Name : Faatihurrizki Prasojo

NIM : 244107020142

Class : TI_1H

Laporan Kuis 1

Pemrograman Berorientasi Objek (PBO)

1. Jelaskan bagaimana inheritance, polymorphism, dan encapsulation digunakan dalam program.
   Answer :

   1. Inheritance (Pewarisan)

   Pada dasarnya, konsep ini digunakan untuk membuat struktur hierarki yang efisien dan menghindari duplikasi kode.

   - **Kelas Induk (Superclass):** Kami mendefinisikan kelas abstrak Character sebagai pondasi utama. Kelas ini menampung atribut umum (name, health, attackPower) dan perilaku dasar (takeDamage()) yang dimiliki oleh semua karakter, baik itu pemain maupun monster. Karena sifatnya abstrak, kami tidak bisa membuat objek langsung dari kelas ini, melainkan harus melalui turunannya.
   - **Kelas Anak (Subclass):** Kami membuat tiga kelas turunan dari Character, yaitu Player, Monster, dan BossMonster. Masing-masing kelas ini **mewarisi** semua atribut dan metode dari Character, sehingga kami tidak perlu mendefinisikan ulang properti dasar seperti health atau name. Selain itu, setiap kelas anak menambahkan atribut dan metode uniknya sendiri, seperti level dan heal() pada Player, atau type pada Monster dan BossMonster. Ini membuat desain program menjadi lebih terstruktur dan mudah dikelola.

   2. Polymorphism (Polimorfisme)

   Polimorfisme memungkinkan kami untuk mengimplementasikan perilaku yang sama (attack()) dengan cara yang berbeda untuk setiap jenis karakter. Ini sangat membantu untuk membuat kode yang fleksibel dan tidak kaku.

   - **Metode Abstrak:** Kami mendefinisikan metode attack(Character target) sebagai abstract di kelas Character. Ini memaksa setiap kelas anak untuk menyediakan implementasi spesifik mereka sendiri untuk metode ini.
   - **Implementasi Beragam:**
     - Kelas Player mengimplementasikan attack() dengan formula super.getAttackPower() + (level * 2).
     - Kelas Monster menggunakan logika serangan acak (Math.random()).
     - Kelas BossMonster memiliki serangan spesial yang memberikan damage dua kali lipat dari attackPower.

   Dengan demikian, di kelas GameTest, kami bisa memanggil karakter.attack(target) tanpa harus memeriksa jenis objeknya terlebih dahulu. Java akan secara otomatis menjalankan metode attack() yang sesuai, tergantung apakah karakter tersebut adalah Player, Monster, atau BossMonster.

Name : Faatihurrizki Prasojo

NIM : 244107020142

Class : TI_1H

3. Encapsulation (Enkapsulasi)

Kami menerapkan enkapsulasi untuk melindungi data internal objek dan memastikan bahwa akses ke data tersebut dilakukan secara terkontrol.

- **Variabel Privat:** Kami mendeklarasikan semua atribut (name, health, attackPower, dll.) sebagai private. Ini mencegah modifikasi langsung dari luar kelas.
- **Metode Publik (Getter & Setter):** Kami menyediakan metode public (getName(), getHealth(), setHealth(), dll.) sebagai satu-satunya "gerbang" untuk mengakses atau memodifikasi atribut privat.
- **Manajemen Data Terkontrol:** Sebagai contoh, atribut health hanya bisa diubah melalui metode takeDamage() atau heal(). Ini memastikan bahwa logika perubahan HP, seperti if (health < 0) health = 0, selalu diterapkan, sehingga tidak ada kondisi yang tidak diinginkan seperti HP bernilai negatif. Konsep ini menjaga integritas data dan membuat kode lebih aman dari kesalahan.

2. Buat class diagramnya