import pygame  
import random  
  
# Инициализация Pygame  
pygame.init()  
  
# Константы  
GRID\_SIZE = 20  
WHITE, BLACK, YELLOW, RED, BLUE, GREEN, ORANGE, GRAY = (255, 255, 255), (0, 0, 0), (255, 255, 0), (255, 0, 0), (  
0, 0, 255), (0, 255, 0), (255, 165, 0), (128, 128, 128)  
COLORS = [WHITE, YELLOW, RED, BLUE, GREEN, ORANGE, GRAY]  
WIDTH, HEIGHT = 400, 600  
  
# Лабиринт  
maze = [  
 "############################",  
 "#oooooooooooo##oooooooooooo#",  
 "#o####o#####o##o#####o####o#",  
 "#o####o#####o##o#####o####o#",  
 "#o####o#####o##o#####o####o#",  
 "#oooooooooooooooooooooooooo#",  
 "#o####o##o######o##o####o###",  
 "#o####o##o######o##o####o###",  
 "#oooooo##oooo##oooo##oooooo#",  
 "######o#####o##o#####o######",  
 "######o#####o##o#####o######",  
 "######o##oooooooo##ooooo####",  
 "######o##o########o##o######",  
 "######o##o########o##o######",  
 "#oooooooooooo##oooooooooooo#",  
 "#o####o#####o##o#####o####o#",  
 "#o####o#####o##o#####o####o#",  
 "#o####o##oooooooo##o####o###",  
 "#o####o##o########o##o####o#",  
 "#oooooo##oooo##oooo##oooooo#",  
 "###########o##o#############",  
 "###########o##o#############",  
 "###########o##o#############",  
 "############################"  
]  
  
# Позиции  
pacman\_pos = [1, 1]  
ghosts = [  
 {"pos": [12, 11], "color": RED, "direction": random.choice([[0, 1], [0, -1], [1, 0], [-1, 0]])},  
 {"pos": [12, 12], "color": GREEN, "direction": random.choice([[0, 1], [0, -1], [1, 0], [-1, 0]])},  
 {"pos": [11, 12], "color": ORANGE, "direction": random.choice([[0, 1], [0, -1], [1, 0], [-1, 0]])}  
]  
dot\_positions = [(x, y) for y, row in enumerate(maze) for x, cell in enumerate(row) if cell == 'o']  
  
# Направления  
direction = [0, 0]  
  
# Время  
clock = pygame.time.Clock()  
  
# Счет  
score = 0  
  
# Шрифты  
font = pygame.font.Font(None, 36)  
  
# Размер окна  
SCREEN\_WIDTH = len(maze[0]) \* GRID\_SIZE  
SCREEN\_HEIGHT = len(maze) \* GRID\_SIZE + 100  
  
# Экран  
screen = pygame.display.set\_mode((SCREEN\_WIDTH, SCREEN\_HEIGHT))  
pygame.display.set\_caption('Pacman')  
  
# Параметры пакмена  
pacman\_image = pygame.image.load('pacmen.png.png')  
pacman\_image = pygame.transform.scale(pacman\_image, (60, 40)) # Уменьшение изображения  
bird\_rect = pacman\_image.get\_rect(center=(WIDTH // 4, HEIGHT // 2))  
velocity = 0  
  
  
# Функция отрисовки лабиринта  
def draw\_maze():  
 for y, row in enumerate(maze):  
 for x, cell in enumerate(row):  
 if cell == "#":  
 pygame.draw.rect(screen, BLUE, pygame.Rect(x \* GRID\_SIZE, y \* GRID\_SIZE, GRID\_SIZE, GRID\_SIZE))  
 elif cell == "o" and (x, y) in dot\_positions:  
 pygame.draw.circle(screen, WHITE, (x \* GRID\_SIZE + GRID\_SIZE // 2, y \* GRID\_SIZE + GRID\_SIZE // 2), 3)  
  
  
# Функция отрисовки счета  
def draw\_score(score):  
 score\_surface = font.render(f'Score: {score}', True, GRAY)  
 score\_rect = score\_surface.get\_rect()  
 score\_rect.midbottom = (SCREEN\_WIDTH // 2, SCREEN\_HEIGHT - 30)  
  
 background\_rect = pygame.Rect(score\_rect.left - 10, score\_rect.top - 5, score\_rect.width + 20,  
 score\_rect.height + 10)  
 pygame.draw.rect(screen, YELLOW, background\_rect)  
 pygame.draw.rect(screen, BLACK, background\_rect, 3)  
  
 screen.blit(score\_surface, score\_rect)  
  
