import pygame, random, sys  
from pygame.locals import \*  
  
WINDOWWIDTH = 1600 #Agrandir la taille en largeur de la fenêtre pour qu'elle match avec l'écran de l'ordinateur.  
WINDOWHEIGHT = 800 #Agrandir la taille en hauteur de la fenêtre pour qu'elle match avec l'écran de l'ordinateur.  
TEXTCOLOR = (0, 0, 0)  
BACKGROUNDCOLOR = (255, 255, 255)  
MENUBACKGROUNDCOLOR = ('tomato') #Couleur d'écran de start changée pour avoir quelque chose plus dans le thème  
FPS = 60 #Le jeu tourne plus vite avec 60 que 40 comme FPS  
BADDIEMINSIZE = 25 #Baddies ont été aggrandi, le code a été modifié en suivant les conseils du livre (Ai Swegart) Ch. 20, Pg. 353-354  
BADDIEMAXSIZE = 60 #La taille maximale des baddies a été augmentée.  
BADDIEMINSPEED = 1 #Vitesse minimale d'ennemi  
BADDIEMAXSPEED = 4 #Vitesse maximale d'ennemi a diminué  
ADDNEWBADDIERATE = 12 #Taux de reproduction de nouveaux ennemis, a été doublé.  
PLAYERMOVERATE = 5 #Nombre de pixel de déplacement pour chaque fois que l'on bouge le personnage.  
  
def terminate():  
 pygame.quit()  
 sys.exit()  
  
  
def waitForPlayerToPressKey(): #Mode de lancement du jeu et esc pour quitter la partie.  
 while True:  
 for event in pygame.event.get():  
 if event.type == QUIT:  
 terminate()  
 if event.type == KEYDOWN:  
 if event.key == K\_ESCAPE: # Pressing ESC quits.  
 terminate()  
 return  
  
  
def playerHasHitBaddie(playerRect, baddies):  
 for b in baddies:  
 if playerRect.colliderect(b['rect']):  
 return True  
 return False  
  
def playerHasHitElf(playerRect, elf):  
 for e in elf:  
 if playerRect.colliderect(e['rect']):  
 return True  
 return False  
  
def playerHasHitGift(playerRect, gift):  
 for g in gift:  
 if playerRect.colliderect(g['rect']):  
 return True  
 return False  
  
def playerHasHitLightining(playerRect, lightning):  
 for l in lightning:  
 if playerRect.colliderect(l['rect']):  
 return True  
 return False  
  
def playerHasHitChimney(playerRect, chimney):  
 for c in chimney:  
 if playerRect.colliderect(c['rect']):  
 return True  
 return False  
  
def playerHasHitCoal(playerRect, coal):  
 for co in coal:  
 if playerRect.colliderect(co['rect']):  
 return True  
 return False  
  
def drawText(text, font, surface, x, y):  
 textobj = font.render(text, 1, TEXTCOLOR)  
 textrect = textobj.get\_rect()  
 textrect.topleft = (x, y)  
 surface.blit(textobj, textrect)  
  
  
# Set up pygame, the window, and the mouse cursor.  
pygame.init()  
mainClock = pygame.time.Clock()  
windowSurface = pygame.display.set\_mode((WINDOWWIDTH, WINDOWHEIGHT))  
pygame.display.set\_caption('Dodger')  
pygame.mouse.set\_visible(False)  
  
  
# Set up the fonts.  
font = pygame.font.SysFont(None, 48)  
  
# Set up sounds.  
gameOverSound = pygame.mixer.Sound('grinch\_gameoversound.mp3')  
pygame.mixer.music.load('KatyPerry-CozyLittleChristmas.mp3')  
pygame.mixer.music.play(-1, 0.0)  
pygame.mixer.music.load('KatyPerry-CozyLittleChristmas.mp3')  
musicPlaying = True  
  
# Set up images.  
playerImage = pygame.image.load('santa-player.png')  
#playerSIZE = pygame.transform.scale(playerImage, (100, 200)) #todo custom-set player Size  
playerRect = playerImage.get\_rect()  
baddieImage = pygame.image.load('gremlin.png')  
  
  
# Set up backgrounds image.  
gameBackground1 = pygame.image.load("background\_snow.png")  
gameOverBackground = pygame.image.load("Grinch end game.png")  
gameBackground2=pygame.image.load("backgroundimagel2.png")  
  
