

**LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK (PBO) – [TUGAS 2]**



Disusun Oleh

Eka Putri Azhari Rtg 123140028

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

INSTITUT TEKNOLOGI SUMATERA

2025

Soal :

- membuat sebuah permainan sederhana tentang pertarungan Robot.
 - Kalian akan membuat kelas Robot yang terdiri dari beberapa **properti** seperti attack, Hp, dll., serta beberapa metode seperti attack_enemy() atau regen_health().
 - Permainan ini akan berakhir ketika salah satu robot memiliki **Hp** = 0.
- Kalian bisa lebih kreatif dengan menambahkan konsep seperti attack_accuracy agar serangan dapat meleset dalam beberapa kesempatan atau menambahkan mekanisme **skill** seperti stun, silence, dll., pada musuh. (Bagian ini opsional).
- Kalian mungkin perlu dua kelas:
 1. **Kelas Robot:** Berisi mekanisme **attack**, **hp**, dan mekanisme pertarungan robot.
 2. **Kelas Game:** Berfungsi untuk menentukan jumlah ronde serta mengatur jalannya permainan.

Source Code

```
import random

class Robot:
    def __init__(self, name, attack, hp):
        self.name = name
        self.attack = attack
        self.hp = hp

    def attack_enemy(self, enemy):
        # Menggunakan attack_accuracy untuk menentukan apakah
        serangan berhasil
        attack_accuracy = random.randint(1, 100)
        if attack_accuracy <= 80: # 80% chance to hit
            damage = self.attack
            enemy.hp -= damage
            print(f"{self.name} menyerang {enemy.name} dan
            memberikan {damage} damage!")
        else:
            print(f"{self.name} menyerang {enemy.name} tetapi
            meleset!")

    def regen_health(self):
        regen_amount = random.randint(5, 15)
        self.hp += regen_amount
        print(f"{self.name} meregenerasi {regen_amount} HP!")

    def is_alive(self):
        return self.hp > 0

class Game:
    def __init__(self, robot1, robot2):
        self.robot1 = robot1
        self.robot2 = robot2
        self.round = 1

    def start_game(self):
        print("Permainan dimulai!")
        while self.robot1.is_alive() and self.robot2.is_alive():
            print(f"\n--- Ronde {self.round} ---")
            self.robot1.attack_enemy(self.robot2)
            if self.robot2.is_alive():
                self.robot2.attack_enemy(self.robot1)
            else:
                print(f"{self.robot2.name} telah kalah!")
                break

        # Mungkin menambahkan regenerasi kesehatan setelah
        setiap ronde
        self.robot1.regen_health()
        self.robot2.regen_health()

        print(f"{self.robot1.name} HP: {self.robot1.hp}")
        print(f"{self.robot2.name} HP: {self.robot2.hp}")

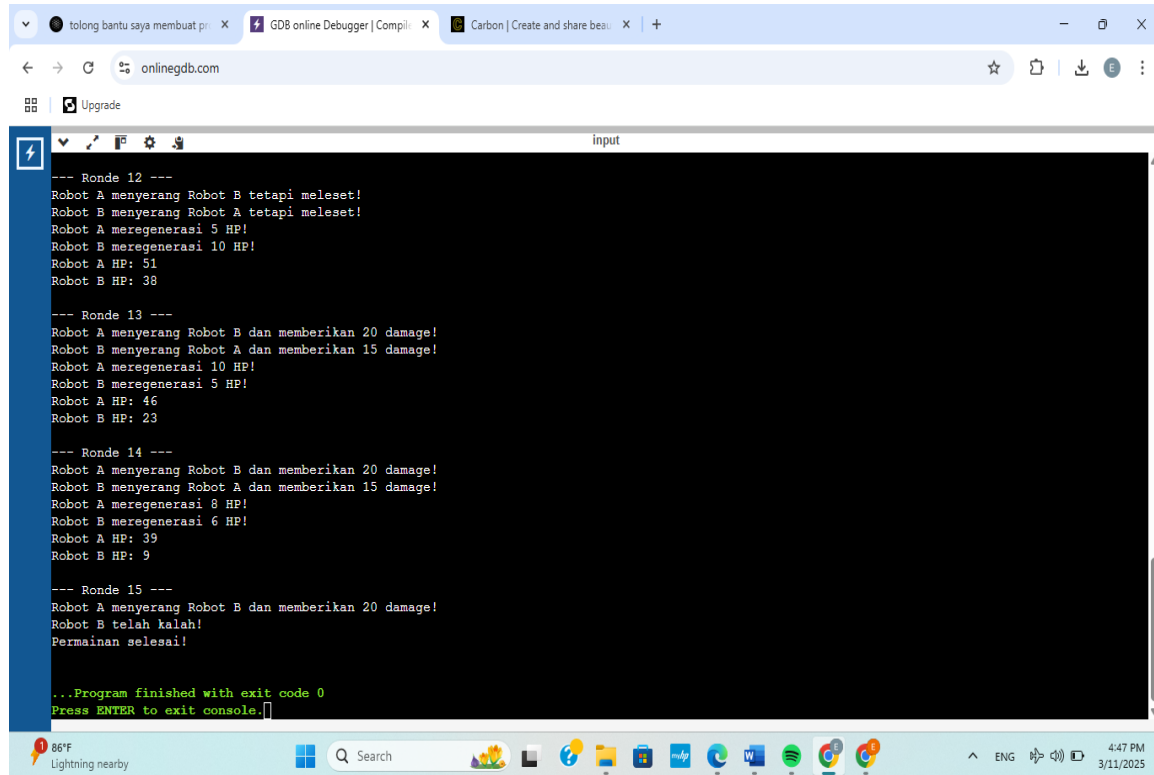
        self.round += 1

        print("Permainan selesai!")

# Contoh penggunaan
robot1 = Robot("Robot A", attack=20, hp=100)
robot2 = Robot("Robot B", attack=15, hp=100)

game = Game(robot1, robot2)
game.start_game()
```

