Jurusan Teknologi Informasi Politeknik Negeri Malang

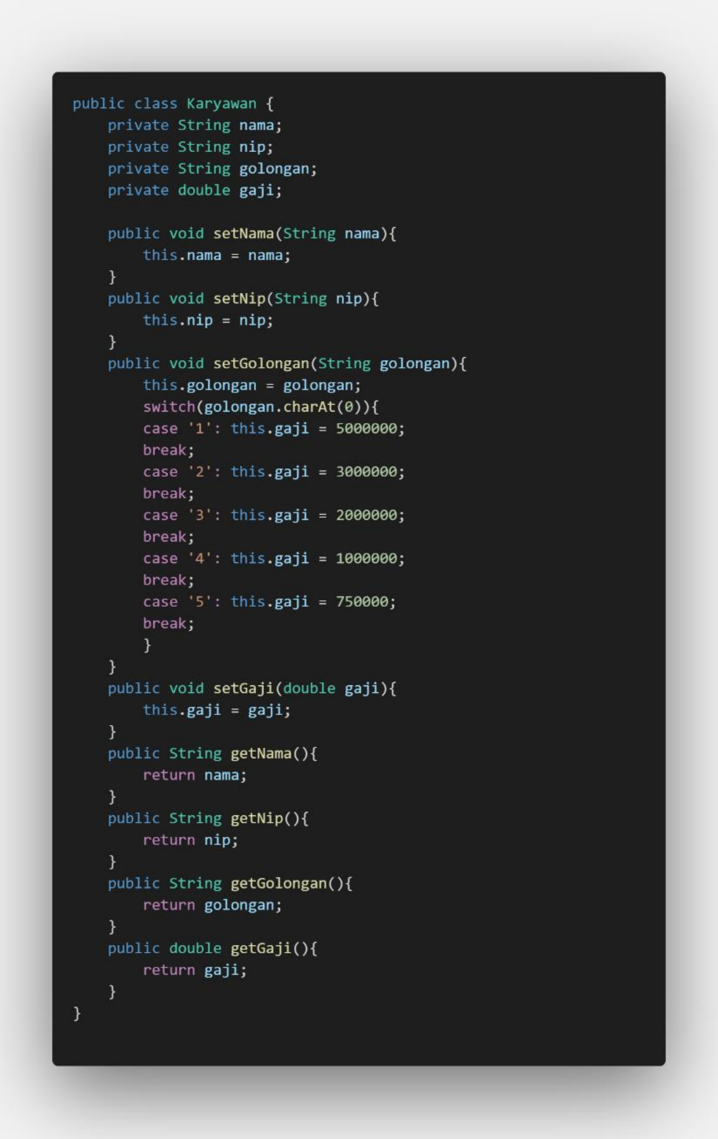
Jobsheet-9: Overloading dan Overriding

Object Oriented Programming

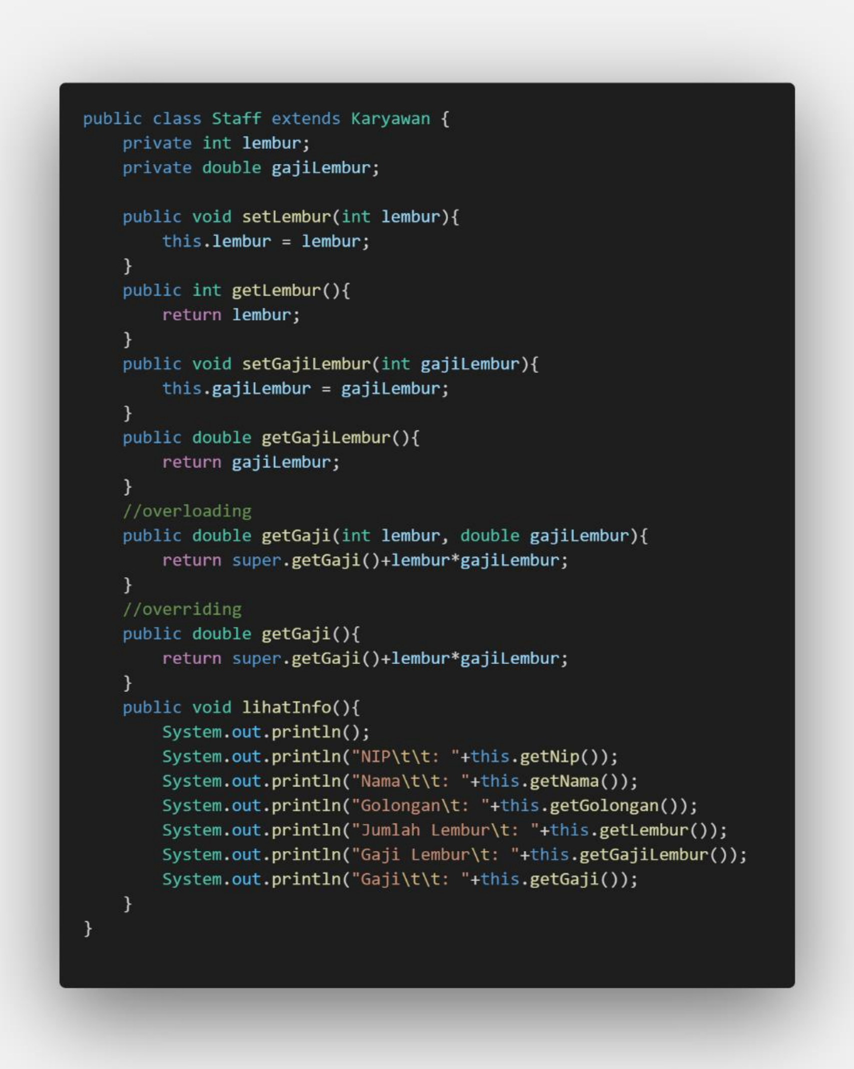
Pengampu : Tim Ajar Object Oriented Programming

