import pygame

import math

import random

# 初始化 Pygame

pygame.init()

# 設置全螢幕模式

screen = pygame.display.set\_mode((0, 0), pygame.FULLSCREEN) # 設置顯示窗口為全屏模式

screen\_width, screen\_height = screen.get\_size() # 獲取當前屏幕的寬度和高度

# 設置顏色

white = (255, 255, 255) # 白色，用於玩家飛機和其他元素

black = (0, 0, 0) # 黑色，用於背景

gray = (100, 100, 100) # 灰色，用於背景方塊

red = (255, 0, 0) # 紅色，用於顯示玩家血量

# 設置玩家飛機屬性

player\_width = 40 # 飛機的寬度

player\_height = 20 # 飛機的高度

player\_x = screen\_width // 2 # 初始位置的X坐標，設為屏幕中央

player\_y = screen\_height // 2 # 初始位置的Y坐標，設為屏幕中央

player\_health = 5 # 玩家初始血量

player\_speed = 6 # 基本飛機速度

player\_speed\_boost = 2 # 速度加倍因子，當按下空格鍵時使用

# 設置子彈屬性

bullet\_width = 10 # 子彈的寬度

bullet\_height = 5 # 子彈的高度

bullet\_speed = 30 # 子彈的速度

bullets = [] # 用於存儲當前屏幕上的所有子彈

# 設置敵機屬性

enemy\_width = 40 # 敵機的寬度

enemy\_height = 20 # 敵機的高度

enemy\_speed = 2 # 敵機的速度

enemies = [] # 用於存儲當前屏幕上的所有敵機

spawn\_enemy\_event = pygame.USEREVENT + 1 # 自定義事件，用於定時生成敵機

pygame.time.set\_timer(spawn\_enemy\_event, 500) # 每500毫秒觸發一次敵機生成事件

# 設置背景方塊，這些方塊將隨著飛機的移動而移動

background\_blocks = [] # 存儲所有背景方塊的位置

for \_ in range(20): # 生成20個隨機位置的背景方塊

block\_x = random.randint(0, screen\_width) # 隨機生成X坐標

block\_y = random.randint(0, screen\_height) # 隨機生成Y坐標

background\_blocks.append([block\_x, block\_y]) # 將生成的方塊加入列表

# 設置分數和計時器

score = 0 # 初始分數為0

start\_time = pygame.time.get\_ticks() # 獲取遊戲開始時的時間，用於計算經過時間

font = pygame.font.SysFont(None, 35) # 設置字體，用於顯示分數、血量和計時器

# 遊戲主循環

running = True # 設置遊戲狀態為運行中

clock = pygame.time.Clock() # 用於控制遊戲幀率

shoot\_cooldown = 0 # 用於控制射擊頻率的冷卻計時器

while running: # 遊戲主循環開始

screen.fill(black) # 每次循環開始時清空屏幕並塗黑

# 處理所有事件

for event in pygame.event.get():

