JOBSHEET 7

OVERLOADING DAN OVERRIDING

Link Github: https://github.com/Majid5654/Semester-3/tree/Main/JAVA%20OOP/Week7

Erwan Majid/08/2i

- **Percobaan 1**

Class karyawan:

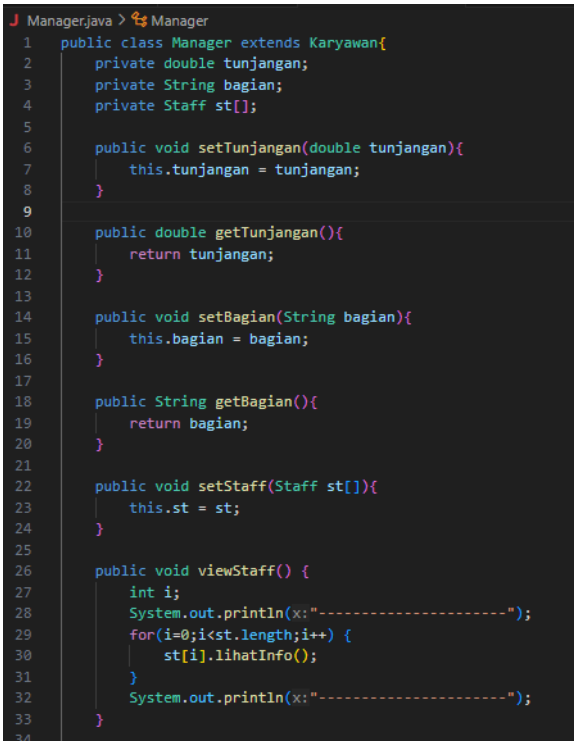
Karyawan.java > Karyawan > setGolongan(String)

```java
public class Karyawan {
    private String nama;
    private String nip;
    private String golongan;
    private double gaji;

    public void setNama(String nama){
        this.nama = nama;
    }

    public void setNip(String nip){
        this.nip = nip;
    }

    public void setGolongan(String golongan){
        this.golongan = golongan;

        switch (golongan.charAt(index: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;
```

Class staff:

```java
Staff.java > Staff > lihatInfo()
public class Staff extends Karyawan {
    private int lembur;
    private double gajiLembur;

    public void setLembur(int lembur){
        this.lembur = lembur;
    }

    public int getLembur(){
        return lembur;
    }

    public void setGajiLembur(double gajiLembur){
        this.gajiLembur = gajiLembur;
    }

    public double getGajiLembur(){
        return gajiLembur;
    }

    public double getGaji (int lembur,double gajiLembur){
        return super.getGaji() + (lembur*gajiLembur);
    }

    public double getGaji(){
        return super.getGaji() + (lembur*gajiLembur);
    }

    public void lihatInfo(){
        System.out.println("NIP :" +this.getNip());
        System.out.println("Nama :" +this.getNama());
        System.out.println("Golongan :" +this.getGolongan());
        System.out.println("Jumlah lembur :" +this.getLembur());
        System.out.printf(format:"Gaji Lembur : %.0f\n" ,this.getGajiLembur());
        System.out.printf(format:"Gaji : %.0f\n",this.getGaji());
    }

```

Class Manager:



```java
Manager.java > Manager
public class Manager extends Karyawan{
    private double tunjangan;
    private String bagian;
    private Staff st[];

    public void setTunjangan(double tunjangan){
        this.tunjangan = tunjangan;
    }

    public double getTunjangan(){
        return tunjangan;
    }

    public void setBagian(String bagian){
        this.bagian = bagian;
    }

    public String getBagian(){
        return bagian;
    }

    public void setStaff(Staff st[]){
        this.st = st;
    }

    public void viewStaff() {
        int i;
        System.out.println(x:"----------------------");
        for(i=0;i<st.length;i++) {
            st[i].lihatInfo();
        }
        System.out.println(x:"----------------------");
    }
```

Main Class:

```java
public class Utama {
    Run | Debug
    public static void main(String[] args) {

        System.out.println(x:"Program Testing Class Manager & Staff");
        Manager man[] = new Manager[2];
        Staff staff1[] = new Staff[2];
        Staff staff2[] = new Staff[3];

        // Pembuatan Manager
        man[0] = new Manager();
        man[0].setNama(nama:"Tedjo");
        man[0].setNip(nip:"101");
        man[0].setGolongan(golongan:"1");
        man[0].setTunjangan(tunjangan:5000000);
        man[0].setBagian(bagian:"Administrasi");

        man[1] = new Manager();
        man[1].setNama(nama:"Atika");
        man[1].setNip(nip:"102");
        man[1].setGolongan(golongan:"1");
        man[1].setTunjangan(tunjangan:2500000);
        man[1].setBagian(bagian:"Pemasaran");