2023

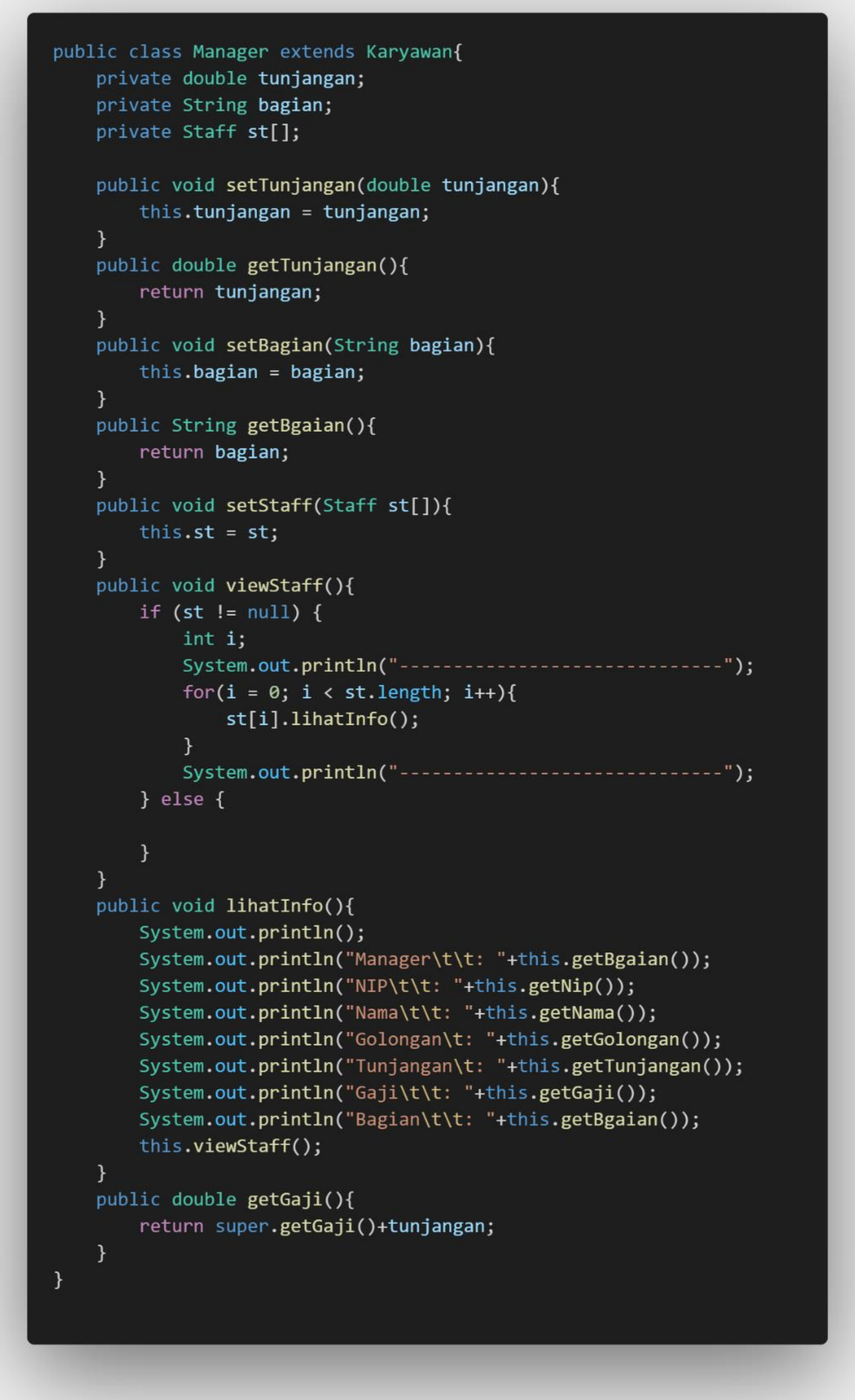
Nama	:	Hawa Esanda
NIM	:	2241720079
Kelas	:	TI-2I

