

Nama : Restu Lestari Mulianingrum

NIM : A11.2022.14668

Kelompok : A11.4415

1. Jelaskan dan berikan contoh apa yang dimaksud dengan :

a. Class, Object, Attribut, Method

- Class

Class adalah cetak biru atau blueprint dari sebuah objek. Class digunakan untuk membuat kerangka dasar. Hasil cetakan dari class adalah objek itu sendiri. Sebagai analogi, kita bisa mengibaratkan class dengan laptop atau notebook, yaitu gambaran umum tentang suatu benda.

Contoh penulisan class dalam bahasa Java:

```
class Laptop {  
    // Isi dari class Laptop...  
}
```

- Object

Objek adalah hasil cetakan dari class. Jika menggunakan analogi class Laptop, maka objek dari class Laptop bisa berupa “laptopRestu”

```
public class Laptop {  
    public static void main(String[] args) {  
        // menciptakan object dengan nama 'laptopRestu'  
        Laptop laptopRestu = new Laptop();  
        laptopRestu.cetakNama("Grey");  
    }  
}
```

- Attribut

Attribut atau kadang juga disebut field adalah data yang terdapat dalam sebuah class. Contoh attribut dari class Laptop bisa berupa merk, warna, jenis processor, ukuran layar, dan lain-lain. Contoh penulisan class dengan attribut:

```
class Laptop {  
    String pemilik;
```

```
String merk;  
double ukuranLayar;  
}
```

- Method

Method adalah tindakan yang bisa dilakukan di dalam class. Contoh method pada class Laptop bisa berupa menghidupkan laptop, mematikan laptop, atau mengganti cover laptop.

```
class Laptop {  
    String merk;  
    String warna;  
    int tahunProduksi;  
  
    void hidupkanLaptop() {  
        System.out.println("Laptop dinyalakan.");  
    }  
  
    void matikanLaptop() {  
        System.out.println("Laptop dimatikan.");  
    }  
  
    void gantiCover(String warnaBaru) {  
        warna = warnaBaru;  
        System.out.println("Cover laptop diganti menjadi " +  
warnaBaru + ".");  
    }  
}  
  
public class Main {  
    public static void main(String[] args) {  
        // Membuat objek laptop  
        Laptop laptopRestu = new Laptop();  
        laptopRestu.merk = "Lenovo";  
        laptopRestu.warna = "Grey";  
        laptopRestu.tahunProduksi = 2022;  
  
        // Menggunakan metode pada objek  
        laptopRestu.hidupkanLaptop();  
    }  
}
```

```
        laptopRestu.gantiCover("Merah");  
        laptopRestu.matikanLaptop();  
    }  
}
```

b. Encapsulation

Encapsulation adalah konsep dalam pemrograman berorientasi objek yang menggabungkan data (attribut) dan metode (method) yang beroperasi pada data tersebut menjadi satu kesatuan. Tujuannya adalah untuk menyembunyikan detail implementasi dan membatasi akses langsung ke data internal suatu objek. Dalam Java, encapsulation dapat dicapai dengan menggunakan access modifiers seperti `private`, `protected`, dan `public`.

Berikut adalah contoh penggunaan encapsulation dalam sebuah kelas `Person`:

```
public class Person {  
    private String name;  
    private int age;  
  
    public String getName() {  
        return name;  
    }  
  
    public void setName(String name) {  
        this.name = name;  
    }  
  
    public int getAge() {  
        return age;  
    }  
  
    public void setAge(int age) {  
        this.age = age;  
    }  
}  
  
public class Main {  
    public static void main(String[] args) {
```

```

    Person person = new Person();
    person.setName("John");
    person.setAge(30);

    System.out.println("Name: " + person.getName());
    System.out.println("Age: " + person.getAge());
}
}

