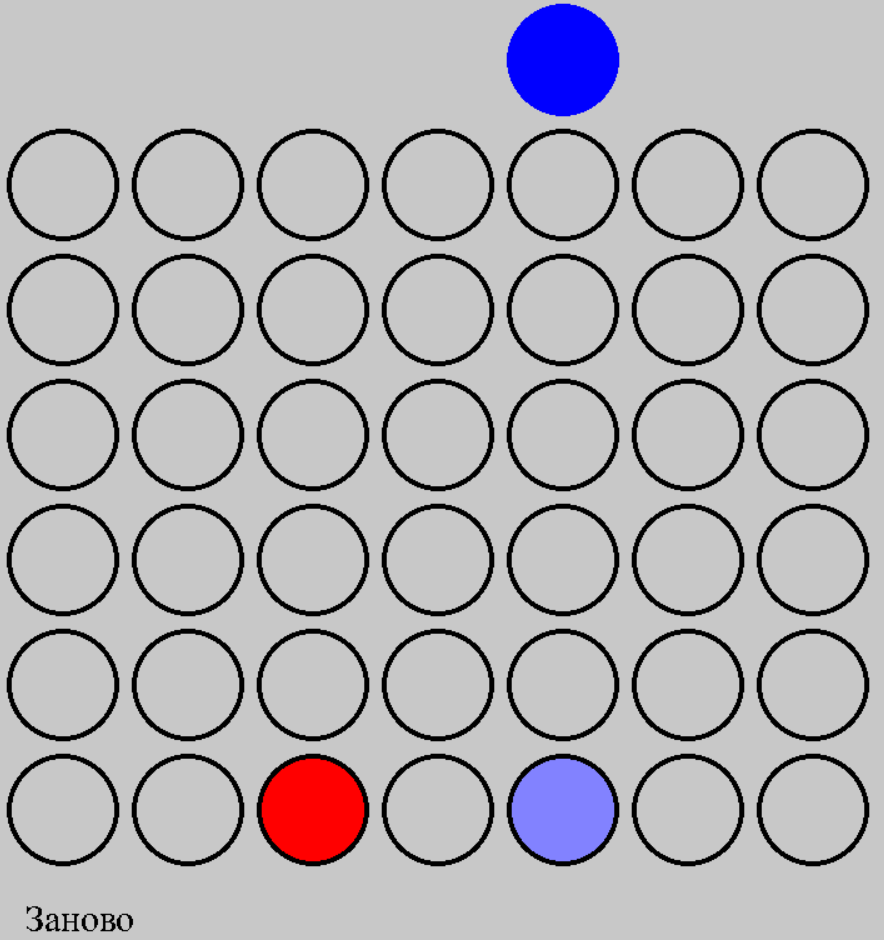


рис.1 «Тень» красной фишки рис.2 «Тень» синей фишки

Игрок может победить в случае размещения 4 фишек одного цвета (своего) по вертикали (рис.1), горизонтали (рис.2), диагонали (рис. 3). В случае победы сверху появляется соответствующая надпись цвета победившего игрока.



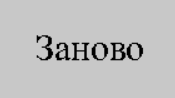
рис.1 Победа красных рис.2 Победа синих рис. 3 Победа красных

(по вертикали) (по горизонтали) (по горизонтали)



В случае ничьей на экране появляется

«Победила дружба!»



Так же на игровом поле присутствует кнопка «заново», она позволяет в любой момент начать игру сначала

## ОПИСАНИЕ КОДА PYTHON

В своём проекте мы создали приложение на основе языка программирования python с использованием библиотек pygame и sys.

Создаём окно приложения, задаём необходимые переменные.

pygame.init()

pygame.display.set\_caption('Four in line')

clock = pygame.time.Clock()

fonty = pygame.font.SysFont('Times new roman', 30)

sc = pygame.display.set\_mode((w, h))

surf = pygame.Surface((w, h))

Создаём массив положений шаров.

for j in range(n):

poses.append([])

for i in range(m):

poses[j].append([])

poses[j][i] = 0

Прописываем функции анализа победы analis\_win\_1, analis\_win\_2, analis\_win\_3, analis\_win\_4. Все функции принимаю по две переменные типа real. Алгоритм рассмотрим на примере первой функции. Вносим в массив massiv позицию только что брошенного пользователем шарика. Идём по диагоналям, проходящим через эту точку и вносим в массив условное обозначение цвета (1 – красный, 2 – синий). Если в массиве оказывается 4 подряд идущих одинаковых цифры, то только что сходивший игрок выиграл и мы обращаемся к функции win()

def analis\_win\_1(xy, const\_xy): # Диагональ \

massiv = [poses[xy[0]][xy[1]]]

kol = 0

while (xy[0] - 1 >= 0) and (xy[1] - 1 >= 0):

xy[0] -= 1

xy[1] -= 1

massiv = [poses[xy[0]][xy[1]]] + massiv

xy = const\_xy

while (xy[0] + 1 <= n - 1) and (xy[1] + 1 <= m - 1):

xy[0] += 1

xy[1] += 1

massiv.append(poses[xy[0]][xy[1]])

for i in range(len(massiv)):

if (massiv[i] == who\_is):

kol += 1

if kol >= 4:

return win(who\_is)

else:

kol = 0

Функция win() возвращает ложь, чтобы передать это в глобальную переменную и остановить цикл игры.