No	Keterangan
Percobaan 1	
	

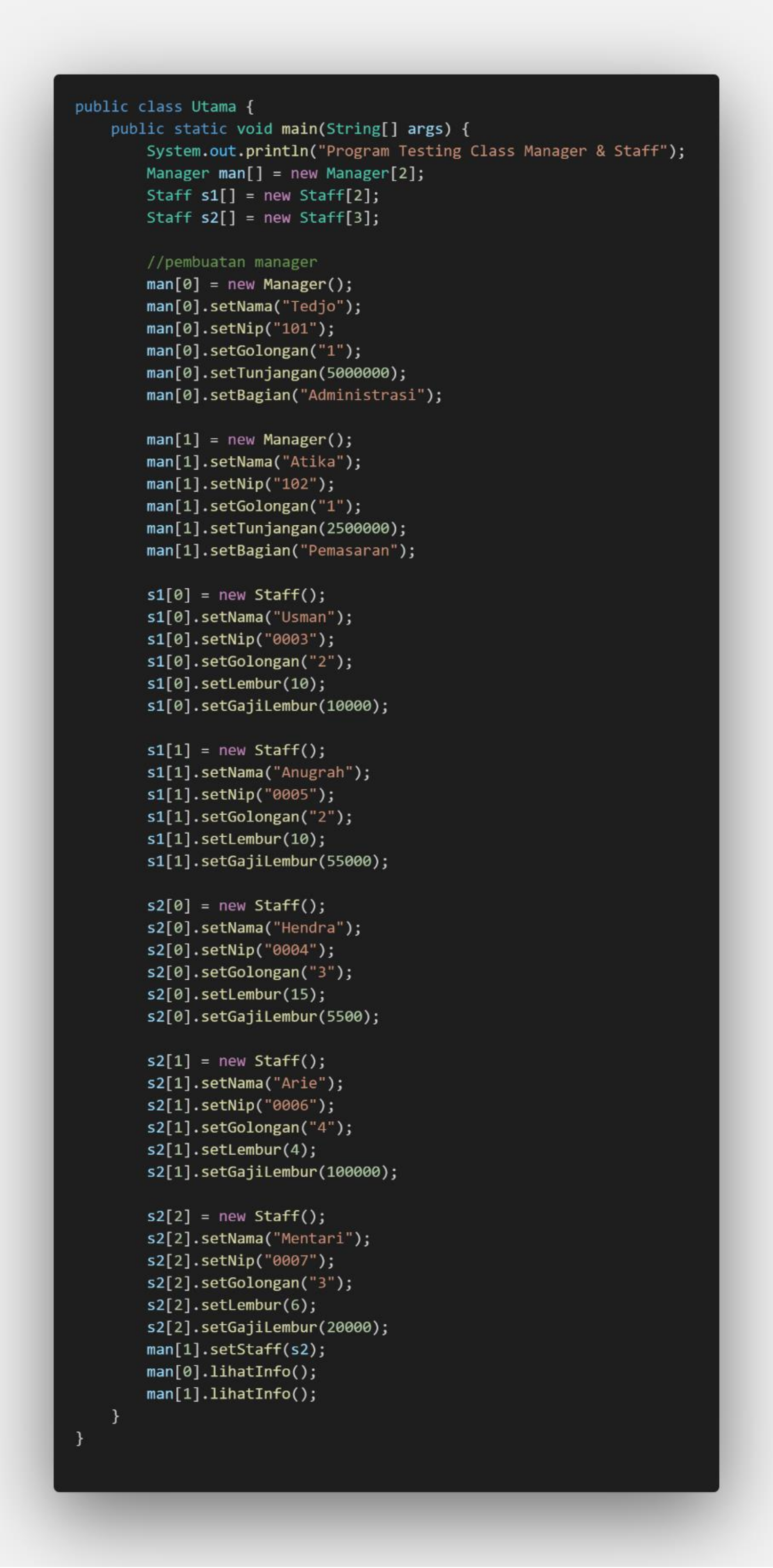
```
public class Karyawan {
    private String nama;
    private String nip;
    private String golongan;
    private double gaji;

    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;
    }
}
```



```java
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(int gajiLembur){
        this.gajiLembur = gajiLembur;
    }
    public double getGajiLembur(){
        return gajiLembur;
    }
    //overloading
    public double getGaji(int lembur, double gajiLembur){
        return super.getGaji()+lembur*gajiLembur;
    }
    //overriding
    public double getGaji(){
        return super.getGaji()+lembur*gajiLembur;
    }
    public void lihatInfo(){
        System.out.println();
        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.println("Gaji Lembur\t: "+this.getGajiLembur());
        System.out.println("Gaji\t\t: "+this.getGaji());
    }
}
```

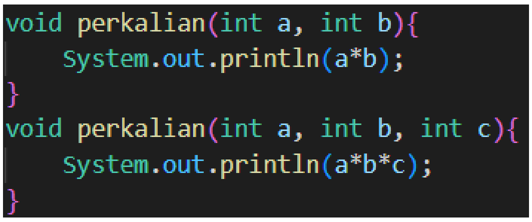


```
public class Manager extends Karyawan{
    private double tunjangan;
    private String bagian;
    private Staff st[];

    public void setTunjangan(double tunjangan){
        this.tunjangan = tunjangan;
    }
    public double getTunjangan(){
        return tunjangan;
    }
    public void setBagian(String bagian){
        this.bagian = bagian;
    }
    public String getBgaian(){
        return bagian;
    }
    public void setStaff(Staff st[]){
        this.st = st;
    }
    public void viewStaff(){
        if (st != null) {
            int i;
            System.out.println("-----------------------------");
            for(i = 0; i < st.length; i++){
                st[i].lihatInfo();
            }
            System.out.println("-----------------------------");
        } else {

        }
    }
    public void lihatInfo(){
        System.out.println();
        System.out.println("Manager\t\t: "+this.getBgaian());
        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("Tunjangan\t: "+this.getTunjangan());
        System.out.println("Gaji\t\t: "+this.getGaji());
        System.out.println("Bagian\t\t: "+this.getBgaian());
        this.viewStaff();
    }
    public double getGaji(){
        return super.getGaji()+tunjangan;
    }
}
```