if event.type == pygame.QUIT: # 當點擊窗口關閉按鈕時，結束遊戲

running = False

if event.type == spawn\_enemy\_event: # 當定時器觸發時生成一個新的敵機

edge = random.choice(["top", "bottom", "left", "right"]) # 隨機選擇一個邊界作為敵機的生成位置

if edge == "top": # 如果選擇了屏幕頂部

enemy\_x = random.randint(0, screen\_width) # 敵機的X坐標為隨機值

enemy\_y = -enemy\_height # 敵機的Y坐標設為屏幕外的上方

elif edge == "bottom": # 如果選擇了屏幕底部

enemy\_x = random.randint(0, screen\_width) # 敵機的X坐標為隨機值

enemy\_y = screen\_height # 敵機的Y坐標設為屏幕外的下方

elif edge == "left": # 如果選擇了屏幕左側

enemy\_x = -enemy\_width # 敵機的X坐標設為屏幕外的左側

enemy\_y = random.randint(0, screen\_height) # 敵機的Y坐標為隨機值

else: # 如果選擇了屏幕右側

enemy\_x = screen\_width # 敵機的X坐標設為屏幕外的右側

enemy\_y = random.randint(0, screen\_height) # 敵機的Y坐標為隨機值

enemies.append([enemy\_x, enemy\_y]) # 將新生成的敵機加入敵機列表

# 檢查鍵盤輸入

keys = pygame.key.get\_pressed() # 獲取當前按下的鍵盤按鍵

speed\_boost = keys[pygame.K\_SPACE] # 如果按下空格鍵，則啟用速度加倍

# 更新背景方塊位置以模擬飛行

mouse\_x, mouse\_y = pygame.mouse.get\_pos() # 獲取滑鼠的當前位置

angle = math.atan2(mouse\_y - player\_y, mouse\_x - player\_x) # 計算飛機朝向滑鼠位置的角度

move\_speed = player\_speed \* (2 if speed\_boost else 1) # 如果速度加倍，則移動速度翻倍

# 更新背景方塊位置

for block in background\_blocks: # 對每個背景方塊進行更新

block[0] += -move\_speed \* math.cos(angle) # 根據飛機的移動方向和速度更新方塊的X坐標

block[1] += -move\_speed \* math.sin(angle) # 根據飛機的移動方向和速度更新方塊的Y坐標

if block[0] < 0: block[0] += screen\_width # 如果方塊超出屏幕左側，將其移動到右側

if block[0] > screen\_width: block[0] -= screen\_width # 如果方塊超出屏幕右側，將其移動到左側

if block[1] < 0: block[1] += screen\_height # 如果方塊超出屏幕上方，將其移動到下方

if block[1] > screen\_height: block[1] -= screen\_height # 如果方塊超出屏幕下方，將其移動到上方

pygame.draw.rect(screen, gray, [block[0], block[1], 10, 10]) # 在屏幕上繪製背景方塊

# 繪製飛機

player\_points = [ # 計算飛機的四個頂點坐標，根據飛機的中心點和角度進行旋轉

(player\_x + player\_width \* math.cos(angle) / 2, player\_y + player\_width \* math.sin(angle) / 2), # 機頭

(player\_x - player\_height \* math.sin(angle) / 2, player\_y + player\_height \* math.cos(angle) / 2), # 左翼

(player\_x - player\_width \* math.cos(angle) / 2, player\_y - player\_width \* math.sin(angle) / 2), # 機尾

(player\_x + player\_height \* math.sin(angle) / 2, player\_y - player\_height \* math.cos(angle) / 2) # 右翼

]

pygame.draw.polygon(screen, white, player\_points) # 繪製飛機，顏色為白色

# 射擊

shoot\_cooldown -= 1 # 減少射擊冷卻時間

if pygame.mouse.get\_pressed()[0] and shoot\_cooldown <= 0: # 如果滑鼠左鍵按下且冷卻時間結束，則發射子彈

for i in range(-15, 15): # 從 -15 到 14 總共發射 30 發子彈

bullet\_angle = angle + math.radians(i \* 1.5) # 每發子彈的角度間隔1.5度

bullet\_dx = bullet\_speed \* math.cos(bullet\_angle) # 計算子彈在X方向上的速度

bullet\_dy = bullet\_speed \* math.sin(bullet\_angle) # 計算子彈在Y方向上的速度

bullets.append([player\_x, player\_y, bullet\_dx, bullet\_dy]) # 將子彈的初始位置和速度加入列表

shoot\_cooldown = 10 # 設置射擊冷卻時間，控制射速

# 更新子彈位置

for bullet in bullets[:]: # 遍歷所有子彈

bullet[0] += bullet[2] # 更新子彈的X坐標

bullet[1] += bullet[3] # 更新子彈的Y坐標

if bullet[0] < 0 or bullet[0] > screen\_width or bullet[1] < 0 or bullet[1] > screen\_height: # 如果子彈超出屏幕範圍

bullets.remove(bullet) # 刪除超出屏幕的子彈

# 更新敵機位置並追蹤飛機

for enemy in enemies[:]: # 遍歷所有敵機

enemy\_angle = math.atan2(player\_y - enemy[1], player\_x - enemy[0]) # 計算敵機朝向玩家飛機的角度

enemy[0] += enemy\_speed \* math.cos(enemy\_angle) # 根據角度更新敵機的X坐標

enemy[1] += enemy\_speed \* math.sin(enemy\_angle) # 根據角度更新敵機的Y坐標

if enemy[0] < 0 or enemy[0] > screen\_width or enemy[1] < 0 or enemy[1] > screen\_height: # 如果敵機超出屏幕範圍

