A. Algoritma Game

1. Memulai program
2. Mengimpor pygame
3. Mengimpor random
4. Menginisiasi pygame.font.init()
5. Menginisiasi s_width = 800
6. Menginisiasi s_height = 650
7. Menginisiasi play_width = 300
8. Menginisiasi play_height = 600
9. Menginisiasi block_size = 30
10. Menginisiasi top_left_x = (s_width - play_width) // 2
11. Menginisiasi top_left_y = s_height - play_height
12. Membuat list format bentuk S, Z, I, O, J, L, T
13. Menginisiasi warna masing-masing bentuk
14. Memanggil fungsi kelas Piece(object):
    - 14.1 Mendefinisikan fungsi __init__(self, x, y, shape):
        - 14.1.1 Menginisiasi self.x = x
        - 14.1.2 Menginisiasi self.y = y
        - 14.1.3 Menginisiasi self.shape = shape
        - 14.1.4 Menginisiasi self.color = shape_colors[shapes.index(shape)]
        - 14.1.5 Menginisiasi self.rotation = 0
15. Mendefinisikan fungsi create_grid(locked_pos={}):
    - 15.1 Menginisiasi grid = [[(0,0,0) for _ in range(10)] for _ in range(20)]
    - 15.2 Untuk i pada range(len(grid)):
        - 15.2.1 Untuk j pada range(len(grid[i])):
            - 15.2.1.1 Jika (j,i) pada locked_pos:
                - 15.2.1.1.1 Menginisiasi c = locked_pos[(j,i)]
                - 15.2.1.1.2 Menginisiasi grid[i][j] = c
    - 15.3 Mengembalikan fungsi grid
16. Mendefinisikan convert_shape_format(shape):
    - 16.1 Menginisiasi positions = []
    - 16.2 Menginisiasi format = shape.shape[shape.rotation % len(shape.shape)]

16.3 Untuk i, line pada enumerate(format):
16.3.1 Menginisiasi row = list(line)
16.3.2 Untuk j, column pada enumerate(row):
16.3.2.1 Jika column == '0':
16.3.2.1.1 Menginisiasi positions.append((shape.x + j, shape.y + i))
16.4 Untuk i, pos pada enumerate(positions):
16.4.1 Menginisiasi positions[i] = (pos[0] - 2, pos[1] - 4)
16.5 Mengembalikan fungsi positions
17. Mendefinisikan valid_space(shape, grid):
17.1 Menginisiasi accepted_pos = [[(j,i) for j in range(10) if grid[i][j] == (0,0,0)] for i in range(20)]
17.2 Menginisiasi accepted_pos = [j for sub in accepted_pos for j in sub]
17.3 Menginisiasi formatted = convert_shape_format(shape)
17.4 Untuk pos in formatted:
17.4.1 Jika pos tidak pada accepted_pos:
17.4.1.1 Jika pos[1] > -1:
17.4.1.1.1 Mengembalikan kondisi False
17.5 Mengembalikan kondisi True
18. Mendefinisikan check_lost(positions):
18.1 Untuk pos pada positions:
18.1.1 Menginisiasi x,y = pos
18.1.2 Jika y < 1:
18.1.2.1 Mengembalikan kondisi True
18.2 Mengembalikan kondisi False
19. Mendefinisikan get_shape():
19.1 Mengembalikan fungsi Piece(5, 0, random.choice(shapes))
20. Mendefinisikan fungsi draw_text_top(text, size, color, surface):
20.1 Menginisiasi font = pygame.font.Font('font/Insanibc.ttf', size)
20.2 Menginisiasi label = font.render(text, 1, color)
20.3 Menginisiasi surface.blit(label, (top_left_x + play_width/3.5 - (label.get_width() / 3), top_left_y + play_height/3 - label.get_height()/3))

