LAPORAN Tugas W6
Inheritance, Abstract Class and Interface

Laporan ini dibuat untuk memenuhi tugas mata kuliah Pemrograman Berbasis Objek

Disusun oleh :

Luthfie Yannuardy 211511019

Program Studi Diploma III Teknik Informatika
Jurusan Teknik Komputer dan Informatika
POLITEKNIK NEGERI BANDUNG
2022

Daftar Isi

Source Code:

Sortable.java

```
package task3;

abstract class Sortable {
    public static void shell_sort(Sortable[] array) {
        int i, j, increment;
        Sortable temp;
        increment = 3;
        while (increment > 0) {
            for (i = 0; i < array.length; i++) {
                j = i;
                temp = array[i];
                while ((j >= increment) && (array[j - increment].compareTo(temp) == 1)) {
                    array[j] = array[j - increment];
                    j = j - increment;
                }
                array[j] = temp;
            }
            if (increment / 2 != 0)
                increment = increment / 2;
            else if (increment == 1)
                increment = 0;
            else
                increment = 1;
        }
    }

    public int compareTo(Sortable other) {
        return 0;
    }
}
```

Employee.java

```
package task3;

class Employee extends Sortable{
    public Employee(String n, double s, int day, int month, int year) {
        name = n;
        salary = s;
        hireday = day;
        hiremonth = month;
        hireyear = year;
    }
```

```
    public void print() {
        System.out.println(name + " " + salary + " " + hireYear());
    }

    public void raiseSalary(double byPercent) {
        salary *= 1 + byPercent / 100;
    }

    public int hireYear() {
        return hireyear;
    }

    @Override
    public int compareTo(Sortable other) {
        Employee otherEmployee = (Employee) other;
        if (this.salary < otherEmployee.salary)
            return -1;
        if (this.salary > otherEmployee.salary)
            return 1;
        return 0;
    }

    private String name;
    private double salary;
    private int hireday;
    private int hiremonth;
    private int hireyear;
}
```

Manager.java

```
package task3;

import java.util.Calendar;
import java.util.GregorianCalendar;

class Manager extends Employee {
    public Manager(String n, double s, int d, int m, int y) {
        super(n, s, d, m, y);
        secretaryName = "";
```

```
    }

    public void raiseSalary(double byPercent) {
        // add 1/2% bonus for every year of service
        GregorianCalendar todaysDate = new GregorianCalendar();
        int currentYear = todaysDate.get(Calendar.YEAR);
        double bonus = 0.5 * (currentYear - hireYear());
        super.raiseSalary(byPercent + bonus);
    }

    public String getSecretaryName() {
        return secretaryName;
    }

    private String secretaryName;
}
```

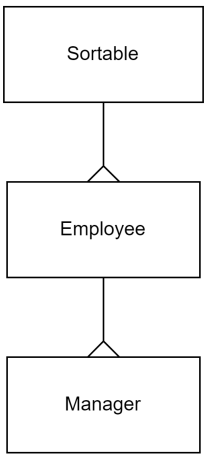
EmployeeTest.java

```
package task3;

public class EmployeeTest {
    public static void main(String[] args) {
        Employee[] staff = new Employee[4];
        staff[0] = new Employee("Antonio Rossi", 2000000, 1, 10, 1989);
        staff[1] = new Employee("Maria Bianchi", 2500000, 1, 12, 1991);
        staff[2] = new Employee("Isabel Vidal", 3000000, 1, 11, 1993);
        staff[3] = new Employee("H2SO4", 2700000, 1, 11, 1993);

        Sortable.shell_sort(staff);
        int i;
        for (i = 0; i < 4; i++)
            staff[i].raiseSalary(5);
        for (i = 0; i < 4; i++)
            staff[i].print();
    }
}
```

ManagerTest.java

```java
package task3;

public class ManagerTest {
    public static void main(String[] args) {
        Employee[] staff = new Employee[3];
        staff[0] = new Employee("Antonio Rossi", 2000000, 1, 10, 1989);
        staff[1] = new Employee("Isabel Vidal", 3000000, 1, 11, 1993);
        staff[2] = new Manager("Maria Bianchi", 2500000, 1, 12, 1991);
        Sortable.shell_sort(staff);
        int i;
        for (i = 0; i < 3; i++)
            staff[i].raiseSalary(5);
        for (i = 0; i < 3; i++)
            staff[i].print();
    }
}
```

