NAMA : Muhamad Faiq Nabil Saputra
NIM : 244107020025
KELAS : TI – 2D
MATKUL : UTS Praktikum PBO

Langkah Langkah Pengerjaan UTS Praktikum PBO
~ Monster Battle ~

1. Kode program

- game/Character.java (Abstract Class, Bagian A & D)

UTS > Game > Character.java > ...

```java
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

public abstract class Character {
    private final String name;
    private int health;
    private int maxHealth;
    private int attackPower;

    private final List<StatusEffect> effects = new ArrayList<>();

    protected List<Skill> skills;

    public Character(String name, int health, int attackPower) {
        if (health < 0 || attackPower < 0) {
            throw new IllegalArgumentException(s:"Health dan Attack Power tidak boleh negatif.");
        }
        this.name = name;
        this.maxHealth = health;
        this.health = health;
        this.attackPower = attackPower;
        this.skills = new ArrayList<>();
    }

    public String getName() { return name; }
    public int getHealth() { return health; }
    public int getMaxHealth() { return maxHealth; }
    public int getAttackPower() { return attackPower; }
    public List<StatusEffect> getEffects() { return Collections.unmodifiableList(effects); }
    public List<Skill> getSkills() { return Collections.unmodifiableList(skills); }

    public void setHealth(int newHealth) {
        if (newHealth < 0) {
            this.health = 0;
        } else if (newHealth > maxHealth) {
            this.health = maxHealth;
        } else {
            this.health = newHealth;
        }
    }

    public void setMaxHealth(int maxHealth) {
        if (maxHealth < 1) {
            throw new IllegalArgumentException(s:"Max Health harus positif.");
        }
        this.maxHealth = maxHealth;
    }
```

NAMA : Muhamad Faiq Nabil Saputra
NIM : 244107020025
KELAS : TI – 2D
MATKUL : UTS Praktikum PBO

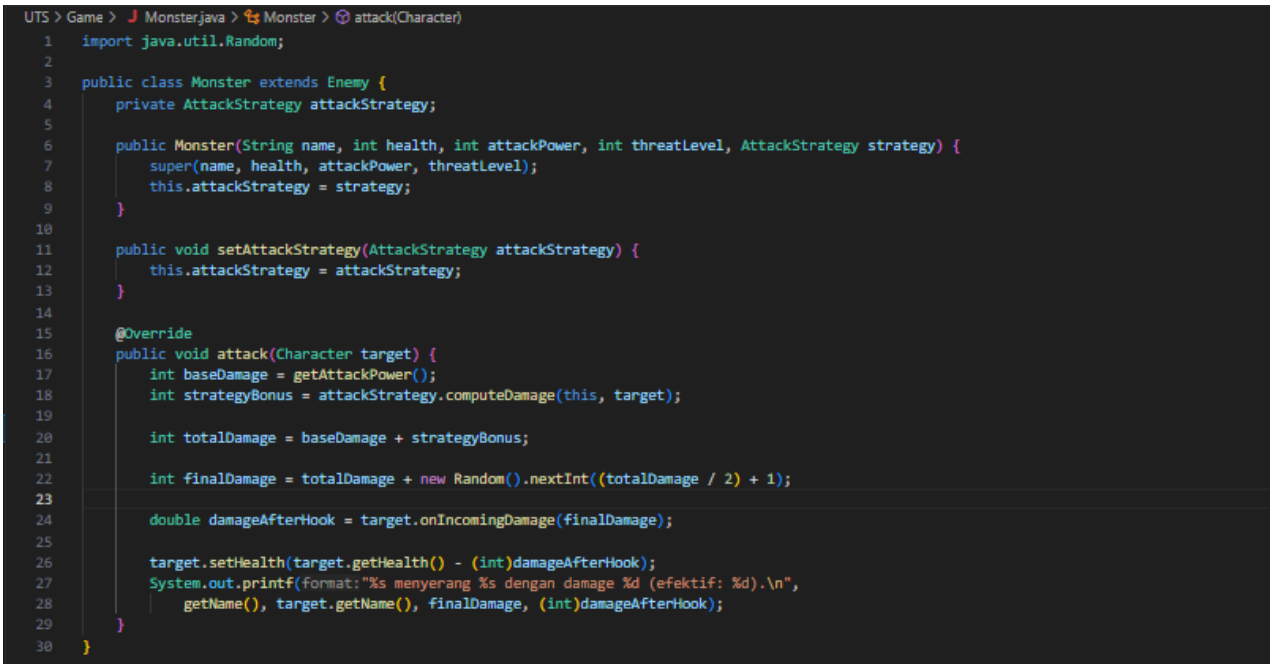
```java
    public abstract void attack(Character target);

    public final boolean isAlive() {
        return this.health > 0;
    }

    public final void performTurn(Character target) {
        if (!isAlive()) {
            System.out.println("[!] " + name + " sudah gugur dan tidak dapat bergerak.\n");
            return;
        }

        System.out.println("--- Giliran " + name + " dimulai ---");

        List<StatusEffect> toRemove = new ArrayList<>();
        for (StatusEffect effect : effects) {
            System.out.println("  > Efek aktif: " + effect.name());
            effect.onTurnStart(this);
            if (effect.isExpired()) {
                toRemove.add(effect);
            }
        }
        effects.removeAll(toRemove);

        if (isAlive() && target.isAlive()) {
            attack(target);
        }

        toRemove.clear();
        for (StatusEffect effect : effects) {
            effect.onTurnEnd(this);
            if (effect.isExpired()) {
                toRemove.add(effect);
            }
        }
        effects.removeAll(toRemove);

        System.out.println("  HP " + name + ": " + health + "/" + maxHealth);
        System.out.println("  HP " + target.getName() + ": " + target.getHealth() + "/" + target.getMaxHealth());
        System.out.println("--- Giliran " + name + " selesai ---");
    }

    public void addEffect(StatusEffect e) {
        if (effects.stream().anyMatch(effect -> effect.name().equals(e.name()))) {
            System.out.println("[*] " + name + " sudah memiliki efek " + e.name());
            return;
        }
        effects.add(e);
        System.out.println("[+] " + name + " mendapatkan efek: " + e.name());
    }

    public double onIncomingDamage(double damage) {
        double finalDamage = damage;
        for (StatusEffect effect : effects) {
            finalDamage = effect.onIncomingDamage(finalDamage);
        }
        return finalDamage;
    }

    public void addSkill(Skill skill) {
        this.skills.add(skill);
    }

    protected void setAttackPower(int attackPower) {
        this.attackPower = attackPower;
    }
}
```

