RESPONSI

PEMOGRAMAN BERIORIENTASI OBJEK

Di Susun Oleh:

ELSA AMBARWATT (5230411238)

Program Studi Informatika
Fakultas Sains dan Teknologi

Universitas Teknologi Yogyakarta
Tahun Akademik 2024/2025

RESPONSI

Soal-soal teori PBO:

1. **Jelaskan perbedaan use case diagram dengan class diagram?**
2. **Jelaskan jenis-jenis dependensi?**
3. **Apa perbedaan pemrograman terstruktur dengan berorientasi objek, jelaskan?**
4. **Jelaskan konsep object dan beri contohnya?**
5. **Jelaskan jenis-jenis access modifier, beri contohnya dalam baris pemrograman?**
6. **Gambarkan contoh pewarisan dalam class diagram?**

Jawaban:

1. perbedaan use case diagram dan class diagram : **use case diagram** Menunjukkan interaksi antara pengguna (aktor) dan sistem. Fokus pada fungsi utama yang bisa dilakukan oleh sistem dari sudut pandang pengguna. Sedangkan, **Class Diagram** Menggambarkan struktur internal sistem dengan menunjukkan kelas-kelas, atribut, metode, dan hubungan antar kelas. Fokus pada bagaimana data dan objek dalam sistem terorganisir.
2. Jenis – jenis dependesi :

   1.) **Dependency**: Hubungan di mana satu elemen bergantung pada elemen lain. Jika yang satu berubah, yang lain mungkin juga harus berubah.

   2.) **Association**: Hubungan antara dua kelas di mana satu kelas berinteraksi atau menggunakan yang lain.

   3.) **Aggregation**: Hubungan "bagian-dari" di mana satu kelas adalah bagian dari kelas lain, tetapi masih bisa berdiri sendiri.

   4.) **Composition**: Hubungan yang lebih kuat daripada aggregation, di mana bagian tidak bisa ada tanpa keseluruhan.

   5.) **Generalization**: Hubungan antara kelas umum (superclass) dan kelas khusus (subclass), di mana subclass mewarisi sifat dari superclass.

3. perbedaan pemrograman terstruktur dengan berorientasi objek : **pemrograman terstruktur** Fokus pada fungsi dan prosedur untuk menyelesaikan masalah. Kode dibagi menjadi bagian-bagian terpisah dengan alur kontrol menggunakan loop dan pernyataan kondisional. Sedangkan, **Pemrograman Berorientasi Objek** Mengorganisir kode dalam bentuk objek yang memiliki data (atribut) dan fungsi (metode). Menggunakan konsep seperti enkapsulasi, pewarisan, dan polimorfisme untuk membuat kode lebih modular dan dapat digunakan kembali.

4. **Object** adalah Instance dari kelas yang memiliki keadaan (atribut) dan perilaku (metode).

   Contoh:

```
class Mobil {
   String warna;
   String merk;
   void nyalakanMesin() {
      System.out.println("Mesin mobil dinyalakan");
   }
}

public class Main {
   public static void main(String[] args) {
      Mobil mobilSaya = new Mobil(); // Membuat objek mobilSaya
      mobilSaya.warna = "Merah";
      mobilSaya.merk = "Toyota";
      mobilSaya.nyalakanMesin(); // Memanggil metode
   }
}
```

5. 1.) **Public**: Akses terbuka untuk semua kelas.

   Contoh :

```
public class Contoh {
   public int angka;
}
```

   2.) **Private**: Akses terbatas hanya untuk kelas itu sendiri.

   Contoh :

```
class Contoh {
   private int angka;
}
```

   3.) **Protected**: Akses terbatas untuk kelas itu sendiri dan kelas yang mewarisi dari kelas tersebut.

Contoh :

```
class Contoh {
   protected int angka;
}
```

4.) **Default** (tanpa modifier): Akses terbatas untuk kelas dalam paket yang sama.

Contoh :

```
class Contoh {
   int angka; // default access
}
```

6. Contoh gambar pewarisan dalam kelas diagram