RESPONSI 1

PEMROGRAMAN BEROTIENTASI OBJEK

Di susun Oleh:

Maulia Hafifatun Solihah

5230411231

PROGRAM STUDI INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS TEKNOLOGI YOGYAKARTA

2024/2025

Soal Teori

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

Jawaban:

1. Use Case Diagram adalah jenis diagram yang digunakan dalam pemodelan system berbasis objek untuk menggambarkan interaksi antara pengguna (aktor) dengan system yang sedang dikembangkan. Tujuannya adalah untuk mendokumentasikan fungsi system dari perspektif pengguna, mendokumentasikan ruang lingkup system, mendokumentasikan interaksi pengguna dan system menggunakan deskripsi use case pendukung (spesifikasi perilaku). Sedangkan class diagram adalah merupakan struktur system yang menggambarkan kelas – kelas, atribut, dan hubungan antar kelas dalam system. Contohnya dalam sistem perpustakaan, class diagram dapat menggambarkan kelas “Buku” dengan atribut seperti “judul”, “pengarang”, dan “ISBN”.

2. Dependensi adalah hubungan antar elemen – elemn dalam suatu sistem. Beberapa jenis dependensi yang umum adalah:
    - Generalisasi:
      Merupakan hubungan "is-a" atau pewarisan. Misalnya, kelas "Mobil" adalah generalisasi dari kelas "Sedan".
    - Realization:
      Merupakan hubungan antara interface dan kelas yang mengimplementasikannya.
    - Association:
      Merupakan hubungan antara dua kelas yang saling terkait. Misalnya, kelas "Mahasiswa" memiliki hubungan "mengambil" dengan kelas "Mata Kuliah".
    - Dependency:
      Merupakan hubungan yang lebih umum, di mana perubahan pada satu elemen dapat mempengaruhi elemen lainnya.

3. Pemrograman terstruktur adalah pemrograman yang lebih berfokus pada urutan langkah-langkah (algoritma) untuk menyelesaikan masalah. Program ini dibagi menjadi fungsi-fungsi yang saling memanggil. Dengan kata lain, pemrograman terstruktur ini melihat masalah sebagai sekumpulan Langkah. Sedangkan Pemrograman berorientasi objek lebih berfokus pada objek-objek yang memiliki atribut dan perilaku. Program dibagi menjadi kelas-kelas yang saling berinteraksi. Dengan kata lain, pemrograman berorientasi objek melihat masalah sebagai sekumpulan objek yang berinteraksi.

4. Konsep objek adalah fondasi dari pemrograman berorientasi objek. Dengan memahami objek, kita dapat membangun aplikasi yang lebih kompleks, terstruktur, dan mudah dipelihara. Yang berarti, dalam pemrograman berorientasi objek, objek adalah representasi dari suatu entitas dalam dunia nyata. Setiap objek memiliki:
    - Atribut
      Atribut adalah karakteristik atau sifat yang dimiliki oleh objek, seperti data yang dimiliki oleh objek tersebut.
    - Metode
      Metode adalah tindakan atau perilaku yang dapat dilakukan oleh objek, seperti fungsi atau prosedur yang terkait dengan objek.

    Contoh-contoh Objek:
    1) Mobil:
       Atribut: Merek, model, warna, tahun pembuatan, jumlah pintu, kapasitas tangki.
       Metode: Menyalakan mesin, mengerem, berbelok, mempercepat, memperlambat.
    2) Buku:
       Atribut: Judul, penulis, penerbit, tahun terbit, jumlah halaman, ISBN.
       Metode: Membuka halaman, menutup buku, mencari kata.
    3) Akun Pengguna:
       Atribut: Username, password, email, nama lengkap, tanggal lahir.
       Metode: Login, logout, mengubah password, mengganti profil.
    4) Lingkaran:
       Atribut: Jari-jari, titik pusat.
       Metode: Menghitung luas, menghitung keliling.
    5) Pegawai:
       Atribut: Nama, nomor pegawai, jabatan, gaji.
       Metode: Masuk kerja, keluar kerja, mengambil cuti.

5. Access Modifiers adalah sebuah konsep di dalam pbo, dimana kita bisa mengatur hak akses suatu atribut dan fungsi pada sebuah class. Konsep ini juga bisa disebut sebagai enkapsulasi, dimana kita akan mendefinisikan mana atribut atau fungsi yang boleh diakses secara terbuka, mana yang bisa diakses secara terbatas, atau mana yang hanya bisa diakses oleh internal kelas alias privat.
   Konvensi Penamaan untuk Access Modifier di Python:
    - Public: Tidak ada penanda khusus. Semua atribut dan metode yang tidak diawali dengan dua underscore (__) dianggap publik dan dapat diakses dari mana saja.
    - Protected: Diawali dengan satu underscore tunggal (_). Menunjukkan bahwa atribut atau metode tersebut sebaiknya tidak diakses dari luar kelas, tetapi masih bisa diakses oleh kelas turunan. Ini lebih merupakan konvensi daripada aturan yang ketat.
    - Private: Diawali dengan dua underscore ganda (__). Menunjukkan bahwa atribut atau metode tersebut benar-benar bersifat privat dan hanya dapat diakses dari dalam kelas itu sendiri.
      Contoh:

```
class Mobil:
    def __init__(self, merek, model, tahun):
        self.merek = merek  # Public
        self._model = model  # Protected
        self.__tahun = tahun  # Private

    def get_model(self):
        return self._model

    def __umur(self):  # Private method
        tahun_sekarang = 2023  # Contoh tahun saat ini
return tahun_sekarang - self.__tahun
```

Penjelasan:

- merek: Atribut publik yang dapat diakses dari mana saja.
- _model: Atribut yang dianggap protected. Sebaiknya diakses melalui metode get_model().
- __tahun: Atribut privat yang hanya dapat diakses dari dalam kelas Mobil. Metode __umur() juga privat dan hanya dapat dipanggil dari dalam kelas.

6. Contoh class diagram pewarisan