LAPORAN TUGAS
PEMROGRAMAN BERORIENTASI OBJECT
INHERITANCE - ENCAPSULATION

Dosen Pengampu:

I Gde Agung Sri Sidhimantra, S.Kom., M.Kom
Moch Deny Pratama, S.Tr.Kom., M.Kom
Dimas Novian Aditia Syahputra, S.Tr.T., M.Tr.T
Binti Kholifah, S.Kom., M.Tr.Kom

Nama:
Muhammad Dafa Alvin Zuhdi (23091397083)

Mata Kuliah Pemrograman Berorientasi Objek
Program Studi D4 Manajemen Informatika
Fakultas Vokasi
Universitas Negeri Surabaya

Penjelasan Kode

Source code lengkap: [Github]

1. Menginstall library game dari python

```
PS D:\Lectures\Semester 3\Pemrograman Berorientasi Object\Kode> pip install pygame
Collecting pygame
  Obtaining dependency information for pygame from https://files.pythonhosted.org/packages/d2/55/ca3eb851aeef4f6f2e98a360c201f0d00bd1ba2eb98e2c7850d80aabc526/pygame-2.6.1-cp311-cp311-win_amd64.whl.metadata
  Downloading pygame-2.6.1-cp311-cp311-win_amd64.whl.metadata (13 kB)
Downloading pygame-2.6.1-cp311-cp311-win_amd64.whl (10.6 MB)
   ---------------------------------------- 10.6/10.6 MB 1.7 MB/s eta 0:00:00
Installing collected packages: pygame
Successfully installed pygame-2.6.1

[notice] A new release of pip is available: 23.2.1 -> 24.3.1
[notice] To update, run: python.exe -m pip install --upgrade pip
PS D:\Lectures\Semester 3\Pemrograman Berorientasi Object\Kode>
```

Perintah pip install pygame digunakan untuk menginstal pustaka **Pygame** pada Python. Dengan menjalankan perintah ini di terminal atau command prompt, Python akan mengunduh dan memasang versi terbaru Pygame dari Python Package Index (PyPI), sehingga pustaka ini siap digunakan untuk membuat game di Python.

2. Import Library

```
import pygame
import sys
import random
import time
```

Kode	Penjelasan
`pygame`	Digunakan untuk membuat tampilan grafis permainan.
`sys`	Digunakan untuk keluar dari permainan.
`random`	Digunakan untuk mengatur posisi apel secara acak.
`time`	Digunakan untuk menunda waktu saat permainan berakhir.

3. Inisialisasi Pygame

```
pygame.init()
```

Memulai semua modul Pygame yang diperlukan.

4. Kelas `GameObject`

```
class GameObject:
    def __init__(self, color, position):
        self.color = color
        self.position = position

    def draw(self, game_window):
        pygame.draw.rect(game_window, self.color, pygame.Rect(self.position[0], self.position[1], 10, 10))
```

Kelas dasar untuk objek dalam permainan yang memiliki atribut color dan position. Metode draw digunakan untuk menggambar objek pada jendela permainan.

5. Kelas `Snake` (Turunan dari `GameObject`)

```
class Snake(GameObject):
    def __init__(self, position):
        super().__init__(pygame.Color(0, 255, 0), position)
        self.body = [list(position), [position[0] - 10, position[1]], [position[0] - 20, position[1]]]
        self.direction = 'RIGHT'
        self.change_to = self.direction
```

Kode	Penjelasan
`class Snake(GameObject):`	Mendeklarasikan kelas Snake, yang merupakan turunan (subclass) dari kelas GameObject. Artinya, Snake mewarisi semua atribut dan metode dari GameObject, serta bisa memiliki atribut dan metode tambahan sendiri.
`def __init__(self, position):`	konstruktor kelas Snake, yang merupakan metode khusus __init__ yang akan dipanggil ketika objek Snake dibuat. Metode ini menerima parameter position, yang merupakan posisi awal kepala ular di game.
`super().__init__(pygame.Color(0, 255, 0), position)`	memanggil konstruktor kelas GameObject menggunakan super(). Konstruksi ini meneruskan warna (RGB pygame.Color(0, 255, 0), yang merupakan hijau untuk ular) dan posisi awal position sebagai parameter ke konstruktor GameObject.
`self.body = [list(position), [position[0] - 10, position[1]], [position[0] - 20, position[1]]]`	Atribut self.body adalah daftar yang menyimpan koordinat posisi tubuh ular. • self.body[0]: posisi kepala ular, dimulai dari position. • self.body[1] dan self.body[2]: dua bagian tubuh ular yang terletak tepat di belakang kepala, dengan jarak 10 piksel ke kiri, membentuk ular dengan panjang tiga segmen.

`self.direction = 'RIGHT'`	self.direction adalah atribut yang menentukan arah pergerakan ular saat ini. Di sini, arah awal ular diatur ke 'RIGHT', sehingga ketika game dimulai, ular akan bergerak ke kanan.
`self.change_to = self.direction`	digunakan untuk menyimpan arah baru jika ada perubahan arah dari pemain, dan dimulai dengan nilai arah awal, yaitu 'RIGHT'. Atribut ini berguna agar ular hanya dapat mengubah arah pada langkah berikutnya, bukan langsung di frame saat tombol ditekan.

