LAPORAN UAS GAME NYAWNYAWNYAW

Mata Kuliah

Pemrograman Berbasis Objek

Oleh

Rahma Aulia Fitriani Arief

24091397067

2024C

PROGRAM STUDI MANAJEMEN INFORMATIKA

FAKULTAS VOKASI

UNIVERSITAS NEGERI SURABAYA

2025

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pemrograman Berorientasi Objek (OOP) merupakan paradigma pemrograman yang banyak digunakan dalam pengembangan perangkat lunak modern. Paradigma ini berfokus pada pemodelan sistem ke dalam bentuk objek yang memiliki atribut dan perilaku tertentu. Dengan pendekatan ini, pengembangan perangkat lunak menjadi lebih terstruktur, modular, mudah dipelihara, serta mudah dikembangkan kembali.

Pada mata kuliah Pemrograman Berorientasi Objek, mahasiswa dituntut tidak hanya memahami konsep secara teori, tetapi juga mampu mengimplementasikannya dalam sebuah aplikasi nyata. Oleh karena itu, pada Ujian Akhir Semester diberikan tugas berupa proyek pengembangan aplikasi berbasis OOP yang disertai dengan dokumentasi lengkap.

Dalam proyek ini, penulis mengembangkan sebuah game sederhana berbasis Python menggunakan library Pygame dengan judul **"NYAWNYAWNYAW – Cat Run"**. Game ini mengangkat konsep survival, di mana pemain mengendalikan karakter kucing yang harus menghindari musuh berupa anjing dan mengumpulkan power-up untuk mempertahankan nyawa. Game dipilih karena memiliki banyak objek yang saling berinteraksi sehingga sangat sesuai untuk menerapkan konsep-konsep OOP seperti encapsulation, inheritance, dan polymorphism.

1.2 Rumusan Masalah

- Bagaimana merancang dan mengimplementasikan aplikasi game sederhana menggunakan OOP?
- Bagaimana menerapkan prinsip encapsulation, inheritance, dan polymorphism dalam struktur class game?

1.3 Tujuan

- Mengembangkan aplikasi game sederhana berbasis OOP menggunakan bahasa pemrograman python.
- Menerapkan dan memahami konsep OOP.

1.4 Manfaat

Menambah pemahaman dan pengalaman dalam penerapan OOP secara nyata serta menjadi bukti capaian pembelajaran mata kuliah pemrograman berorientasi objek.

BAB II

LANDASAN TEORI

2.1 Pemrograman Berorientasi Objek (OOP)

Pemrograman Berorientasi Objek (Object-Oriented Programming/OOP) merupakan paradigma pemrograman yang berfokus pada konsep **objek** sebagai representasi dari dunia nyata. Setiap objek memiliki **atribut** (data) dan **method** (perilaku) yang saling berkaitan. Pendekatan ini bertujuan untuk meningkatkan keterbacaan kode, kemudahan pemeliharaan, serta kemampuan pengembangan sistem dalam skala besar.

Dalam OOP, program dibangun menggunakan kumpulan kelas (class) yang saling berinteraksi. Setiap class berfungsi sebagai cetak biru (blueprint) untuk membentuk objek. Paradigma ini sangat sesuai untuk pengembangan aplikasi modern, termasuk game, karena mampu memodelkan entitas seperti pemain, musuh, dan objek pendukung secara terstruktur. Penggunaan OOP dalam pengembangan game memberikan keuntungan berupa:

- Struktur kode yang lebih rapi
- Kemudahan pengembangan fitur baru
- Pengurangan duplikasi kode

2.2 Konsep Dasar OOP

- **Encapsulation**

Encapsulation adalah konsep OOP yang bertujuan untuk membungkus data dan method dalam satu class serta membatasi akses langsung terhadap data tersebut. Dengan encapsulation, detail implementasi suatu objek dapat disembunyikan dari luar class, sehingga keamanan dan integritas data lebih terjaga. Dalam bahasa pemrograman Python, encapsulation dapat diterapkan dengan penggunaan atribut private (menggunakan awalan __) dan method khusus untuk mengakses atau memodifikasi nilai tersebut. Pada aplikasi game NYAWNYAWNYAW – Cat Run, encapsulation diterapkan pada pengelolaan nyawa (health) pemain. Atribut nyawa disimpan sebagai atribut private dan hanya dapat diubah melalui method tertentu seperti take_damage() dan heal(). Hal ini mencegah perubahan nilai nyawa secara sembarangan dari luar class.