NAMA : Muhamad Faiq Nabil Saputra
NIM : 244107020025
KELAS : TI – 2D
MATKUL : UTS Praktikum PBO

- game/Enemy.java (Abstract Class, Bagian B)

```java
public abstract class Enemy extends Character {
    private final int threatLevel;

    public Enemy(String name, int health, int attackPower, int threatLevel) {
        super(name, health, attackPower);
        if (threatLevel < 1 || threatLevel > 5) {
            throw new IllegalArgumentException("Threat Level harus antara 1 sampai 5.");
        }
        this.threatLevel = threatLevel;
    }

    public int getThreatLevel() {
        return threatLevel;
    }
}
```

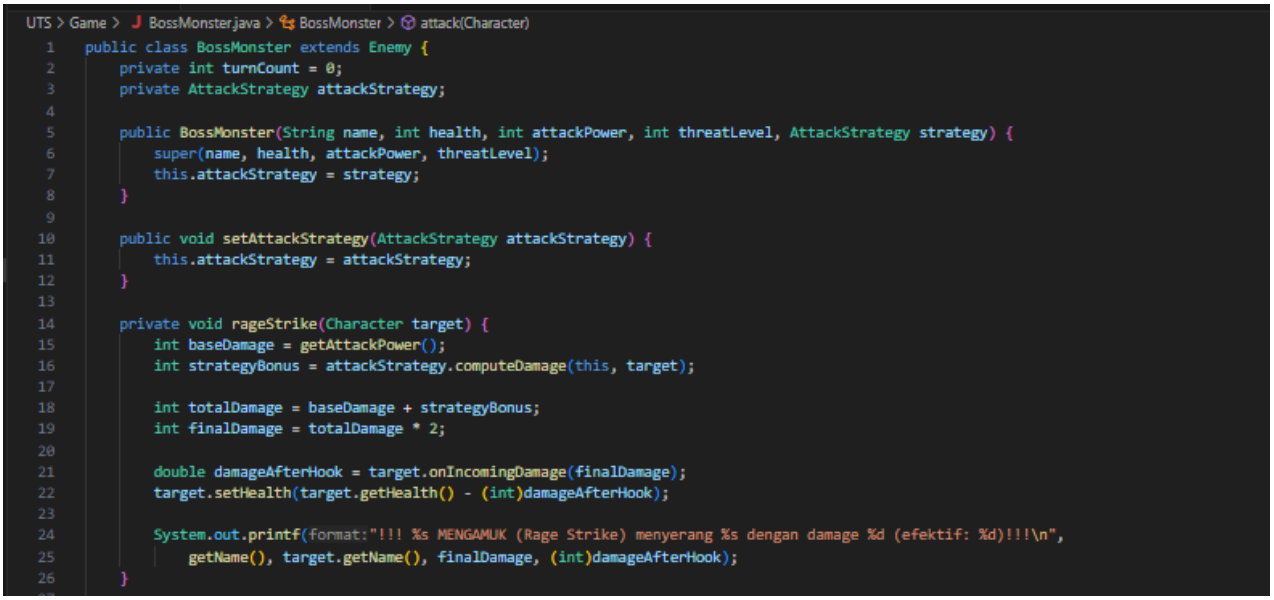
- game/Monster.java (Class, Bagian B)



