import pygame

import math

import random

def **calculate\_x\_velocity**(position, target, velocity):

direction = math.atan2(target[1] - position[1], target[0] - position[0])

return velocity \* math.cos(direction)

def **calculate\_y\_velocity**(position, target, velocity):

direction = math.atan2(target[1] - position[1], target[0] - position[0])

return velocity \* math.sin(direction)

class **PlayerInput**:

def **\_\_init\_\_**(self):

self.stop = False

self.left = False

self.right = False

self.fire = False

def **update**(self):

events = pygame.event.get()

for e in events:

if e.type == pygame.QUIT:

self.stop = True

elif e.type == pygame.KEYDOWN:

if e.key == pygame.K\_LEFT:

self.left = True

if e.key == pygame.K\_RIGHT:

self.right = True

elif e.type == pygame.KEYUP:

if e.key == pygame.K\_LEFT:

self.left = False

if e.key == pygame.K\_RIGHT:

self.right = False

self.fire = pygame.mouse.get\_pressed()[0]

class **Player**:

def **\_\_init\_\_**(self, city):

self.playerImage = pygame.image.load("player.png").convert\_alpha()

self.playerShotImage = pygame.image.load("shot.png").convert\_alpha()

self.crosshairImage = pygame.image.load("crosshair.png").convert\_alpha()

self.rect = self.playerImage.get\_rect(bottomleft=(0, window.get\_height() - city.get\_height()))

self.alive = True

self.shots = []

self.player\_has\_fired = False

def **move**(self, player\_input):

if player\_input.left and self.rect.left > window.get\_rect().left:

self.rect.x = self.rect.x - 2

if player\_input.right and self.rect.right < window.get\_rect().right:

self.rect.x = self.rect.x + 2

for shot in list(self.shots):

shot.move()

if not shot.rect.colliderect(window.get\_rect()):

self.shots.remove(shot)

if not player\_input.fire:

self.player\_has\_fired = False

def **hit**(self):

self.alive = False

class **Shot**:

def **\_\_init\_\_**(self, rect, x\_speed, y\_speed):

self.rect = rect

self.x = rect.x

self.y = rect.y

self.x\_speed = x\_speed

self.y\_speed = y\_speed

def **move**(self):

self.x = self.x + self.x\_speed

self.y = self.y + self.y\_speed

self.rect.x = self.x

self.rect.y = self.y

class **Alien**:

def **\_\_init\_\_**(self, speed\_x):

self.alienImage = pygame.image.load("alien.png").convert\_alpha()

self.alienShotImage = pygame.image.load("alien\_shot.png").convert\_alpha()

self.crash\_sound = pygame.mixer.Sound("Flashbang-Kibblesbob-899170896.wav")

self.rect = self.alienImage.get\_rect(topright=(0,0))

self.x = self.rect.x

self.speed\_x = speed\_x

self.bounds = window.get\_rect()

self.direction = 1

self.has\_been\_hit = False

self.alive = True

self.stone\_dead = False

self.time\_to\_shoot = random.randint(100, 1000)

def **move**(self):

self.x = self.x + self.direction \* self.speed\_x

self.rect.x = self.x

if self.rect.right > self.bounds.right:

self.direction = -1

if self.rect.left < self.bounds.left:

self.direction = 1

def **hit**(self):

self.alive = False

self.has\_been\_hit = True

def **maybe\_shoot**(self, player, shot\_list):

self.time\_to\_shoot = self.time\_to\_shoot - 1

if self.time\_to\_shoot <= 0 and player.alive and self.alive:

self.time\_to\_shoot = random.randint(100, 1000)

rect = self.alienShotImage.get\_rect(center=self.rect.midbottom)

speed = 1

x\_speed = calculate\_x\_velocity(self.rect.midbottom, player.rect.center, speed)

y\_speed = calculate\_y\_velocity(self.rect.midbottom, player.rect.center, speed)

shot = Shot(rect, x\_speed, y\_speed)

shot\_list.append(shot)

class **City**:

def **\_\_init\_\_**(self):

self.cityImage = pygame.image.load("grey\_city.png").convert\_alpha()

def **set\_rect**(self, rect):

self.rect = rect

class **Bomb**:

def **\_\_init\_\_**(self, pos, x\_speed, y\_speed):

self.enemyBomb = pygame.image.load("enemy\_bomb.png").convert\_alpha()

self.rect = self.enemyBomb.get\_rect(center=pos)

self.x = self.rect.x

self.y = self.rect.y

self.x\_speed = x\_speed

self.y\_speed = y\_speed

def **move**(self):

self.x = self.x + self.x\_speed

self.y = self.y + self.y\_speed

self.rect.x = self.x

self.rect.y = self.y

class **GameState**:

def **\_\_init\_\_**(self):

self.cities = []

