

**LAPORAN JOBSHEET 09**  
**PEMROGRAMAN BERBASIS OBJEK**



**Disusun Oleh:**

Fadhlurohman Al Farabi

TI-2C

NIM. 2241720081

**PROGRAM STUDI D-IV TEKNIK INFORMATIKA**  
**JURUSAN TEKNOLOGI INFORMASI**  
**POLITEKNIK NEGERI MALANG**  
**2023**

## **Praktikum**

### 1. Karyawan

Source Code

```
1 package Percobaan;
2
3 /**
4  * Karyawan
5  */
6 public class Karyawan {
7
8     private String nama;
9     private String nip;
10    private String golongan;
11    private double gaji;
12
13    public void setNama(String nama) {
14        this.nama = nama;
15    }
16
17    public void setNip(String nip) {
18        this.nip = nip;
19    }
20
21    public void setGolongan(String golongan) {
22        this.golongan = golongan;
23
24        switch (golongan.charAt(0)) {
25            case '1':
26                this.gaji = 5000000;
27                break;
28            case '2':
29                this.gaji = 3000000;
30                break;
31            case '3':
32                this.gaji = 2000000;
33                break;
34            case '4':
35                this.gaji = 1000000;
36                break;
37            case '5':
38                this.gaji = 750000;
39                break;
40        }
41    }
42
43    public void setGaji(double gaji) {
44        this.gaji = gaji;
45    }
46
47    public String getNama() {
48        return nama;
49    }
50
51    public String getNip() {
52        return nip;
53    }
54
55    public String getGolongan() {
56        return golongan;
57    }
58
59    public double getGaji() {
60        return gaji;
61    }
62
63 }
```

## 2. Staff

```
1  package Percobaan;
2
3  public class Staff extends Karyawan {
4      private int lembur;
5      private double gajiLembur;
6
7      public void setLembur(int lembur) {
8          this.lembur = lembur;
9      }
10
11     public int getLembur() {
12         return lembur;
13     }
14
15     public void setGajiLembur(double gajiLembur) {
16         this.gajiLembur = gajiLembur;
17     }
18
19     public double getGajiLembur() {
20         return gajiLembur;
21     }
22
23     // Overloading
24     public double getGaji(int lembur, double gajiLembur) {
25         return super.getGaji() + lembur * gajiLembur;
26     }
27
28     // Overriding
29     public double getGaji() {
30         return super.getGaji() + lembur * gajiLembur;
31     }
32
33     public void lihatInfo() {
34         System.out.println("NIP          : " + this.getNip());
35         System.out.println("Nama       : " + this.getNama());
36         System.out.println("Golongan  : " + this.getGolongan());
37         System.out.println("Jml Lembur : " + this.getLembur());
38         System.out.printf("Gaji Lembur :%.0f\n ", this.getGajiLembur());
39         System.out.printf("Gaji      :%.0f\n ", this.getGaji());
40     }
41 }
42
```

## 3. Manager

```
1 package Percobaan;
2
3 public class Manager extends Karyawan {
4     private double tunjangan;
5     private String bagian;
6     private Staff st[];
7
8     public void setTunjangan(double tunjangan) {
9         this.tunjangan = tunjangan;
10    }
11
12    public double getTunjangan() {
13        return tunjangan;
14    }
15
16    public void setBagian(String bagian) {
17        this.bagian = bagian;
18    }
19
20    public String getBagian() {
21        return bagian;
22    }
23
24    public void setStaff(Staff st[]) {
25        this.st = st;
26    }
27
28    public void viewStaff() {
29        int i;
30        System.out.println("-----");
31        for (i = 0; i < st.length; i++) {
32            st[i].lihatInfo();
33        }
34        System.out.println("-----");
35    }
36
37    public void lihatInfo() {
38        System.out.println("Manager      :" + this.getBagian());
39        System.out.println("NIP        :" + this.getNip());
40        System.out.println("Nama       :" + this.getNama());
41        System.out.println("Golongan   :" + this.getGolongan());
42        System.out.println("Tunjangan  :" + this.getTunjangan());
43        System.out.println("Gaji       :" + this.getGaji());
44        System.out.println("Bagian     :" + this.getBagian());
45        this.viewStaff();
46    }
47
48    public double getGaji() {
49        return super.getGaji() + tunjangan;
50    }
51 }
52
```

