Object Oriented Programming Overloading And Overriding

Name

Virza Aulia Rachman

NIM

2241720078

Class

1i

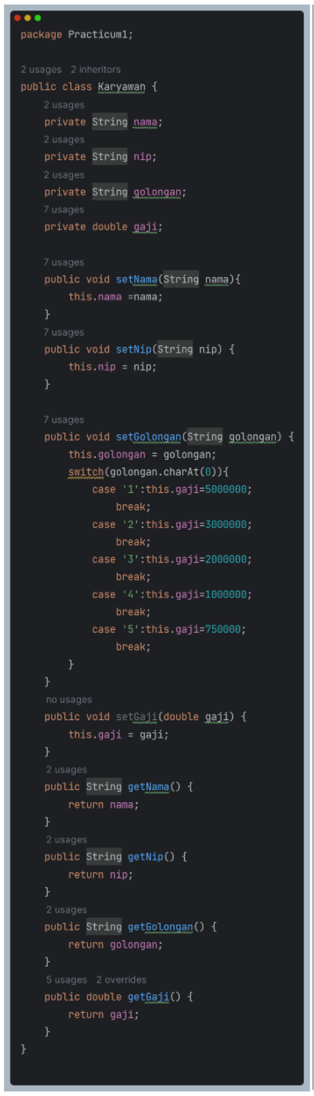
Department

Information Technology

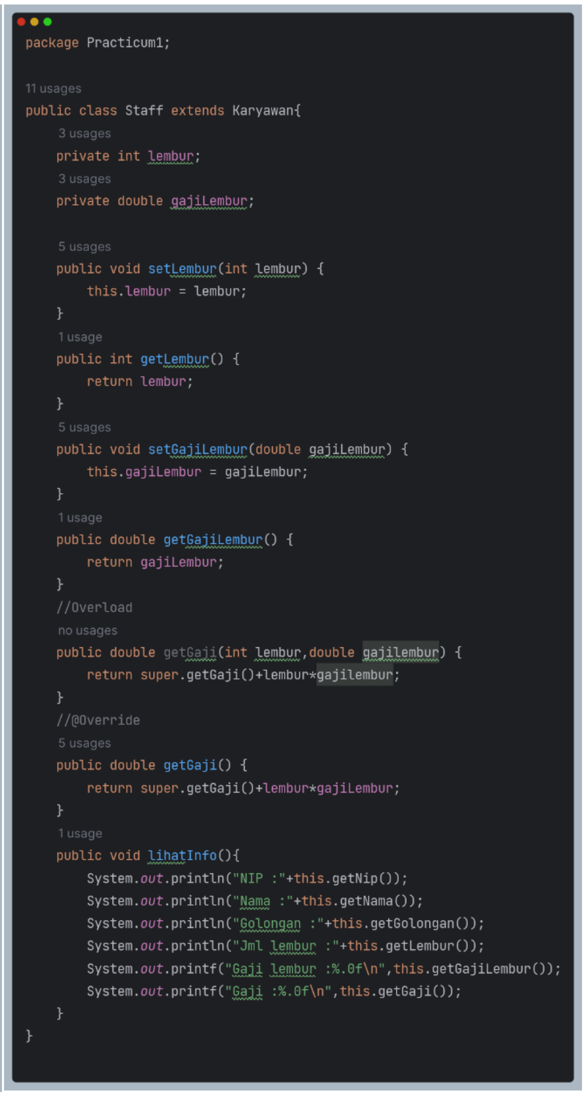
Study Program

D4 Informatics Engineering

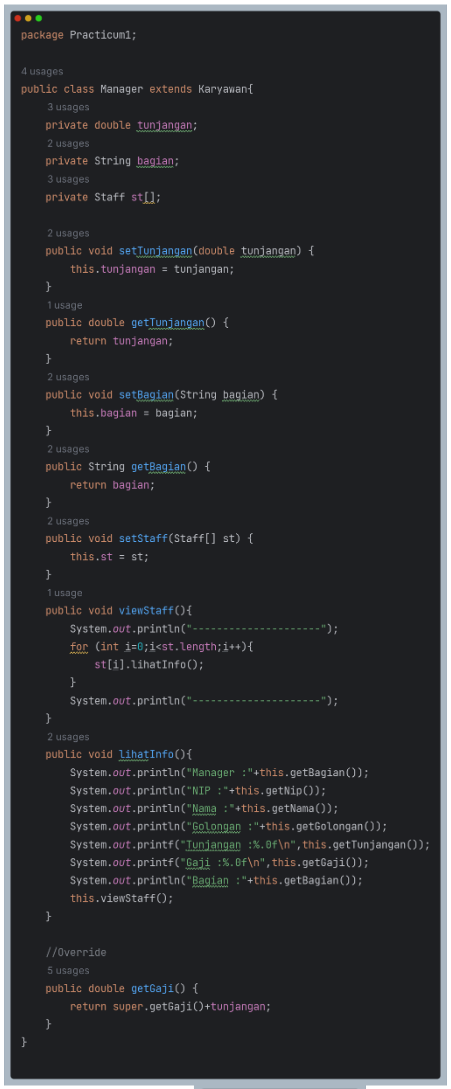
Practicum 1



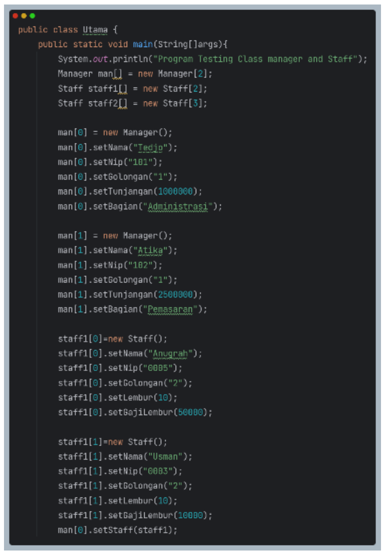
```java
package Practicum1;

2 usages   2 inheritors
public class Karyawan {
    2 usages
    private String nama;
    2 usages
    private String nip;
    2 usages
    private String golongan;
    7 usages
    private double gaji;

    7 usages
    public void setNama(String nama){
        this.nama =nama;
    }
    7 usages
    public void setNip(String nip) {
        this.nip = nip;
    }

    7 usages
    public void setGolongan(String golongan) {
        this.golongan = golongan;
        switch(golongan.charAt(0)){
            case '1':this.gaji=5000000;
                break;
            case '2':this.gaji=3000000;
                break;
            case '3':this.gaji=2000000;
                break;
            case '4':this.gaji=1000000;
                break;
            case '5':this.gaji=750000;
                break;
        }
    }
    no usages
    public void setGaji(double gaji) {
        this.gaji = gaji;
    }
    2 usages
    public String getNama() {
        return nama;
    }
    2 usages
    public String getNip() {