  
# Проверка движения  
def move(pos, direction):  
 new\_pos = [pos[0] + direction[0], pos[1] + direction[1]]  
 if 0 <= new\_pos[0] < len(maze[0]) and 0 <= new\_pos[1] < len(maze) and maze[new\_pos[1]][new\_pos[0]] != "#":  
 return new\_pos  
 return pos  
  
  
# Проверка победы  
def check\_win():  
 return len(dot\_positions) == 0  
  
  
# Функция для генерации градиентного фона  
def draw\_gradient\_background():  
 for y in range(SCREEN\_HEIGHT):  
 r = int(255 \* (1 - y / SCREEN\_HEIGHT)) # Красный (градиент вверх)  
 g = int(255 \* (y / SCREEN\_HEIGHT)) # Зеленый (градиент вниз)  
 b = int(255 \* (1 - abs(y / SCREEN\_HEIGHT - 0.5) \* 2)) # Синий (градиент к середине)  
 pygame.draw.line(screen, (r, g, b), (0, y), (SCREEN\_WIDTH, y))  
  
  
# Экран старта  
def start\_screen():  
 screen.fill(BLACK)  
 # Список для хранения данных о случайных кругах  
 circles = []  
  
 # Генерация случайных кругов  
 for \_ in range(100):  
 radius = random.randint(5, 20)  
 color = random.choice(COLORS)  
 pos = [random.randint(0, SCREEN\_WIDTH), random.randint(0, SCREEN\_HEIGHT)] # Используем список вместо кортежа  
 circles.append([pos, radius, color])  
  
 title\_text = font.render('PACMAN', True, YELLOW)  
 start\_text = font.render('Press SPACE to Start', True, WHITE)  
 instruction\_text = font.render('Нажмите клавишу пробел для старта', True, WHITE)  
  
 screen.blit(title\_text, (  
 SCREEN\_WIDTH // 2 - title\_text.get\_width() // 2, SCREEN\_HEIGHT // 2 - title\_text.get\_height() // 2 - 40))  
 screen.blit(start\_text,  
 (SCREEN\_WIDTH // 2 - start\_text.get\_width() // 2, SCREEN\_HEIGHT // 2 - start\_text.get\_height() // 2))  
 screen.blit(instruction\_text, (SCREEN\_WIDTH // 2 - instruction\_text.get\_width() // 2,  
 SCREEN\_HEIGHT // 2 - instruction\_text.get\_height() // 2 + 40))  
  
 # Анимация случайных кругов  
 while not pygame.key.get\_pressed()[pygame.K\_SPACE]:  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 pygame.quit()  
 return  
  
 screen.fill(BLACK)  
 for circle in circles:  
 pos, radius, color = circle  
 pygame.draw.circle(screen, color, pos, radius)  
 # Обновление позиции круга  
 pos[0] += random.randint(-2, 2)  
 pos[1] += random.randint(-2, 2)  
 # Удаление круга, если он вышел за границы экрана  
 if pos[0] < 0 or pos[0] > SCREEN\_WIDTH or pos[1] < 0 or pos[1] > SCREEN\_HEIGHT:  
 circles.remove(circle)  
  
 screen.blit(title\_text, (  
 SCREEN\_WIDTH // 2 - title\_text.get\_width() // 2, SCREEN\_HEIGHT // 2 - title\_text.get\_height() // 2 - 40))  
 screen.blit(start\_text, (  
 SCREEN\_WIDTH // 2 - start\_text.get\_width() // 2, SCREEN\_HEIGHT // 2 - start\_text.get\_height() // 2))  
 screen.blit(instruction\_text, (SCREEN\_WIDTH // 2 - instruction\_text.get\_width() // 2,  
 SCREEN\_HEIGHT // 2 - instruction\_text.get\_height() // 2 + 40))  
  
 pygame.display.flip()  
 clock.tick(30)  
  