- Inheritance

Inheritance (pewarisan) merupakan konsep OOP yang memungkinkan sebuah class turunan (child class) mewarisi atribut dan method dari class induk (parent class). Konsep ini bertujuan untuk **mengurangi pengulangan kode** dan meningkatkan efisiensi pengembangan. Dalam game yang dikembangkan, class Entity digunakan sebagai class induk yang menyimpan atribut umum seperti posisi, ukuran, kecepatan, dan method dasar. Class Player, Dog, dan PowerUp merupakan class turunan dari Entity yang mewarisi atribut tersebut, kemudian menambahkan atau memodifikasi perilaku sesuai dengan perannya masing-masing. Dengan penerapan inheritance, pengelolaan objek dalam game menjadi lebih sederhana dan terstruktur, serta memudahkan pengembangan fitur baru di masa mendatang.

- Polymorphism

  Polymorphism adalah kemampuan objek untuk memiliki perilaku yang berbeda meskipun menggunakan method yang sama**.** Konsep ini memungkinkan pemanggilan method yang sama pada objek yang berbeda dengan hasil yang berbeda pula. Dalam aplikasi game ini, polymorphism diterapkan pada method update() dan draw() yang dimiliki oleh setiap entity. Meskipun method yang digunakan sama, perilaku yang dijalankan berbeda tergantung pada jenis objek, seperti pemain, musuh, atau power-up. Penerapan polymorphism membuat kode menjadi lebih fleksibel dan mendukung pengelolaan banyak objek secara bersamaan dalam satu struktur program.

2.3 Pengenalan Game Development dengan Pygame

Pygame merupakan library Python yang digunakan untuk pengembangan game dua dimensi. Library ini menyediakan berbagai fitur penting seperti pengelolaan tampilan grafis, input keyboard, suara, serta manajemen waktu (frame rate).

Pemilihan Pygame pada proyek ini didasarkan pada beberapa pertimbangan, antara lain:

- Mudah digunakan
- Mendukung pengembangan game berbasis OOP

Dalam game NYAWNYAWNYAW – Cat Run, Pygame digunakan untuk menangani tampilan karakter, pergerakan objek, deteksi tabrakan (collision detection), serta pemutaran suara latar dan efek suara.

2.4 Konsep Game Arcade

Game arcade merupakan jenis permainan yang menuntut refleks cepat dan fokus pemain dalam menghindari rintangan serta mencapai skor tertinggi. Konsep utama dari game arcade adalah kesederhanaan mekanisme permainan namun tetap menantang. Game NYAWNYAWNYAW – Cat Run mengadopsi konsep arcade dengan mekanisme:

- Pemain mengendalikan karakter kucing
- Musuh berupa anjing yang harus dihindari
- Sistem skor yang terus bertambah seiring waktu
- Power up untuk menambah nyawa

2.5 Struktur Data Pendukung

Dalam pengembangan game ini beberapa struktur data digunakan untuk mendukung jalannya game, antara lain:

- List, digunakan untuk menyimpan kumpulan objek musuh dan entity dalam game
- Dictionary, digunakan untuk menyimpan data highscore
- JSON, digunakan sebagai media penyimpanan data skor tertinggi secara permanen