```

Output:

```
Program Testing Class Manager & Staff
Manager :Administrasi
NIF :101
Nama :Tedjo
Golongan :1
Tunjangan : 5000000
Gaji : 10000000
Bagian :Administrasi
----------------------
NIP :0003
Nama :Usman
Golongan :2
Jumlah lembur :10
Gaji Lembur : 10000
Gaji : 3100000
NIP :0005
Nama :Anugrah
Golongan :2
Jumlah lembur :10
Gaji Lembur : 55000
Gaji : 3550000
----------------------
```

- **Exercise**

```
Perkalianku.java > Perkalianku > perkalian(int, int, int)
1   public class Perkalianku {
2       public int a,b,c;
3
4       void perkalian(int a, int b){
5           this.a=a;
6           this.b=b;
7           System.out.println("Hasil perkalian dari " + a + " dan " + b + " adalah: " + (a * b));
8       }
9
10
11
12
13      void perkalian(int a, int b, int c){
14          this.a=a;
15          this.b=b;
16          this.c=c;
17          System.out.println("Hasil perkalian dari " + a + ", " + b + " dan " + c + " adalah: " + (a * b * c));
18      }
19
        Run | Debug
20      public static void main(String args []){
21          Perkalianku objek = new Perkalianku();
22          objek.perkalian(a:25, b:43);
23          objek.perkalian(a:34, b:23, c:56);
24      }
25
26  }
```

Output:

```
Hasil perkalian dari 25 dan 43 adalah: 1075
Hasil perkalian dari 34, 23 dan 56 adalah: 43792
PS D:\Semester 3\JAVA OOP\Week7>
```

4.1 From the above source code, where is the overloading?

-

```
void perkalian(int a, int b){
    this.a=a;
    this.b=b;
    System.out.println("Hasil perkalian dari " + a + " dan " + b + " adalah: " + (a * b));
}
```

```
void perkalian(int a, int b, int c){
    this.a=a;
    this.b=b;
    this.c=c;
    System.out.println("Hasil perkalian dari " + a + ", " + b + " dan " + c + " adalah: " + (a * b * c));
}
```

4.2 If there any overloading, how many parameters are different?

-The first perkalian method has 2 parameters:

```
void perkalian(int a, int b){
    this.a=a;
    this.b=b;
    System.out.println("Hasil perkalian dari " + a + " dan " + b + " adalah: " + (a * b));
}
```

-The second perkalian method has 3 parameters:

```
void perkalian(int a, int b, int c){
    this.a=a;
    this.b=b;
    this.c=c;
    System.out.println("Hasil perkalian dari " + a + ", " + b + " dan " + c + " adalah: " + (a * b * c));
}
```

-The difference between the overloaded methods is 1 parameter. The first method accepts 2 parameters, while the second accepts 3.

-

Perkalianku.java > Perkalianku > perkalian(int, int)

```
1   public class Perkalianku {
2
3       void perkalian(int a, int b) {
4           System.out.println(a * b);
5       }
6
7       void perkalian(double a, double b) {
8           System.out.println(a * b);
9       }
10
        Run | Debug
11      public static void main(String[] args) {
12          Perkalianku objek = new Perkalianku();
13
14          objek.perkalian(a:25, b:43);
15          objek.perkalian(a:34.56, b:23.7);
16      }
17  }
18
```

4.3 From the above source code, where is the overloading?

```
void perkalian(int a, int b) {
    System.out.println(a * b);
}
```

```java
void perkalian(double a, double b) {
    System.out.println(a * b);
}
```

4.4 If there any overloading, how many parameters are different?

-No Different

-

Ikan.java > Ikan

```java
public class Ikan {
    public void swim(){
        System.out.println(x:"Ikan bisa berenang");
    }

}
```

Piranha.java > Piranha

```java
public class Piranha extends Ikan {
        public void swim(){
            System.out.println(x:"Piranha bisa makan daging");
        }
    }
```

Fish.java > Fish

```java
public class Fish {
    Run | Debug
    public static void main(String[] args) {
        Ikan a = new Ikan();
        Ikan b = new Piranha();

        a.swim();
        b.swim();
    }
}
```

4.5 From the above source code, where is the overriding?

```java
Piranha.java > Piranha
1   public class Piranha extends Ikan {
2       public void swim(){
3           System.out.println(x:"Piranha bisa makan daging");
4       }
5   }
```

- class Piranha but extends it as an Ikan (as seen in the line Ikan b = new Piranha();), the overridden swim() method in the Piranha class will be called, not the one in the Ikan class.

So the overriding happens in the Piranha class with the swim() method:

4.6 If there any overloading, how many parameters are different?

-In method overriding, the number of parameters must remain the same between the method in the superclass and the method in the subclass.

- method swim() in both the Ikan class and the Piranha class has no parameters. Therefore, the number of parameters is the same in both the superclass (Ikan) and the subclass (Piranha), meaning there are no differences in the number of parameters between the two methods.