        return nip;
    }
    2 usages
    public String getGolongan() {
        return golongan;
    }
    5 usages   2 overrides
    public double getGaji() {
        return gaji;
    }
}
```



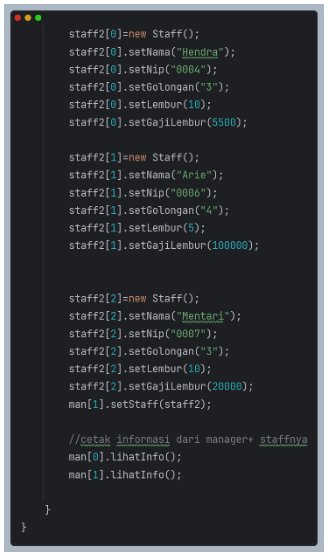
```java
package Practicum1;

11 usages
public class Staff extends Karyawan{
    3 usages
    private int lembur;
    3 usages
    private double gajiLembur;

    5 usages
    public void setLembur(int lembur) {
        this.lembur = lembur;
    }
    1 usage
    public int getLembur() {
        return lembur;
    }
    5 usages
    public void setGajiLembur(double gajiLembur) {
        this.gajiLembur = gajiLembur;
    }
    1 usage
    public double getGajiLembur() {
        return gajiLembur;
    }
    //Overload
    no usages
    public double getGaji(int lembur,double gajilembur) {
        return super.getGaji()+lembur*gajilembur;
    }
    //@Override
    5 usages
    public double getGaji() {
        return super.getGaji()+lembur*gajiLembur;
    }
    1 usage
    public void lihatInfo(){
        System.out.println("NIP :"+this.getNip());
        System.out.println("Nama :"+this.getNama());
        System.out.println("Golongan :"+this.getGolongan());
        System.out.println("Jml lembur :"+this.getLembur());
        System.out.printf("Gaji lembur :%.0f\n",this.getGajiLembur());
        System.out.printf("Gaji :%.0f\n",this.getGaji());
    }
}
```