BAB III

PERANCANGAN SISTEM

3.1 Gambaran Umum Sistem

Aplikasi yang dikembangkan pada proyek ini adalah sebuah game 2D berbasis Python menggunakan library Pygame dengan judul "NYAWNYAWNYAW – Cat Run". Game ini mengusung konsep endless survival, di mana pemain harus bertahan selama mungkin dengan menghindari musuh dan menjaga nyawa karakter. Sistem game dirancang menggunakan pendekatan Pemrograman Berorientasi Objek (OOP) agar setiap komponen memiliki tanggung jawab yang jelas. Dengan pendekatan ini, game menjadi lebih modular, mudah dikembangkan, serta meminimalkan duplikasi kode. Secara umum sistem terdiri dari:

- Objek pemain (Cat)
- Objek musuh (Dog)
- Objek Power up (Fish Bone)
- Sistem pengendali game
- Sistem pendukung (skor, nyawa, dan highscore)

3.2 Perancangan Sistem

Arsitektur sistem game menggunakan pola object-based design, di mana setiap objek di layar direpresentasikan sebagai sebuah class. Class Game berperan sebagai controller yang mengatur alur permainan, sedangkan class lain berperan sebagai model objek dalam game. Hubungan antar class dirancang sebagai berikut:

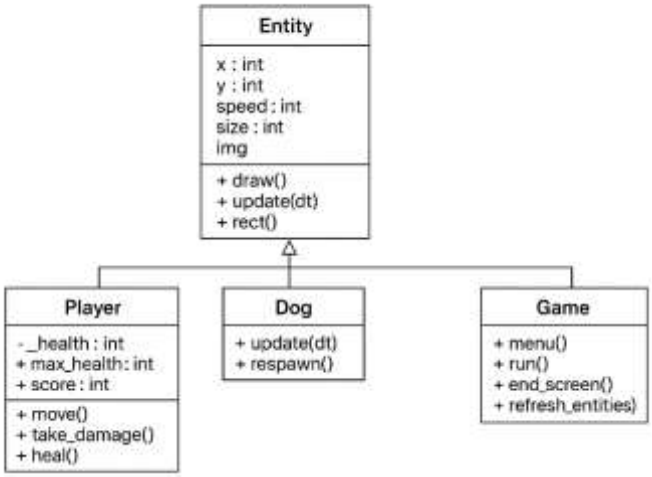
- Entity, menjadi class induk
- Cat, dog, power up sebagai class turunan
- Game memiliki relasi asosiasi dengan seluruh objek

3.3 Alur Proses Aplikasi

Alur kerja aplikasi dapat dijelaskan sebagai berikut:

- Program dijalankan dan system melakukan inisialisasi asset (gambar, suara, font)
- Sistem menampilkan menu utama
- Game loop dijalankan, meliputi:
    - Pengambilan input pengguna
    - Pergerakan objek
    - Deteksi tabrakan
    - Perhitungan skor
    - Update tampilan layar
- Jika nyawa pemain habis, sistem akan menampilkan layar "Game Over"
- Pemain dapat memilih untuk mengulang permainan atau keluar

3.4 Class Diagram



Class diagram menjelaskan struktur program berbasis OOP yang digunakan pada permainan ini. Kelas *Entity* berperan sebagai kelas dasar yang digunakan oleh semua objek yang memiliki posisi, kecepatan, ukuran, serta gambar. Kelas ini menyediakan fungsi dasar seperti *draw()*, *update()*, dan *rect()*.

Kelas *Player*, *Dog*, dan *PowerUp* merupakan turunan dari *Entity*. Kelas *Player* memiliki atribut tambahan seperti *health*, *max_health*, serta *score*, dan menyediakan fungsi untuk bergerak, menerima damage, memulihkan nyawa, dan mengecek kondisi hidup. Kelas *Dog* dan *PowerUp* mewarisi perilaku dasar dari *Entity*, namun memiliki fungsi khusus seperti *update()* dan *respawn()*.

Kelas *Game* berperan sebagai pengontrol utama, yang mengatur menu, loop permainan, layar akhir, serta pembaruan semua entitas. Diagram ini menggambarkan relasi antar kelas dan menjelaskan bagaimana setiap bagian saling berinteraksi dalam sistem permainan.

