LAPORAN PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK

JOBSHEET 9: OVERLOADING & OVERRIDING

oleh :
Halim Teguh Saputro
2E
2141762122

PROGRAM STUDI D-IV SISTEM INFORMASI BISNIS
JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG
Jl. Soekarno Hatta No .9, Jatimulyo, Kec. Lowokwaru, Kota Malang,
JawaTimur 65141

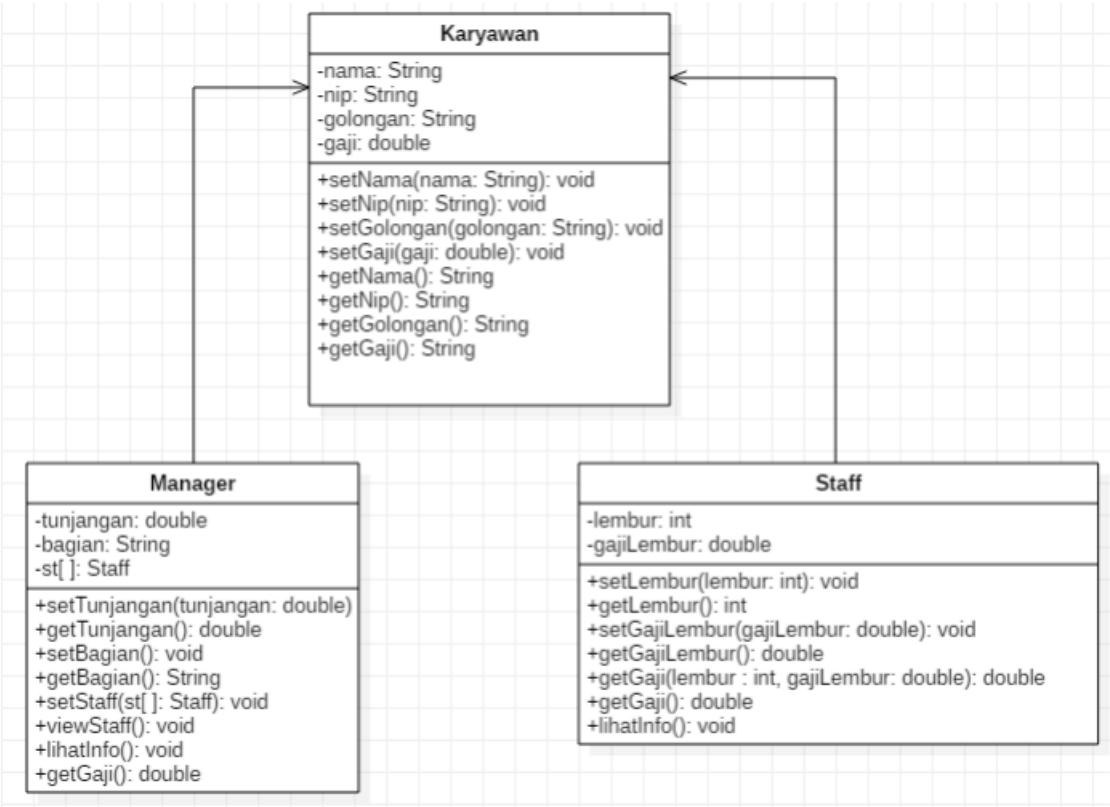
KOMPETENSI

Setelah melakukan percobaan pada modul ini, mahasiswa memahami konsep:

1. Memahami konsep overloading dan overriding
2. Memahami perbedaan overloading dan overriding
3. Ketepatan dalam mengidentifikasi method overriding dan overloading
4. Ketepatan dalam mempraktekkan instruksi pada jobsheet
5. Mengimplementasikan method overloading dan overriding

PRAKTIKUM 1

Untuk kasus contoh berikut ini, terdapat tig akelas, yaitu Karyawan, Manager, dan Staff, Class Karyawan merupakan superclass dari Manager dan Staff dimana subclass Manager dan Staff memiliki method untuk menghitung gaji yang berbeda



A. Karyawan

```
package Praktikum;
```

```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(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;
        }
    }

    public void setGaji(double gaji) {
        this.gaji = gaji;
    }

    public String getNama() {
        return nama;
    }

    public String getNip() {
        return nip;
    }
```

```
    public String getGolongan() {
        return golongan;
    }

    public double getGaji() {
        return gaji;
    }

}
```

B. Staff

```
package Praktikum;

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\t\t: " + this.getNip());
        System.out.println("Nama\t\t: " + this.getNama());
        System.out.println("Golongan\t: " + this.getGolongan());
```

```
        System.out.println("Jumlah Lembur\t: " + this.getLembur());
        System.out.printf("Gaji Lembur\t: %.0f\n", this.getGajiLembur());
        System.out.printf("Gaji\t\t: %.0f\n", this.getGaji());
        System.out.println("");
    }
}
```