def win(who\_is):

return False

Далее код основной программы. Создаём кнопку начать игру. При нажатии на неё пользователю открывается окно с игрой, где нарисованы круги 6 по вертикали и 7 по горизонтали. Снизу создаётся кнопка «Заново», которая перезапускает игровой процесс. Это всё происходит, когда значение переменной is\_win равняется правде. Иначе мы можем запустить код игры.

fonty = pygame.font.SysFont('Times new roman', 30)

mouse = pygame.mouse.get\_pos()

start\_button\_2 = fonty.render('Заново', False, 'black')

start\_button\_2\_rect = start\_button\_2.get\_rect(topleft=(30, h - 70))

if not start\_button\_2\_rect.collidepoint(mouse):

start\_button\_2 = fonty.render('Заново', False, 'black')

if start\_button\_2\_rect.collidepoint(mouse):

start\_button\_2 = fonty.render('Заново', False, 'white')

sc.blit(start\_button\_2, start\_button\_2\_rect)

if start\_button\_2\_rect.collidepoint(mouse) and pygame.mouse.get\_pressed()[0]:

non\_stop = True

for j in range(n):

for i in range(m):

poses[j][i] = 0

Используем pygame.mouse.get\_pressed()[0] и pygame.mouse.get\_ pos()[0],

чтобы анализировать, нажата ли левая кнопка и где находится курсор мыши.

Если нажато в столбце, куда возможно кинуть мяч, нижней свободной ячейке присваивается значение цвета текущего игрока. Далее идёт анализ победы вышеупомянутыми функциями.

if (now\_event != 0) and (event.type == pygame.MOUSEBUTTONDOWN) and (lastEvent != now\_event) and (poses[0][k] != 2) and (poses[0][k] != 1):

if mouse\_pos[1] < 720:

if k > 6:

k = 6

for j in range(n):

if (poses[0][k] == 2) or (poses[0][k] == 1):

break

elif poses[n - 1 - j][k] == 0:

k2 = n - 1 - j

break

poses[k2][k] = who\_is

analis = analis\_win\_1([k2, k], [k2, k])

if analis == False:

non\_stop = False

Иначе, если столбец переполнен, пользователь щёлкнул вне игрового поля или щелчка не было, то в ячейке над всеми кругами рисуется круг цвета текущего игрока, демонстрируя, чей ход и куда вероятно произойдёт падение шарика.

mouse\_pos = pygame.mouse.get\_pos()

if mouse\_pos[1] < 720:

k = abs(mouse\_pos[0] - 10) // 100

if k > 6:

k = 6

for j in range(n):

if (poses[0][k] == 2) or (poses[0][k] == 1):

break

elif poses[n - 1 - j][k] == 0:

k2 = n - 1 - j

if who\_is == 1:

poses[k2][k] = 3

else:

poses[k2][k] = 4

break

Далее идёт отрисовка в зависимости от номера в ячейке.

pygame.draw.rect(surf, (200, 200, 200), (0, 0, w, h))

for j in range(n):

for i in range(m):

if poses[j][i] == 1:

color = (255, 0, 0)

elif poses[j][i] == 2:

color = (0, 0, 255)

pygame.draw.circle(surf, color, (60 + i \* 100, 160 + j \* 100), 45)

pygame.draw.circle(surf, (0, 0, 0), 60 + i \* 100, 160 + j \* 100), 45, 4).

blit(surf, (0,0))

# **ЗАКЛЮЧЕНИЕ**

За время выполнения курсовой работы, были изучены принципы работы с библиотекой Pygame. На основе полученных знаний была реализована и визуализирована компьютерная игра «Четыре в ряд». В дальнейшем существует возможность реализации игры в онлайн формате, в качестве последующей доработки программы и улучшения динамической и визуальной составляющих игры.

В заключении хотелось бы сказать, что нашей команде понравилось работать над данным проектом, представленные в ходе работы задачи были выполнены, и мы получили полезный опыт в работе над созданием игры в Python и написанием курсовых работ.

# **СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

[1] Бизли Д. Python. Подробный справочник. – Пер. с англ. – СПб.: Символ-Плюс, 2010.

[2] Wikipedia