enemies.remove(enemy) # 刪除超出屏幕的敵機

# 碰撞檢測

for enemy in enemies[:]: # 遍歷所有敵機

enemy\_rect = pygame.Rect(enemy[0], enemy[1], enemy\_width, enemy\_height) # 創建敵機的碰撞矩形

player\_rect = pygame.Rect(player\_x - player\_width // 2, player\_y - player\_height // 2, player\_width, player\_height) # 創建玩家飛機的碰撞矩形

if enemy\_rect.colliderect(player\_rect): # 如果敵機與玩家飛機發生碰撞

enemies.remove(enemy) # 刪除該敵機

player\_health -= 1 # 玩家血量減少

if player\_health <= 0: # 如果玩家血量為0，結束遊戲

running = False

for bullet in bullets[:]: # 遍歷所有子彈

for enemy in enemies[:]: # 遍歷所有敵機

enemy\_rect = pygame.Rect(enemy[0], enemy[1], enemy\_width, enemy\_height) # 創建敵機的碰撞矩形

if enemy\_rect.collidepoint(bullet[0], bullet[1]): # 如果子彈與敵機發生碰撞

bullets.remove(bullet) # 刪除該子彈

enemies.remove(enemy) # 刪除該敵機

score += 1 # 分數增加

# 繪製子彈

for bullet in bullets: # 遍歷所有子彈

pygame.draw.rect(screen, white, [bullet[0], bullet[1], bullet\_width, bullet\_height]) # 繪製子彈，顏色為白色

# 繪製敵機

for enemy in enemies: # 遍歷所有敵機

pygame.draw.rect(screen, white, [enemy[0], enemy[1], enemy\_width, enemy\_height]) # 繪製敵機，顏色為白色

# 顯示分數

score\_text = font.render(f"Score: {score}", True, white) # 將分數渲染為文字

screen.blit(score\_text, [10, 10]) # 將分數顯示在屏幕左上角

# 顯示血量

health\_text = font.render(f"Health: {player\_health}", True, red) # 將血量渲染為紅色文字

screen.blit(health\_text, [screen\_width - 150, 10]) # 將血量顯示在屏幕右上角

# 計時器

current\_time = pygame.time.get\_ticks() # 獲取當前時間（從遊戲開始以來的毫秒數）

elapsed\_time = (current\_time - start\_time) // 1000 # 計算經過的秒數

timer\_text = font.render(f"Time: {elapsed\_time}s", True, white) # 將計時器渲染為文字

screen.blit(timer\_text, [screen\_width // 2 - 50, 10]) # 將計時器顯示在屏幕中央上方

# 更新屏幕

pygame.display.flip() # 更新整個螢幕顯示

# 控制遊戲幀率

clock.tick(60) # 控制每秒幀數為60

# 遊戲結束，退出 Pygame

pygame.quit()

import pygame

import math

import random

pygame.init()

screen = pygame.display.set\_mode((0, 0), pygame.FULLSCREEN)

screen\_width, screen\_height = screen.get\_size()

white = (255, 255, 255)

black = (0, 0, 0)

gray = (100, 100, 100)

red = (255, 0, 0)

player\_width = 40

player\_height = 20

player\_x = screen\_width // 2

player\_y = screen\_height // 2

player\_health = 5

player\_speed = 6

player\_speed\_boost = 2

bullet\_width = 10

bullet\_height = 5

bullet\_speed = 30

bullets = []

enemy\_width = 40

enemy\_height = 20

enemy\_speed = 2

enemies = []

spawn\_enemy\_event = pygame.USEREVENT + 1

pygame.time.set\_timer(spawn\_enemy\_event, 500)

background\_blocks = []

for \_ in range(20):

block\_x = random.randint(0, screen\_width)

block\_y = random.randint(0, screen\_height)

background\_blocks.append([block\_x, block\_y])

score = 0

start\_time = pygame.time.get\_ticks()

font = pygame.font.SysFont(None, 35)

running = True

clock = pygame.time.Clock()

shoot\_cooldown = 0

while running:

screen.fill(black)

for event in pygame.event.get():

if event.type == pygame.QUIT:

running = False

if event.type == spawn\_enemy\_event:

edge = random.choice(["top", "bottom", "left", "right"])

if edge == "top":

enemy\_x = random.randint(0, screen\_width)

enemy\_y = -enemy\_height

elif edge == "bottom":

enemy\_x = random.randint(0, screen\_width)

