# Create Performance Task

import pygame

import random

import intersects

pygame.init()

# Window settings

WIDTH = 1100

HEIGHT = 800

TITLE = "Buzz Lightyear: Attack of the Zurgs"

FPS = 60

# Make the window

screen = pygame.display.set\_mode([WIDTH, HEIGHT])

pygame.display.set\_caption(TITLE)

clock = pygame.time.Clock()

# Colors

BLACK = (0, 0, 0)

WHITE = (255, 255, 255)

RED = (175, 0, 0)

YELLOW = (255, 255, 0)

#Stages

START = 0

PLAYING = 1

END = 2

PAUSE = 3

# Images

space2\_img = pygame.image.load("img/space.png")

astronaut\_img = pygame.image.load("img/astronaut.png")

ground\_img = pygame.image.load("img/ground.jpg")

coin\_img = pygame.image.load("img/coin.png")

enemy\_img = pygame.image.load("img/enemy.png")

laser\_img = pygame.image.load("img/laser.png")

button\_img = pygame.image.load("img/button.png")

laserbeam\_img = pygame.image.load("img/laserbeam.png")

meme1\_img = pygame.image.load("img/meme1.jpg")

# Transforms images to desired size

space2\_img = pygame.transform.scale(space2\_img, [WIDTH, 700])

astronaut\_img = pygame.transform.scale(astronaut\_img, [60, 85])

ground\_img = pygame.transform.scale(ground\_img, [WIDTH, 100])

coin\_img = pygame.transform.scale(coin\_img, [50, 50])

enemy\_img = pygame.transform.scale(enemy\_img, [70, 85])

laser\_img = pygame.transform.scale(laser\_img, [50, 35])

button\_img = pygame.transform.scale(button\_img, [40, 40])

laserbeam\_img = pygame.transform.scale(laserbeam\_img, [(WIDTH - 50), 40])

meme1\_img= pygame.transform.scale(meme1\_img, [230, 250])

# Physics

H\_SPEED = 6

JUMP\_POWER = 12

GRAVITY = 0.4

# Fonts

FONT\_LG = pygame.font.Font(None, 60)

FONT\_SM = pygame.font.Font(None, 30)

score = 0

lives = 3

class SpaceMan():

def \_\_init\_\_(self, x, y, img):

self.x = x

self.y = y

self.img = img

self.w = self.img.get\_width()

self.h = self.img.get\_height()

self.vx = 0

self.vy = 0

def get\_rect(self):

return [self.x, self.y, self.w, self.h]

def jump(self, ground, platforms):

can\_jump = False

self.y += 1

if intersects.rect\_rect(self.get\_rect(), ground.get\_rect()):

can\_jump = True

spaceman\_rect = self.get\_rect()

for p in platforms:

platform\_rect = p.get\_rect()

if intersects.rect\_rect(spaceman\_rect, platform\_rect):

can\_jump = True

if can\_jump:

self.vy = -JUMP\_POWER

self.y -= 1

def move(self, vx):

self.vx = vx

def stop(self):

self.vx = 0

def apply\_gravity(self):

self.vy += GRAVITY

def move\_and\_process\_platforms(self, platforms):

self.x += self.vx

spaceman\_rect = self.get\_rect()

for p in platforms:

platform\_rect = p.get\_rect()

if intersects.rect\_rect(spaceman\_rect, platform\_rect):

if self.vx > 0:

self.x = p.x - self.w

elif self.vx < 0:

self.x = p.x + p.w

self.y += self.vy

spaceman\_rect = self.get\_rect()

for p in platforms:

platform\_rect = p.get\_rect()

if intersects.rect\_rect(spaceman\_rect, platform\_rect):

if self.vy > 0:

self.y = p.y - self.h

if self.vy < 0:

self.y = p.y + p.h

self.vy = 0

def check\_screen\_edges(self):

if self.x < 0:

self.x = 0

elif self.x + self.w > WIDTH:

self.x = WIDTH - self.w

def check\_ground(self):

if self.y + self.h > ground.y:

self.y = ground.y - self.h

self.vy = 0

def process\_coins(self, coins):

global score

spaceman\_rect = self.get\_rect()

coins\_to\_remove = []

for c in coins:

coin\_rect = c.get\_rect()

if intersects.rect\_rect(spaceman\_rect, coin\_rect):

coins\_to\_remove.append(c)

score += 1

for c in coins\_to\_remove:

coins.remove(c)

def process\_enemies(self, enemies):

spaceman\_rect = self.get\_rect()

global lives

for p in enemies:

enemy\_rect = p.get\_rect()

if intersects.rect\_rect(spaceman\_rect, enemy\_rect):

print("skadoosh")

lives -= 1

self.x = 0

self.y = 615

def process\_button(self, buttons, laserbeams):

hit = False

spaceman\_rect = self.get\_rect()

for b in buttons:

button\_rect = b.get\_rect()

if intersects.rect\_rect(spaceman\_rect, button\_rect):

self.x = 0

self.y = 615

hit = True

if hit:

for l in laserbeams:

l.shoot()

def update(self, ground, platforms, coins, enemies, buttons):

self.apply\_gravity()

self.move\_and\_process\_platforms(platforms)

self.check\_screen\_edges()

self.check\_ground()

self.process\_coins(coins)

self.process\_enemies(enemies)

self.process\_button(buttons, laserbeams)

def draw(self):

screen.blit(self.img, [self.x, self.y])

class Meme1():

def \_\_init\_\_(self, x, y, img):

self.x = x

self.y = y

self.img = img

self.w = self.img.get\_width()

self.h = self.img.get\_height()

def get\_rect(self):

return [self.x, self.y, self.w, self.h]

def update(self):

pass

def draw(self):

screen.blit(self.img, [self.x, self.y])

class Ground():

def \_\_init\_\_(self, x, y, img):

self.x = x

self.y = y

self.img = img

self.w = self.img.get\_width()

self.h = self.img.get\_height()

def get\_rect(self):

return [self.x, self.y, self.w, self.h]

def draw(self):

screen.blit(self.img, [self.x, self.y])

class Planet():

def \_\_init\_\_(self, x, y, img):

self.x = x

self.y = y

self.img = img

self.w = self.img.get\_width()

self.h = self.img.get\_height()

def get\_rect(self):

return [self.x, self.y, self.w, self.h]

def update(self):

pass

def draw(self):

screen.blit(self.img, [self.x, self.y])

class Stars():

def \_\_init\_\_(self, num\_stars):

self.stars = []

for i in range(num\_stars):

x = random.randrange(0, WIDTH)

y = random.randrange(0, HEIGHT)

r = random.randrange(1, 3)

self.stars.append([x, y, r])

def draw(self):

for s in self.stars:

pygame.draw.circle(screen, WHITE, [s[0], s[1]], s[2])

class Platform():

def \_\_init\_\_(self, x, y, w, h):

self.x = x

self.y = y

self.w = w

self.h = h

def get\_rect(self):

return [self.x, self.y, self.w, self.h]

def draw(self):

pygame.draw.rect(screen, RED, [self.x, self.y, self.w, self.h])

class Coin():

def \_\_init\_\_(self, x, y, img):

self.x = x

self.y = y

self.img = img

self.w = self.img.get\_width()

self.h = self.img.get\_height()

self.vx = 3

self.vy = 0

def get\_rect(self):

return [self.x, self.y, self.w, self.h]

def update(self):

pass

def draw(self):

screen.blit(self.img, [self.x, self.y])

class Space():

def \_\_init\_\_(self, x, y, img):

self.x = x

self.y = y

self.img = img

self.w = self.img.get\_width()

self.h = self.img.get\_height()

def get\_rect(self):

return [self.x, self.y, self.w, self.h]

def update(self):

pass

def draw(self):

screen.blit(self.img, [self.x, self.y])

class Enemy():

def \_\_init\_\_(self, x, y, img):

self.x = x

self.y = y

self.img = img

self.w = self.img.get\_width()

self.h = self.img.get\_height()

self.vx = 5

self.vy = 0

def get\_rect(self):

return [self.x, self.y, self.w, self.h]

def move(self):

self.x += self.vx

def check\_screen\_edges(self):

if self.x < 0:

self.x = 0

self.vx \*= -1

elif self.x + self.w > WIDTH:

self.x = WIDTH - self.w

self.vx \*= -1

def die(self):

self.x = 1200

self.y = 1000

self.vx = 0

def process\_laserbeam(self, enemies, laserbeams):

enemy\_lives = 1

enemies\_rect = self.get\_rect()

for n in laserbeams:

laserbeams\_rect = n.get\_rect()

if n.on() and intersects.rect\_rect(enemies\_rect, laserbeams\_rect):

self.die()