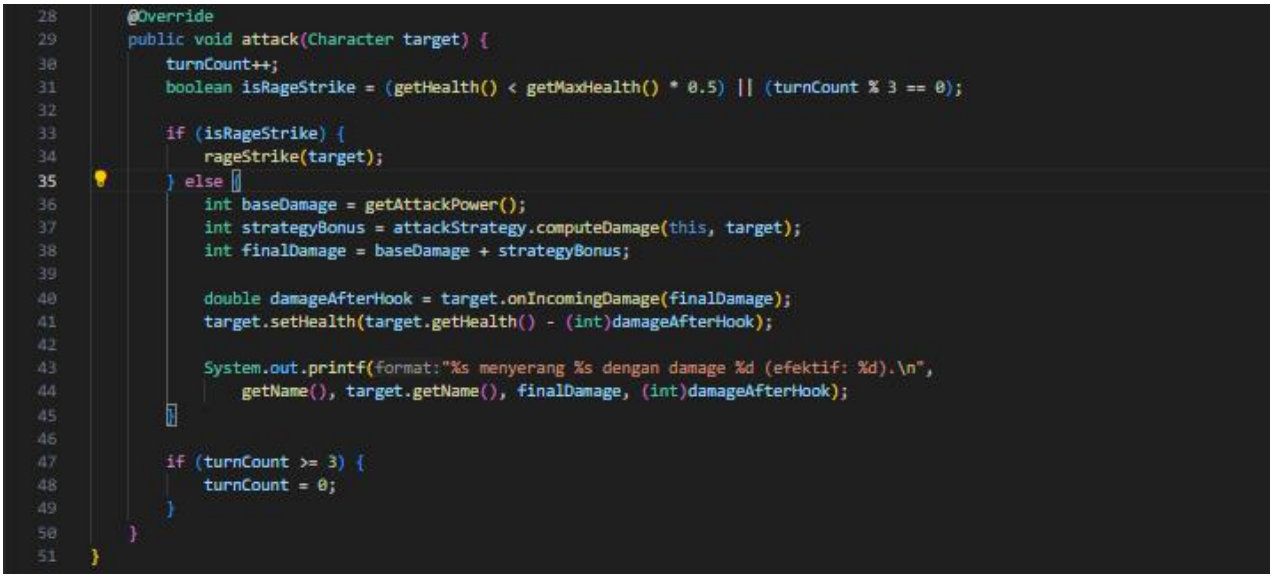
```java
import java.util.Random;

public class Monster extends Enemy {
    private AttackStrategy attackStrategy;

    public Monster(String name, int health, int attackPower, int threatLevel, AttackStrategy strategy) {
        super(name, health, attackPower, threatLevel);
        this.attackStrategy = strategy;
    }

    public void setAttackStrategy(AttackStrategy attackStrategy) {
        this.attackStrategy = attackStrategy;
    }

    @Override
    public void attack(Character target) {
        int baseDamage = getAttackPower();
        int strategyBonus = attackStrategy.computeDamage(this, target);

        int totalDamage = baseDamage + strategyBonus;

        int finalDamage = totalDamage + new Random().nextInt((totalDamage / 2) + 1);

        double damageAfterHook = target.onIncomingDamage(finalDamage);

        target.setHealth(target.getHealth() - (int)damageAfterHook);
        System.out.printf("%s menyerang %s dengan damage %d (efektif: %d).\n",
            getName(), target.getName(), finalDamage, (int)damageAfterHook);
    }
}
```

NAMA : Muhamad Faiq Nabil Saputra
NIM : 244107020025
KELAS : TI – 2D
MATKUL : UTS Praktikum PBO

- game/BossMonster.java (Class, Bagian B)