C. Manager

```
package Praktikum;

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("---------------------");
        for (i = 0; i < st.length; i++) {
            st[i].lihatInfo();
        }
        System.out.println("---------------------");
    }

    public void lihatInfo() {
        System.out.println("Manage\t\t: " + this.getBagian());
```

```
        System.out.println("NIP\t\t: " + this.getNip());
        System.out.println("Nama\t\t: " + this.getNama());
        System.out.println("Golongan\t: " + this.getGolongan());
        System.out.printf("Tunjangan\t: %.0f\n", this.getTunjangan());
        System.out.printf("Gaji\t\t: %.0f\n", this.getGaji());
        System.out.println("Bagian\t\t: " + this.getBagian());
        this.viewStaff();
    }

    public double getGaji() {
        return super.getGaji() + tunjangan;
    }
}
```

D. Utama (main program)

```
package Praktikum;

public class Utama {
    public static void main(String[] args) {
        System.out.println("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("Halim");
        man[0].setNip("101");
        man[0].setGolongan("1");
        man[0].setTunjangan(5000000);
        man[0].setBagian("Administrasi");

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

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

```
        staff1[1] = new Staff();
        staff1[1].setNama("Budi");
        staff1[1].setNip("0005");
        staff1[1].setGolongan("2");
        staff1[1].setLembur(10);
        staff1[1].setGajiLembur(55000);
        man[0].setStaff(staff1);

        staff2[0] = new Staff();
        staff2[0].setNama("Cece");
        staff2[0].setNip("0004");
        staff2[0].setGolongan("3");
        staff2[0].setLembur(15);
        staff2[0].setGajiLembur(5500);

        staff2[1] = new Staff();
        staff2[1].setNama("Doni");
        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(6);
        staff2[2].setGajiLembur(20000);
        man[1].setStaff(staff2);

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

LATIHAN 1

```
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 di atas terletak dimanakah overloading?
   Jawab: terletak pada method void perkalian
2. Jika terdapat overloading ada berapa jumlah parameter yang berbeda?
   Jawab: method perkalian yang pertama memiliki 2 parameter dan method perkalian yang kedua memiliki 3 parameter.

```
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);
 }
}
```

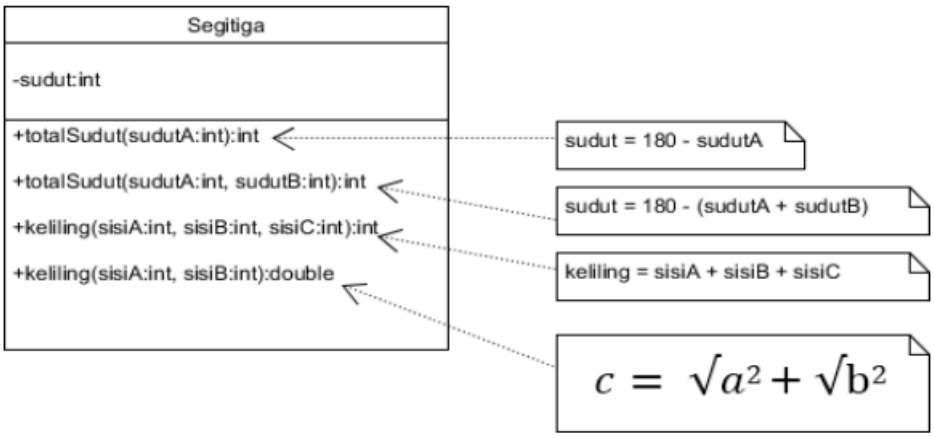
3. Dari source coding diatas terletak dimanakah overloading?
   Jawab: terdapat pada method perkalian
4. Jika terdapat overloading pada berapa tipe parameter yang berbeda?
   Jawab: terdapat 2 parameter yang berbeda, pada method perkalian pertama kedua parameternya menggunakan tipe data int. Sedangkan, pada method perkalian kedua, kedua parameternya menggunakan tipe data double.

```
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 di atas terletak dimanakah overriding?
   Jawab: Terletak pada method swim pada class piranha.
6. Jabarkanlah apabila source coding di atas jika terdapat overriding?
   Jawab: Method swim pada class piranha memodifikasi method superclassnya yaitu Class Ikan. Method swim pada class piranha memodifikasi outputnya menjadi “Piranha bisa makan daging”.

TUGAS

A. Overloading
   Implementasikan konsep overloading pada class diagram dibawah ini:



Jawab:

a. Segitiga

```java
package Tugas;

public class Segitiga {
    private int sudut;

    public void setSudut(int sudut) {
        this.sudut = sudut;
    }

    public int getSudut() {
        return sudut;
    }

    public int totalSudut(int sudutA) {
        sudut = 180 - sudutA;
        return sudut;
    }

    public int totalSudut(int sudutA, int sudutB) {
        sudut = 180 - (sudutA + sudutB);
        return sudut;
    }

    public int keliling(int sisiA, int sisiB, int sisiC) {
        return sisiA + sisiB + sisiC;
    }

    public double keliling(int sisiA, int sisiB) {
        return Math.sqrt(sisiA * sisiA) + Math.sqrt(sisiB * sisiB);
    }
}
```