def update(self):

self.move()

self.check\_screen\_edges()

self.process\_laserbeam(enemies, laserbeams)

def draw(self):

screen.blit(self.img, [self.x, self.y])

class Laser():

def \_\_init\_\_(self, x, y, img):

self.x = x

self.y = y

self.img = img

self.w = self.img.get\_width()

self.h = self.img.get\_height()

def get\_rect(self):

return [self.x, self.y, self.w, self.h]

def update(self):

pass

def draw(self):

screen.blit(self.img, [self.x, self.y])

class Button():

def \_\_init\_\_(self, x, y, img):

self.x = x

self.y = y

self.img = img

self.w = self.img.get\_width()

self.h = self.img.get\_height()

def get\_rect(self):

return [self.x, self.y, self.w, self.h]

def update(self):

pass

def draw(self):

screen.blit(self.img, [self.x, self.y])

class Laserbeam():

def \_\_init\_\_(self, x, y, img):

self.x = x

self.y = y

self.img = img

self.w = self.img.get\_width()

self.h = self.img.get\_height()

self.ticks = 0

def get\_rect(self):

return [self.x, self.y, self.w, self.h]

def shoot(self):

self.ticks = 60

def update(self):

if self.ticks > 0:

self.ticks-= 1

def on(self):

return self.ticks > 0

def draw(self):

if self.on():

screen.blit(self.img, [self.x, self.y])

# Make game objects

space2 = Space(0, 0, space2\_img)

player = SpaceMan(0, 0, astronaut\_img)

ground = Ground(0, 700, ground\_img)

meme1 = Meme1(400, 10, meme1\_img)

b1 = Button(1000, 75, button\_img)

b2 = Button(1400, 75, button\_img)

buttons = [b1, b2]

s1 = Laser(1050, 305, laser\_img)

s2 = Laser(1050, 430, laser\_img)

s3 = Laser(1050, 555, laser\_img)

s4 = Laser(1050, 180, laser\_img)

lasers = [s1, s2, s3, s4]

c1 = Coin(800, 475, coin\_img)

c2 = Coin(920, 325, coin\_img)

c3 = Coin(400, 475, coin\_img)

c4 = Coin(600, 325, coin\_img)

c5 = Coin(265, 325, coin\_img)

c6 = Coin(500, 200, coin\_img)

c7 = Coin(730, 200, coin\_img)

c8 = Coin(630, 470, coin\_img)

c9 = Coin(895, 615, coin\_img)

c10 = Coin(170, 200, coin\_img)

c11 = Coin(80, 470, coin\_img)

c12 = Coin(350, 45, coin\_img)

c13 = Coin(690, 70, coin\_img)

c14 = Coin(950, 200, coin\_img)

coins = [c1, c2, c3, c4, c5, c6, c7, c8, c9, c10, c11, c12, c13, c14]

p1 = Platform(790, 550, 70, 10)

p2 = Platform(880, 400, 70, 10)

p3 = Platform(400, 550, 70, 10)

p4 = Platform(600, 400, 70, 10)

p5 = Platform(265, 400, 70, 10)

p6 = Platform(500, 275, 70, 10)

p7 = Platform(730, 275, 70, 10)

p8 = Platform(630, 535, 70, 10)

p9 = Platform(895, 690, 70, 10)

p10 = Platform(170, 275, 70, 10)

p11 = Platform(80, 545, 70, 10)

p12 = Platform(350, 120, 70, 10)

p13 = Platform(690, 145, 70, 10)

p14 = Platform(950, 275, 70, 10)

platforms = [p1, p2, p3, p4, p5, p6, p7, p8, p9, p10, p11, p12, p13, p14]

n1 = Enemy(990, 275, enemy\_img)

n2 = Enemy(990, 400, enemy\_img)

n3 = Enemy(990, 525, enemy\_img)

n4 = Enemy(990, 150, enemy\_img)

enemies = [n1, n2, n3, n4]

m1 = Laserbeam(0, 295, laserbeam\_img)

m2 = Laserbeam(0, 420, laserbeam\_img)

m3 = Laserbeam(0, 545, laserbeam\_img)

m4 = Laserbeam(0, 170, laserbeam\_img)

laserbeams = [m1, m2, m3, m4]

def setup():

global player, stage

player = SpaceMan(0, 0, astronaut\_img)

stage = START

# game loop

setup()

done = False

seconds = 30

time = seconds \* 60

while not done:

