

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



Disusun Oleh

Abel Fortino 123140111

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI SUMATERA
2025**

Soal

- Minggu ini hanya terdiri dari 1 Problem Set. Kalian perlu 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:
 - **Kelas Robot:** Berisi mekanisme **attack**, **hp**, dan mekanisme pertarungan robot.
 - **Kelas Game:** Berfungsi untuk menentukan jumlah ronde serta mengatur jalannya permainan.
- Kalian bisa melihat contoh cara kerja program di bawah ini. ini cuma contoh, kalian bisa buat versi kalian sendiri, yang lebih **sederhana** atau **kompleks**, tergantung kreativitas kalian

```
1  Round-1 =====
2  Atreus [500|10]
3  Daedalus [750|8]
4
5  1. Attack    2. Defense    3. Giveup
6  Atreus, pilih aksi: 1
7
8  1. Attack    2. Defense    3. Giveup
9  Daedalus, pilih aksi: 1
10
11 ----- Daedalus gagal menyerang -----
12
13 Round-2 =====
14 Atreus [500|10]
15 Daedalus [666|7]
16
17 1. Attack    2. Defense    3. Giveup
18 Atreus, pilih aksi: 3
19
20 1. Attack    2. Defense    3. Giveup
21 Daedalus, pilih aksi: 1
22
23 Daedalus menang!
```

Penjelasan

- Import random : menghasilkan angka acak antara 0 dan 1.
- Class Robot : konstruktor untuk menginialisasi objek yang terdiri dari nama, attack, dan hp.
 - Def attack_enemy : untuk menyerang musuh, dengan tambahan adanya chance.
 - Def regen_health : untuk heal diri sendiri maupun musuh, dengan adanya chance antara 5 sampai 10 hp.
 - Def give_up : opsi menyerah untuk musuh/ diri sendiri
- Class game : Konstruktor untuk menginialisasi yang terdiri dari round, robot1, dan robot2.
 - Def start_game : merupakan menu kecil untuk memulai permainan
- Radint(a,b) : menghasilkan angka acak antara a dan b, sebagai contoh radint(5, 10) maka angka acak yang dihasilkan antara 5 sampai 10.

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):
        if random.random() > 0.3:
            enemy.hp -= self.attack
            print(f"{self.name} berhasil menyerang {enemy.name} dengan {self.attack} damage")
        else:
            print(f"{self.name} gagal menyerang!")

    def regen_health(self):
        heal = random.randint(5, 10)
        if random.random() > 0.3:
            self.hp += heal
            if self.hp > 100:
                self.hp = 100
            print(f"{self.name} berhasil memulihkan {heal} health")
        else:
            print(f"{self.name} gagal memulihkan health")

    def give_up(self):
        self.hp = 0
        print(f"{self.name} menyerah")

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

    def __str__(self):
        return f"{self.name} dengan hp {self.hp}"

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

    def start_game(self):
        while self.robot1.is_alive() and self.robot2.is_alive():
            print(f"\nRound-{self.round}")
            print(f"{self.robot1.name} [{self.robot1.hp}]")
            print(f"{self.robot2.name} [{self.robot2.hp}]")

            action1 = input(f"{self.robot1.name}, pilih aksi (1. Attack 2. Regen 3. Give up): ")
            if action1 == "1":
                self.robot1.attack_enemy(self.robot2)
            elif action1 == "2":
                self.robot1.regen_health()
            elif action1 == "3":
                self.robot1.give_up()
            if not self.robot2.is_alive():
                print(f"{self.robot1.name} menang!")
                return

            action2 = input(f"{self.robot2.name}, pilih aksi (1. Attack 2. Regen 3. Give up): ")
            if action2 == "1":
                self.robot2.attack_enemy(self.robot1)
            elif action2 == "2":
                self.robot2.regen_health()
            elif action2 == "3":
                self.robot2.give_up()
            if not self.robot1.is_alive():
                print(f"{self.robot2.name} menang!")
                return

            self.round += 1


robot1 = Robot("Finn", random.randint(5, 15), 100)
robot2 = Robot("Ice King", random.randint(5, 15), 100)
game = Game(robot1, robot2, 1)
game.start_game()
```

Output

```
Round-1
Finn [100]
Ice King [100]
Finn, pilih aksi (1. Attack 2. Regen 3. Give up): 1
Finn berhasil menyerang Ice King dengan 6 damage
Ice King, pilih aksi (1. Attack 2. Regen 3. Give up): 1
Ice King berhasil menyerang Finn dengan 15 damage


Round-2
Finn [85]
Ice King [94]
Finn, pilih aksi (1. Attack 2. Regen 3. Give up): 3
Finn menyerah
Ice King, pilih aksi (1. Attack 2. Regen 3. Give up): 1
Ice King berhasil menyerang Finn dengan 15 damage
Ice King menang!
PS C:\Users\ASUS\Documents\Kuliah\PBO> |
```

Lampiran

 AbelFortino

apa kegunaan dari import random?

`import random.py:1-50`

 GitHub Copilot

> Used 1 reference

`import random`

digunakan untuk mengimpor modul `random` di Python, yang menyediakan berbagai fungsi untuk menghasilkan angka acak. Dalam kode Anda, modul `random` digunakan untuk beberapa tujuan:

- Menentukan keberhasilan serangan dan regenerasi kesehatan:**
 - Fungsi `random.random()` menghasilkan angka acak antara 0 dan 1. Ini digunakan untuk menentukan apakah serangan atau regenerasi kesehatan berhasil.

```
if random.random() > 0.3:
    # Berhasil
else:
    # Gagal
```

- Menghasilkan nilai acak untuk regenerasi kesehatan:**
 - Fungsi `random.randint(a, b)` menghasilkan angka acak antara `a` dan `b` (inklusif). Ini digunakan untuk menentukan jumlah kesehatan yang dipulihkan saat regenerasi.

```
heal = random.randint(5, 10)
```

- Menghasilkan nilai acak untuk serangan robot saat inisialisasi:**
 - Fungsi `random.randint(a, b)` juga digunakan untuk menentukan nilai serangan robot saat objek `Robot` dibuat.

```
robot1 = Robot("Finn", random.randint(5, 15), 100)
robot2 = Robot("Ice King", random.randint(5, 15), 100)
```