# Show the "Start" screen.  
windowSurface.fill(MENUBACKGROUNDCOLOR)  
drawText('X-Mas Dodger', font, windowSurface, (WINDOWWIDTH / 3), (WINDOWHEIGHT / 3))  
drawText('Press a key to start', font, windowSurface, (WINDOWWIDTH / 3)-40, (WINDOWHEIGHT / 3) + 50)  
drawText('saving Christmas', font, windowSurface, (WINDOWWIDTH / 3)-35, (WINDOWHEIGHT / 3) + 100)  
pygame.display.update()  
waitForPlayerToPressKey()  
  
  
  
  
  
class GameLevel():  
 def \_\_init\_\_(self):  
 self.level="level1"  
  
  
 def level1(self):  
 baddies = []  
 score = 0  
 playerRect.topleft = (WINDOWWIDTH / 2, WINDOWHEIGHT - 50)  
 moveLeft = moveRight = moveUp = moveDown = False  
 reverseCheat = slowCheat = False  
 baddieAddCounter = 0  
 pygame.mixer.music.play(-1, 0.0)  
  
 topScore = 0  
 while True: # The game loop runs while the game part is playing.  
  
 score += 1 # Increase score.  
 for event in pygame.event.get():  
 if event.type == QUIT:  
 terminate()  
 if event.type == KEYDOWN:  
 if event.key == K\_z:  
 reverseCheat = True  
 if event.key == K\_x:  
 slowCheat = True  
 if event.key == K\_LEFT or event.key == K\_a:  
 moveRight = False  
 moveLeft = True  
 if event.key == K\_RIGHT or event.key == K\_d:  
 moveLeft = False  
 moveRight = True  
 if event.key == K\_UP or event.key == K\_w:  
 moveDown = False  
 moveUp = True  
 if event.key == K\_DOWN or event.key == K\_s:  
 moveUp = False  
 moveDown = True  
 # option mute pour enlever le son du jeu. Par contre le son du Game Over reste toujours  
 if event.key == K\_m:  
 if musicPlaying:  
 pygame.mixer.music.stop()  
 else:  
 pygame.mixer.music.play(-1, 0.0) # quand la musique arrive à la fin, elle recommence  
 musicPlaying = not musicPlaying # le code a ete adapte depuis le livre de cours (Ai Swegart) Ch. 19 Page 325-326  
  
 if event.type == KEYUP:  
 if event.key == K\_z:  
 reverseCheat = False  
 score = 0  
 if event.key == K\_x:  
 slowCheat = False  
 score = 0  
 if event.key == K\_ESCAPE:  
 terminate()  
  
 if event.key == K\_LEFT or event.key == K\_a:  
 moveLeft = False  
 if event.key == K\_RIGHT or event.key == K\_d:  
 moveRight = False  
 if event.key == K\_UP or event.key == K\_w:  
 moveUp = False  
 if event.key == K\_DOWN or event.key == K\_s:  
 moveDown = False  
  
 if event.type == MOUSEMOTION:  
 # If the mouse moves, move the player where to the cursor.  
 playerRect.centerx = event.pos[0]  
 playerRect.centery = event.pos[1]  
 # Add new baddies at the top of the screen, if needed.  
 if not reverseCheat and not slowCheat:  
 baddieAddCounter += 1  
 if baddieAddCounter == ADDNEWBADDIERATE:  
 baddieAddCounter = 0  
 baddieSize = random.randint(BADDIEMINSIZE, BADDIEMAXSIZE)  
 newBaddie = {  
 'rect': pygame.Rect(random.randint(0, WINDOWWIDTH - baddieSize), 0 - baddieSize, baddieSize,  
 baddieSize),  
 'speed': random.randint(BADDIEMINSPEED, BADDIEMAXSPEED),  
 'surface': pygame.transform.scale(baddieImage, (baddieSize, baddieSize)),  
 }  
  
 baddies.append(newBaddie)  
  
 # Move the player around.  
 if moveLeft and playerRect.left > 0:  
 playerRect.move\_ip(-1 \* PLAYERMOVERATE, 0)  
 if moveRight and playerRect.right < WINDOWWIDTH:  
 playerRect.move\_ip(PLAYERMOVERATE, 0)  
 if moveUp and playerRect.top > 0:  
 playerRect.move\_ip(0, -1 \* PLAYERMOVERATE)  
 if moveDown and playerRect.bottom < WINDOWHEIGHT:  
 playerRect.move\_ip(0, PLAYERMOVERATE)  
  