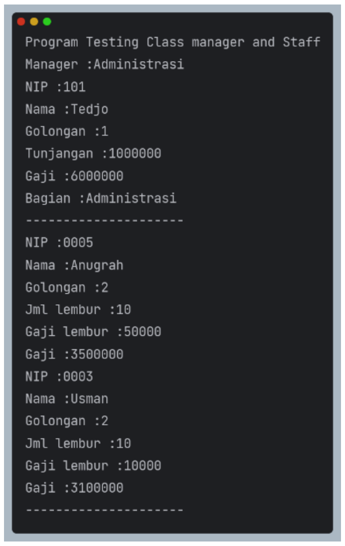
```java
package Practicum1;

4 usages
public class Manager extends Karyawan{
    3 usages
    private double tunjangan;
    2 usages
    private String bagian;
    3 usages
    private Staff st[];

    2 usages
    public void setTunjangan(double tunjangan) {
        this.tunjangan = tunjangan;
    }
    1 usage
    public double getTunjangan() {
        return tunjangan;
    }
    2 usages
    public void setBagian(String bagian) {
        this.bagian = bagian;
    }
    2 usages
    public String getBagian() {
        return bagian;
    }
    2 usages
    public void setStaff(Staff[] st) {
        this.st = st;
    }
    1 usage
    public void viewStaff(){
        System.out.println("--------------------");
        for (int i=0;i<st.length;i++){
            st[i].lihatInfo();
        }
        System.out.println("--------------------");
    }
    2 usages
    public void lihatInfo(){
        System.out.println("Manager :"+this.getBagian());
        System.out.println("NIP :"+this.getNip());
        System.out.println("Nama :"+this.getNama());
        System.out.println("Golongan :"+this.getGolongan());
        System.out.printf("Tunjangan :%.0f\n",this.getTunjangan());
        System.out.printf("Gaji :%.0f\n",this.getGaji());
        System.out.println("Bagian :"+this.getBagian());
        this.viewStaff();
    }

    //Override
    5 usages
    public double getGaji() {
        return super.getGaji()+tunjangan;
    }
}
```



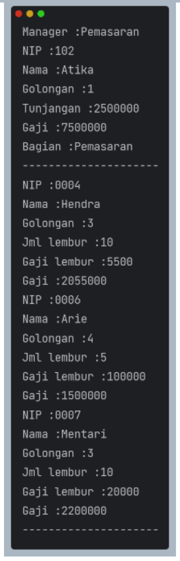
```java
public class Utama {
    public static void main(String[]args){
        System.out.println("Program Testing Class manager and Staff");
        Manager man[] = new Manager[2];
        Staff staff1[] = new Staff[2];
        Staff staff2[] = new Staff[3];

        man[0] = new Manager();
        man[0].setNama("Tedjo");
        man[0].setNip("101");
        man[0].setGolongan("1");
        man[0].setTunjangan(1000000);
        man[0].setBagian("Administrasi");

        man[1] = new Manager();
        man[1].setNama("Atika");
        man[1].setNip("102");
        man[1].setGolongan("1");
        man[1].setTunjangan(2500000);
        man[1].setBagian("Pemasaran");

        staff1[0]=new Staff();
        staff1[0].setNama("Anugrah");
        staff1[0].setNip("0005");
        staff1[0].setGolongan("2");
        staff1[0].setLembur(10);
        staff1[0].setGajiLembur(50000);

        staff1[1]=new Staff();
        staff1[1].setNama("Usman");
        staff1[1].setNip("0003");
        staff1[1].setGolongan("2");
        staff1[1].setLembur(10);
        staff1[1].setGajiLembur(10000);
        man[0].setStaff(staff1);
```



```java
        staff2[0]=new Staff();
        staff2[0].setNama("Hendra");
        staff2[0].setNip("0004");
        staff2[0].setGolongan("3");
        staff2[0].setLembur(10);
        staff2[0].setGajiLembur(5500);

        staff2[1]=new Staff();
        staff2[1].setNama("Arie");
        staff2[1].setNip("0006");
        staff2[1].setGolongan("4");
        staff2[1].setLembur(5);
        staff2[1].setGajiLembur(100000);


        staff2[2]=new Staff();
        staff2[2].setNama("Mentari");
        staff2[2].setNip("0007");
        staff2[2].setGolongan("3");
        staff2[2].setLembur(10);
        staff2[2].setGajiLembur(20000);
        man[1].setStaff(staff2);

        //cetak informasi dari manager+ staffnya
        man[0].lihatInfo();
        man[1].lihatInfo();

    }
}
```