self.bombs = []

self.player\_shots = []

self.alien\_shots = []

number\_of\_cities = 5

city\_distance = window.get\_width() // (number\_of\_cities + 1)

for city\_number in **range**(number\_of\_cities):

new\_city = City()

left = (city\_number + 1) \* city\_distance - new\_city.cityImage.get\_width() / 2

rect = new\_city.cityImage.get\_rect(bottomleft=(left, window.get\_height()))

new\_city.set\_rect(rect)

self.cities.append(new\_city)

self.player = Player(self.cities[0].cityImage)

self.alien = Alien(1.0)

self.player\_has\_fired = False

def **update**(self, player\_input):

self.player.move(player\_input)

if self.cities and random.randint(1, 1000) > 999:

x = random.randint(0, window.get\_width())

y = 0

pos = (x, y)

target = random.choice(self.cities).rect.midbottom

velocity = random.uniform(0.1, 0.5)

velocity\_x = calculate\_x\_velocity(pos, target, velocity)

velocity\_y = calculate\_y\_velocity(pos, target, velocity)

new\_bomb = Bomb(pos, velocity\_x, velocity\_y)

self.bombs.append(new\_bomb)

for bomb in self.bombs:

bomb.move()

self.alien.move()

self.alien.maybe\_shoot(self.player, self.alien\_shots)

for shot in list(self.player\_shots):

shot.move()

if not shot.rect.colliderect(window.get\_rect()):

self.player\_shots.remove(shot)

for shot in list(self.alien\_shots):

shot.move()

for shot in self.alien\_shots:

if shot.rect.colliderect(self.player.rect):

self.player.hit()

for shot in list(self.player\_shots):

if self.alien.rect.colliderect(shot.rect):

self.alien.hit()

for bomb in list(self.bombs):

for shot in self.player\_shots:

if bomb.rect.colliderect(shot.rect):

self.bombs.remove(bomb)

for bomb in list(self.bombs):

for city in list(self.cities):

if bomb.rect.colliderect(city.rect):

self.cities.remove(city)

self.bombs.remove(bomb)

if bomb.rect.colliderect(self.player.rect) and self.player.alive:

self.player.hit()

self.bombs.remove(bomb)

if player\_input.fire and not self.player\_has\_fired and **len**(self.player\_shots) < 3 and self.player.alive:

self.player\_has\_fired = True

self.add\_shot()

if not player\_input.fire:

self.player\_has\_fired = False

def **add\_shot**(self):

pos = self.player.rect.center

new\_shot\_rect = self.player.playerShotImage.get\_rect(center=pos)

target = pygame.mouse.get\_pos()

velocity = 1.5

velocity\_x = calculate\_x\_velocity(pos, target, velocity)

velocity\_y = calculate\_y\_velocity(pos, target, velocity)

new\_shot = Shot(new\_shot\_rect, velocity\_x, velocity\_y)

self.player\_shots.append(new\_shot)

def **paint\_screen**(window, start\_ticks):

ticks = pygame.time.get\_ticks()-start\_ticks

if ticks < 1000 and ticks > 0:

remainder = ticks % 3

if remainder == 0:

window.fill(black)

elif remainder == 1:

window.fill(white)

elif remainder == 2:

window.fill(lilla)

else:

window.fill(black)

if game\_state.player.alive:

window.blit(game\_state.player.playerImage, game\_state.player.rect)

if game\_state.alien.alive:

window.blit(game\_state.alien.alienImage, game\_state.alien.rect)

for shot in game\_state.player\_shots:

window.blit(game\_state.player.playerShotImage, shot.rect)

for city in game\_state.cities:

window.blit(city.cityImage, city.rect)

for bomb in game\_state.bombs:

window.blit(bomb.enemyBomb, bomb.rect)

for shot in game\_state.alien\_shots:

window.blit(game\_state.alien.alienShotImage, shot.rect)

window.blit(game\_state.player.crosshairImage,

game\_state.player.crosshairImage.get\_rect(center=pygame.mouse.get\_pos()))

pygame.display.flip()

def **main\_loop**():

pygame.mouse.set\_visible(False)

start\_ticks = 10000000

while not player\_input.stop:

pygame.time.delay(5)

player\_input.update()

game\_state.update(player\_input)

if game\_state.alien.has\_been\_hit and not game\_state.alien.stone\_dead:

pygame.mixer.Sound.play(game\_state.alien.crash\_sound)

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

game\_state.alien.stone\_dead = True

paint\_screen(window,start\_ticks)

pygame.display.quit()

pygame.quit()

pygame.init()

black = (0,0,0)

white = (255, 255, 255)

lilla = (255, 0, 255)

screen\_width = 800

screen\_height = 600

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

player\_input = PlayerInput()

game\_state = GameState()

main\_loop()