BAB IV
IMPLEMENTASI SISTEM

4.1 Implementasi Class

- **Entity**
  Class Entity merupakan class induk yang merepresentasikan semua objek dalam game. Class ini berisi atribut umum seperti posisi, ukuran, kecepatan, dan gambar. Class ini digunakan untuk mengurangi pengulangan kode, menyediakan method umum seperti draw() dan rect()
  Penggunaan class Entity merupakan penerapan konsep abstraction dan inheritance.
- **Player (cat)**
  Class Player merupakan turunan dari class Entity yang merepresentasikan karakter utama pemain. Class ini memiliki atribut private __health untuk menyimpan jumlah nyawa. Penggunaan atribut private ini merupakan penerapan encapsulation, karena nilai nyawa hanya dapat diubah melalui method tertentu seperti take_damage() dan heal().
- **Enemy (dog)**
  Class Dog merepresentasikan musuh dalam game. Class ini juga merupakan turunan dari Entity dan memiliki perilaku khusus pada method update(). Setiap objek Dog bergerak dari atas ke bawah layar dan akan muncul kembali secara acak setelah keluar layar. Implementasi ini menunjukkan penerapan polymorphism, karena method update() memiliki perilaku berbeda dibandingkan class lain.
- **Power Up**
  Class PowerUp berfungsi sebagai objek pendukung yang dapat menambah nyawa pemain. Class ini memiliki perilaku update yang mirip dengan musuh, tetapi efeknya berbeda ketika terjadi collision dengan pemain.
- **Game**
  Class game merupakan pusat pengendali aplikasi. Class ini bertanggung jawab untuk:
  - Menampilkan menu
  - Menjalankan game loop
  - Mengatur skor dan highscore
  - Menangani kondisi pause dan game over

  Class ini berperan penting dalam menghubungkan selurug objek game.

4.2 Implementasi Fitur Utama

Fitur utama yang berhasil diimplementasikan dalam game ini meliputi:

- Sistem pergerakan pemain berbasis input keyboard
- Sistem collision antara pemain, musuh, dan power-up
- Sistem nyawa dan skor berbasis waktu
- Sistem highscore menggunakan file eksternal
- Efek suara dan musik latar
- Menu utama dan layar Game Over

4.3 Keterkaitan Implementasi dengan Konsep OOP

Seluruh implementasi kode telah disesuaikan dengan prinsip OOP:

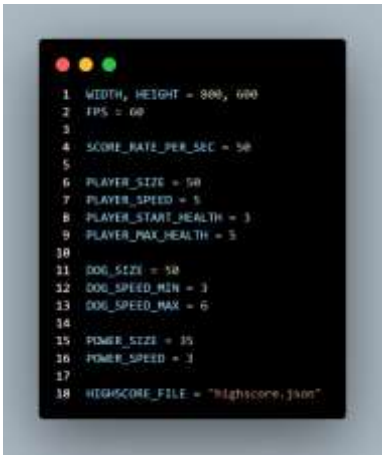
- Encapsulation diterapkan pada atribut nyawa
- Inheritance digunakan untuk struktur class
- Polymorphism diterapkan pada method update()
- Abstraction diterapkan melalui class induk Entity

Dengan penerapan ini, aplikasi menjadi lebih terstruktur dan mudah dikembangkan.

BAB V
PENJELASAN KODE PROGRAM DAN TAMPILAN GAME

```
1  import pygame
2  import random
3  import sys
4  import os
5  import json
```