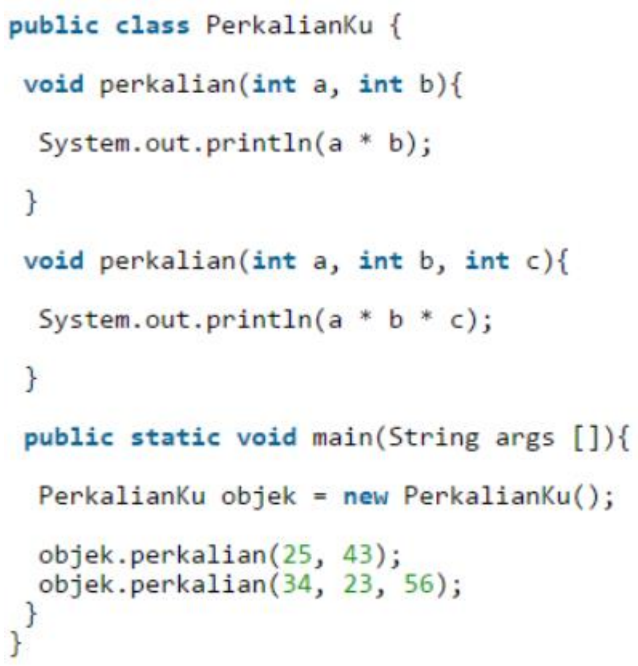


```
Program Testing Class manager and Staff
Manager :Administrasi
NIP :101
Nama :Tedjo
Golongan :1
Tunjangan :1000000
Gaji :6000000
Bagian :Administrasi
--------------------
NIP :0005
Nama :Anugrah
Golongan :2
Jml lembur :10
Gaji lembur :50000
Gaji :3500000
NIP :0003
Nama :Usman
Golongan :2
Jml lembur :10
Gaji lembur :10000
Gaji :3100000
--------------------
```



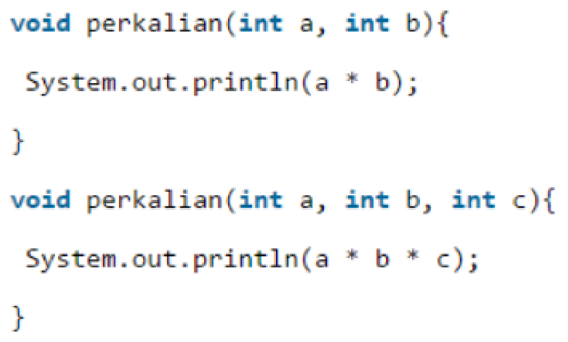
```
Manager :Pemasaran
NIP :102
Nama :Atika
Golongan :1
Tunjangan :2500000
Gaji :7500000
Bagian :Pemasaran
--------------------
NIP :0004
Nama :Hendra
Golongan :3
Jml lembur :10
Gaji lembur :5500
Gaji :2055000
NIP :0006
Nama :Arie
Golongan :4
Jml lembur :5
Gaji lembur :100000
Gaji :1500000
NIP :0007
Nama :Mentari
Golongan :3
Jml lembur :10
Gaji lembur :20000
Gaji :2200000
--------------------
```

