import pygame

import random

# Inisialisasi pygame

pygame.init()

# Ukuran layar dan grid

WIDTH, HEIGHT = 300, 600

BLOCK\_SIZE = 30

COLS = WIDTH // BLOCK\_SIZE

ROWS = HEIGHT // BLOCK\_SIZE

# Warna

BLACK = (0, 0, 0)

GRAY = (128, 128, 128)

COLORS = [

(0, 255, 255), # I

(255, 255, 0), # O

(128, 0, 128), # T

(0, 255, 0), # S

(255, 0, 0), # Z

(0, 0, 255), # J

(255, 165, 0), # L

]

# Bentuk‑bentuk tetromino (matriks 2D)

SHAPES = [

[[1, 1, 1, 1]], # I

[[1, 1],

[1, 1]], # O

[[0, 1, 0],

[1, 1, 1]], # T

[[1, 1, 0],

[0, 1, 1]], # S

[[0, 1, 1],

[1, 1, 0]], # Z

[[1, 0, 0],

[1, 1, 1]], # J

[[0, 0, 1],

[1, 1, 1]], # L

]

# Fungsi rotasi (searah jarum jam)

def rotate\_shape(shape):

# transpose + reverse baris

return [list(row) for row in zip(\*shape[::-1])]

# Cek tabrakan

def check\_collision(grid, shape, offset\_x, offset\_y):

for y, row in enumerate(shape):

for x, cell in enumerate(row):

if cell:

gx = x + offset\_x

gy = y + offset\_y

# jika di luar grid

if gx < 0 or gx >= COLS or gy >= ROWS:

return True

if gy >= 0 and grid[gy][gx]:

return True

return False

# Menggabungkan tetromino ke grid

def merge\_shape(grid, shape, offset\_x, offset\_y, color):

for y, row in enumerate(shape):

for x, cell in enumerate(row):

if cell:

gx = x + offset\_x

gy = y + offset\_y

if 0 <= gy < ROWS and 0 <= gx < COLS:

grid[gy][gx] = color

# Menghapus baris penuh

def clear\_rows(grid):

new\_grid = [row for row in grid if any(cell == 0 for cell in row)]

cleared = ROWS - len(new\_grid)

# Tambahkan baris kosong di atas

for \_ in range(cleared):

new\_grid.insert(0, [0]\*COLS)

return new\_grid, cleared

# Setup layar

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

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

def draw\_grid(grid):

for y in range(ROWS):

for x in range(COLS):

color = grid[y][x]

if color:

pygame.draw.rect(screen, color,

(x\*BLOCK\_SIZE, y\*BLOCK\_SIZE, BLOCK\_SIZE, BLOCK\_SIZE))

# garis pinggir

pygame.draw.rect(screen, GRAY,

(x\*BLOCK\_SIZE, y\*BLOCK\_SIZE, BLOCK\_SIZE, BLOCK\_SIZE), 1)

def draw\_shape(shape, offset\_x, offset\_y, color):

for y, row in enumerate(shape):

for x, cell in enumerate(row):

if cell:

px = (offset\_x + x) \* BLOCK\_SIZE

py = (offset\_y + y) \* BLOCK\_SIZE

pygame.draw.rect(screen, color,

(px, py, BLOCK\_SIZE, BLOCK\_SIZE))

pygame.draw.rect(screen, GRAY,

(px, py, BLOCK\_SIZE, BLOCK\_SIZE), 1)

def tetris\_game():

grid = [[0]\*COLS for \_ in range(ROWS)]

current\_shape = random.choice(SHAPES)

current\_color = random.choice(COLORS)

shape\_x = COLS // 2 - len(current\_shape[0]) // 2

shape\_y = -2 # mulai agak di atas grid

clock = pygame.time.Clock()

fall\_time = 0

fall\_speed = 0.5 # kecepatan jatuh (detik)

score = 0

running = True

while running:

dt = clock.tick(30) / 1000 # waktu per frame (detik)

fall\_time += dt

# Input

for event in pygame.event.get():

if event.type == pygame.QUIT:

running = False

elif event.type == pygame.KEYDOWN:

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

if not check\_collision(grid, current\_shape, shape\_x - 1, shape\_y):

shape\_x -= 1

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

if not check\_collision(grid, current\_shape, shape\_x + 1, shape\_y):

shape\_x += 1

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

if not check\_collision(grid, current\_shape, shape\_x, shape\_y + 1):

shape\_y += 1

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

# rotate

new\_shape = rotate\_shape(current\_shape)

if not check\_collision(grid, new\_shape, shape\_x, shape\_y):

current\_shape = new\_shape

# jatuh otomatis

if fall\_time > fall\_speed:

fall\_time = 0

if not check\_collision(grid, current\_shape, shape\_x, shape\_y + 1):

shape\_y += 1

else:

# gabungkan ke grid

merge\_shape(grid, current\_shape, shape\_x, shape\_y, current\_color)

# clear rows jika bisa

grid, cleared = clear\_rows(grid)

score += cleared \* 10

# buat tetromino baru

current\_shape = random.choice(SHAPES)

current\_color = random.choice(COLORS)

shape\_x = COLS // 2 - len(current\_shape[0]) // 2

shape\_y = -2

# jika tumpukan sudah tinggi → game over

if check\_collision(grid, current\_shape, shape\_x, shape\_y):

print("Game Over! Skor:", score)

running = False

# Gambar

screen.fill(BLACK)

draw\_grid(grid)

draw\_shape(current\_shape, shape\_x, shape\_y, current\_color)

pygame.display.update()

pygame.quit()

if \_\_name\_\_ == "\_\_main\_\_":

tetris\_game()