Bagian ini digunakan untuk mengimpor library yang mendukung jalannya aplikasi. Library pygame berfungsi sebagai framework utama pengembangan game 2D, random digunakan untuk menghasilkan nilai acak, sedangkan os dan json digunakan untuk pengelolaan file penyimpanan highscore. Bagian ini bersifat pendukung sistem dan belum menerapkan konsep OOP secara langsung, namun menjadi fondasi utama agar program dapat berjalan dengan baik.



```
1  WIDTH, HEIGHT = 800, 600
2  FPS = 60
3
4  SCORE_RATE_PER_SEC = 50
5
6  PLAYER_SIZE = 50
7  PLAYER_SPEED = 5
8  PLAYER_START_HEALTH = 3
9  PLAYER_MAX_HEALTH = 5
10
11 DOG_SIZE = 50
12 DOG_SPEED_MIN = 3
13 DOG_SPEED_MAX = 6
14
15 POWER_SIZE = 35
16 POWER_SPEED = 3
17
18 HIGHSCORE_FILE = "highscore.json"
```

Bagian ini mendefinisikan konstanta yang digunakan secara global, seperti ukuran layar, kecepatan permainan, dan ukuran objek. Penggunaan konstanta bertujuan untuk mempermudah pengaturan dan pemeliharaan kode karena nilai dapat diubah tanpa harus memodifikasi banyak bagian program.

```
def safe_image_load(path, size):
    try:
        img = pygame.image.load(path).convert_alpha()
        return pygame.transform.scale(img, size)
    except:
        surf = pygame.Surface(size, pygame.SRCALPHA)
        surf.fill((200,200,200,255))
        pygame.draw.rect(surf,(100,100,100),surf.get_rect(),2)
        return surf
```

Fungsi ini digunakan untuk memuat gambar secara aman agar program tidak berhenti ketika file tidak ditemukan. Jika terjadi kesalahan, sistem akan menampilkan gambar pengganti. Pendekatan ini meningkatkan keandalan aplikasi namun belum termasuk konsep OOP karena masih berupa fungsi prosedural.

```
class Entity:
    def __init__(self,x,y,speed,size,img):
        self.x=x; self.y=y; self.speed=speed
        self.size=size; self.img=img
```

Class Entity merupakan class induk yang digunakan sebagai dasar seluruh objek dalam game. Class ini menyimpan atribut umum seperti posisi, kecepatan, ukuran, dan gambar. Penerapan class ini menunjukkan konsep inheritance, karena class lain akan mewarisi atribut dan method dari class ini.

```
class Player(Entity):
    def __init__(self,x,y):
        super().__init__(x,y,PLAYER_SPEED,PLAYER_SIZE,cat_img)
        self.__health = PLAYER_START_HEALTH
```

Class Player merupakan turunan dari class Entity yang merepresentasikan karakter utama. Konsep inheritance diterapkan melalui pewarisan dari Entity, sedangkan encapsulationditerapkan dengan penggunaan atribut private __health untuk melindungi data nyawa pemain agar tidak dapat diakses langsung dari luar class.

```
class Dog(Entity):
    def __init__(self):
        speed = random.randint(DOG_SPEED_MIN, DOG_SPEED_MAX)
        x = random.randint(0, WIDTH - DOG_SIZE)
        y = random.randint(-400, -40)
        super().__init__(x,y,speed,DOG_SIZE,dog_img)

```

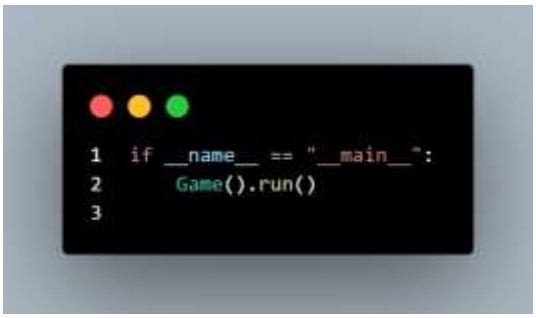
Class Dog merupakan class turunan dari Entity yang berperan sebagai musuh. Class ini memiliki perilaku khusus berupa pergerakan otomatis dan respawn ketika keluar layar. Metode update() pada class ini mengubah perilaku method induk, sehingga menunjukkan konsep polymorphism.

```
class PowerUp(Entity):
    def __init__(self):
        x = random.randint(0,WIDTH-POWER_SIZE)
        y = random.randint(-800,-40)
        super().__init__(x,y,POWER_SPEED,POWER_SIZE,power_img)

    def update(self,dt):
        self.y += self.speed
        if self.y > HEIGHT: self.respawn()

    def respawn(self):
        self.y = random.randint(-800,-40)
        self.x = random.randint(0,WIDTH-self.size)
```

Class PowerUp digunakan sebagai item pendukung untuk menambah nyawa pemain. Class ini juga merupakan turunan dari Entity dan memiliki perilaku berbeda pada method update(), yang memperkuat penerapan konsep polymorphism dalam aplikasi.