## 4. Utama

```
1 package Percobaan;
2
3 public class Utama {
4     public static void main(String[] args) {
5         System.out.println("Program Testing Class Manager & Staff");
6         Manager man[] = new Manager[2];
7         Staff staff1[] = new Staff[2];
8         Staff staff2[] = new Staff[3];
9
10        // pembuatan manager
11        man[0] = new Manager();
12        man[0].setNama("Tedjo");
13        man[0].setNip("101");
14        man[0].setGolongan("1");
15        man[0].setTunjangan(5000000);
16        man[0].setBagian("Administrasi");
17
18        man[1] = new Manager();
19        man[1].setNama("Atika");
20        man[1].setNip("102");
21        man[1].setGolongan("1");
22        man[1].setTunjangan(2500000);
23        man[1].setBagian("Pemasaran");
24
25        staff1[0] = new Staff();
26        staff1[0].setNama("Usman");
27        staff1[0].setNip("0003");
28        staff1[0].setGolongan("2");
29        staff1[0].setLembur(10);
30        staff1[0].setGajiLembur(10000);
31
32        staff1[1] = new Staff();
33        staff1[1].setNama("Anugrah");
34        staff1[1].setNip("0005");
35        staff1[1].setGolongan("2");
36        staff1[1].setLembur(10);
37        staff1[1].setGajiLembur(55000);
38        man[0].setStaff(staff1);
39
40        staff2[0] = new Staff();
41        staff2[0].setNama("Hendra");
42        staff2[0].setNip("0004");
43        staff2[0].setGolongan("3");
44        staff2[0].setLembur(15);
45        staff2[0].setGajiLembur(5500);
46
47        staff2[1] = new Staff();
48        staff2[1].setNama("Arie");
49        staff2[1].setNip("0006");
50        staff2[1].setGolongan("4");
51        staff2[1].setLembur(5);
52        staff2[1].setGajiLembur(100000);
53
54        staff2[2] = new Staff();
55        staff2[2].setNama("Mentari");
56        staff2[2].setNip("0007");
57        staff2[2].setGolongan("3");
58        staff2[2].setLembur(6);
59        staff2[2].setGajiLembur(20000);
60        man[1].setStaff(staff2);
61
62        // cetak informasi dari manager + staffnya
63        man[0].lihatInfo();
64        man[1].lihatInfo();
65    }
66 }
67
```

## Output

```
Program Testing Class Manager & Staff
Manager      :Administrasi
NIP          :101
Nama         :Tedjo
Golongan     :1
Tunjangan    :5000000.0
Gaji         :1.0E7
Bagian       :Administrasi
-----
NIP          : 0003
Nama         : Usman
Golongan     : 2
Jml Lembur   : 10
Gaji Lembur  :10000
Gaji         :3100000
NIP          : 0005
Nama         : Anugrah
Golongan     : 2
Jml Lembur   : 10
Gaji Lembur  :55000
Gaji         :3550000
-----
Manager      :Pemasaran
NIP          :102
Nama         :Atika
Golongan     :1
Tunjangan    :2500000.0
Gaji         :7500000.0
Bagian       :Pemasaran
-----
```

```
Manager      :Pemasaran
NIP           :102
Nama          :Atika
Golongan      :1
Tunjangan     :2500000.0
Gaji          :7500000.0
Bagian        :Pemasaran
```

```
-----
NIP           : 0004
Nama          : Hendra
Golongan      : 3
Jml Lembur    : 15
Gaji Lembur   :5500
  Gaji        :2082500
  NIP         : 0006
Nama          : Arie
Golongan      : 4
Jml Lembur    : 5
Gaji Lembur   :100000
  Gaji        :1500000
  NIP         : 0007
Nama          : Mentari
Golongan      : 3
Jml Lembur    : 6
Gaji Lembur   :20000
  Gaji        :2120000
-----
```

```
PS E:\SEMESTER 3\PRAKTIKUM PEMROGRAMAN BERBASIS OBJEK\PERTEMUAN 11>
```