```

Dalam contoh di atas, kita menggunakan encapsulation dengan mendeklarasikan variabel instance kelas Person sebagai private, sehingga hanya dapat diakses di dalam kelas tersebut. Kemudian kita menggunakan getter dan setter untuk mengakses dan mengubah nilai variabel tersebut dari luar kelas

c. Inheritance

Inheritance (pewarisan) adalah konsep di mana sebuah class dapat mewarisi atribut dan metode dari class lain. Class yang mewarisi disebut subclass atau child class, sedangkan class yang memberikan warisan disebut superclass atau parent class. Contoh:

```

class Kendaraan {
    // Atribut dan metode untuk semua kendaraan
}

class Mobil extends Kendaraan {
    // Atribut dan metode khusus untuk mobil
}

```

d. Overloading

Overloading adalah kemampuan untuk mendefinisikan beberapa metode dengan nama yang sama di dalam sebuah class. Metode-metode ini memiliki parameter yang berbeda. Java membedakan metode berdasarkan jumlah dan tipe parameter. Contoh:

```

class Kalkulator {
    int tambah(int a, int b) {

```

```
        return a + b;
    }

    double tambah(double a, double b) {
        return a + b;
    }
}
```

2. Perhatikan code program berikut :

<pre>class Mhs{ String nim,nm; float tgs, uts, uas; Mhs(float a, float b, float c){ tgs=a;uts=b;uas=c; } void cetak() { System.out.println("Nama :"+ nm); System.out.println("Tgs :"+ tgs); System.out.println("Uts :"+ uts); System.out.println("Uts :"+ uas); } void cetak(String s){ System.out.println(s); } } class Nilai extends Mhs{ float na; Nilai(float l, float m, float n) { super(l,m,n);na=0; } }</pre>	<pre>void cetak(){ super.cetak("Mahasiswa"); super.cetak(); } void cetak(String x){ System.out.println(x); System.out.println("Nim = " + super.nim); System.out.println("Nama = " + super.nm); System.out.println("Tgs = " + tgs); System.out.println("Uts = " + uts); System.out.println("Uas = " + uas); System.out.println("NA = " + na); } public static void main(String args[]){ Nilai obj = new Nilai(90,90,85); Obj.nim="A11.";obj.nm="Najwa"; obj.na=(obj.tgs+obj.uts+obj.uas)/3; obj.cetak(); obj.cetak("Nilai"); }</pre>
---	---

Tentukan :

a. Konstruktor

- Kelas Mhs memiliki konstruktor Mhs(float a, float b, float c) yang menginisialisasi atribut tgs, uts, dan uas.
- Kelas Nilai memiliki konstruktor Nilai(float l, float m, float n) yang memanggil konstruktor kelas induk Mhs menggunakan super(l, m, n).

b. Inheritance

Kelas Nilai meng-extend (mewarisi) dari kelas Mhs. Ini berarti bahwa kelas Nilai mewarisi semua properti (field dan method) dari kelas Mhs. Inheritance memungkinkan kelas Nilai menggunakan kembali kode yang didefinisikan di kelas Mhs tanpa menggandakannya.

c. Overloading dan Overriding:

- Method Overloading:

Kelas Mhs mendefinisikan dua metode cetak:

1. Metode 'cetak()' tanpa parameter, mencetak nama mahasiswa, nilai tugas (tgs), dan nilai ujian (uts dan uas).
2. Metode 'cetak(String s)' dengan satu parameter bertipe String.

Kelas Nilai mendefinisikan dua metode cetak:

1. Metode 'cetak()' tanpa parameter yang diwarisi dari kelas Mhs.
 2. Metode 'cetak(String x)' yang didefinisikan di kelas Nilai.
- Method Overriding

Kelas Nilai melakukan overriding terhadap metode cetak yang diwarisi dari kelas Mhs. Metode cetak yang di-override di kelas Nilai mencetak “Mahasiswa” (menandakan jenis mahasiswa) dan kemudian memanggil metode cetak dari superclass untuk mencetak nama mahasiswa dan nilai. Metode cetak(String x) di kelas Nilai juga mencetak informasi tambahan terkait nilai mahasiswa.

