import pygame

import random

import sys

from typing import List, Tuple

# Инициализация Pygame

pygame.init()

# Константы

SCREEN\_WIDTH = 800

SCREEN\_HEIGHT = 600

GRID\_SIZE = 20

GRID\_WIDTH = SCREEN\_WIDTH // GRID\_SIZE

GRID\_HEIGHT = SCREEN\_HEIGHT // GRID\_SIZE

# Цвета

BOARD\_BACKGROUND\_COLOR = (0, 0, 0)

SNAKE\_COLOR = (0, 255, 0)

APPLE\_COLOR = (255, 0, 0)

TEXT\_COLOR = (255, 255, 255)

class GameObject:

"""Базовый класс для игровых объектов."""

def init(self, position: Tuple[int, int] = None):

"""

Инициализирует игровой объект.

Args:

position: Начальная позиция объекта. Если None, генерируется случайная позиция.

"""

if position is None:

self.position = self.generate\_random\_position()

else:

self.position = position

def generate\_random\_position(self) -> Tuple[int, int]:

"""Генерирует случайную позицию на игровом поле."""

return (

random.randint(0, GRID\_WIDTH - 1) \* GRID\_SIZE,

random.randint(0, GRID\_HEIGHT - 1) \* GRID\_SIZE

)

def draw(self, surface: pygame.Surface):

"""Абстрактный метод для отрисовки объекта."""

raise NotImplementedError("Метод draw должен быть реализован в дочернем классе")

class Apple(GameObject):

"""Класс яблока - еды для змейки."""

def init(self):

"""Инициализирует яблоко со случайной позицией."""

super().init()

def draw(self, surface: pygame.Surface):

"""Отрисовывает яблоко на поверхности."""

rect = pygame.Rect(

(self.position[0], self.position[1]),

(GRID\_SIZE, GRID\_SIZE)

)

pygame.draw.rect(surface, APPLE\_COLOR, rect)

pygame.draw.rect(surface, (255, 255, 255), rect, 1)

class Snake(GameObject):

"""Класс змейки."""

def init(self):

"""Инициализирует змейку в начальном состоянии."""

super().init(((SCREEN\_WIDTH // 2), (SCREEN\_HEIGHT // 2)))

self.positions = [self.position]

self.length = 1

self.direction = random.choice([(GRID\_SIZE, 0), (-GRID\_SIZE, 0), (0, GRID\_SIZE), (0, -GRID\_SIZE)])

self.last = None

def get\_head\_position(self) -> Tuple[int, int]:

"""Возвращает позицию головы змейки."""

return self.positions[0]

def move(self):

"""Перемещает змейку в текущем направлении."""

head\_x, head\_y = self.get\_head\_position()

dx, dy = self.direction

new\_x = (head\_x + dx) % SCREEN\_WIDTH

new\_y = (head\_y + dy) % SCREEN\_HEIGHT

new\_position = (new\_x, new\_y)

# Проверка столкновения с собой

if new\_position in self.positions[1:]:

self.reset()

return

self.positions.insert(0, new\_position)

if len(self.positions) > self.length:

self.last = self.positions.pop()

def reset(self):

"""Сбрасывает змейку в начальное состояние."""

self.positions = [((SCREEN\_WIDTH // 2), (SCREEN\_HEIGHT // 2))]

self.length = 1

self.direction = random.choice([(GRID\_SIZE, 0), (-GRID\_SIZE, 0), (0, GRID\_SIZE), (0, -GRID\_SIZE)])

self.last = None

def draw(self, surface: pygame.Surface):

"""Отрисовывает змейку на поверхности."""

for position in self.positions:

rect = pygame.Rect(

(position[0], position[1]),

(GRID\_SIZE, GRID\_SIZE)

)

pygame.draw.rect(surface, SNAKE\_COLOR, rect)

pygame.draw.rect(surface, (255, 255, 255), rect, 1)

def draw\_score(surface: pygame.Surface, score: int):

"""Отрисовывает счет на поверхности."""

font = pygame.font.Font(None, 36)

text = font.render(f'Счет: {score}', True, TEXT\_COLOR)

surface.blit(text, (10, 10))

def main():

"""Основная функция игры."""

screen = pygame.display.set\_mode((SCREEN\_WIDTH, SCREEN\_HEIGHT))

pygame.display.set\_caption('Змейка')

clock = pygame.time.Clock()

snake = Snake()

apple = Apple()

score = 0

while True:

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

sys.exit()

elif event.type == pygame.KEYDOWN:

if event.key == pygame.K\_UP and snake.direction != (0, GRID\_SIZE):

snake.direction = (0, -GRID\_SIZE)

elif event.key == pygame.K\_DOWN and snake.direction != (0, -GRID\_SIZE):

snake.direction = (0, GRID\_SIZE)

elif event.key == pygame.K\_LEFT and snake.direction != (GRID\_SIZE, 0):

snake.direction = (-GRID\_SIZE, 0)

elif event.key == pygame.K\_RIGHT and snake.direction != (-GRID\_SIZE, 0):

snake.direction = (GRID\_SIZE, 0)

snake.move()

# Проверка съедания яблока

if snake.get\_head\_position() == apple.position:

snake.length += 1

score += 1

apple = Apple()

# Убедимся, что яблоко не появилось на змейке

while apple.position in snake.positions:

apple = Apple()

screen.fill(BOARD\_BACKGROUND\_COLOR)

snake.draw(screen)

apple.draw(screen)

draw\_score(screen, score)

pygame.display.update()

clock.tick(10) # Скорость игры

if name == "main":

main()