Exercise



```
public class PerkalianKu {

 void perkalian(int a, int b){

  System.out.println(a * b);

 }

 void perkalian(int a, int b, int c){

  System.out.println(a * b * c);

 }

 public static void main(String args []){

  PerkalianKu objek = new PerkalianKu();

  objek.perkalian(25, 43);
  objek.perkalian(34, 23, 56);
 }
}
```

1. Dari source coding diatas terletak dimanakah overloading?

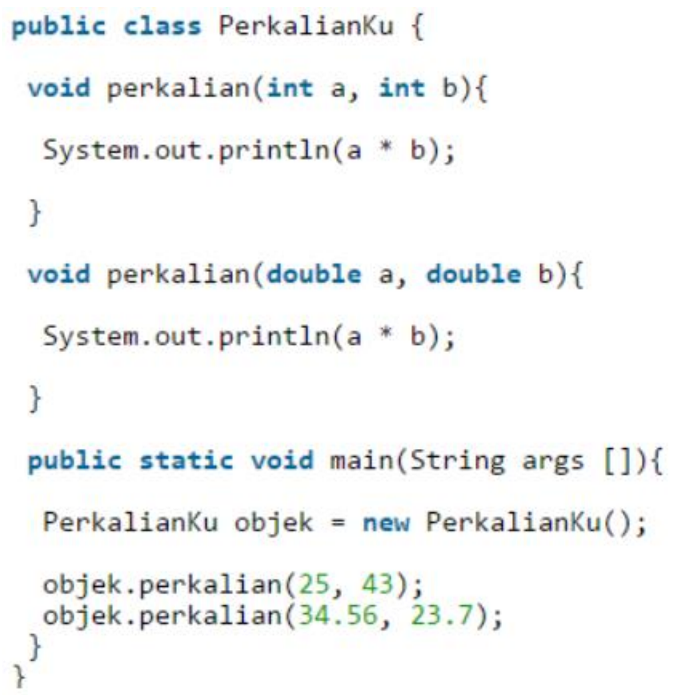


```
void perkalian(int a, int b){

 System.out.println(a * b);

}

void perkalian(int a, int b, int c){

 System.out.println(a * b * c);

}
```

2. Jika terdapat overloading ada berapa jumlah parameter yang berbeda?

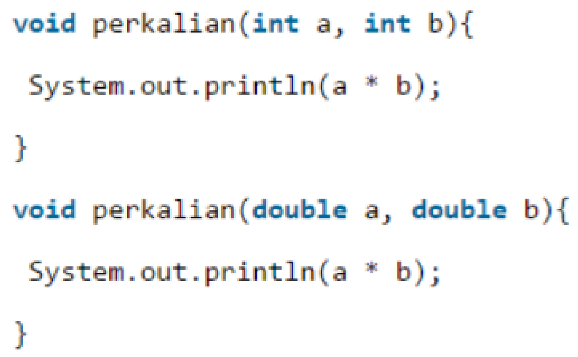
There are two different parameters for the overloaded perkalian() method:

- Two int parameters
- Three int parameters



```java
public class PerkalianKu {

 void perkalian(int a, int b){

  System.out.println(a * b);

 }

 void perkalian(double a, double b){

  System.out.println(a * b);

 }

 public static void main(String args []){

  PerkalianKu objek = new PerkalianKu();

  objek.perkalian(25, 43);
  objek.perkalian(34.56, 23.7);
 }
}
```