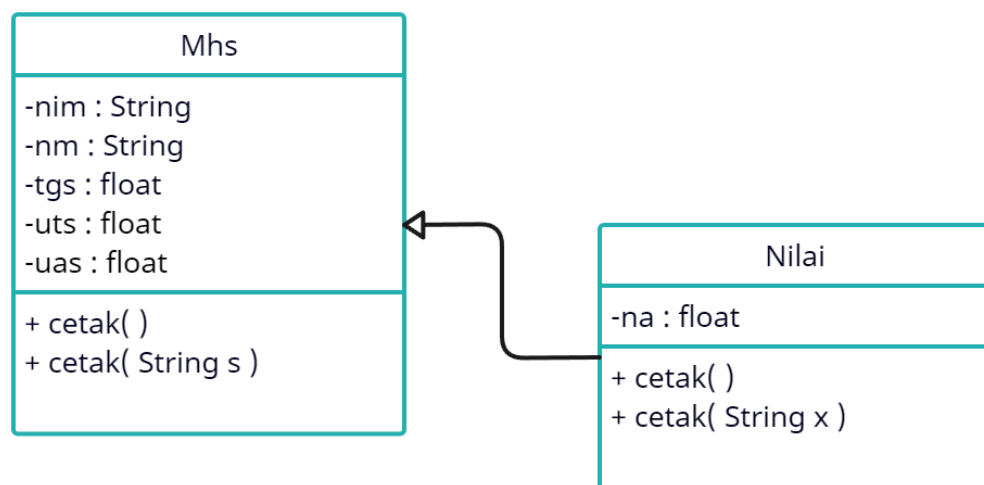
d. Hasil Program :

```
Command Prompt
D:\Kuliah\Semester 4\PB0>javac Nilai.java

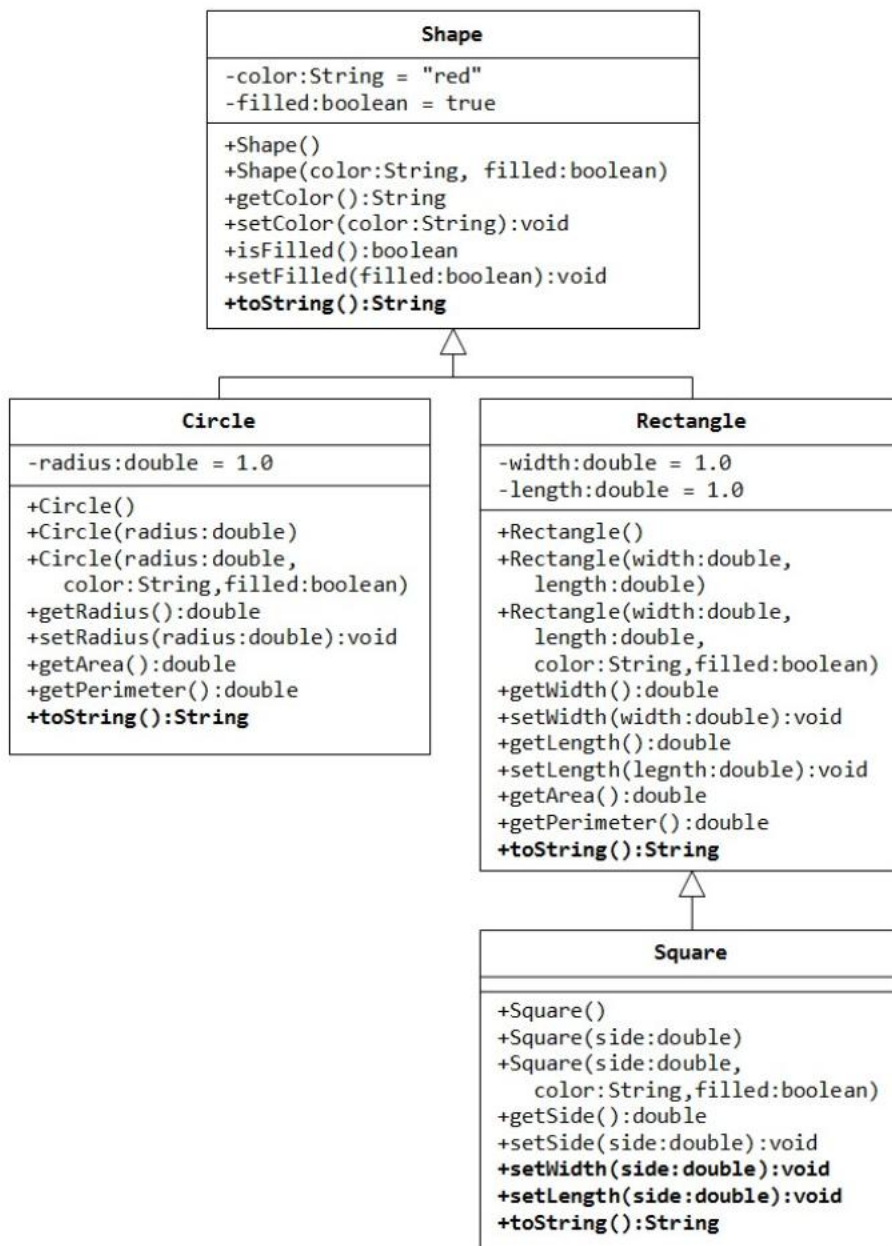
D:\Kuliah\Semester 4\PB0>java Nilai
Mahasiswa
Nama: Najwa
Tgs: 90.0
Uts: 90.0
Uas: 85.0
Nilai
Nim = A11
Nama = Najwa
Tgs = 90.0
Uts = 90.0
Uas = 85.0
NA = 88.333336

D:\Kuliah\Semester 4\PB0>
```