Output Hasil (Screenshot)



```
--- Ronde 12 ---
Robot A menyerang Robot B tetapi meleset!
Robot B menyerang Robot A tetapi meleset!
Robot A meregenerasi 10 HP!
Robot B meregenerasi 5 HP!
Robot A HP: 51
Robot B HP: 38

--- Ronde 13 ---
Robot A menyerang Robot B dan memberikan 20 damage!
Robot B menyerang Robot A dan memberikan 15 damage!
Robot A meregenerasi 10 HP!
Robot B meregenerasi 5 HP!
Robot A HP: 46
Robot B HP: 23

--- Ronde 14 ---
Robot A menyerang Robot B dan memberikan 20 damage!
Robot B menyerang Robot A dan memberikan 15 damage!
Robot A meregenerasi 8 HP!
Robot B meregenerasi 6 HP!
Robot A HP: 39
Robot B HP: 9

--- Ronde 15 ---
Robot A menyerang Robot B dan memberikan 20 damage!
Robot B telah kalah!
Permainan selesai!

...Program finished with exit code 0
Press ENTER to exit console.
```

Penjelasan :

Berdasarkan pada soal diatas, Program ini adalah permainan sederhana yang mensimulasikan pertarungan antara dua robot menggunakan konsep pemrograman berorientasi objek (OOP) dalam Python. Terdapat dua kelas utama: `Robot`, yang merepresentasikan setiap robot dengan atribut seperti nama, nilai serangan, dan kesehatan (HP), serta metode untuk menyerang musuh, meregenerasi kesehatan, dan memeriksa apakah robot masih hidup; dan `Game`, yang mengatur jalannya permainan dengan mengelola dua robot dan menjalankan ronde pertarungan hingga salah satu robot kehabisan HP. Setiap serangan memiliki kemungkinan meleset, dan setelah setiap ronde, kedua robot dapat meregenerasi kesehatan mereka, menciptakan dinamika yang menarik dalam permainan. Program ini diakhiri dengan mencetak hasil pertarungan dan menyatakan robot yang kalah.

Lampiran

1. [Link Percakapan LLM](#)
2. [Web Referensi - DuniaIlkom](#)