21. Mendefinisikan fungsi draw_text_middle(surface,text, size, color):
    21.1 Menginisiasi font = pygame.font.SysFont("font/Poppins-Medium.otf", size, bold = True
    21.2 Menginisiasi label = font.render(text, 1, color)
    21.3 Menginisiasi surface.blit(label, (top_left_x + play_width/1.9 - (label.get_width()/2), top_left_y + play_height/2 - label.get_height()/2))
22. Mendefinisikan fungsi draw_grid(surface, grid):
    22.1 Menginisiasi sx = top_left_x
    22.2 Menginisiasi sy = top_left_y
    22.3 Untuk i pada range(len(grid)):
        22.3.1 Menginisiasi pygame.draw.line(surface, (255,255,255), (sx, sy + i*block_size), (sx+play_width, sy+ i*block_size))
        22.3.2 Untuk j pada range(len(grid[i])):
            22.3.2.1 Menginisiasi pygame.draw.line(surface, (255,255,255), (sx + j*block_size, sy), (sx+ j*block_size, sy+ play_height))
23. Mendefinisikan fungsi clear_rows(grid, locked):
    23.1 Menginisiasi inc = 0
    23.2 Untuk i pada range(len(grid)-1, -1, -1):
        23.2.1 Menginisiasi row = grid[i]
        23.2.2 Jika (0,0,0) tidak pada row:
            23.2.2.1 Menginisiasi inc += 1
            23.2.2.2 Menginisiasi ind = i
            23.2.2.3 Untuk j pada range(len(row)):
                23.2.2.3.1 Menginisiasi try:
                    23.2.2.3.1.1 Menginisiasi del locked[(j,i)]
                23.2.2.3.2 Menginisiasi except:
                    23.2.2.3.2.1 Menginisasi continue
    23.3 Jika inc < 0:
        23.3.1 Untuk key pada sorted(list(locked), key = lambda x: x[1]) [::-1]:
            23.3.1.1 Menginisiasi x, y = key
            23.3.1.2 Jika y < ind:
                23.3.1.2.1 Menginisiasi newKey = (x,y + inc)

23.3.1.2.2 Menginisiasi locked[newKey] = locked.pop(key)

23.4 Mengembalikan fungsi inc

24. Mendefinisikan draw_next_shape(shape, surface):
    24.1 Menginisiasi font = pygame.font.Font('font/Poppins-Medium.otf', 20)
    24.2 Menginisiasi label = font.render('Next Shape', 1, (255,250,88))
    24.3 Menghitung sx = top_left_x + play_width + 50
    24.4 Menghitung sy = top_left_y + play_height/2 -100
    24.5 Menginisiasi format = shape.shape[shape.rotation % len(shape.shape)]
    24.6 Untuk i, line pada enumerate(format):
        24.6.1 Menginisiasi row = list(line)
        24.6.2 Untuk j, column pada enumerate(row):
            24.6.2.1 Jika column == '0':
                24.6.2.1.1 Memanggil fungsi dengan mendeklarasikan pygame.draw.rect(surface,shape.color, (sx + j*block_size, sy + i*block_size, block_size, block_size), 0)
    24.7 Menginisiasi surface.blit(label, (sx+-40, sy+-30))
25. Mendefinisikan draw_window(surface, grid, score):
    25.1 Menginisiasi surface.fill((0,0,0))
    25.2 Menginisiasi pygame.font.init()
    25.3 Menginisiasi font = pygame.font.Font(font/Insanibc.ttf', 30)
    25.4 Menginisiasi label = font.render("TETRIS GAME", 1, (255,250,88))
    25.5 Menginisiasi surface.blit(label, (top_left_x + play_width/2 - (label.get_width()/2), 10))
    25.6 Menginisiasi font = pygame.font.Font('font/Poppins-Bold.otf', 45, bold = True)
    25.7 Menginisiasi label = font.render('Score ' + str(score), 1, (222,255,88))
    25.8 Menghitung sx = top_left_x + play_width + 25
    25.9 Menghitung sy = top_left_y + play_height/2 -50
    25.10 Menginisiasi surface.blit(label, (sx + 10, sy + 160))
    25.11 Untuk i pada range(len(grid)):