## 5. Latihan

```
public class PerkalianKu {
    // latihan 1
    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) {
    // latihan 1
    PerkalianKu objek = new PerkalianKu();
    objek.perkalian(a:25, b:43);
    objek.perkalian(a:34, b:23, c:56);
}
```

a. Dari Source Code diatas terletak dimanakah overloading ?

Jawab : Overloading terletak pada deklarasi fungsi perkalian, di mana ada dua versi fungsi dengan nama yang sama tetapi dengan jumlah parameter yang berbeda.



- b. Jika terdapat overloading ada berapa jumlah parameter yang berbeda ?

Jawab : Terdapat dua versi fungsi perkalian dengan jumlah parameter yang berbeda. Versi pertama memiliki dua parameter (int a, int b), sedangkan versi kedua memiliki tiga parameter (int a, int b, int c). Jadi, terdapat perbedaan dalam jumlah parameter antara dua versi fungsi tersebut.

```
// latihan 2
void perkalian(int a, int b) {
    System.out.println(a * b);
}

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

```
// latihan 2
PerkalianKu objek = new PerkalianKu();
objek.perkalian(a:25, b:43);
objek.perkalian(a:34.56, b:23.7);
```

- c. Dari Source Code diatas terletak dimanakah overloading ?

Jawab : Overloading terletak pada deklarasi fungsi perkalian, di mana ada dua versi fungsi dengan nama yang sama tetapi dengan jumlah parameter yang berbeda.

- d. Jika terdapat overloading ada berapa tipe parameter yang berbeda ?

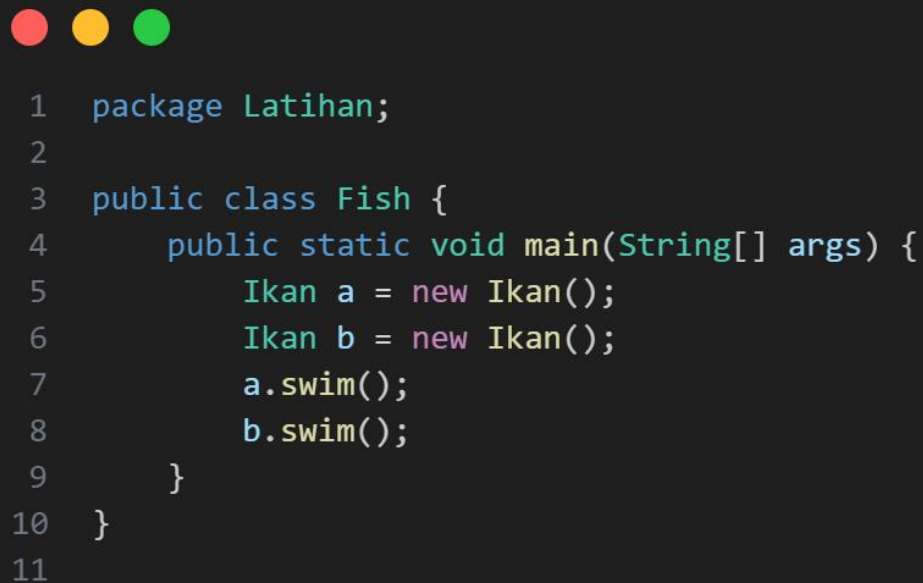
Jawab : Terdapat dua versi fungsi perkalian dengan tipe parameter yang berbeda. Versi pertama menggunakan tipe data integer (int a, int b) dan versi kedua menggunakan tipe data double (double a, double b).