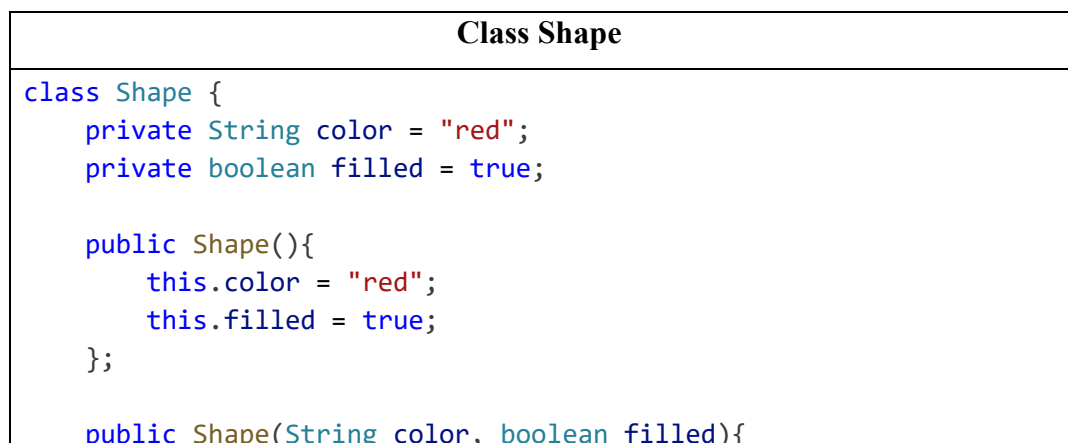
e. Gambar Class Diagram



3. Perhatikan class diagram berikut :



a. Buatlah design class dari class diagram di atas!




```

        this.color = color;
        this.filled = filled;
    }

    public String getColor(){
        return color;
    }

    public void setColor (String color){
        this.color = color;
    }

    public boolean isFilled(){
        return filled;
    }

    public void setFilled (boolean filled){
        this.filled = filled;
    }

    public String toString() {
        return "Shape[color=" + color + ", filled=" + filled +
        "]\n";
    }
}