25.11.1 Untuk j pada range(len(grid[i])):

25.11.1.1 Menginisiasi pygame.draw.rect(surface, grid[i][j], (top_left_x + j*block_size, top_left_y + i*block_size, block_size, block_size), 0)

25.12 Menginisiasi pygame.draw.rect(surface, (255,255,255), (top_left_x, top_left_y, play_width, play_height), 5)

25.13 Menginisiasi draw_grid(surface, grid)

26. Mendefinisikan main(win):

26.1 Menginisiasi locked_positions = {}

26.2 Menginisiasi grid = create_grid(locked_positions)

26.3 Mengondisikan change_piece = False

26.4 Mengondisikan run = True

26.5 Menginisiasi current_piece = get_shape()

26.6 Menginisiasi next_piece = get_shape()

26.7 Menginisiasi clock = pygame.time.Clock()

26.8 Menginisiasi fall_time = 0

26.9 Menginisiasi fall_speed = 0.45

26.10 Menginisiasi level_time = 0

26.11 Menginisiasi score = 0

26.11.1 Membuat while run:

26.11.1 Menginisiasi grid = create_grid(locked_positions)

26.11.2 Menginisiasi fall_time += clock.get_rawtime()

26.11.3 Menginisiasi level_time += clock.get_rawtime()

26.11.4 Menginisiasi clock.tick()

26.11.5 Jika level_time/1000 > 5:

26.11.5.1 Menginisiasi level_time = 0

26.11.5.2 Jika fall_speed > 0.12:

26.11.5.2.1 Menginisiasi fall_speed -= 0.005

26.11.6 Jika fall_time/1000 > fall_speed:

26.11.6.1 Menginisiasi fall_time = 0

26.11.6.2 Menginisiasi current_piece.y += 1

26.11.6.3 Jika tidak (valid_space(current_piece, grid)) dan current_piece.y > 0:

26.11.6.4 Menginisiasi current_piece.y -= 1

26.11.6.5 Mengondisikan change_piece = True

26.11.7 Untuk event pada pygame.event.get():

26.11.7.1 Jika event.type == pygame.QUIT:

26.11.7.1.1 Mengondisikan run = False

26.11.7.2 Jika event.type == pygame. KEYDOWN:

26.11.7.2.1 Jika event.key == pygame.K_LEFT:

26.11.7.2.1.1 Menginisiasi current_piece.x -= 1

26.11.7.2.1.2 Jika tidak (valid_space(current_piece, grid)):

26.11.7.2.1.2.1 Menginisiasi current_piece.x += 1

26.11.7.2.2 Jika event.key == pygame.K_RIGHT:

26.11.7.2.2.1 Menginisiasi current_piece.x += 1

26.11.7.2.2.2 Jika tidak (valid_space(current_piece, grid)):

26.11.7.2.2.2.1 Menginisiasi current_piece.x -= 1

26.11.7.2.3 Jika event.key == pygame.K_DOWN:

26.11.7.2.3.1 Menginisiasi current_piece.y += 1

26.11.7.2.3.2 Jika tidak (valid_space(current_piece, grid)):

26.11.7.2.3.2.1 Menginisiasi current_piece.y -= 1

26.11.7.2.4 Jika event.key == pygame.K_UP:

26.11.7.2.4.1 Menginisiasi current_piece.rotation += 1

26.11.7.2.4.2 Jika tidak (valid_space(current_piece, grid)):

26.11.7.2.4.2.1 Menginisiasi current_piece.rotation -= 1

26.11.7.2.5 Jika event.key == pygame.K_SPACE:

26.11.7.2.5.1 Membuat while valid_space(current_piece, grid):

26.11.7.2.5.1.1 Menginisiasi current_piece.y += 1

26.11.7.2.5.2 Menginisiasi current_piece.y -= 1

26.11.7.2.5.3 Mencetak (convert_shape_format(current_piece))