6. Method **`change_direction`** pada **`class Snake`**

```
def change_direction(self, direction):
	if direction == 'UP' and self.direction != 'DOWN':
		self.change_to = 'UP'
	if direction == 'DOWN' and self.direction != 'UP':
		self.change_to = 'DOWN'
	if direction == 'LEFT' and self.direction != 'RIGHT':
		self.change_to = 'LEFT'
	if direction == 'RIGHT' and self.direction != 'LEFT':
		self.change_to = 'RIGHT'
```

Metode **`change_direction`** digunakan untuk mengatur arah pergerakan ular sesuai dengan input dari pemain. Namun, terdapat aturan tambahan yang mencegah ular untuk langsung berbalik arah ke belakang. Misalnya, jika ular sedang bergerak ke kanan, ia tidak bisa langsung berbalik ke kiri, untuk menghindari tabrakan langsung dengan tubuhnya sendiri.

7. Method **`move`** pada **`Class Snake`**

```
def move(self):
	self.direction = self.change_to
	if self.direction == 'UP':
		self.position[1] -= 10
	if self.direction == 'DOWN':
		self.position[1] += 10
	if self.direction == 'LEFT':
		self.position[0] -= 10
	if self.direction == 'RIGHT':
		self.position[0] += 10
	self.body.insert(0, list(self.position))
```

Metode **`move`** digunakan untuk mengatur pergerakan ular dengan mengubah posisinya berdasarkan arah yang ditentukan.

8. Method **shrink** pada **Class Snake**

```
    def shrink(self):
        if len(self.body) > 1:
            self.body.pop()
```

Method **shrink** digunakan untuk mengurangi panjang tubuh ular dengan menghapus segmen terakhir dari self.body.

9. Method draw pada Class Snake:

```
    def draw(self, game_window):
            for pos in self.body:
                    pygame.draw.rect(game_window, self.color, pygame.Rect(pos[0], pos[1], 10, 10))
```

Method **draw** digunakan untuk menggambar ular pada jendela permainan (game_window) menggunakan setiap posisi segmen tubuh dalam self.body.

10. Method check_collision pada Class Snake:

```
def check_collision(self, frame_size_x, frame_size_y):
        if self.position[0] < 0 or self.position[0] > frame_size_x - 10 or self.position[1] < 0 or self.position[1] > frame_size_y - 10:
            return True
        for block in self.body[1:]:
            if self.position[0] == block[0] and self.position[1] == block[1]:
                return True
        return False
```

Method **check_collison** digunakan untuk memeriksa apakah ular mengalami tabrakan, baik dengan dinding maupun dengan tubuhnya sendiri.

11. Kelas **Apple** (Turunan dari **GameObject**)

```
class Apple(GameObject):
    def __init__(self, frame_size_x, frame_size_y, color):
        position = [random.randrange(1, (frame_size_x // 10)) * 10, random.randrange(1, (frame_size_y // 10)) * 10]
        super().__init__(color, position)

    def respawn(self, frame_size_x, frame_size_y):
        self.position = [random.randrange(1, (frame_size_x // 10)) * 10, random.randrange(1, (frame_size_y // 10)) * 10]
```

Kelas Apple adalah turunan dari GameObject yang digunakan untuk menghasilkan apel dalam game Snake pada posisi acak dan memungkinkan apel muncul kembali setelah dimakan oleh ular. Konstruktor __init__ menerima parameter frame_size_x, frame_size_y, dan color untuk menginisialisasi warna dan posisi apel secara acak dalam jendela permainan. Posisi apel diatur menggunakan koordinat acak yang selalu kelipatan 10 agar sejajar dengan grid permainan, yang penting agar apel selalu berada di jalur

gerakan ular. Metode respawn memungkinkan apel untuk berpindah ke posisi baru ketika dimakan, dengan menghasilkan koordinat acak baru dalam batas area permainan.Hal ini menciptakan tantangan bagi pemain untuk terus mengejar apel yang berpindah-pindah di lokasi acak dalam area permainan.

12. Inisialisasi Kelas `Game`

```
class Game:
    def __init__(self):
        self.frame_size_x = 720
        self.frame_size_y = 480
        self.game_window = pygame.display.set_mode((self.frame_size_x, self.frame_size_y))
        pygame.display.set_caption('Snake Game')
        self.fps_controller = pygame.time.Clock()
        self.snake = Snake([100, 50])
        self.apple1 = Apple(self.frame_size_x, self.frame_size_y, pygame.Color(255, 0, 0))  # Apple merah
        self.apple2 = Apple(self.frame_size_x, self.frame_size_y, pygame.Color(0, 0, 255))  # Apple biru
        self.score = 0
```

Metode ini adalah konstruktor yang menginisialisasi berbagai atribut penting dalam game. Di sini, ukuran jendela permainan ditentukan dengan frame_size_x dan frame_size_y, serta jendela permainan dibuat menggunakan pygame.display.set_mode. fps_controller digunakan untuk mengatur kecepatan permainan, sementara atribut snake, apple1, dan apple2 adalah objek Snake dan Apple yang mewakili ular serta dua apel dalam game (merah dan biru). Skor awal ditetapkan menjadi 0.