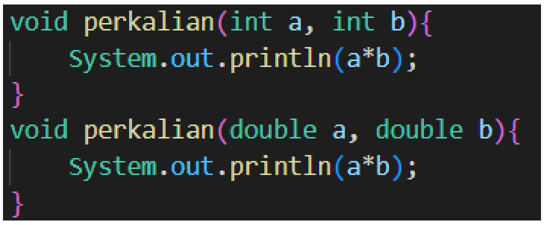


```java
public class Utama {
    public static void main(String[] args) {
        System.out.println("Program Testing Class Manager & Staff");
        Manager man[] = new Manager[2];
        Staff s1[] = new Staff[2];
        Staff s2[] = new Staff[3];

        //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");

        s1[0] = new Staff();
        s1[0].setNama("Usman");
        s1[0].setNip("0003");
        s1[0].setGolongan("2");
        s1[0].setLembur(10);
        s1[0].setGajiLembur(10000);

        s1[1] = new Staff();
        s1[1].setNama("Anugrah");
        s1[1].setNip("0005");
        s1[1].setGolongan("2");
        s1[1].setLembur(10);
        s1[1].setGajiLembur(55000);

        s2[0] = new Staff();
        s2[0].setNama("Hendra");
        s2[0].setNip("0004");
        s2[0].setGolongan("3");
        s2[0].setLembur(15);
        s2[0].setGajiLembur(5500);

        s2[1] = new Staff();
        s2[1].setNama("Arie");
        s2[1].setNip("0006");
        s2[1].setGolongan("4");
        s2[1].setLembur(4);
        s2[1].setGajiLembur(100000);

        s2[2] = new Staff();
        s2[2].setNama("Mentari");
        s2[2].setNip("0007");
        s2[2].setGolongan("3");
        s2[2].setLembur(6);
        s2[2].setGajiLembur(20000);
        man[1].setStaff(s2);
        man[0].lihatInfo();
        man[1].lihatInfo();
    }
}
```

Latihan	

4.1	Dari source coding diatas terletak dimanakah overloading?
	Jawab :
	
	`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);`
	`}`

4.2	Jika terdapat overloading ada berapa jumlah parameter yang berbeda? **Jawab :** Ada 3 parameter yang berbeda

4.3	Dari source coding diatas terletak dimanakah overloading?
	Jawab :
	
	`void perkalian(int a, int b){`
	`    System.out.println(a*b);`
	`}`
	`void perkalian(double a, double b){`
	`    System.out.println(a*b);`
	`}`

4.4	Jika terdapat overloading ada berapa tipe parameter yang berbeda? **Jawab:** ada 2 tipe parameter yang berbeda
4.5	Dari source coding diatas terletak dimanakah overriding? **Jawab :** overriding terjadi pada method **swim** di kelas **Piranha**, karena **Piranha** merupakan subclass dari **Ikan** dan meng-override (menggantikan) method **swim** yang ada di superclass.
4.6	Jabarkanlah apabila sourcoding diatas jika terdapat overriding? **Jawab :** jika terdapat overriding, kita dapat melihat bahwa kelas **Piranha** memiliki method **swim** yang sama dengan kelas **Ikan**. Overriding terjadi ketika subclass mengimplementasikan kembali method yang sudah ada di superclass dengan nama, tipe data, dan parameter yang sama.