26.11.8 Menginisiasi shape_pos = convert_shape_format(current_piece)

26.11.9 Untuk i pada range(len(shape_pos)):

26.11.9.1 Menginisiasi x,y = shape_pos[i]

26.11.9.2 Jika y > -1:

26.11.9.2.1 Menginisiasi grid[y][x] = current_piece.color

26.11.10 Jika change_piece:

26.11.10.1 Untuk pos pada shape_pos:

26.11.10.1.1 Menginisiasi p = (pos[0], pos[1])

26.11.10.1.2 Menginisiasi locked_positions[p] = current_piece.color

26.11.10.2 Menginisiasi current_piece = next_piece

26.11.10.3 Menginisiasi next_piece = get_shape()

26.11.10.4 Menginisiasi change_piece = False

26.11.10.5 Menginisiasi score += clear_rows(grid, locked_positions) * 10

26.11.11 Menginisiasi draw_window(win, grid, score)

26.11.12 Menginisiasi draw_next_shape(next_piece, win)

26.11.13 Menginisiasi pygame.display.update()

26.11.14 Jika check_lost(locked_positions):

26.11.14.1 Menginisiasi draw_text_middle(win, "YOU LOSE!", 40, (255,255,255))

26.11.14.2 Menginisiasi pygame.display.update()

26.11.14.3 Menginisiasi pygame.time.delay(3000)

26.11.14.4 Mengondisikan run = False

27. Mendefinisikan main_menu(win):

27.1 Mengondisikan run = True

27.2 Membuat while run:

27.2.1 Menginisiasi win.fill((0,0,0))

27.2.2 Menginisiasi draw_text_top('TETRIS GAME’, 80, (203, 253, 65), win)

27.2.3 Menginisiasi draw_text_middle(win, 'Press Any Key To Play!', 30, (203, 253, 65))

27.2.4 Menginisiasi pygame.display.update()

27.2.5 Untuk event pada pygame.event.get():

27.2.5.1 Jika event.type == pygame.QUIT:

27.2.5.1.1 Mengondisikan run = False

27.2.5.2 Jika event.type == pygame.KEYDOWN:

27.2.5.2.1 Menginisiasi main(win)

27.3 Menginisiasi pygame.display.quit()

28. Menginisiasi win = pygame.display.set_mode((s_width, s_height))
29. Menginisiasi pygame.display.set_caption('TETRIS GAME')
30. Menginisiasi main_menu(win)
31. Mengakhiri Program

B. Algoritma scoring

26.11.10 Untuk i pada range(len(shape_pos)):

26.11.10.1 Menginisiasi x,y = shape_pos[i]

26.11.10.2 Jika y > -1:

26.11.10.2.1 Menginisiasi grid[y][x] = current_piece.color

26.11.15 Jika change_piece:

26.11.15.1 Untuk pos pada shape_pos:

26.11.15.1.1 Menginisiasi p = (pos[0], pos[1])

26.11.15.1.2 Menginisiasi locked_positions[p] = current_piece.color

26.11.15.2 Menginisiasi current_piece = next_piece

26.11.15.3 Menginisiasi next_piece = get_shape()

26.11.15.4 Menginisiasi change_piece = False

26.11.15.5 Menginisiasi score += clear_rows(grid, locked_positions) * 10

26.11.16 Menginisiasi draw_window(win, grid, score)

26.11.17 Menginisiasi draw_next_shape(next_piece, win)

26.11.18 Menginisiasi pygame.display.update()

26.11.19 Jika check_lost(locked_positions):

- 26.11.19.1 Menginisiasi draw_text_middle(win, "YOU LOSE!", 40, (255,0,0))
- 26.11.19.2 Menginisiasi pygame.display.update()
- 26.11.19.3 Menginisiasi pygame.time.delay(1500)
- 26.11.19.4 Mengondisikan run = False