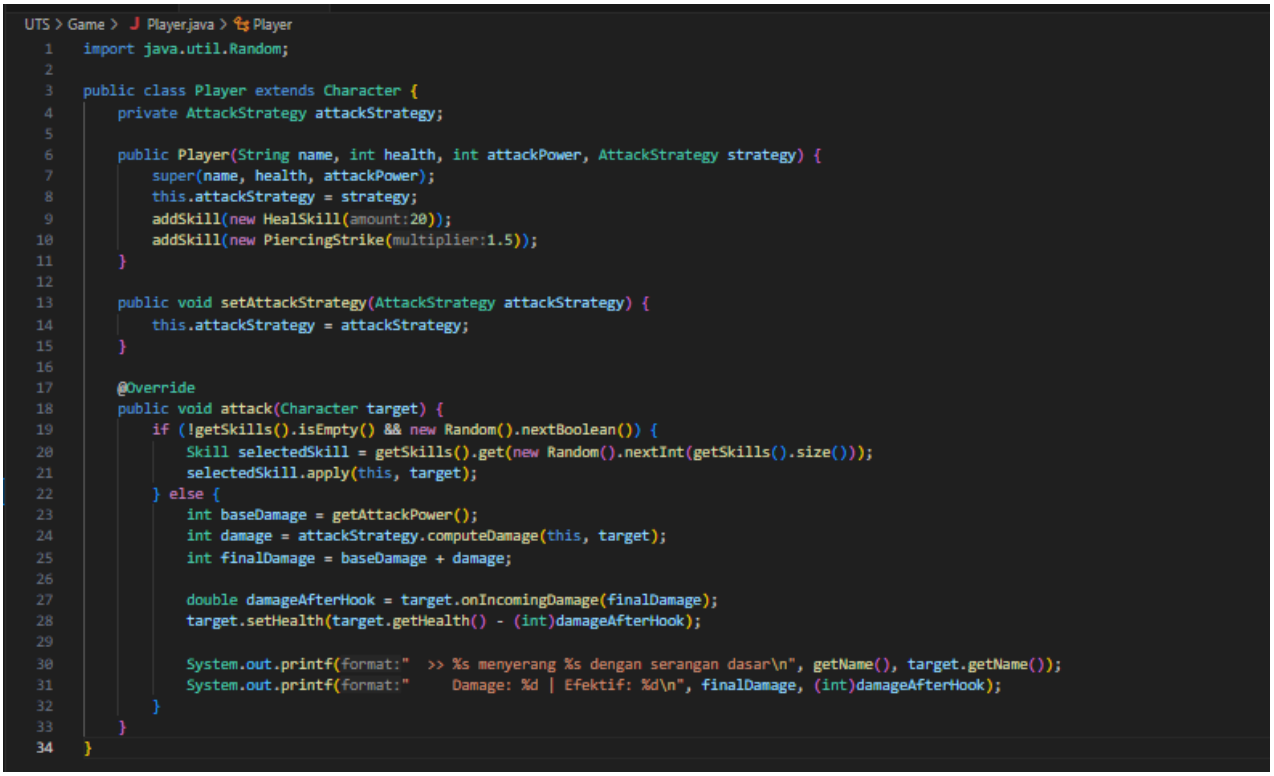
```
UTS > Game > BossMonster.java > BossMonster > attack(Character)
1   public class BossMonster extends Enemy {
2       private int turnCount = 0;
3       private AttackStrategy attackStrategy;
4
5       public BossMonster(String name, int health, int attackPower, int threatLevel, AttackStrategy strategy) {
6           super(name, health, attackPower, threatLevel);
7           this.attackStrategy = strategy;
8       }
9
10      public void setAttackStrategy(AttackStrategy attackStrategy) {
11          this.attackStrategy = attackStrategy;
12      }
13
14      private void rageStrike(Character target) {
15          int baseDamage = getAttackPower();
16          int strategyBonus = attackStrategy.computeDamage(this, target);
17
18          int totalDamage = baseDamage + strategyBonus;
19          int finalDamage = totalDamage * 2;
20
21          double damageAfterHook = target.onIncomingDamage(finalDamage);
22          target.setHealth(target.getHealth() - (int)damageAfterHook);
23
24          System.out.printf(format:"!!! %s MENGAMUK (Rage Strike) menyerang %s dengan damage %d (efektif: %d)!!!\n",
25              getName(), target.getName(), finalDamage, (int)damageAfterHook);
26      }
```



```
28      @Override
29      public void attack(Character target) {
30          turnCount++;
31          boolean isRageStrike = (getHealth() < getMaxHealth() * 0.5) || (turnCount % 3 == 0);
32
33          if (isRageStrike) {
34              rageStrike(target);
35          } else {
36              int baseDamage = getAttackPower();
37              int strategyBonus = attackStrategy.computeDamage(this, target);
38              int finalDamage = baseDamage + strategyBonus;
39
40              double damageAfterHook = target.onIncomingDamage(finalDamage);
41              target.setHealth(target.getHealth() - (int)damageAfterHook);
42
43              System.out.printf(format:"%s menyerang %s dengan damage %d (efektif: %d).\n",
44                  getName(), target.getName(), finalDamage, (int)damageAfterHook);
45          }
46
47          if (turnCount >= 3) {
48              turnCount = 0;
49          }
50      }
51  }
```

NAMA : Muhamad Faiq Nabil Saputra
NIM : 244107020025
KELAS : TI – 2D
MATKUL : UTS Praktikum PBO

- game/Player.java (Class, Bagian B, C, & E)



```java
import java.util.Random;

public class Player extends Character {
    private AttackStrategy attackStrategy;

    public Player(String name, int health, int attackPower, AttackStrategy strategy) {
        super(name, health, attackPower);
        this.attackStrategy = strategy;
        addSkill(new HealSkill(amount:20));
        addSkill(new PiercingStrike(multiplier:1.5));
    }

    public void setAttackStrategy(AttackStrategy attackStrategy) {
        this.attackStrategy = attackStrategy;
    }

    @Override
    public void attack(Character target) {
        if (!getSkills().isEmpty() && new Random().nextBoolean()) {
            Skill selectedSkill = getSkills().get(new Random().nextInt(getSkills().size()));
            selectedSkill.apply(this, target);
        } else {
            int baseDamage = getAttackPower();
            int damage = attackStrategy.computeDamage(this, target);
            int finalDamage = baseDamage + damage;

            double damageAfterHook = target.onIncomingDamage(finalDamage);
            target.setHealth(target.getHealth() - (int)damageAfterHook);

            System.out.printf(format:"  >> %s menyerang %s dengan serangan dasar\n", getName(), target.getName());
            System.out.printf(format:"     Damage: %d | Efektif: %d\n", finalDamage, (int)damageAfterHook);
        }
    }
}
```

- game/Skill.java (Interface, Bagian C)

```java
public interface Skill {
    String name();
    void apply(Character self, Character target);
}
```

- game/HealSkill.java (Class, Bagian C & G)

```java
public class HealSkill implements Skill {
    private final int amount;

    public HealSkill(int amount) {
        this.amount = amount;
    }

    @Override
    public String name() {
        return "HealSkill";
    }

    @Override
    public void apply(Character self, Character target) {
        int oldHealth = self.getHealth();
        self.setHealth(oldHealth + amount);
        int healedAmount = self.getHealth() - oldHealth;

        System.out.printf(format:"%s menggunakan HealSkill, memulihkan HP sebesar %d. HP sekarang: %d/%d.\n",
            self.getName(), healedAmount, self.getHealth(), self.getMaxHealth());
    }
}
```

