LAPORAN PRAKTIKUM PEMROGRAMAN BERBASIS OBJEK (PBO)

JOBSHEET 7

OVERLOADING & OVERRIDING

Oleh :

VINCENTIUS LEONANDA PRABOWO

2341720149 / 28

TI 2A

POLITEKNIK NEGERI MALANG

JURUSAN TEKNOLOGI INFORMASI

PRODI D-IV TEKNIK INFORMATIKA

- Percobaan 1
  - Class Karyawan

```
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;
            default:
            this.gaji = 750000;
        }
    }

    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;
    }
}
```

- Class Staff

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

- Class 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(" ------------------------- ");
        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;
    }
}
```

- Class Main

```
public class testKaryawan {
    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];

        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();
    }
}
```

- Hasil :

```
run:
Program Testing Class Manager & Staff
Manager : Administrasi
NIP     : 101
Nama    : Tedjo
Golongan: 1
Tunjangan: 5000000
Gaji     : 10000000
Bagian   : Administrasi
---------------------------
NIP        : 0003
Nama       : Usman
Golongan  : 2
Jml Lembur     : 10Gaji Lembur: 10000
Gaji       : 3100000
------------------------------
NIP        : 0005
Nama       : Anugrah
Golongan  : 2
Jml Lembur     : 10Gaji Lembur: 55000
Gaji       : 3550000
------------------------------
Manager : Pemasaran
NIP     : 102
Nama    : Atika
Golongan: 1
Tunjangan: 2500000
Gaji     : 7500000
Bagian   : Pemasaran
---------------------------
NIP        : 0004
Nama       : Hendra
Golongan  : 3
Jml Lembur     : 15Gaji Lembur: 5500
Gaji       : 2082500
------------------------------
NIP        : 0006
Nama       : Arie
Golongan  : 4
Jml Lembur     : 5Gaji Lembur: 100000
Gaji       : 1500000
------------------------------
NIP        : 0007
Nama       : Mentari
Golongan  : 3
Jml Lembur     : 6Gaji Lembur: 20000
Gaji       : 2120000
------------------------------
```

- Latihan

Kode Perkalianku 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 class testPerkalianku {
   public static void main(String[] args) {
      Perkalianku objek = new Perkalianku();

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

Soal :

1. Dari source coding diatas terletak dimanakah overloading?
   Terletak pada metode perkalian yang dideklarasikan dua kali dalam kelas Perkalianku. Keduanya memiliki nama yang sama, yaitu perkalian, tetapi perbedaannya ada pada jumlah parameter yang diterima.
2. Jika terdapat overloading ada berapa jumlah parameter yang berbeda?
   Terdapat dua metode yang overload, yaitu:
   - void perkalian(int a, int b) dengan dua parameter.
   - void perkalian(int a, int b, int c) dengan tiga parameter.

Kode Perkalianku 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 class testPerkalianku {
     public static void main(String[] args) {
     Perkalianku objek = new Perkalianku();
     objek.perkalian(25, 43);
     objek.perkalian(34.56, 23.7);
   }
}
```

Soal :

1. Dari source coding diatas terletak dimanakah overloading?
   Terletak pada metode perkalian. Kedua metode memiliki nama yang sama, yaitu perkalian, tetapi tipe data parameternya berbeda. Metode pertama menerima parameter bertipe int, sedangkan metode kedua menerima parameter bertipe double.

2. Jika terdapat overloading ada berapa tipe parameter yang berbeda?
   Terdapat dua tipe parameter yang berbeda pada overloading tersebut, yaitu:
   - int untuk metode pertama: void perkalian(int a, int b)
   - double untuk metode kedua: void perkalian(double a, double b)

Kode Ikan :

```
public 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();
}
}
```

Soal :

1. Dari source coding diatas terletak dimanakah overloading?
   Tidak terdapat Overloading di source code di atas, karena tidak ada method dengan nama yang sama tetapi memiliki parameter yang berbeda.
2. Jabarkanlah apabila sourcoding diatas jika terdapat overriding?
   Overriding terjadi ketika metode swim() pada kelas Piranha menimpa (override) metode swim() yang diwarisi dari kelas Ikan. Keduanya memiliki nama metode yang sama, parameter yang sama, tetapi implementasi berbeda.

- Tugas

1. Overloading
   - Class Segitiga

```
public class Segitiga {
    int sudut;

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

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

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

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

- Class Main

```
public class testSegitiga {
    public static void main(String[] args) {
        Segitiga segitiga = new Segitiga();

        System.out.println("Total Sudut (satu sudut): " +
segitiga.totalSudut(60));
        System.out.println("Total Sudut (dua sudut): " +
segitiga.totalSudut(60, 40));
        System.out.println("Keliling : " + segitiga.keliling(3, 4, 5));
        System.out.println("Keliling : " + segitiga.keliling(3.0, 4.0));
    }
}
```

Hasil :

```
run:
Total Sudut (satu sudut): 120
Total Sudut (dua sudut): 80
Keliling : 12
Keliling : 5.0
```

2. Overriding
    - Class Manusia

```
public class Manusia {
    public void bernapas() {
        System.out.println("Manusia bisa bernapas.");
    }

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

```
class Dosen extends Manusia {
    @Override
    public void makan() {
        System.out.println("Dosen sedang makan sambil membaca
makalah.");
    }

    public void lembur() {
        System.out.println("Dosen sering lembur untuk mengoreksi tugas.");
    }
}

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

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

- Class Main

```
public class testManusia {
    public static void main(String[] args) {
        Manusia manusia;

        manusia = new Dosen();
        manusia.bernapas();
        manusia.makan();

        manusia = new Mahasiswa();
        manusia.bernapas();
        manusia.makan();
    }
}
```

Hasil :

```
run:
Manusia bisa bernapas.
Dosen sedang makan sambil membaca makalah.
Manusia bisa bernapas.
Mahasiswa sedang makan di kantin kampus.
```