```

Class Circle

```

class Circle extends Shape{
    private double radius = 1.0;

    public Circle(){
        this.radius = 1.0;
    }

    public Circle(double radius){
        this.radius = radius;
    }

    public Circle(double radius, String color, boolean filled){
        super(color, filled);
        this.radius = radius;
    }

    public double getRadius(){
        return radius;
    }
}

```

```

    public void setRadius(double radius){
        this.radius = radius;
    }

    public double getArea(){
        return Math.PI * radius * radius;
    }

    public double getPerimeter() {
        return 2 * Math.PI * radius;
    }

    public String toString() {
        return "Circle[Shape" + super.toString() + ", radius=" +
radius + "]";
    }
}

```

Class Rectangle

```

class Rectangle extends Shape {
    private double width;
    private double length;

    public Rectangle() {
        this.width = 1.0;
        this.length = 1.0;
    }

    public Rectangle(double width, double length) {
        this.width = width;
        this.length = length;
    }

    public Rectangle(double width, double length, String color,
boolean filled) {
        super(color, filled);
        this.width = width;
        this.length = length;
    }

    public double getWidth() {
        return width;
    }

    public void setWidth(double width) {

```

```

        this.width = width;
    }

    public double getLength() {
        return length;
    }

    public void setLength(double length) {
        this.length = length;
    }

    public double getArea() {
        return width * length;
    }

    public double getPerimeter() {
        return 2 * (width + length);
    }

    @Override
    public String toString() {
        return "Rectangle[Shape" + super.toString() + ", width=" +
width + ", length=" + length + "];"
    }
}

```

Class Square

```

class Square extends Rectangle {
    public Square() {
        // Konstruktor tanpa parameter
    }

    public Square(double side) {
        super(side, side);
    }

    public Square(double side, String color, boolean filled) {
        super(side, side, color, filled);
    }
}

```

- b. Berikan contoh penerapan untuk mencetak Rectangle yang ada di design class pada soal (a) dengan output :

Warna :

Tinggi :

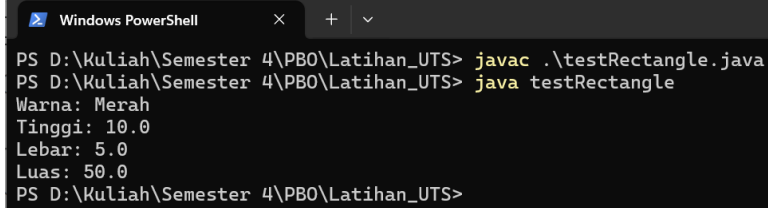
Lebar :

Luas :

testRectangle.java

```
public class testRectangle {  
    public static void main(String[] args) {  
        Rectangle rectangle = new Rectangle();  
        rectangle.setColor("Merah");  
        rectangle.setWidth(5.0);  
        rectangle.setLength(10.0);  
  
        System.out.println("Warna: " + rectangle.getColor());  
        System.out.println("Tinggi: " + rectangle.getLength());  
        System.out.println("Lebar: " + rectangle.getWidth());  
        System.out.println("Luas: " + rectangle.getArea());  
    }  
}
```

Output



```
Windows PowerShell  
PS D:\Kuliah\Semester 4\PBO\Latihan_UTS> javac .\testRectangle.java  
PS D:\Kuliah\Semester 4\PBO\Latihan_UTS> java testRectangle  
Warna: Merah  
Tinggi: 10.0  
Lebar: 5.0  
Luas: 50.0  
PS D:\Kuliah\Semester 4\PBO\Latihan_UTS>
```