13. Method `game_over` pada kelas `Game`

```
 def game_over(self):
        my_font = pygame.font.SysFont('Arial', 90)
        game_over_surface = my_font.render('YOU DIED', True, pygame.Color(255, 0, 0))
        game_over_rect = game_over_surface.get_rect()
        game_over_rect.midtop = (self.frame_size_x / 2, self.frame_size_y / 4)
        self.game_window.fill(pygame.Color(0,  0, 0))
        self.game_window.blit(game_over_surface, game_over_rect)
        pygame.display.flip()
        time.sleep(3)
        pygame.quit()
        sys.exit()
```

Method `game_over` digunakan untuk menampilkan layar akhir permainan ketika ular bertabrakan dengan dinding atau tubuhnya sendiri. Tulisan "YOU DIED" ditampilkan di tengah layar dengan font besar berwarna merah. Setelah itu, layar ditampilkan selama 3 detik sebelum game berhenti dan program keluar.

14. Method `show_screen` pada kelas `Game`

```
def show_screen(self):
       score_font = pygame.font.SysFont('Arial', 20)
       score_surface = score_font.render('Score : ' + str(self.score), True, pygame.Color(0, 0, 0))
       score_rect = score_surface.get_rect()
       score_rect.midtop = (72, 15)
```

```
        self.game_window.blit(score_surface, score_rect)
```

Bertugas menampilkan skor di layar selama permainan berlangsung. Menggunakan font kecil dan warna hitam, skor pemain ditampilkan di sudut kiri atas jendela permainan, memperbarui informasi skor setiap kali pemain memakan apel.

15. Method `run` pada kelas `Game`

```
def run(self):
        while True:
            for event in pygame.event.get():
                if event.type == pygame.QUIT:
                    pygame.quit()
                    sys.exit()
                elif event.type == pygame.KEYDOWN:
                    if event.key == pygame.K_UP:
                        self.snake.change_direction('UP')
                    if event.key == pygame.K_DOWN:
                        self.snake.change_direction('DOWN')
                    if event.key == pygame.K_LEFT:
                        self.snake.change_direction('LEFT')
                    if event.key == pygame.K_RIGHT:
                        self.snake.change_direction('RIGHT')
                    if event.key == pygame.K_ESCAPE:
                        pygame.event.post(pygame.event.Event(pygame.QUIT))

            self.snake.move()

            if self.snake.position == self.apple1.position or self.snake.position == self.apple2.position:
                self.score += 1
                if self.snake.position == self.apple1.position:
                    self.apple1.respawn(self.frame_size_x, self.frame_size_y)
                else:
                    self.apple2.respawn(self.frame_size_x, self.frame_size_y)
            else:
                self.snake.shrink()

            self.game_window.fill(pygame.Color(255, 255, 255))

            self.snake.draw(self.game_window)
            self.apple1.draw(self.game_window)
            self.apple2.draw(self.game_window)
            self.show_screen()

           if self.snake.check_collision(self.frame_size_x, self.frame_size_y):
                self.game_over()

            pygame.display.update()
            self.fps_controller.tick(10)
```

Method run adalah loop utama permainan yang menjalankan seluruh logika game. Dalam loop ini, setiap input keyboard diperiksa untuk mengatur arah ular. Metode self.snake.move() menggerakkan ular sesuai arah yang diinginkan pemain. Ketika kepala ular mencapai posisi apel merah atau biru, skor bertambah, dan apel yang dimakan dipindahkan ke posisi acak baru menggunakan metode respawn. Jika tidak ada apel yang dimakan, metode self.snake.shrink() dipanggil untuk menghapus segmen terakhir dari tubuh ular. Semua objek permainan (ular dan apel) serta skor kemudian digambar di layar. Terakhir, check_collision memeriksa apakah ular mengalami tabrakan dengan dinding atau tubuhnya sendiri, dan jika terjadi, game_over dipanggil. Setiap loop diakhiri dengan pygame.display.update() untuk

memperbarui layar dan self.fps_controller.tick(10) untuk mengatur kecepatan permainan pada 10 frame per detik.

16. Fungsi utama

```
if __name__ == "__main__":
 game = Game()
 game.run()
```

Kode ini memastikan bahwa blok kode hanya dijalankan jika skrip dijalankan langsung, bukan diimpor sebagai modul. Di dalam blok ini, objek Game dibuat dengan memanggil game = Game(), yang menginisialisasi seluruh elemen permainan, termasuk ular, apel, layar, dan pengontrol FPS. Setelah itu, game.run() dijalankan untuk memulai loop utama permainan, di mana seluruh logika dan tampilan game Snake berjalan terus menerus hingga pemain kalah atau menutup jendela permainan. Blok ini menjadi titik awal untuk menjalankan game Snake secara interaktif.

Hasil scene game