Class Game berfungsi sebagai pengatur utama alur permainan, termasuk inisialisasi objek, pengelolaan menu, deteksi tabrakan, serta pengaturan skor dan kondisi game over. Class ini mengoordinasikan interaksi antar objek sehingga sistem berjalan secara terstruktur.

Bagian ini digunakan untuk menyimpan skor tertinggi pemain ke dalam file JSON agar dapat digunakan kembali saat game dijalankan ulang. Fitur ini merupakan pengembangan tambahan yang meningkatkan fungsionalitas aplikasi.



Bagian ini merupakan titik awal eksekusi program yang menjalankan class Game. Seluruh konsep OOP yang telah dirancang akan bekerja secara terintegrasi pada bagian ini.

Tampilan game


NYAWNYAWNYAW
Press ENTER to Play
Press ESC to Quit
Highscore: 2835


Score: 174


PAUSED


GAME OVER
Your Score: 917
Highscore: 2835
Press ENTER to Play Again
Press ESC to Quit

BAB VI
KESIMPULAN DAN SARAN

6.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian aplikasi, dapat disimpulkan bahwa game “NYAWNYAWNYAW – Cat Run” berhasil dikembangkan menggunakan bahasa pemrograman Python dengan menerapkan konsep Pemrograman Berorientasi Objek (OOP) secara baik dan terstruktur. Game ini mampu berjalan dengan stabil serta memenuhi tujuan utama sebagai aplikasi game arcade sederhana berbasis survival.

Penerapan konsep OOP pada game ini terlihat jelas melalui penggunaan class dan objek. Konsep **encapsulation** diterapkan pada pengelolaan nyawa pemain dengan penggunaan atribut private sehingga data lebih aman. Konsep inheritance diterapkan dengan baik melalui class induk *Entity* yang diwarisi oleh class *Player*, *Dog*, dan *PowerUp*. Sementara itu, konsep polymorphism terlihat pada penggunaan method update() dan draw() yang memiliki perilaku berbeda pada setiap class turunan.

Selain itu, game ini juga dilengkapi dengan berbagai fitur pendukung seperti sistem skor berbasis waktu, sistem nyawa, power-up, menu utama, layar game over, serta penyimpanan highscore menggunakan file JSON. Dengan adanya fitur-fitur tersebut, game menjadi lebih interaktif.

6.2 Saran

Meskipun game NYAWNYAWNYAW – Cat Run telah berjalan dengan baik, masih terdapat beberapa pengembangan yang dapat dilakukan ke depannya. Salah satunya adalah penambahan variasi musuh dengan pola pergerakan yang lebih kompleks agar tantangan permainan semakin meningkat. Selain itu, fitur level atau tingkat kesulitan juga dapat ditambahkan untuk memberikan pengalaman bermain yang lebih beragam.

Dari sisi teknis, pengembangan antarmuka pengguna (UI) dapat ditingkatkan dengan penggunaan desain grafis yang lebih menarik dan animasi yang lebih halus. Penggunaan konsep OOP yang lebih lanjut, seperti penerapan interface atau design pattern tertentu, juga dapat dipertimbangkan untuk meningkatkan kualitas struktur kode.

Secara keseluruhan, game ini sudah layak digunakan sebagai media pembelajaran OOP. Dengan pengembangan lebih lanjut, aplikasi ini berpotensi menjadi game edukatif yang lebih kompleks dan menarik.