Nama : Hafizh Habiibi Lubis

NPM : 5230411296

Kelas : Pemrograman Berbasis Objek VIII

Prodi : Informatika

Jawab

1. Jelaskan perbedaan use case diagram dengan class diagram?

   Use case diagram
   Use case diagram merupakan salah satu jenis diagram UML (Unified Modelling Language) yang digunakan untuk menggambarkan interaksi antara pengguna atau user dengan sistem. Use case diagram memiliki berbagai macam notasi yaitu:

   - Aktor
     Notasi aktor digambarkan dengan orang stik dengan nama yang digunakan untuk menggambarkan pengguna atau user yang berinteraksi atau mengakses sistem.

   - Use Case
     Notasi use case digambarkan dengan oval atau elips dan terdapat nama di dalam maupun di bawah setiap oval atau elips. Notasi ini digunakan untuk menggambarkan fungsi yang ada pada sistem yang nantinya akan digunakan untuk berinteraksi dengan aktor.

   - Asosiasi Komunikasi
     Notasi asosiasi komunikasi digambarkan dengan sebuah garis yang menghubungkan antara aktor dengan use case. Notasi ini digunakan untuk menunjukkan interaksi antara aktor dan use case.

   - Subjek
     Notasi subjek digambarkan dengan sebuah persegi panjang yang didalamnya terdapat use case yang termasuk pada subjek yang sama. Notasi ini digunakan untuk mewakili batasan sistem yang sedang dimodelkan.

   - Dependensi
     Notasi dependensi digambarkan dengan garis putus putus dan panah. Notasi ini digunakan untuk menunjukan hubungan extend dan iclude antara use case (ketergantungan).

   Class diagram
   Class diagram merupakan salah satu diagram UML (Unified Modeling Language) yang digunakan untuk menggambarkan struktur sebuah sistem yang berfokus pada kelas yang ada di dalam sistem tersebut, selain itu juga class diagram digunakan untuk merancang struktur sistem secara detail. Class diagram memiliki notasi seperti kelas yang digambarkan sebagai persegi panjang yang dibagi menjadi tiga bagian yaitu (nama kelas, atribut, dan method), notasi hubungan seperti (asosiasi, agregasi, komposisi, dan generalisasi), serta notasi visibilitas yaitu (public (+), private (-), dan protected (#).

2. Jelaskan jenis-jenis dependensi?
    - Usage Dependency
      Digunakan untuk menggambarkan bahwa sebuah elemen memerlukan elemen lain untuk berfungsi dengan baik. Dependeny ini digambarkan degan garis putus putus dengan panah dari elemen yang mengarah ke elemen yang dibutuhkan.

    - Abstraction Dependency
      Digunakan untuk menunjukan hubungan antar elemen yang abstarak dengan elemen yang konkret. Dependency ini digambarkan dengan garis putus putus dengan anah dari elemen konkret menuju elemen abstarak.

    - Realization Dependency
      Digunakan untuk menunukan suatu elemen (antarmuka) direalisasikan oleh elemen lain (kelas atau komponen), Dependency ini digambarkan dengan garis putuh [utus dengan panah dari elemen yang direalisasika ke elemen yang merealisasikan.

    - Permission Dependency
      Digunakan untuk menunjukan bahwa suatu elemen memberikan izin kepada elemen lain utuk mengakses elemen tersebu. Dependency ini digambarkan dengan garis putus putus dan panah dari elemen yang memberikan izin ke yang menerima izin.

    - Deployment Dependency
      Digunakan untuk menunjukan hubungan antara element perangkat lunak dan perangkat keras yang merupakan tempat elemen perangkat lunak tersebut dideploy nantinya. Dependency ini digambarkan dengan garis putus putus dengan panah dari elemen perangkat lunka ke elemen perangkat keras.

    - Trace Dependency
      Digunakan untuk menunjukan hubungan pelacakan atntara dua elemen yang biasanya digunakan untuk melacak perubahan suatu elemen. Dependency ini digambarkan dengan gars putus putus denganpanah dari elemen yang dilacak ke elemen yang melacak.