 # Move the baddies down.  
 for b in baddies:  
 if not reverseCheat and not slowCheat:  
 b['rect'].move\_ip(0, b['speed'])  
 elif reverseCheat:  
 b['rect'].move\_ip(0, -5)  
 elif slowCheat:  
 b['rect'].move\_ip(0, 1)  
  
 # Delete baddies that have fallen past the bottom.  
 for b in baddies[:]:  
 if b['rect'].top > WINDOWHEIGHT:  
 baddies.remove(b)  
  
 # Draw the game world on the window.  
 windowSurface.fill(BACKGROUNDCOLOR)  
 # add the background image  
 windowSurface.blit(gameBackground1, (0, 0))  
  
 # Draw the score and top score.  
 drawText('Score: %s' % (score), font, windowSurface, 10, 0)  
 drawText('Top Score: %s' % (topScore), font, windowSurface, 10, 40)  
  
 # Draw the player's rectangle.  
 windowSurface.blit(playerImage, playerRect)  
  
 # Draw each baddie.  
 for b in baddies:  
 windowSurface.blit(b['surface'], b['rect'])  
  
 pygame.display.update()  
  
 # Check if any of the baddies have hit the player.  
 if playerHasHitBaddie(playerRect, baddies):  
 if score > topScore:  
 topScore = score  
 # set new top score and inform the player  
 break  
 # Background game over set up  
 windowSurface.fill(BACKGROUNDCOLOR)  
 # add the background image  
 windowSurface.blit(gameOverBackground, (0, 0))  
 self.level="level2"  
  
 def level2(self):  
 baddies = []  
 score = 0  
 playerRect.topleft = (WINDOWWIDTH / 2, WINDOWHEIGHT - 50)  
 moveLeft = moveRight = moveUp = moveDown = False  
 reverseCheat = slowCheat = False  
 baddieAddCounter = 0  
 pygame.mixer.music.play(-1, 0.0)  
  
 topScore = 0  
 while True: # The game loop runs while the game part is playing.  
  
 score += 1 # Increase score.  
 for event in pygame.event.get():  
 if event.type == QUIT:  
 terminate()  
 if event.type == KEYDOWN:  
 if event.key == K\_z:  
 reverseCheat = True  
 if event.key == K\_x:  
 slowCheat = True  
 if event.key == K\_LEFT or event.key == K\_a:  
 moveRight = False  
 moveLeft = True  
 if event.key == K\_RIGHT or event.key == K\_d:  
 moveLeft = False  
 moveRight = True  
 if event.key == K\_UP or event.key == K\_w:  
 moveDown = False  
 moveUp = True  
 if event.key == K\_DOWN or event.key == K\_s:  
 moveUp = False  
 moveDown = True  
 # option mute pour enlever le son du jeu. Par contre le son du Game Over reste toujours  
 if event.key == K\_m:  
 if musicPlaying:  
 pygame.mixer.music.stop()  
 else:  
 pygame.mixer.music.play(-1, 0.0) # quand la musique arrive à la fin, elle recommence  
 musicPlaying = not musicPlaying # le code a ete adapte depuis le livre de cours (Ai Swegart) Ch. 19 Page 325-326  
  
 if event.type == KEYUP:  
 if event.key == K\_z:  
 reverseCheat = False  
 score = 0  
 if event.key == K\_x:  
 slowCheat = False  
 score = 0  
 if event.key == K\_ESCAPE:  
 terminate()  
  
 if event.key == K\_LEFT or event.key == K\_a:  
 moveLeft = False  
 if event.key == K\_RIGHT or event.key == K\_d:  
 moveRight = False  
 if event.key == K\_UP or event.key == K\_w:  
 moveUp = False  
 if event.key == K\_DOWN or event.key == K\_s:  
 moveDown = False  
  
 #for e in pygame.event.get():  
 # if e.type == QUIT: raise SystemExit, "QUIT"  
 # if e.type == KEYDOWN and e.key == K\_ESCAPE:  
 # raise SystemExit, "ESCAPE"  
  