3. Dari source coding diatas terletak dimanakah overloading?

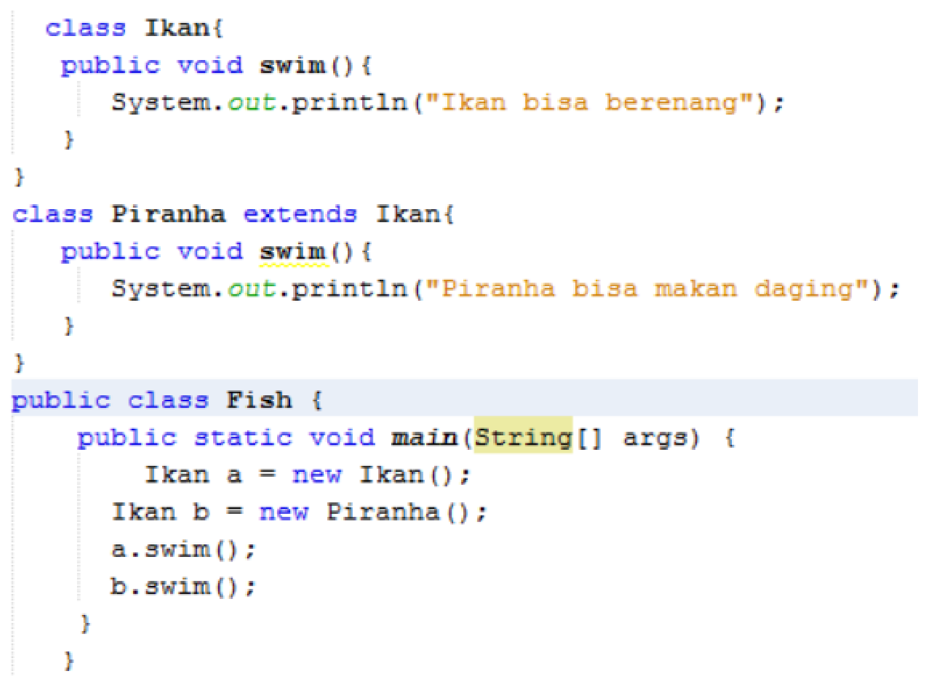


```java
void perkalian(int a, int b){

 System.out.println(a * b);

}

void perkalian(double a, double b){

 System.out.println(a * b);

}
```

4. Jika terdapat overloading ada berapa tipe parameter yang berbeda?

There are two different parameters for the overloaded perkalian() method:

- Two int parameters
- Two double parameters



```
  class Ikan{
   public void swim(){
      System.out.println("Ikan bisa berenang");
   }
}
class Piranha extends Ikan{
   public void swim(){
      System.out.println("Piranha bisa makan daging");
   }
}
public class Fish {
    public static void main(String[] args) {
        Ikan a = new Ikan();
      Ikan b = new Piranha();
      a.swim();
      b.swim();
    }
   }
```

5. Dari source coding diatas terletak dimanakah overriding?

The overriding is located in the Piranha class, where it provides its own implementation of the swim method, which overrides the swim method in the superclass Ikan.

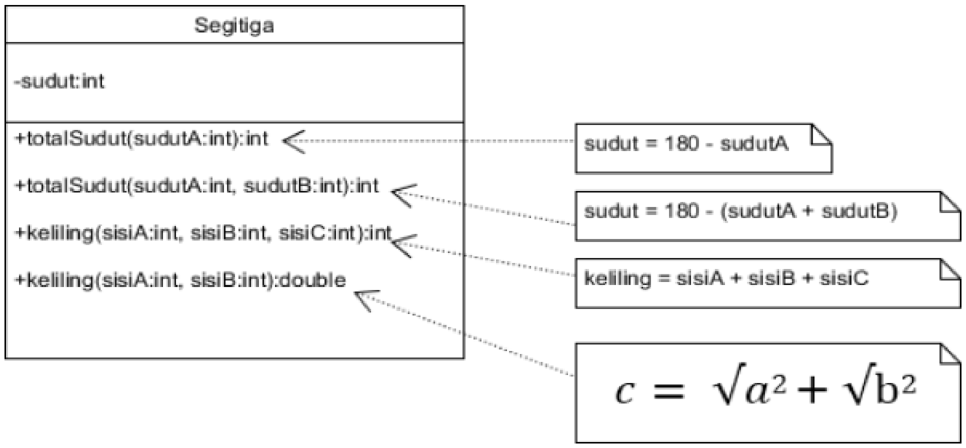
6. Jabarkanlah apabila sourcoding diatas jika terdapat overriding?

The Piranha class extends the Ikan class and provides its own implementation of the swim method. When you create an instance of Piranha and call the swim method on it, it executes the overridden method in the Piranha class. This is an example of method overriding in Java, where a subclass provides a specific implementation for a method that is already defined in its superclass. In this case, the swim method is overridden in the Piranha class to provide a more specialized behavior for piranhas.