NAMA : Muhamad Faiq Nabil Saputra
NIM : 244107020025
KELAS : TI – 2D
MATKUL : UTS Praktikum PBO

- game/PiercingStrike.java (Class, Bagian C)

UTS > Game > PiercingStrike.java > PiercingStrike

```java
public class PiercingStrike implements Skill {
    private final double multiplier;

    public PiercingStrike(double multiplier) {
        this.multiplier = multiplier;
    }

    @Override
    public String name() {
        return "PiercingStrike";
    }

    @Override
    public void apply(Character self, Character target) {
        int baseDamage = self.getAttackPower();
        int finalDamage = (int) (baseDamage * multiplier);

        double ignoredReduction = 0.25;

        double initialDamage = finalDamage;

        double damageAfterHook = target.onIncomingDamage(initialDamage);

        double reducedAmount = initialDamage - damageAfterHook;

        double piercingAmount = reducedAmount * ignoredReduction;

        double effectiveDamage = damageAfterHook + piercingAmount;

        target.setHealth(target.getHealth() - (int)effectiveDamage);

        System.out.printf("%s menggunakan PiercingStrike pada %s, damage %d (efektif: %d). " +
            "Mengabaikan 25%% dari damage reduction lawan.\n",
            self.getName(), target.getName(), finalDamage, (int)effectiveDamage);
    }
}
```

- game/StatusEffect.java (Interface, Bagian D)

UTS > Game > StatusEffect.java > StatusEffect > isExpired()

```java
public interface StatusEffect {
    String name();
    void onTurnStart(Character self);
    void onTurnEnd(Character self);
    boolean isExpired();
    double onIncomingDamage(double damage);
}
```

NAMA : Muhamad Faiq Nabil Saputra
NIM : 244107020025
KELAS : TI – 2D
MATKUL : UTS Praktikum PBO

- game/Regen.java (Class, Bagian D)

UTS > Game > Regen.java > Regen > onIncomingDamage(double)

```java
public class Regen implements StatusEffect {
    private final int perTurn;
    private int duration;

    public Regen(int perTurn, int duration) {
        this.perTurn = perTurn;
        this.duration = duration;
    }

    @Override
    public String name() {
        return "Regen (" + duration + "T)";
    }

    @Override
    public void onTurnStart(Character self) {
        if (duration > 0) {
            int oldHealth = self.getHealth();
            self.setHealth(oldHealth + perTurn);
            int healedAmount = self.getHealth() - oldHealth;
            System.out.printf(format:"Regen: %s memulihkan %d HP. HP sekarang: %d/%d.\n",
                self.getName(), healedAmount, self.getHealth(), self.getMaxHealth());
        }
    }

    @Override
    public void onTurnEnd(Character self) {
        duration--;
        if (isExpired()) {
            System.out.println("Efek Regen pada " + self.getName() + " berakhir.");
        }
    }

    @Override
    public boolean isExpired() {
        return duration <= 0;
    }

    @Override
    public double onIncomingDamage(double damage) {
        return damage;
    }
}
```

NAMA : Muhamad Faiq Nabil Saputra
NIM : 244107020025
KELAS : TI – 2D
MATKUL : UTS Praktikum PBO

- game/Shield.java (Class, Bagian D & G)

```java
public class Shield implements StatusEffect {
    private final int flatReduce;
    private int duration;

    public Shield(int flatReduce, int duration) {
        this.flatReduce = flatReduce;
        this.duration = duration;
    }

    @Override
    public String name() {
        return "Shield (" + duration + "T, -" + flatReduce + " dmg)";
    }

    @Override
    public void onTurnStart(Character self) {
    }

    @Override
    public void onTurnEnd(Character self) {
        duration--;
        if (isExpired()) {
            System.out.println("Efek Shield pada " + self.getName() + " berakhir.");
        }
    }

    @Override
    public boolean isExpired() {
        return duration <= 0;
    }

    @Override
    public double onIncomingDamage(double damage) {
        double reducedDamage = damage - flatReduce;
        if (reducedDamage < 0) {
            reducedDamage = 0;
        }
        System.out.printf(format:"Shield: Damage %d dikurangi %d (flat reduce) menjadi %d.\n",
            (int)damage, flatReduce, (int)reducedDamage);
        return reducedDamage;
    }
}
```

- game/AttackStrategy.java (Interface, Bagian E)

```java
public interface AttackStrategy {
    int computeDamage(Character self, Character target);
}
```

- game/FixedStrategy.java (Class, Bagian E)

```java
public class FixedStrategy implements AttackStrategy {

    @Override
    public int computeDamage(Character self, Character target) {
        return 10;
    }
}
```

NAMA : Muhamad Faiq Nabil Saputra
NIM : 244107020025
KELAS : TI – 2D
MATKUL : UTS Praktikum PBO

- game/LevelScaledStrategy.java (Class, Bagian E)