#pressed = pygame.key.get\_pressed()  
#up, left, right = [pressed[key] for key in (K\_UP, K\_LEFT, K\_RIGHT)]  
  
 if event.type == MOUSEMOTION:  
 # If the mouse moves, move the player where to the cursor.  
 playerRect.centerx = event.pos[0]  
 playerRect.centery = event.pos[1]  
 # Add new baddies at the top of the screen, if needed.  
 if not reverseCheat and not slowCheat:  
 baddieAddCounter += 1  
 if baddieAddCounter == ADDNEWBADDIERATE:  
 baddieAddCounter = 0  
 baddieSize = random.randint(BADDIEMINSIZE, BADDIEMAXSIZE)  
 newBaddie = {  
 'rect': pygame.Rect(random.randint(0, WINDOWWIDTH - baddieSize), 0 - baddieSize, baddieSize,  
 baddieSize),  
 'speed': random.randint(BADDIEMINSPEED, BADDIEMAXSPEED),  
 'surface': pygame.transform.scale(baddieImage, (baddieSize, baddieSize)),  
 }  
  
 baddies.append(newBaddie)  
  
 # Move the player around.  
 if moveLeft and playerRect.left > 0:  
 playerRect.move\_ip(-1 \* PLAYERMOVERATE, 0)  
 if moveRight and playerRect.right < WINDOWWIDTH:  
 playerRect.move\_ip(PLAYERMOVERATE, 0)  
 if moveUp and playerRect.top > 0:  
 playerRect.move\_ip(0, -1 \* PLAYERMOVERATE)  
 if moveDown and playerRect.bottom < WINDOWHEIGHT:  
 playerRect.move\_ip(0, PLAYERMOVERATE)  
  
 # Move the baddies down.  
 for b in baddies:  
 if not reverseCheat and not slowCheat:  
 b['rect'].move\_ip(0, b['speed'])  
 elif reverseCheat:  
 b['rect'].move\_ip(0, -5)  
 elif slowCheat:  
 b['rect'].move\_ip(0, 1)  
  
 # Delete baddies that have fallen past the bottom.  
 for b in baddies[:]:  
 if b['rect'].top > WINDOWHEIGHT:  
 baddies.remove(b)  
  
 # Draw the game world on the window.  
 windowSurface.fill(BACKGROUNDCOLOR)  
 # add the background image  
 windowSurface.blit(gameBackground2, (0, 0))  
  
 # Draw the score and top score.  
 drawText('Score: %s' % (score), font, windowSurface, 10, 0)  
 drawText('Top Score: %s' % (topScore), font, windowSurface, 10, 40)  
  
 # Draw the player's rectangle.  
 windowSurface.blit(playerImage, playerRect)  
  
 # Draw each baddie.  
 for b in baddies:  
 windowSurface.blit(b['surface'], b['rect'])  
  
 pygame.display.update()  
  
 # Check if any of the baddies have hit the player.  
 if playerHasHitBaddie(playerRect, baddies):  
 if score > topScore:  
 topScore = score  
 # set new top score and inform the player  
 break  
 # Background game over set up  
 windowSurface.fill(BACKGROUNDCOLOR)  
 # add the background image  
 windowSurface.blit(gameOverBackground, (0, 0))  
  
 def level\_manager(self):  
 if self.level== "level1":  
 self.level1()  
 if self.level=="level2":  
 self.level2()  
  
  
  
#set up level  
game\_level= GameLevel()  
  
  
  
  
while True:  
 game\_level.level\_manager()  
 # Set up the start of the game.  
 mainClock.tick(FPS)  
  
 # Stop the game and show the "Game Over" screen.  
 pygame.mixer.music.stop()  
 gameOverSound.play()  
  
  
  
 drawText('GAME OVER', font, windowSurface, (WINDOWWIDTH / 3), (WINDOWHEIGHT / 3))  
 drawText('Press a key to retry', font, windowSurface, (WINDOWWIDTH / 3) - 45, (WINDOWHEIGHT / 3) + 50)  
 drawText('to save Christmas', font, windowSurface, (WINDOWWIDTH / 3) - 45,(WINDOWHEIGHT / 3) + 100)  
 pygame.display.update()  
 waitForPlayerToPressKey()  
  
 gameOverSound.stop()