Assignment

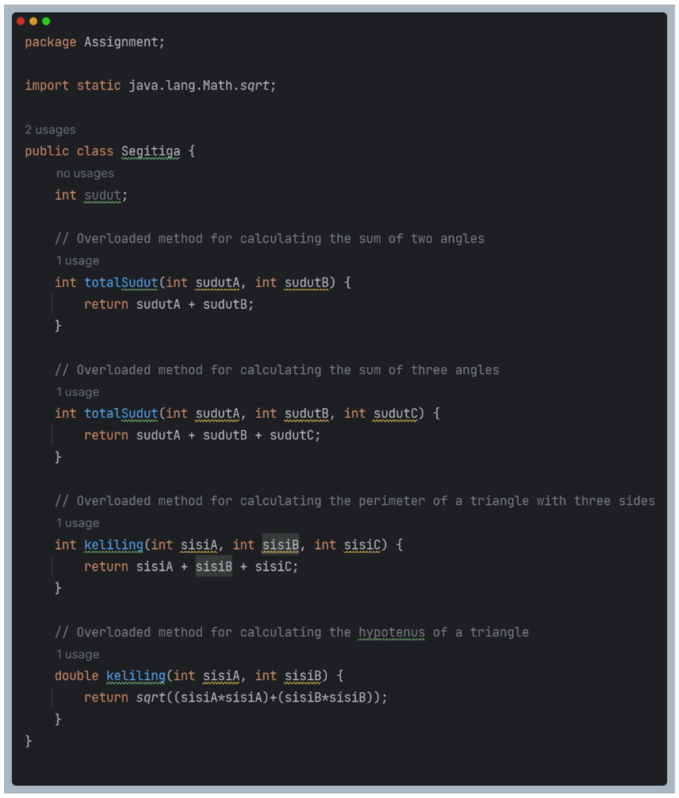
1. Overloading

Implementasikan konsep overloading pada class diagram dibawah ini :


Segitiga
-sudut:int
+totalSudut(sudutA:int):int
+totalSudut(sudutA:int, sudutB:int):int
+keliling(sisiA:int, sisiB:int, sisiC:int):int
+keliling(sisiA:int, sisiB:int):double
sudut = 180 - sudutA
sudut = 180 - (sudutA + sudutB)
keliling = sisiA + sisiB + sisiC
$c = \sqrt{a^2} + \sqrt{b^2}$

Answer:

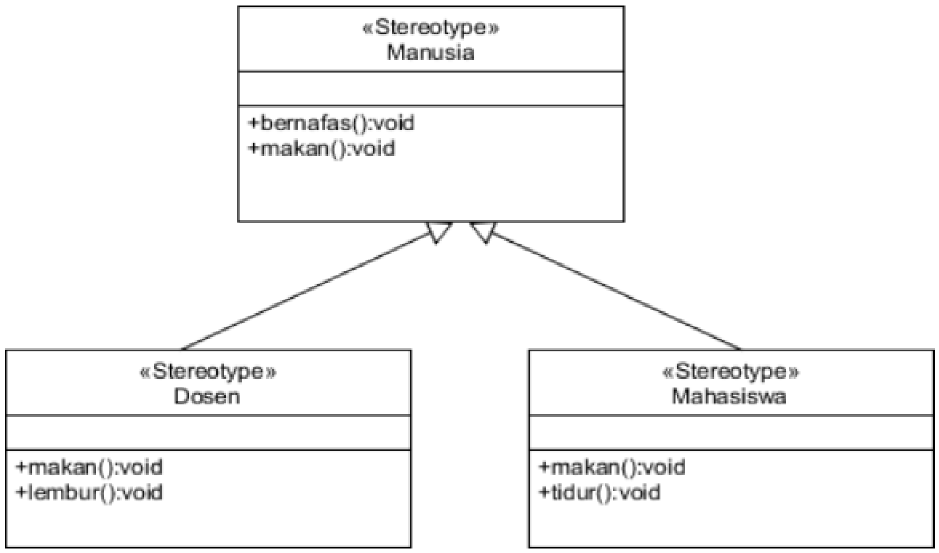
Segitiga.java



```
package Assignment;

import static java.lang.Math.sqrt;

2 usages
public class Segitiga {
    no usages
    int sudut;

    // Overloaded method for calculating the sum of two angles
    1 usage
    int totalSudut(int sudutA, int sudutB) {
        return sudutA + sudutB;
    }

    // Overloaded method for calculating the sum of three angles
    1 usage
    int totalSudut(int sudutA, int sudutB, int sudutC) {
        return sudutA + sudutB + sudutC;
    }

    // Overloaded method for calculating the perimeter of a triangle with three sides
    1 usage
    int keliling(int sisiA, int sisiB, int sisiC) {
        return sisiA + sisiB + sisiC;
    }

    // Overloaded method for calculating the hypotenus of a triangle
    1 usage
    double keliling(int sisiA, int sisiB) {
        return sqrt((sisiA*sisiA)+(sisiB*sisiB));
    }
}
```

