import pygame

import random

colors = [

(0, 0, 0),

(120, 37, 179),

(100, 179, 179),

(80, 34, 22),

(80, 134, 22),

(180, 34, 22),

(180, 34, 122),

]

class Figure:

x = 0

y = 0

figures = [

[[1, 5, 9, 13], [4, 5, 6, 7]],

[[4, 5, 9, 10], [2, 6, 5, 9]],

[[6, 7, 9, 10], [1, 5, 6, 10]],

[[1, 2, 5, 9], [0, 4, 5, 6], [1, 5, 9, 8], [4, 5, 6, 10]],

[[1, 2, 6, 10], [5, 6, 7, 9], [2, 6, 10, 11], [3, 5, 6, 7]],

[[1, 4, 5, 6], [1, 4, 5, 9], [4, 5, 6, 9], [1, 5, 6, 9]],

[[1, 2, 5, 6]],

]

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

self.x = x

self.y = y

self.type = random.randint(0, len(self.figures) - 1)

self.color = random.randint(1, len(colors) - 1)

self.rotation = 0

def image(self):

return self.figures[self.type][self.rotation]

def rotate(self):

self.rotation = (self.rotation + 1) % len(self.figures[self.type])

class Tetris:

def \_\_init\_\_(self, height, width):

self.level = 2

self.score = 0

self.state = "start"

self.field = []

self.height = 0

self.width = 0

self.x = 100

self.y = 60

self.zoom = 20

self.figure = None

self.height = height

self.width = width

self.field = []

self.score = 0

self.state = "start"

for i in range(height):

new\_line = []

for j in range(width):

new\_line.append(0)

self.field.append(new\_line)

def new\_figure(self):

self.figure = Figure(3, 0)

def intersects(self):

intersection = False

for i in range(4):

for j in range(4):

if i \* 4 + j in self.figure.image():

if i + self.figure.y > self.height - 1 or \

j + self.figure.x > self.width - 1 or \

j + self.figure.x < 0 or \

self.field[i + self.figure.y][j + self.figure.x] > 0:

intersection = True

return intersection

def break\_lines(self):

lines = 0

for i in range(1, self.height):

zeros = 0

for j in range(self.width):

if self.field[i][j] == 0:

zeros += 1

if zeros == 0:

lines += 1

for i1 in range(i, 1, -1):

for j in range(self.width):

self.field[i1][j] = self.field[i1 - 1][j]

self.score += lines \*\* 2

def go\_space(self):

while not self.intersects():

self.figure.y += 1

self.figure.y -= 1

self.freeze()

def go\_down(self):

self.figure.y += 1

if self.intersects():

self.figure.y -= 1

self.freeze()

def freeze(self):

for i in range(4):

for j in range(4):

if i \* 4 + j in self.figure.image():

self.field[i + self.figure.y][j + self.figure.x] = self.figure.color

self.break\_lines()

self.new\_figure()

if self.intersects():

self.state = "gameover"

def go\_side(self, dx):

old\_x = self.figure.x

self.figure.x += dx

if self.intersects():

self.figure.x = old\_x

def rotate(self):

old\_rotation = self.figure.rotation

self.figure.rotate()

if self.intersects():

self.figure.rotation = old\_rotation

# Initialize the game engine

pygame.init()

# Define some colors

BLACK = (0, 0, 0)

WHITE = (255, 255, 255)

GRAY = (128, 128, 128)

size = (400, 500)

screen = pygame.display.set\_mode(size)

pygame.display.set\_caption("Tetris")

# Loop until the user clicks the close button.

done = False

clock = pygame.time.Clock()

fps = 25

game = Tetris(20, 10)

counter = 0

pressing\_down = False

while not done:

if game.figure is None:

game.new\_figure()

counter += 1

if counter > 100000:

counter = 0

if counter % (fps // game.level // 2) == 0 or pressing\_down:

if game.state == "start":

game.go\_down()

for event in pygame.event.get():

if event.type == pygame.QUIT:

done = True

if event.type == pygame.KEYDOWN:

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

game.rotate()

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

pressing\_down = True

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

game.go\_side(-1)

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

game.go\_side(1)

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

game.go\_space()

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

game.\_\_init\_\_(20, 10)

if event.type == pygame.KEYUP:

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

pressing\_down = False

screen.fill(BLACK)

for i in range(game.height):

for j in range(game.width):

pygame.draw.rect(screen, WHITE, [game.x + game.zoom \* j, game.y + game.zoom \* i, game.zoom, game.zoom], 1)

if game.field[i][j] > 0:

pygame.draw.rect(screen, colors[game.field[i][j]],

[game.x + game.zoom \* j + 1, game.y + game.zoom \* i + 1, game.zoom - 2, game.zoom - 1])

if game.figure is not None:

for i in range(4):

for j in range(4):

p = i \* 4 + j

if p in game.figure.image():

pygame.draw.rect(screen, colors[game.figure.color],

[game.x + game.zoom \* (j + game.figure.x) + 1,

game.y + game.zoom \* (i + game.figure.y) + 1,

game.zoom - 2, game.zoom - 2])

font = pygame.font.SysFont('Calibri', 25, True, False)

font1 = pygame.font.SysFont('Calibri', 65, True, False)

text = font.render("Score: " + str(game.score), True, WHITE)

text\_game\_over = font1.render("Game Over", True, (255, 125, 0))

text\_game\_over1 = font1.render("Press ESC", True, (255, 215, 0))

screen.blit(text, [0, 0])

if game.state == "gameover":

screen.blit(text\_game\_over, [20, 200])

screen.blit(text\_game\_over1, [25, 265])

pygame.display.flip()

clock.tick(fps)

pygame.quit()