# event handling

for event in pygame.event.get():

if event.type == pygame.QUIT:

done=True

elif event.type == pygame.KEYDOWN:

if stage == START:

if event.key == pygame.K\_SPACE:

stage = PLAYING

elif stage == END:

if event.key == pygame.K\_SPACE:

setup()

elif stage == PLAYING:

if event.key == pygame.K\_p:

stage = PAUSE

elif event.key == pygame.K\_UP:

player.jump(ground, platforms)

elif stage == PAUSE:

if event.key == pygame.K\_p:

stage = PLAYING

if stage == PLAYING:

pressed = pygame.key.get\_pressed()

if pressed[pygame.K\_RIGHT]:

player.move(H\_SPEED)

elif pressed[pygame.K\_LEFT]:

player.move(-H\_SPEED)

else:

player.stop()

for e in enemies:

e.update()

# game logic

for l in laserbeams:

l.update()

if stage == PLAYING:

time -= 1

player.update(ground, platforms, coins, enemies, buttons)

if time == 0 and score < 14:

done = True

if score == 14:

stage = END

print("Congratulations! You Win!")

if lives == 0:

stage = END

if stage == PLAYING:

space2.draw()

ground.draw()

player.draw()

for b in buttons:

b.draw()

for n in enemies:

n.draw()

for p in platforms:

p.draw()

for c in coins:

c.draw()

for s in lasers:

s.draw()

for m in laserbeams:

m.draw()

SCORE\_NUMBER = FONT\_SM.render(str(score), True, YELLOW)

SCORE\_TEXT = FONT\_SM.render("Coins: ", True, YELLOW)

TIMER\_TEXT = FONT\_SM.render("Time Left: ", True, WHITE)

TIMER = FONT\_SM.render(str(time/60), True, WHITE)

LIVES = FONT\_SM.render(str(lives), True, WHITE)

LIVES\_TEXT = FONT\_SM.render("Lives: ", True, WHITE)

LASER = FONT\_SM.render("Click Here For Laser:", True, RED)

TITLE = FONT\_LG.render("BUZZ LIGHTYEAR I: ATTACK OF THE ZURGS", True, YELLOW)

SPACE = FONT\_SM.render("Press Space To Start", True, WHITE)

DEFEAT = FONT\_SM.render("You have defeated the Zurgs!", True, YELLOW)

LINE1 = FONT\_SM.render("Every time you hit a Zurg, you lose a life, but if you ", True, WHITE)

LINE2 = FONT\_SM.render("hit the button that activates the laser, you kill the Zurgs.", True, WHITE)

LINE3 = FONT\_SM.render("You have 30 seconds to collect all the coins.", True, WHITE)

LINE4 = FONT\_LG.render("GOOD LUCK!", True, YELLOW)

PAUSE\_MSG = FONT\_LG.render("The game is paused. Press p to play.", True, WHITE)

TO\_PAUSE = FONT\_SM.render("Press p to pause.", True, YELLOW)

END\_MSG = FONT\_LG.render("The game is finished.", True, YELLOW)

LOSE = FONT\_SM.render("You lose.", True, YELLOW)

WIN = FONT\_SM.render("You win.", True, YELLOW)

if stage == PLAYING:

screen.blit(SCORE\_TEXT, [10, 10])

screen.blit(SCORE\_NUMBER, [90, 10])

screen.blit(TIMER\_TEXT, [950, 10])

screen.blit(TIMER, [1050, 10])

screen.blit(LIVES, [75, 30])

screen.blit(LIVES\_TEXT, [10, 30])

screen.blit(LASER, [780, 85])

screen.blit(TO\_PAUSE, [900, 780])

elif stage == START:

screen.fill(BLACK)

screen.blit(TITLE, [100, 300])

screen.blit(SPACE, [395, 350])

screen.blit(LINE1, [260, 390])

screen.blit(LINE2, [260, 410])

screen.blit(LINE3, [280, 430])

screen.blit(LINE4, [390, 460])

meme1.draw()

elif stage == PAUSE:

screen.blit(PAUSE\_MSG, [200, 430])

elif stage == END:

screen.blit(END\_MSG, [200, 430])

if lives == 0:

screen.blit(LOSE, [200, 500])

elif lives > 0:

screen.blit(WIN, [200, 570])

# update screen

pygame.display.update()

clock.tick(FPS)

# close window on quit

pygame.quit ()