3. Apa perbedaan pemrograman terstruktur dengan berorientasi objek, jelaskan?

    - Pemrograman terstruktur berfokus pada pemecahan masalah dengan membagi progam menjadi prosedur atau fungsi yang kecil kecil dan setiap prosedur atau fungsi ini memiliki tugas yang spesifik, Pemrograman terstruktur biasanya digunakan pada masalah yang lebih sederhana karena mudah untuk depitakan dalam alur yang jelas.

    - Pemrograman berorietasi objek (OOP) berfokus pada manipulasi objek dengan mengelompokkan kode berdasarkan objek yang merupakan gabungan antara atribt dan method. OOP cocok untuk digunakan dalam pembuatan program yang berskala besar dan kompleks karena OOP memiliki empat konsep inti yatu enkapsulasi, abstraksi, pewarisan, dan polimorfisme. Enkapsulasi digunaan untuk menggambungkan atribut dan method dalam satu Kelas dan memberikan batas aksesnya. Abstraksi digunakan untuk menyederhanakan dan menyembunyikan detail tertentu, pewarisan digunakan untuk memungkinkan sebuah kelas anak untuk mewarisi sifat dari kelas parentnya. Polimorfisme digunakan untuk memungkinkan sebuah objek untuk berperilaku berbeda ketia dipanggil dalam konteks berbeda pula.

4. Jelaskan konsep objek dan beri contohnya?
   Kosep objek dalam OOP yaitu representasi dari sebuah hal yang memiliki dua karakteristik utama yaitu atribut dan method, atribut merupakan informasi yang dimiliki oleh sebuah objek sedangkan method meruakan fungsi atau hal yang dapat dilakukan oleh objek tersebut. Objek dibuat berdasarkan cetakan atau blueprint yang disebut sebagai kelas. Contoh dari sebuah objek yaitu sebuah kelas bernama proyektor yang memiliki atribut merek, warna, dan resolusi selain itu kelas ini memiliki method hidupkan(), matikan(), freeze(). Ketika kita membuat objek contohnya proyektor1 maka objek tersebut dapat memiliki atribut merek = “Samsung”, warna = “putih”, dan resolusi = “4K” dan ketika suatu method misalnya hidupkan() dipanggil pada objek proyektor 1 maka objek tersebut akan menjalankan instruksi untuk menyalakan proyektor tersebut.

5. Jelaskan jenis-jenis access modifier beri contohnya dalam baris pemrograman
   - Public
     Atribut atau method yang ditandai sebagai Public dapat diakses dari mana saja dalam sebuah program.

```python
class Proyektor:
    def __init__(self, warna, merek, resolusi):
        self.warna = warna
        self.merek = merek
        self.resolusi = resolusi

    def tampil(self):
        print(f"Proyektor dengan Warna : {self.warna}, Merek : {self.merek}, Resolusi {self.resolusi}")

proyektor1 = Proyektor("Putih", "Samsung", "8K")
proyektor1.tampil()
```

   - Private
     Atribut atau method yang ditandai sebagai Private hanya dapat diakses di dalam kelas dari atribut atau method tersebut, untuk mengakses dari luar kelas dibutuhkan cara tambahan seperti getter dan setter.

```python
class Elektronik:
    def __init__(self, tipe):
        self.__tipe = tipe

class Motor(Elektronik):
    def __init__(self, tipe):
        super().__init__(tipe)

    def set_tipe(self, tipe):
        self.__tipe = tipe
        print(f"Kendaraan bertipe {self.__tipe}")

motor = Motor("Bensin")
motor.set_tipe("Listrik")
```

- Protected
  Atribut atau method yang ditandai sebagai Protected hanya dapat diakses pada kelas dari atribut atau method tersebut dan juga kelas turunannya.

```python
class Elektronik:
    def __init__(self, tipe):
        self._tipe = tipe

class Motor(Elektronik):
    def __init__(self, tipe):
        super().__init__(tipe)

    def set_tipe(self, tipe):
        self._tipe = tipe
        print(f"Kendaraan bertipe {self._tipe}")

motor = Motor("Bensin")
motor.set_tipe("Listrik")
```

6. Gambarkan contoh pewarisan dalam diagram class?