UTS > Game > LevelScaledStrategy.java > LevelScaledStrategy > computeDamage(Character, Character)

```java
public class LevelScaledStrategy implements AttackStrategy {
    private final int bonusPerLevel;

    public LevelScaledStrategy(int bonusPerLevel) {
        this.bonusPerLevel = bonusPerLevel;
    }

    @Override
    public int computeDamage(Character self, Character target) {
        int level;
        if (self instanceof Enemy) {
            level = ((Enemy) self).getThreatLevel();
        } else if (self instanceof Player) {
            level = self.getMaxHealth() / 100;
            if (level < 1) level = 1;
        } else {
            level = 1;
        }

        int calculatedBonus = level * bonusPerLevel;
        System.out.println(self.getName() + " menggunakan LevelScaledStrategy. Bonus damage: " + calculatedBonus);
        return calculatedBonus;
    }
}
```

- game/Battle.java (Class, Bagian F)

UTS > Game > Battle.java > Battle

```java
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
import java.util.stream.Collectors;

public class Battle {
    private final List<Character> teamA;
    private final List<Character> teamB;
    private final List<Character> allCharacters;
    private int turn = 0;

    public Battle(List<Character> teamA, List<Character> teamB) {
        this.teamA = new ArrayList<>(teamA);
        this.teamB = new ArrayList<>(teamB);
        this.allCharacters = new ArrayList<>();
        this.allCharacters.addAll(teamA);
        this.allCharacters.addAll(teamB);
    }
```

NAMA : Muhamad Faiq Nabil Saputra
NIM : 244107020025
KELAS : TI – 2D
MATKUL : UTS Praktikum PBO

```java
private boolean isTeamDefeated(List<Character> team) {
    return team.stream().noneMatch(Character::isAlive);
}

private Character autoTarget(Character attacker, List<Character> targetTeam) {
    List<Character> aliveTargets = targetTeam.stream()
        .filter(Character::isAlive)
        .collect(Collectors.toList());

    if (aliveTargets.isEmpty()) {
        return null;
    }

    if (attacker instanceof Enemy) {
        List<Character> alivePlayers = aliveTargets.stream()
            .filter(c -> c instanceof Player)
            .collect(Collectors.toList());

        if (!alivePlayers.isEmpty()) {
            return alivePlayers.stream()
                .max((c1, c2) -> Integer.compare(c1.getHealth(), c2.getHealth()))
                .orElse(other:null);
        } else {
            return aliveTargets.get(new Random().nextInt(aliveTargets.size()));
        }
    } else if (attacker instanceof Player) {
        List<Enemy> aliveEnemies = aliveTargets.stream()
            .filter(c -> c instanceof Enemy)
            .map(c -> (Enemy) c)
            .collect(Collectors.toList());

        if (!aliveEnemies.isEmpty()) {
            return aliveEnemies.stream()
                .max((e1, e2) -> {
                    int threatCompare = Integer.compare(e1.getThreatLevel(), e2.getThreatLevel());
                    if (threatCompare != 0) {
                        return threatCompare;
                    }
                    return Integer.compare(e2.getHealth(), e1.getHealth());
                })
                .orElse(other:null);
        } else {
            return aliveTargets.get(new Random().nextInt(aliveTargets.size()));
        }
    }

    return aliveTargets.get(new Random().nextInt(aliveTargets.size()));
```

```java
public void run() {
    System.out.println(x:"\n======================================");
    System.out.println(x:"     SIMULASI PERTARUNGAN DIMULAI");
    System.out.println(x:"======================================\n");

    printTeamStatus();

    while (!isTeamDefeated(teamA) && !isTeamDefeated(teamB)) {
        turn++;
        System.out.println(x:"\n==========================================");
        System.out.println("              GILIRAN " + turn);
        System.out.println(x:"==========================================\n");

        List<Character> currentTurnOrder = new ArrayList<>(allCharacters);
        List<Character> aliveCharacters = currentTurnOrder.stream()
            .filter(Character::isAlive)
            .collect(Collectors.toList());

        for (Character attacker : aliveCharacters) {
            List<Character> targetTeam = teamA.contains(attacker) ? teamB : teamA;

            if (isTeamDefeated(targetTeam)) break;

            Character target = autoTarget(attacker, targetTeam);

            if (target != null) {
                attacker.performTurn(target);
                System.out.println();
            }
        }

        if (isTeamDefeated(teamA) || isTeamDefeated(teamB)) {
            break;
        }
    }
```

