INHERITANCE, ABSTRACT CLASS AND INTERFACE

LAPORAN

Diajukan untuk memenuhi Tugas Mata Kuliah Pemrograman Berorientasi Objek

Disusun Oleh
MUHAMMAD HAFIZH AULIANSYAH
211511047

PROGRAM DIPLOMA III TEKNIK INFORMATIKA
POLITEKNIK NEGERI BANDUNG
BANDUNG
2022

PERSOALAN

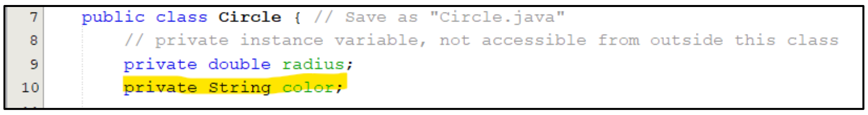
Link Repository : https://github.com/HafizhAuliansyah/211511047_M-Hafizh-A_PraktikumPBO.git

1.1. Exercise 1

Jawaban Soal :

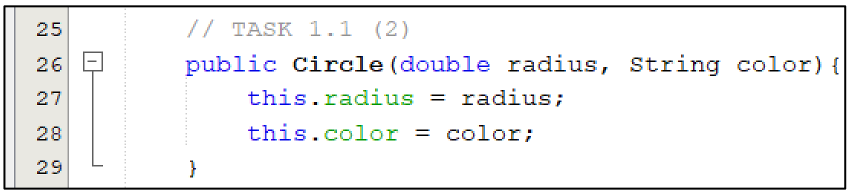
- Task 1.1 *(in Circle.java)*

  (1) Add variabel String Color



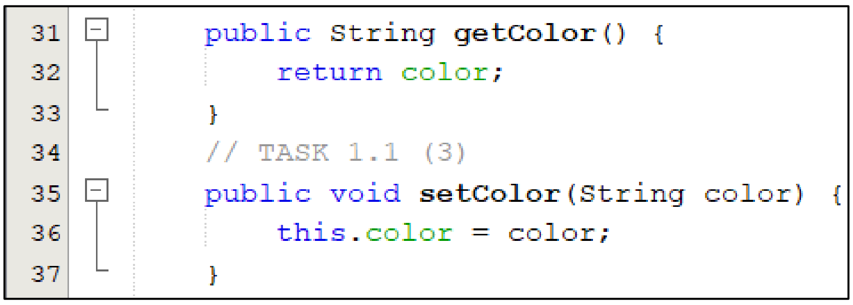
```java
public class Circle { // Save as "Circle.java"
    // private instance variable, not accessible from outside this class
    private double radius;
    private String color;
```

  (2) Constructor Circle(radius : double, color : string)



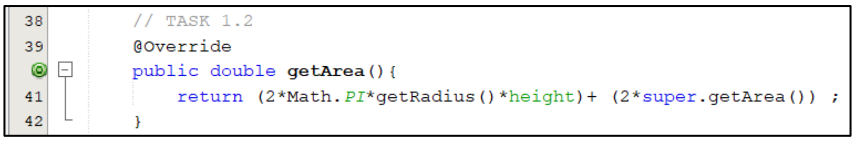
```java
    // TASK 1.1 (2)
    public Circle(double radius, String color){
        this.radius = radius;
        this.color = color;
    }
```

  (3) Getter and setter for color



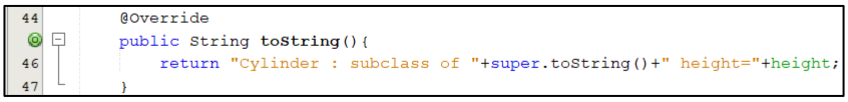
```java
    public String getColor() {
        return color;
    }
    // TASK 1.1 (3)
    public void setColor(String color) {
        this.color = color;
    }
```