5. Task

5.1 Overloading

Segitiga Class:

```java
Welcome | Karyawan.java U | Staff.java U | Manager.java U | Utama
Segitiga.java > Segitiga > keliling(int, int, int)
1   public class Segitiga {
2       private int sudut;
3
4       public int totalSudut(int sudutA) {
5           sudut = 180 - sudutA;
6           return sudut;
7       }
8
9       public int totalSudut(int sudutA, int sudutB) {
10          sudut = 180 - (sudutA + sudutB);
11          return sudut;
12      }
13
14      public int keliling(int sisiA, int sisiB, int sisiC) {
15          return sisiA + sisiB + sisiC;
16      }
17
18      public double keliling(int sisiA, int sisiB) {
19          return Math.sqrt(Math.pow(sisiA, b:2) + Math.pow(sisiB, b:2));
20      }
21
22      public int getSudut() {
23          return sudut;
24      }
25  }
```

Main Segitiga:

```java
public class MainSegitiga {
    Run | Debug
    public static void main(String[] args) {
        Segitiga segitiga = new Segitiga();

        int sudut1 = segitiga.totalSudut(sudutA:60);
        System.out.println("Total sudut with one angle: " + sudut1);
        System.out.println("Value of sudut field: " + segitiga.getSudut());

        int sudut2 = segitiga.totalSudut(sudutA:60, sudutB:30);
        System.out.println("Total sudut with two angles: " + sudut2);
        System.out.println("Value of sudut field: " + segitiga.getSudut());

        int keliling1 = segitiga.keliling(sisiA:3, sisiB:4, sisiC:5);
        System.out.println("Perimeter of the triangle: " + keliling1);

        double keliling2 = segitiga.keliling(sisiA:3, sisiB:4);
        System.out.println("Hypotenuse: " + keliling2);
    }
}
```

Output:

```
Total sudut with one angle: 120
Value of sudut field: 120
Total sudut with two angles: 90
Value of sudut field: 90
Perimeter of the triangle: 12
Hypotenuse: 5.0
PS D:\Semester 3\JAVA OOP\Week7>
```

5.2 Overriding

Class Manusia:

```java
public class Manusia {
    public void bernafas() {
        System.out.println(x:"Manusia sedang bernafas.");
    }

    public void makan() {
        System.out.println(x:"Manusia sedang makan.");
    }
}
```

Class Mahasiswa:

```java
public class Mahasiswa extends Manusia {
    @Override
    public void makan() {
        System.out.println("Mahasiswa sedang makan mie instan.");
    }

    public void tidur() {
        System.out.println("Mahasiswa sedang tidur setelah belajar semalaman.");
    }
}
```

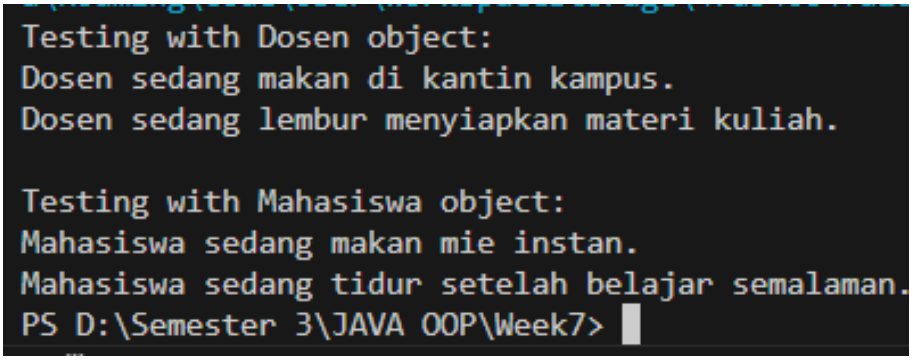
Class dosen:

```java
public class Dosen  extends Manusia{
        @Override
        public void makan() {
            System.out.println("Dosen sedang makan di kantin kampus.");
        }

        public void lembur() {
            System.out.println("Dosen sedang lembur menyiapkan materi kuliah.");
        }
}
```

Main:

```java
public class MainTask2 {
    public static void aktivitasMakan(Manusia manusia) {
        manusia.makan();
    }

    Run | Debug
    public static void main(String[] args) {
        System.out.println("Testing with Dosen object:");
        Manusia dosen = new Dosen();
        aktivitasMakan(dosen);
        ((Dosen) dosen).lembur();

        System.out.println("\nTesting with Mahasiswa object:");
        Manusia mahasiswa = new Mahasiswa();
        aktivitasMakan(mahasiswa);
        if (mahasiswa instanceof Mahasiswa) {
            ((Mahasiswa) mahasiswa).tidur();
        }
    }
}
```

Output:


Testing with Dosen object:
Dosen sedang makan di kantin kampus.
Dosen sedang lembur menyiapkan materi kuliah.
Testing with Mahasiswa object:
Mahasiswa sedang makan mie instan.
Mahasiswa sedang tidur setelah belajar semalaman.
PS D:\Semester 3\JAVA OOP\Week7>