NAMA : Muhamad Faiq Nabil Saputra
NIM : 244107020025
KELAS : TI – 2D
MATKUL : UTS Praktikum PBO

```
        System.out.println("\n====================================");
        System.out.println("         PERTARUNGAN SELESAI");
        System.out.println("====================================\n");

        if (isTeamDefeated(teamA)) {
            System.out.println(">>> PEMENANG: TIM B (Enemy) <<<\n");
        } else if (isTeamDefeated(teamB)) {
            System.out.println(">>> PEMENANG: TIM A (Player) <<<\n");
        } else {
            System.out.println("Pertarungan berakhir tanpa pemenang.\n");
        }

        printFinalStats();
    }

    private void printTeamStatus() {
        System.out.println("TIM A (Players):");
        teamA.forEach(c -> {
            System.out.printf("  - %-15s | HP: %3d/%3d\n",
                c.getName(), c.getHealth(), c.getMaxHealth());
        });

        System.out.println("\nTIM B (Enemies):");
        teamB.forEach(c -> {
            System.out.printf("  - %-15s | HP: %3d/%3d\n",
                c.getName(), c.getHealth(), c.getMaxHealth());
        });
        System.out.println();
    }

    private void printFinalStats() {
        System.out.println("====================================");
        System.out.println("          STATISTIK AKHIR");
        System.out.println("====================================\n");

        System.out.println("TIM A (Players):");
        teamA.forEach(c -> {
            String status = c.isAlive() ? "[HIDUP]" : "[GUGUR]";
            System.out.printf("  - %-15s | HP: %3d/%3d | %s\n",
                c.getName(), c.getHealth(), c.getMaxHealth(), status);
        });
```

```
        System.out.println("\nTIM B (Enemies):");
        teamB.forEach(c -> {
            String status = c.isAlive() ? "[HIDUP]" : "[GUGUR]";
            System.out.printf("  - %-15s | HP: %3d/%3d | %s\n",
                c.getName(), c.getHealth(), c.getMaxHealth(), status);
        });

        System.out.println("\n============================================\n");
    }
}
```

- game/Main.java (Main Class untuk menjalankan simulasi)

```
UTS > Game > Main.java > Main > main(String[])
import java.util.Arrays;
import java.util.List;

public class Main {
    Run | Debug
    public static void main(String[] args) {
        AttackStrategy fixedStrategy = new FixedStrategy();
        AttackStrategy levelScaledStrategy = new LevelScaledStrategy(bonusPerLevel:5);

        Player p1 = new Player(name:"Ksatria", health:300, attackPower:30, fixedStrategy);
        Player p2 = new Player(name:"Penyihir", health:150, attackPower:45, levelScaledStrategy);

        p1.addEffect(new Shield(flatReduce:15, duration:3));
        p2.addEffect(new Regen(perTurn:10, duration:5));

        Monster m1 = new Monster(name:"Goblin", health:100, attackPower:20, threatLevel:1, fixedStrategy);
        Monster m2 = new Monster(name:"Orc", health:180, attackPower:25, threatLevel:3, levelScaledStrategy);
        BossMonster boss = new BossMonster(name:"Naga", health:500, attackPower:50, threatLevel:5, levelScaledStrategy);

        List<Character> teamA = Arrays.asList(p1, p2);
        List<Character> teamB = Arrays.asList(m1, m2, boss);

        Battle battle = new Battle(teamA, teamB);
        battle.run();
    }
}
```

NAMA : Muhamad Faiq Nabil Saputra
NIM : 244107020025
KELAS : TI – 2D
MATKUL : UTS Praktikum PBO

2. Tampilan Output

```
[+] Ksatria mendapatkan efek: Shield (3T, -15 dmg)
[+] Penyihir mendapatkan efek: Regen (5T)

======================================
     SIMULASI PERTARUNGAN DIMULAI
======================================

TIM A (Players):
  - Ksatria         | HP: 300/300
  - Penyihir        | HP: 150/150

TIM B (Enemies):
  - Goblin          | HP: 100/100
  - Orc             | HP: 180/180
  - Naga            | HP: 500/500


==========================================
             GILIRAN 1
==========================================

--- Giliran Ksatria dimulai ---
  > Efek aktif: Shield (3T, -15 dmg)
Ksatria menggunakan PiercingStrike pada Naga, damage 45 (efektif: 45). Mengabaikan 25% dari damage reduction lawan.
  HP Ksatria: 300/300
  HP Naga: 455/500
--- Giliran Ksatria selesai ---

--- Giliran Penyihir dimulai ---
  > Efek aktif: Regen (5T)
Regen: Penyihir memulihkan 0 HP. HP sekarang: 150/150.
Penyihir menggunakan LevelScaledStrategy. Bonus damage: 5
  >> Penyihir menyerang Naga dengan serangan dasar
     Damage: 50 | Efektif: 50
  HP Penyihir: 150/150
  HP Naga: 405/500
--- Giliran Penyihir selesai ---

--- Giliran Goblin dimulai ---
Shield: Damage 31 dikurangi 15 (flat reduce) menjadi 16.
Goblin menyerang Ksatria dengan damage 31 (efektif: 16).
  HP Goblin: 100/100
  HP Ksatria: 284/300
--- Giliran Goblin selesai ---

--- Giliran Orc dimulai ---
Orc menggunakan LevelScaledStrategy. Bonus damage: 15
Shield: Damage 58 dikurangi 15 (flat reduce) menjadi 43.
Orc menyerang Ksatria dengan damage 58 (efektif: 43).
  HP Orc: 180/180
  HP Ksatria: 241/300
--- Giliran Orc selesai ---
```