```
1 package Latihan;  
2  
3 public class Ikan {  
4     public void swim() {  
5         System.out.println("Ikan bisa berenang");  
6     }  
7 }  
8
```



```
1 package Latihan;  
2  
3 public class Piranha extends Ikan {  
4     public void swim() {  
5         System.out.println("Piranha bisa makan daging");  
6     }  
7 }  
8
```



```
1  package Latihan;
2
3  public class Fish {
4      public static void main(String[] args) {
5          Ikan a = new Ikan();
6          Ikan b = new Ikan();
7          a.swim();
8          b.swim();
9      }
10 }
11
```

- e. Dari source coding diatas terletak dimanakah overriding?  
Jawab : Overriding terletak pada fungsi swim.
- f. Jabarkanlah apabila sourcoding diatas jika terdapat overriding?  
Jawab : Fungsi swim yang terletak pada super class atau parent class (Ikan) dituliskan kembali pada subclass atau child class (Piranha) namun memiliki eksekusi yang berbeda atau hasil yang berbeda.

## Tugas

### Overloading – Source Code

```
1 package Tugas.Overloading;
2
3 public class Segitiga {
4     private int sudut;
5
6     public int totalSudut(int sudutA) {
7         return this.sudut = 180 - sudutA;
8     }
9
10    public int totalSudut(int sudutA, int sudutB) {
11        return this.sudut = 180 - (sudutA + sudutB);
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, 2) + Math.pow(sisiB, 2));
20    }
21
22    public static void main(String[] args) {
23        Segitiga objek = new Segitiga();
24
25        // method totalSudut()
26        System.out.println("Method totalSudut yang pertama : " + objek.totalSudut(50));
27        System.out.println("Method totalSudut yang kedua : " + objek.totalSudut(50, 30));
28
29        // method keliling()
30        System.out.println("Metod keliling yang pertama : " + objek.keliling(1, 2, 3));
31        System.out.println("Method keliling yang kedua : " + objek.keliling(2, 5));
32    }
33 }
34
35 }
36
```

### Overloading – Output

```
Method totalSudut yang pertama : 130
Method totalSudut yang kedua : 100
Metod keliling yang pertama : 6
Method keliling yang kedua : 5.385164807134504
PS E:\SEMESTER 3\PRAKTIKUM PEMROGRAMAN BERBASIS OBJEK\PERTEMUAN 11>
```

## Overriding – Source Code

```
1 package Tugas.Overriding;
2
3 public class Manusia {
4     public void bernafas() {
5         System.out.println("Manusia harus bernafas");
6     }
7
8     public void makan() {
9         System.out.println("Manusia harus makan agar mempunyai energi");
10    }
11 }
12
```

```
1 package Tugas.Overriding;
2
3 public class Dosen extends Manusia {
4     public void makan() {
5         System.out.println("Dosen makan-makanan yang bergizi");
6     }
7
8     public void lembur() {
9         System.out.println("Dosen selalu lembur untuk mengecek tugas-tugas yang diinputkan oleh Mahasiswanya");
10    }
11 }
12
```

```
1 package Tugas.Overriding;
2
3 public class Mahasiswa extends Manusia {
4     public void makan() {
5         System.out.println("Mahasiswa selalu makan-makanan yang praktis yang paling penting adalah perutnya kenyang");
6     }
7
8     public void tidur() {
9         System.out.println("Mahasiswa memiliki waktu tidur yang kurang karena harus mengerjakan tugas jobsheet");
10    }
11 }
12
```

## Overriding – Output

```
=====
Super Class
Manusia harus bernafas
Manusia harus makan agar mempunyai energi
=====
Parent Class Dosen
Manusia harus bernafas
Dosen makan-makanan yang bergizi
Dosen selalu lembur untuk mengecek tugas-tugas yang diinputkan oleh Mahasiswanya
=====
Parent Class Mahasiswa
Manusia harus bernafas
Mahasiswa selalu makan-makanan yang praktis yang paling penting adalah perutnya kenyang
Mahasiswa memiliki waktu tidur yang kurang karena harus mengerjakan tugas jobsheet
=====
PS E:\SEMESTER 3\PRAKTIKUM PEMROGRAMAN BERBASIS OBJEK\PERTEMUAN 11>
```