  
# Экран Game Over  
def game\_over\_screen(message, score):  
 draw\_gradient\_background()  
  
 game\_over\_text = font.render(message, True, WHITE)  
 score\_text = font.render(f'Score: {score}', True, WHITE)  
 restart\_text = font.render('Press R to Restart', True, WHITE)  
 screen.blit(game\_over\_text, (  
 SCREEN\_WIDTH // 2 - game\_over\_text.get\_width() // 2, SCREEN\_HEIGHT // 2 - game\_over\_text.get\_height() // 2 - 20))  
 screen.blit(score\_text, (  
 SCREEN\_WIDTH // 2 - score\_text.get\_width() // 2, SCREEN\_HEIGHT // 2 - score\_text.get\_height() // 2 + 20))  
 screen.blit(restart\_text, (  
 SCREEN\_WIDTH // 2 - restart\_text.get\_width() // 2, SCREEN\_HEIGHT // 2 - restart\_text.get\_height() // 2 + 60))  
 pygame.display.flip()  
  
  
# Основной цикл игры  
def game():  
 global pacman\_pos, ghosts, direction, score, dot\_positions  
 pacman\_pos = [1, 1]  
 ghosts = [  
 {"pos": [12, 11], "color": RED, "direction": random.choice([[0, 1], [0, -1], [1, 0], [-1, 0]])},  
 {"pos": [12, 12], "color": GREEN, "direction": random.choice([[0, 1], [0, -1], [1, 0], [-1, 0]])},  
 {"pos": [11, 12], "color": ORANGE, "direction": random.choice([[0, 1], [0, -1], [1, 0], [-1, 0]])}  
 ]  
 direction = [0, 0]  
 score = 0  
 dot\_positions = [(x, y) for y, row in enumerate(maze) for x, cell in enumerate(row) if cell == 'o']  
  
 game\_over = False  
 paused = False # Флаг для остановки движения Pacman  
 while not game\_over:  
 screen.fill(BLACK)  
  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 game\_over = True  
 pygame.quit()  
 return  
 if event.type == pygame.KEYDOWN:  
 if event.key == pygame.K\_LEFT:  
 direction = [-1, 0]  
 if event.key == pygame.K\_RIGHT:  
 direction = [1, 0]  
 if event.key == pygame.K\_UP:  
 direction = [0, -1]  
 if event.key == pygame.K\_DOWN:  
 direction = [0, 1]  
 if event.key == pygame.K\_SPACE:  
 return  
 if event.key == pygame.K\_s: # Проверка нажатия клавиши 'S'  
 paused = not paused # Переключение состояния паузы  
  
 if not paused: # Движение Pacman только если не на паузе  
 pacman\_pos = move(pacman\_pos, direction)  
  
 # Поедание точек  
 if (pacman\_pos[0], pacman\_pos[1]) in dot\_positions:  
 dot\_positions.remove((pacman\_pos[0], pacman\_pos[1]))  
 score += 10  
  
 # Проверка на победу  
 if check\_win():  
 game\_over = True  
 game\_over\_screen("Success!", score)  
 return  
  
 if not paused: # Движение привидений только если не на паузе  
 for ghost in ghosts:  
 ghost['pos'] = move(ghost['pos'], ghost['direction'])  
 if random.random() < 0.2:  
 ghost['direction'] = random.choice([[0, 1], [0, -1], [1, 0], [-1, 0]])  
  
 for ghost in ghosts:  
 if ghost['pos'] == pacman\_pos:  
 game\_over = True  
  
 draw\_maze()  
  
 # Отрисовка Pacman  
 pygame.draw.circle(screen, YELLOW,  
 (pacman\_pos[0] \* GRID\_SIZE + GRID\_SIZE // 2, pacman\_pos[1] \* GRID\_SIZE + GRID\_SIZE // 2),  
 GRID\_SIZE // 2)  
  
 # Отрисовка привидений  
 for ghost in ghosts:  
 pygame.draw.rect(screen, ghost['color'],  
 pygame.Rect(ghost['pos'][0] \* GRID\_SIZE, ghost['pos'][1] \* GRID\_SIZE, GRID\_SIZE,  
 GRID\_SIZE))  
  
 draw\_score(score)  
  
 pygame.display.flip()  
 clock.tick(10)  
  
  
# Главный цикл  
running = True  
while running:  
 start\_screen()  
 start = False  
 while not start:  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 running = False  
 start = True  
 if event.type == pygame.KEYDOWN:  
 if event.key == pygame.K\_SPACE:  
 start = True  
  
 if running:  
 game()  
 game\_over\_screen("Game Over", score)  
 restart = False  
 while not restart:  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 running = False  
 restart = True  
 if event.type == pygame.KEYDOWN:  
 if event.key == pygame.K\_r:  
 restart = True  
  
pygame.quit()