NAMA : Muhamad Faiq Nabil Saputra
NIM : 244107020025
KELAS : TI – 2D
MATKUL : UTS Praktikum PBO

```
--- Giliran Naga dimulai ---
Naga menggunakan LevelScaledStrategy. Bonus damage: 25
Shield: Damage 75 dikurangi 15 (flat reduce) menjadi 60.
Naga menyerang Ksatria dengan damage 75 (efektif: 60).
  HP Naga: 405/500
  HP Ksatria: 181/300
--- Giliran Naga selesai ---


==========================================
             GILIRAN 2
==========================================

--- Giliran Ksatria dimulai ---
  > Efek aktif: Shield (2T, -15 dmg)
  >> Ksatria menyerang Naga dengan serangan dasar
     Damage: 40 | Efektif: 40
  HP Ksatria: 181/300
  HP Naga: 365/500
--- Giliran Ksatria selesai ---

--- Giliran Penyihir dimulai ---
  > Efek aktif: Regen (4T)
Regen: Penyihir memulihkan 0 HP. HP sekarang: 150/150.
Penyihir menggunakan LevelScaledStrategy. Bonus damage: 5
  >> Penyihir menyerang Naga dengan serangan dasar
     Damage: 50 | Efektif: 50
  HP Penyihir: 150/150
  HP Naga: 315/500
--- Giliran Penyihir selesai ---

--- Giliran Goblin dimulai ---
Shield: Damage 37 dikurangi 15 (flat reduce) menjadi 22.
Goblin menyerang Ksatria dengan damage 37 (efektif: 22).
  HP Goblin: 100/100
  HP Ksatria: 159/300
--- Giliran Goblin selesai ---

--- Giliran Orc dimulai ---
Orc menggunakan LevelScaledStrategy. Bonus damage: 15
Shield: Damage 50 dikurangi 15 (flat reduce) menjadi 35.
Orc menyerang Ksatria dengan damage 50 (efektif: 35).
  HP Orc: 180/180
  HP Ksatria: 124/300
--- Giliran Orc selesai ---

--- Giliran Naga dimulai ---
Naga menggunakan LevelScaledStrategy. Bonus damage: 25
Naga menyerang Penyihir dengan damage 75 (efektif: 75).
  HP Naga: 315/500
  HP Penyihir: 75/150
--- Giliran Naga selesai ---
```

```
==========================================
             GILIRAN 3
==========================================

--- Giliran Ksatria dimulai ---
  > Efek aktif: Shield (1T, -15 dmg)
  >> Ksatria menyerang Naga dengan serangan dasar
     Damage: 40 | Efektif: 40
Efek Shield pada Ksatria berakhir.
  HP Ksatria: 124/300
  HP Naga: 275/500
--- Giliran Ksatria selesai ---

--- Giliran Penyihir dimulai ---
  > Efek aktif: Regen (3T)
Regen: Penyihir memulihkan 10 HP. HP sekarang: 85/150.
Penyihir menggunakan LevelScaledStrategy. Bonus damage: 5
  >> Penyihir menyerang Naga dengan serangan dasar
     Damage: 50 | Efektif: 50
  HP Penyihir: 85/150
  HP Naga: 225/500
--- Giliran Penyihir selesai ---

--- Giliran Goblin dimulai ---
Goblin menyerang Ksatria dengan damage 45 (efektif: 45).
  HP Goblin: 100/100
  HP Ksatria: 79/300
--- Giliran Goblin selesai ---

--- Giliran Orc dimulai ---
Orc menggunakan LevelScaledStrategy. Bonus damage: 15
Orc menyerang Penyihir dengan damage 60 (efektif: 60).
  HP Orc: 180/180
  HP Penyihir: 25/150
--- Giliran Orc selesai ---

--- Giliran Naga dimulai ---
Naga menggunakan LevelScaledStrategy. Bonus damage: 25
!!! Naga MENGAMUK (Rage Strike) menyerang Ksatria dengan damage 150 (efektif: 150)!!!
  HP Naga: 225/500
  HP Ksatria: 0/300
--- Giliran Naga selesai ---
```

NAMA : Muhamad Faiq Nabil Saputra
NIM : 244107020025
KELAS : TI – 2D
MATKUL : UTS Praktikum PBO

```
==========================================
             GILIRAN 4
==========================================

--- Giliran Penyihir dimulai ---
  > Efek aktif: Regen (2T)
Regen: Penyihir memulihkan 10 HP. HP sekarang: 35/150.
Penyihir menggunakan PiercingStrike pada Naga, damage 67 (efektif: 67). Mengabaikan 25% dari damage reduction lawan.
  HP Penyihir: 35/150
  HP Naga: 158/500
--- Giliran Penyihir selesai ---

--- Giliran Goblin dimulai ---
Goblin menyerang Penyihir dengan damage 33 (efektif: 33).
  HP Goblin: 100/100
  HP Penyihir: 2/150
--- Giliran Goblin selesai ---

--- Giliran Orc dimulai ---
Orc menggunakan LevelScaledStrategy. Bonus damage: 15
Orc menyerang Penyihir dengan damage 50 (efektif: 50).
  HP Orc: 180/180
  HP Penyihir: 0/150
--- Giliran Orc selesai ---


======================================
         PERTARUNGAN SELESAI
======================================

>>> PEMENANG: TIM B (Enemy) <<<

==================================
         STATISTIK AKHIR
==================================

TIM A (Players):
  - Ksatria         | HP:   0/300 | [GUGUR]
  - Penyihir        | HP:   0/150 | [GUGUR]

TIM B (Enemies):
  - Goblin          | HP: 100/100 | [HIDUP]
  - Orc             | HP: 180/180 | [HIDUP]
  - Naga            | HP: 158/500 | [HIDUP]

==========================================
```