Jurusan Teknologi Informasi Politeknik Negeri Malang

Jobsheet-9: OVERLOADING DAN OVERRIDING

Mata Kuliah Praktikum Pemograman Berbasis Objek

Pengampu: Tim Ajar Praktikum Pemograman Berbasis Objek

Oktober 2021

Nama	:	Annisa Aulia Nadhila
Nim	:	2041720023
Kelas	:	TI – 2C

3. Praktikum

3.1 Percobaan 1

Untuk kasus contoh berikut ini, terdapat tiga kelas, 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.

3.2 Karyawan

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

    public String getNama() {
        return nama;
    }
    public void setNama(String nama) {
        this.nama = nama;
    }
    public String getNip() {
        return nip;
    }
    public void setNip(String nip) {
        this.nip = nip;
    }
    public String getGolongan() {
        return golongan;
    }
    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 double getGaji() {
```

```
        return gaji;
    }

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

3.3 Staff

```
public class Staff extends Karyawan{
   private int lembur;
   private double gajiLembur;

    public int getLembur() {
        return lembur;
    }

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

    public double getGajiLembur() {
        return gajiLembur;
    }

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

    public double getGaji(int lembut, 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("Jml Lembur  :"+this.gajiLembur);
        System.out.printf("Gaji Lembur :%.0f\n", gajiLembur);
        System.out.printf("Gaji        :%.0f\n",this.getGaji());
}
}
```

3.4 Manager

```
ublic class Manager extends Karyawan{
  private double tunjangan;
    private String bagian;
    private Staff st[];

    public double getTunjangan() {
        return tunjangan;
    }

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

    public String getBagian() {
        return bagian;
    }

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

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

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

```
        System.out.println("---------------------------------------");
    }

    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();
    }

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

3.5 Utama

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

        //Pembuatan Manager
        man[0] = new Manager();
        man[0].setNama("Tedjo");
        man[0].setNip("101");
        man[0].setGolongan("1");
        man[0].setTunjangan(5000000);
        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("Usman");
        staff1[0].setNip("0003");
        staff1[0].setGolongan("2");
        staff1[0].setLembur(10);
        staff1[0].setGajiLembur(10000);

        staff1[1] = new Staff();
        staff1[1].setNama("Anugrah");
        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("Hendra");
        staff2[0].setNip("0004");
        staff2[0].setGolongan("3");
        staff2[0].setLembur(15);
        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(6);
        staff2[2].setGajiLembur(20000);
```

```
        man[1].setStaff(staff2);

        man[0].lihatInfo();
        man[1].lihatInfo();
    }
}
```

4. Latihan

```
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);

    }
}
```

4.1 Dari source coding diatas terletak dimanakah overloading?

Jawab : method perkalian

4.2 Jika terdapat overloading ada berapa jumlah parameter yang berbeda?

Jawab : ada 2

```
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);

    }
}
```

4.3 Dari source coding diatas terletak dimanakah overloading?

Jawab : perkalian

4.4 Jika terdapat overloading ada berapa tipe parameter yang berbeda?

Jawab : ada dua

```
public class Fish {
    public static void main(String[] args) {
        ikan a = new ikan();
        ikan b = new Piranha();
        a.swim();
        b.swim();
    }
}
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");
    }
}
```

4.5 Dari source coding diatas terletak dimanakah overriding?

Jawab: pada method swim

4.6 Jabarkanlah apabila sourcoding diatas jika terdapat overriding?

Jawab : method swim sama" memiliki nama yang sama hanya saja data yang dikembalikan / outpunya beda.

5. Tugas

5.1 Overloading

Implementasikan konsep overloading pada class diagram dibawah ini :

```
public class Segitiga {
    private int sudut;
    public int totalSudut(int sudutA){
        return sudut = 180 - sudutA;
    }
    public int totalSudut(int sudutA, int sudutB){
        return sudut = 180 - (sudutA - sudutB);
    }
    public int keliling(int sisiA, int sisiB, int sisiC){
        int keliling = sisiA + sisiB + sisiC;
        return keliling;
    }
    public double keliling(int sisiA, int sisiB){
    double keliling = Math.sqrt( Math.pow(sisiA,2) + Math.pow(sisiB,2) );
    return keliling;
    }

    public static void main(String[] args) {
        Segitiga s = new Segitiga();
        System.out.println("totalSudut pertama: " + s.totalSudut(45));
        System.out.println("totalSudut kedua: "+s.totalSudut(45, 45));
        System.out.println("Keliling pertama: "+ s.keliling(3, 4, 5));
        System.out.println("Keliling kedua: "+ s.keliling(3, 4));
    }
}
```

5.2 Overriding

Implementasikan class diagram dibawah ini dengan menggunakan teknik dynamic

method dispatch :

```
public class Manusia {
    public void bernafas(){
        System.out.println("Manusia bisa bernapas");
    }
    public void makan(){
        System.out.println("Manusia bisa makan");
    }
}
class Dosen{
    public void makan(){
        System.out.println("Dosen juga makan seperti manusia biasa");
    }
    public void lembur(){
        System.out.println("Dosen bekerja lembur");
    }
}
class Mahasiswa{
    public void makan(){
        System.out.println("Mahasiswa juga makan seperti manusia biasa");
    }
    public void tidur(){
        System.out.println("Mahasiswa tidur teratur");
    }
}
```