- Task 1.2 *(in Cylinder.java)*



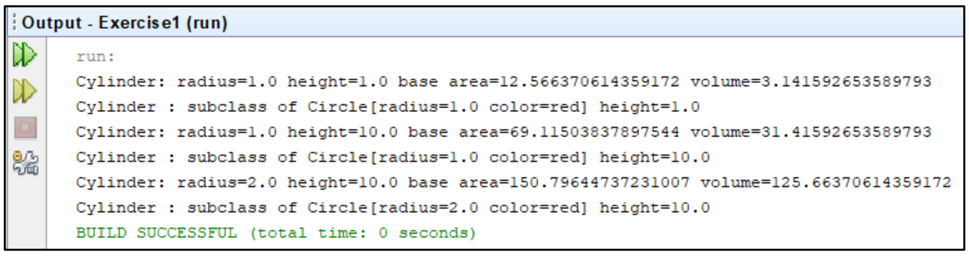
```java
    // TASK 1.2
    @Override
    public double getArea(){
        return (2*Math.PI*getRadius()*height)+ (2*super.getArea()) ;
    }
```

- Task 1.3 *(in Cylinder.java)*



```java
    @Override
    public String toString(){
        return "Cylinder : subclass of "+super.toString()+" height="+height;
    }
```

Hasil Akhir *(Run TestCylinder.java)* :



Output - Exercise1 (run)