b. MainSegitiga

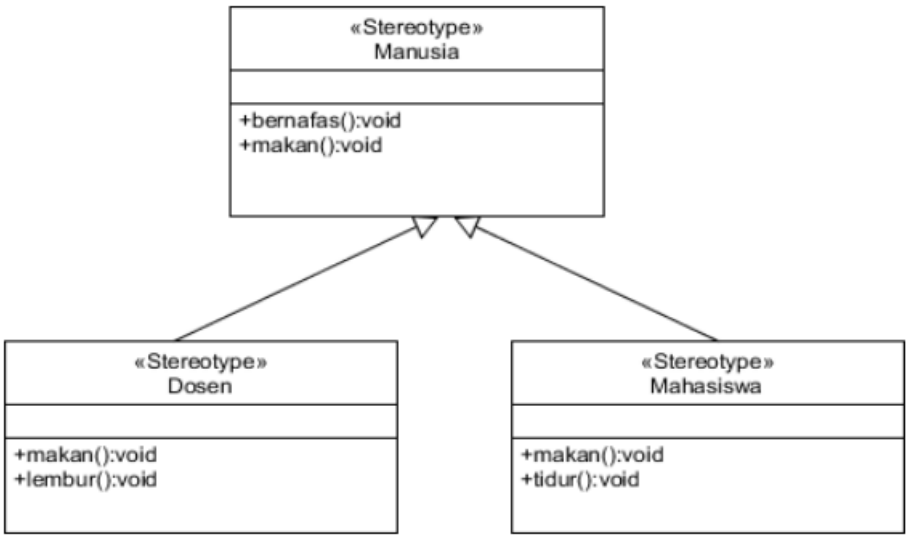
```
package Tugas;

import java.util.Scanner;

public class MainSegitiga {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Segitiga data = new Segitiga();

        System.out.print("Masukkan sudut A: ");
        int sudutA = sc.nextInt();
        System.out.print("Masukkan sudut B: ");
        int sudutB = sc.nextInt();
        System.out.println("");
        System.out.println("Hasil Method 1 : " + data.totalSudut(sudutA));
        System.out.println("Hasil Method 2 : " + data.totalSudut(sudutA,
sudutB));

        System.out.println("");
        System.out.println("");

        System.out.print("Masukkan sisi A: ");
        int sisiA = sc.nextInt();
        System.out.print("Masukkan sisi B: ");
        int sisiB = sc.nextInt();
        System.out.print("Masukkan sisi C: ");
        int sisiC = sc.nextInt();
        System.out.println("");

        System.out.println("Hasil Method 3 : " + data.keliling(sisiA, sisiB,
sisiC));
        data.keliling(sisiA, sisiB, sisiC);
        System.out.println("Hasil Method 4 : " + data.keliling(sisiA, sisiB));

    }

}
```

c. Output

```
Masukkan sudut A: 30
Masukkan sudut B: 60

Hasil Method 1 : 150
Hasil Method 2 : 90


Masukkan sisi A: 4
Masukkan sisi B: 5
Masukkan sisi C: 6

Hasil Method 3 : 15
Hasil Method 4 : 9.0
```

B. Overriding

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



Jawab:

a. Manusia

```java
package Tugas;

public class Manusia {
    public Manusia() {

    }

    public void Bernafas() {
        System.out.println("Sedang bernafas");
    }

    public void makan() {
        System.out.println("Sedang Makan");
```

```
    }
}
```

b. Dosen

```
package Tugas;

public class Dosen extends Manusia {
    Dosen() {

    }

    public void makan() {
        System.out.println("Dosen sedang makan steak");
    }

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

c. Mahasiswa

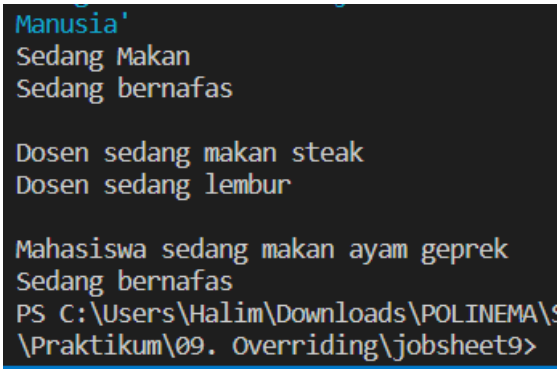
```
package Tugas;

public class Mahasiswa extends Manusia {
    Mahasiswa() {

    }

    public void makan() {
        System.out.println("Mahasiswa sedang makan ayam geprek");
    }

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

d. MainManusia (main program)

```
package Tugas;

public class MainManusia {
```

```
    public static void main(String[] args) {
        Manusia manusia = new Manusia();
        Dosen dosen = new Dosen();
        Mahasiswa mhs = new Mahasiswa();

        manusia.makan();
        manusia.Bernafas();
        System.out.println("");

        dosen.makan();
        dosen.lembur();
        System.out.println("");

        mhs.makan();
        mhs.Bernafas();
    }
}
```

e. Output



LINK GITHUB

https://github.com/HalimTeguh/Praktikum/tree/master/09.%20Overriding