1. **Diagram Hirarki Kelas**



2. **Case 1**

   Hal yang dilakukan:

   1) Menambahkan class Sortable yang berfungsi untuk melakukan sorting, class ini merupakan superclass dari class Employee.
   2) Melakukan overriding pada method compareTo di class Employee. Pada kasus ini overriding method bertujuan untuk melakukan komparasi atribut salary, agar proses sorting pada class Sortable dilakukan berdasarkan atribut salary.

```
@Override
public int compareTo(Sortable other) {
    Employee otherEmployee = (Employee) other;
    if (this.salary < otherEmployee.salary)
        return -1;
    if (this.salary > otherEmployee.salary)
        return 1;
    return 0;
}
```

   Method compareTo yang di override ini nantinya akan dipanggil pada method shell_sort() di class Sortable.

```java
public class Sortable {
    public static void shell_sort(Sortable[] array) {
        int i, j, increment;
        Sortable temp;
        increment = 3;
        while (increment > 0) {
            for (i = 0; i < array.length; i++) {
                j = i;
                temp = array[i];
                while ((j >= increment) && (array[j - increment].compareTo(temp) == 1)) {
                    array[j] = array[j - increment];
                    j = j - increment;
                }
                array[j] = temp;
            }
            if (increment / 2 != 0)
                increment = increment / 2;
            else if (increment == 1)
                increment = 0;
            else
                increment = 1;
        }
    }

    public int compareTo(Sortable other) {
        return 0;
    }
}
```

Di override pada Class Employee

3) Sehingga array dari object Employee dapat di sort berdasarkan salary seperti pada contoh berikut:

```java
public class EmployeeTest {
    Run | Debug
    public static void main(String[] args) {
        Employee[] staff = new Employee[4];
        staff[0] = new Employee(n: "Antonio Rossi", s: 2000000, day: 1, month: 10, year: 1989);
        staff[1] = new Employee(n: "Maria Bianchi", s: 2500000, day: 1, month: 12, year: 1991);
        staff[2] = new Employee(n: "Isabel Vidal", s: 3000000, day: 1, month: 11, year: 1993);
        staff[3] = new Employee(n: "H2SO4", s: 2700000, day: 1, month: 11, year: 1993);

        Sortable.shell_sort(staff);
        int i;
        for (i = 0; i < 4; i++)
            staff[i].raiseSalary(byPercent: 5);
        for (i = 0; i < 4; i++)
            staff[i].print();
    }
}
```

Output:

```
Antonio Rossi 2100000.0 1989
Maria Bianchi 2625000.0 1991
H2SO4 2835000.0 1993
Isabel Vidal 3150000.0 1993
```

3. Case 2

1) Kita dapat membuat object Manager pada variable yang bertipe Employee karena Manager adalah subclass dari Employee (Polymorphism).
2) Karena class Manager merupakan subclass dari class Employee, class ini dapat mengakses method shell_sort() dan method compareTo yang sudah di override tadi. Oleh karena itu, object dari Manager dapat juga di sort berdasarkan salary.

```java
public class ManagerTest {
    Run | Debug
    public static void main(String[] args) {
        Employee[] staff = new Employee[3];
        staff[0] = new Employee(n: "Antonio Rossi", s: 2000000, day: 1, month: 10, year: 1989);
        staff[1] = new Employee(n: "Isabel Vidal", s: 3000000, day: 1, month: 11, year: 1993);
        staff[2] = new Manager(n: "Maria Bianchi", s: 2500000, d: 1, m: 12, y: 1991);
        Sortable.shell_sort(staff);
        int i;
        for (i = 0; i < 3; i++)
            staff[i].raiseSalary(byPercent: 5);
        for (i = 0; i < 3; i++)
            staff[i].print();
    }
}
```

Output:

```
Antonio Rossi 2100000.0 1989
Maria Bianchi 3012500.0 1991
Isabel Vidal 3150000.0 1993
```

4. **Link Source Code:**
   https://github.com/LuthfieY/Latihan-Java-Minggu-6