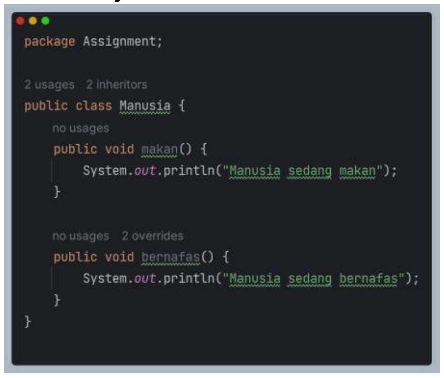
2. Overriding

Implementasikan class diagram dibawah ini dengan menggunakan teknik dynamic method dispatch :



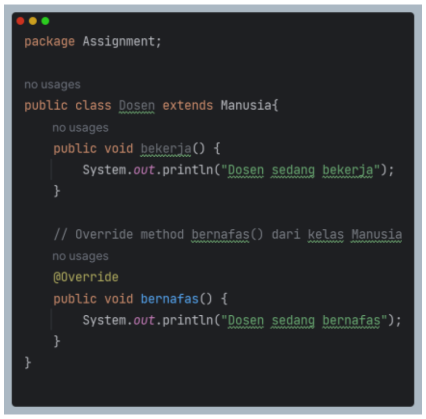
Answer:

Manusia.java



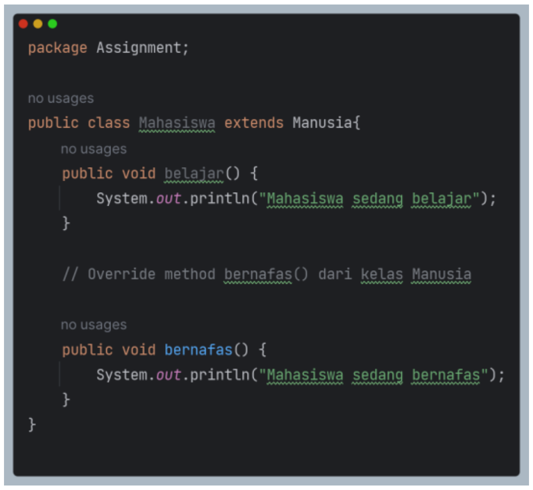
```
package Assignment;

2 usages  2 inheritors
public class Manusia {
    no usages
    public void makan() {
        System.out.println("Manusia sedang makan");
    }

    no usages  2 overrides
    public void bernafas() {
        System.out.println("Manusia sedang bernafas");
    }
}
```

Dosen.java



```
package Assignment;

no usages
public class Dosen extends Manusia{
    no usages
    public void bekerja() {
        System.out.println("Dosen sedang bekerja");
    }

    // Override method bernafas() dari kelas Manusia
    no usages
    @Override
    public void bernafas() {
        System.out.println("Dosen sedang bernafas");
    }
}
```

Mahasiswa.java



```
package Assignment;

no usages
public class Mahasiswa extends Manusia{
    no usages
    public void belajar() {
        System.out.println("Mahasiswa sedang belajar");
    }

    // Override method bernafas() dari kelas Manusia

    no usages
    public void bernafas() {
        System.out.println("Mahasiswa sedang bernafas");
    }
}
```