enemy\_y = screen\_height

elif edge == "left":

enemy\_x = -enemy\_width

enemy\_y = random.randint(0, screen\_height)

else:

enemy\_x = screen\_width

enemy\_y = random.randint(0, screen\_height)

enemies.append([enemy\_x, enemy\_y])

keys = pygame.key.get\_pressed()

speed\_boost = keys[pygame.K\_SPACE]

mouse\_x, mouse\_y = pygame.mouse.get\_pos()

angle = math.atan2(mouse\_y - player\_y, mouse\_x - player\_x)

move\_speed = player\_speed \* (2 if speed\_boost else 1)

for block in background\_blocks:

block[0] += -move\_speed \* math.cos(angle)

block[1] += -move\_speed \* math.sin(angle)

if block[0] < 0: block[0] += screen\_width

if block[0] > screen\_width: block[0] -= screen\_width

if block[1] < 0: block[1] += screen\_height

if block[1] > screen\_height: block[1] -= screen\_height

pygame.draw.rect(screen, gray, [block[0], block[1], 10, 10])

player\_points = [

(player\_x + player\_width \* math.cos(angle) / 2, player\_y + player\_width \* math.sin(angle) / 2),

(player\_x - player\_height \* math.sin(angle) / 2, player\_y + player\_height \* math.cos(angle) / 2),

(player\_x - player\_width \* math.cos(angle) / 2, player\_y - player\_width \* math.sin(angle) / 2),

(player\_x + player\_height \* math.sin(angle) / 2, player\_y - player\_height \* math.cos(angle) / 2)

]

pygame.draw.polygon(screen, white, player\_points)

shoot\_cooldown -= 1

if pygame.mouse.get\_pressed()[0] and shoot\_cooldown <= 0:

for i in range(-15, 15):

bullet\_angle = angle + math.radians(i \* 1.5)

bullet\_dx = bullet\_speed \* math.cos(bullet\_angle)

bullet\_dy = bullet\_speed \* math.sin(bullet\_angle)

bullets.append([player\_x, player\_y, bullet\_dx, bullet\_dy])

shoot\_cooldown = 10

for bullet in bullets[:]:

bullet[0] += bullet[2]

bullet[1] += bullet[3]

if bullet[0] < 0 or bullet[0] > screen\_width or bullet[1] < 0 or bullet[1] > screen\_height:

bullets.remove(bullet)

for enemy in enemies[:]:

enemy\_angle = math.atan2(player\_y - enemy[1], player\_x - enemy[0])

enemy[0] += enemy\_speed \* math.cos(enemy\_angle)

enemy[1] += enemy\_speed \* math.sin(enemy\_angle)

if enemy[0] < 0 or enemy[0] > screen\_width or enemy[1] < 0 or enemy[1] > screen\_height:

enemies.remove(enemy)

for enemy in enemies[:]:

enemy\_rect = pygame.Rect(enemy[0], enemy[1], enemy\_width, enemy\_height)

player\_rect = pygame.Rect(player\_x - player\_width // 2, player\_y - player\_height // 2, player\_width, player\_height)

if enemy\_rect.colliderect(player\_rect):

enemies.remove(enemy)

player\_health -= 1

if player\_health <= 0:

running = False

for bullet in bullets[:]:

for enemy in enemies[:]:

enemy\_rect = pygame.Rect(enemy[0], enemy[1], enemy\_width, enemy\_height)

if enemy\_rect.collidepoint(bullet[0], bullet[1]):

bullets.remove(bullet)

enemies.remove(enemy)

score += 1

for bullet in bullets:

pygame.draw.rect(screen, white, [bullet[0], bullet[1], bullet\_width, bullet\_height])

for enemy in enemies:

pygame.draw.rect(screen, white, [enemy[0], enemy[1], enemy\_width, enemy\_height])

score\_text = font.render(f"Score: {score}", True, white)

screen.blit(score\_text, [10, 10])

health\_text = font.render(f"Health: {player\_health}", True, red)

screen.blit(health\_text, [screen\_width - 150, 10])

current\_time = pygame.time.get\_ticks()

elapsed\_time = (current\_time - start\_time) // 1000

timer\_text = font.render(f"Time: {elapsed\_time}s", True, white)

screen.blit(timer\_text, [screen\_width // 2 - 50, 10])

pygame.display.flip()

clock.tick(60)

pygame.quit()