Quiz OOP 1

Oleh :

NAME : Maulana Dwi Cahyono

CLASS : 1I

NO.ABSENT: 14

Major : Information Technology

STUDY PROGRAM : Information Engineering

Quiz 1

Class and Object:

1. Dalam pemrograman berorientasi objek, "class" adalah blueprint atau cetak biru yang mendefinisikan struktur atau karakteristik suatu objek. Class berisi atribut (variabel) dan metode (fungsi) yang digunakan untuk membuat objek. Dengan kata lain, class adalah entitas yang digunakan untuk membuat objek, dan itu menentukan bagaimana objek tersebut akan bekerja dan berinteraksi.
2. NamaClass namaObjek = new NamaClass();
   Di mana "NamaClass" adalah nama class yang ingin Anda instansiasi menjadi objek, "namaObjek" adalah nama yang Anda berikan kepada objek yang baru dibuat, dan "new NamaClass()" adalah perintah untuk membuat objek baru berdasarkan class tersebut.
3. Barang laptop = new Barang();
   Dengan ini, Anda telah membuat objek "laptop" yang merupakan instance dari class "Barang" dan dapat mengakses atribut dan metode yang telah didefinisikan dalam class tersebut untuk mengelola informasi dan perilaku objek "laptop" tersebut dalam sistem inventaris.

Encapsulation

1. Konsep encapsulation dalam pemrograman berorientasi objek adalah praktik mengenkapsulasi atau mengemas atribut (variabel) dan metode (fungsi) dalam sebuah class sehingga hanya dapat diakses melalui metode yang ditentukan dan dibatasi oleh class

tersebut. Hal ini penting dalam pengembangan sistem informasi inventaris barang karena melalui encapsulation, Anda dapat menjaga integritas data dan menerapkan kontrol akses yang ketat terhadap atribut. Dengan cara ini, Anda dapat memastikan bahwa data inventaris barang hanya dapat diubah melalui metode yang telah ditetapkan, sehingga mengurangi risiko kesalahan atau perubahan yang tidak diinginkan. Ini juga membantu dalam mengamankan data dan mencegah manipulasi yang tidak sah.

2. - Nama Barang: Menggunakan encapsulation memungkinkan Anda untuk memastikan bahwa nama barang hanya dapat diakses dan diubah melalui metode yang sesuai, mencegah perubahan yang tidak diinginkan.
   - Harga Barang: Data harga barang harus di-encapsulate untuk menghindari perubahan yang tidak sah dan memastikan bahwa perubahan hanya dapat dilakukan melalui prosedur yang telah ditentukan.
   - Jumlah Stok: Atribut ini perlu di-encapsulate untuk menghindari masalah sinkronisasi dan memastikan bahwa perubahan jumlah stok hanya terjadi dengan benar melalui metode yang memperhitungkan validasi dan logika bisnis.
   - Kategori Barang: Menggunakan encapsulation dapat membantu dalam memastikan bahwa setiap barang hanya termasuk dalam kategori yang sah, menghindari kategori yang salah atau tidak valid.

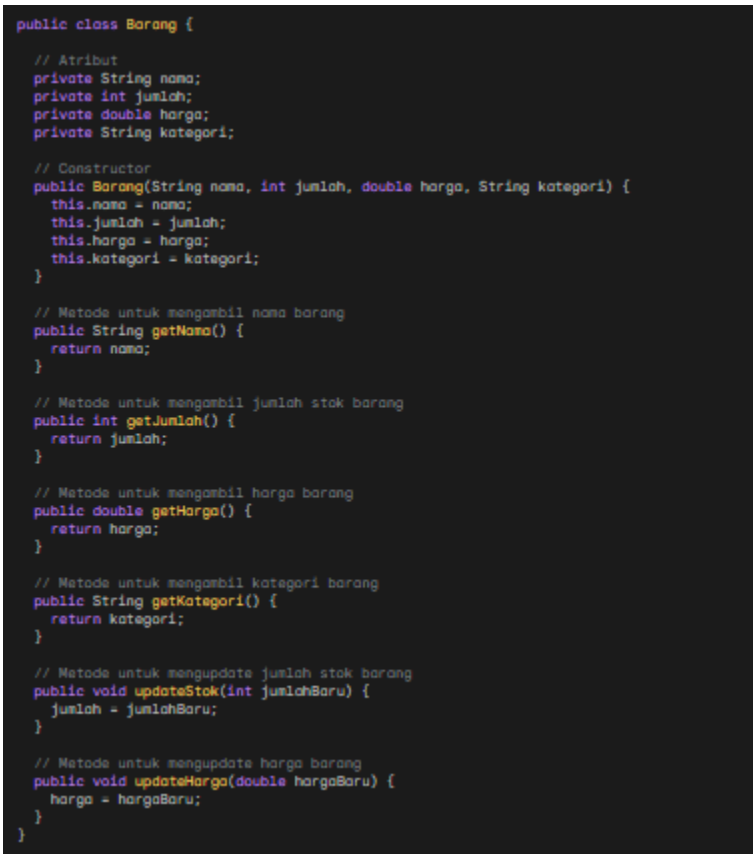
Relasi Kelas:

1. Relasi antara kelas dalam pemrograman berorientasi objek mengacu pada hubungan atau interaksi yang terjadi antara dua

atau lebih kelas dalam sebuah program. Ini mencerminkan cara kelas-kelas tersebut berkolaborasi atau berinteraksi satu sama lain untuk mencapai tujuan tertentu dalam aplikasi. Relasi kelas dapat berupa hubungan "has-a" (komposisi), "is-a" (inheritance), atau hubungan lain yang mencerminkan bagaimana kelas-kelas tersebut saling terkait dan berkontribusi dalam sistem.

2. Komposisi (Has-a Relationship): Setiap objek dari kelas "Barang" dapat memiliki atribut "Kategori" yang merujuk pada objek dari kelas "Kategori" yang mencerminkan kategori barang tersebut. Dengan kata lain, setiap barang terkait dengan satu kategori tertentu. Ini dapat diimplementasikan dengan menambahkan atribut "Kategori" ke dalam kelas "Barang" yang merupakan objek dari kelas "Kategori". Contohnya, setiap objek "Barang" memiliki atribut "Kategori" yang merupakan objek "Kategori" yang merujuk pada kategori barang tersebut, seperti "Elektronik", "Pakaian", atau "Alat Rumah Tangga". Dengan demikian, kita dapat melacak dan mengelola kategori barang dengan lebih baik dalam sistem inventaris.

PBL

1. 

```java
public class Barang {

  // Atribut
  private String nama;
  private int jumlah;
  private double harga;
  private String kategori;

  // Constructor
  public Barang(String nama, int jumlah, double harga, String kategori) {
    this.nama = nama;
    this.jumlah = jumlah;
    this.harga = harga;
    this.kategori = kategori;
  }

  // Metode untuk mengambil nama barang
  public String getNama() {
    return nama;
  }

  // Metode untuk mengambil jumlah stok barang
  public int getJumlah() {
    return jumlah;
  }

  // Metode untuk mengambil harga barang
  public double getHarga() {
    return harga;
  }

  // Metode untuk mengambil kategori barang
  public String getKategori() {
    return kategori;
  }

  // Metode untuk mengupdate jumlah stok barang
  public void updateStok(int jumlahBaru) {
    jumlah = jumlahBaru;
  }

  // Metode untuk mengupdate harga barang
  public void updateHarga(double hargaBaru) {
    harga = hargaBaru;
  }
}
```

2. Untuk menggunakan encapsulation dan melindungi atribut dalam class "Barang", atribut-atribut tersebut dinyatakan sebagai private, sehingga hanya dapat diakses melalui metode publik yang telah ditentukan, seperti getNama(), getJumlah(), getHarga(), dan getKategori(). Ini memastikan bahwa nilai-nilai atribut hanya dapat diubah melalui metode-metode yang telah disediakan, seperti updateStok() dan updateHarga(), yang memungkinkan validasi dan kontrol lebih lanjut atas perubahan nilai.
3. Hierarki class atau hubungan antar class dalam sistem informasi inventaris barang di jurusan Teknologi Informasi dapat mencakup beberapa kelas, seperti:

   Barang: Representasi objek barang dengan atribut seperti nama, jumlah, harga, dan kategori.

Kategori: Representasi objek kategori barang dengan atribut seperti nama kategori.
Penyedia: Representasi objek penyedia barang dengan atribut seperti nama penyedia, alamat, dan kontak.
Transaksi: Representasi objek transaksi dengan atribut seperti tanggal, daftar barang yang dibeli, dan total pembayaran.
Contoh relasi antar class dalam konteks ini bisa mencakup:

Association: Barang dapat berhubungan dengan Kategori melalui relasi "has-a" yang mengindikasikan setiap barang terkait dengan satu kategori tertentu.
Inheritance: Anda bisa memiliki kelas Turunan seperti "Laptop" yang merupakan subkelas dari "Barang" jika ingin menggambarkan jenis barang tertentu yang memiliki atribut tambahan atau metode khusus.