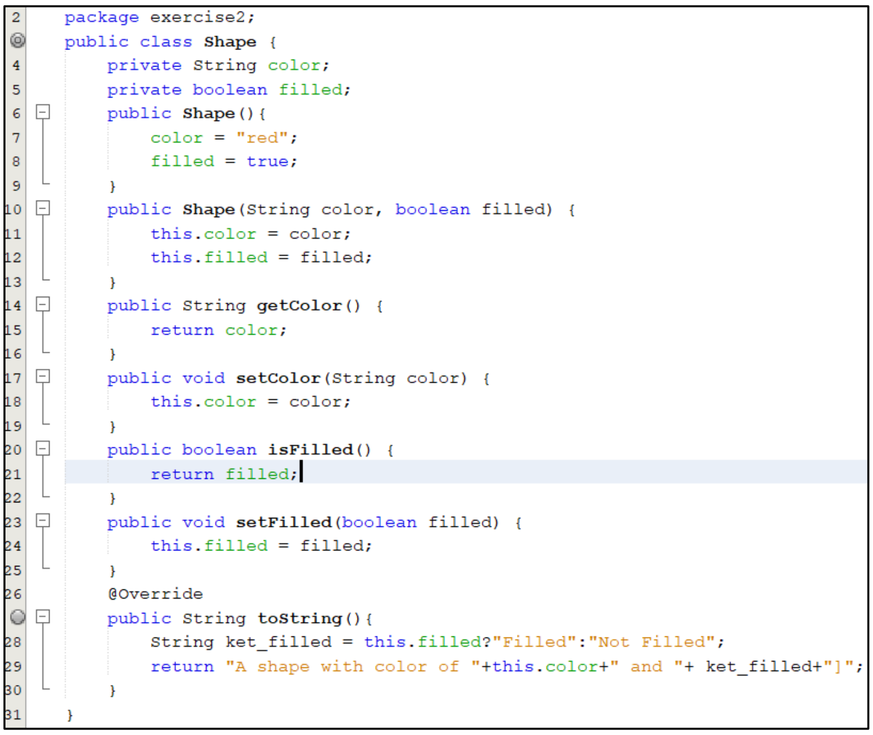
```
run:
Cylinder: radius=1.0 height=1.0 base area=12.566370614359172 volume=3.141592653589793
Cylinder : subclass of Circle[radius=1.0 color=red] height=1.0
Cylinder: radius=1.0 height=10.0 base area=69.11503837897544 volume=31.41592653589793
Cylinder : subclass of Circle[radius=1.0 color=red] height=10.0
Cylinder: radius=2.0 height=10.0 base area=150.79644737231007 volume=125.66370614359172
Cylinder : subclass of Circle[radius=2.0 color=red] height=10.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

Permasalahan yang dihadapi : -
Solusi : -
Teman yang membantu : -

1.2. Exercise 2

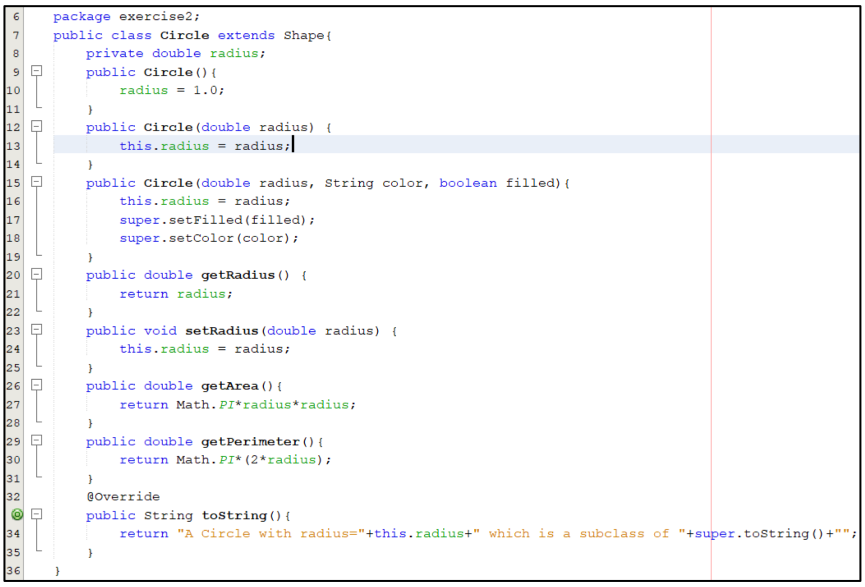
Jawaban Soal :

In Shape.java



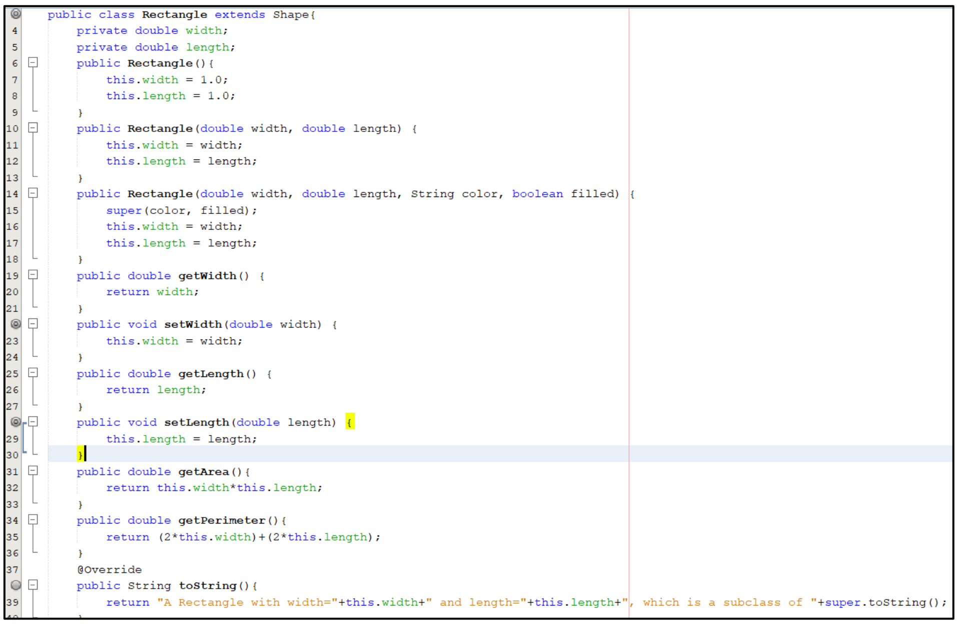
```java
package exercise2;
public class Shape {
    private String color;
    private boolean filled;
    public Shape(){
        color = "red";
        filled = true;
    }
    public Shape(String color, boolean filled) {
        this.color = color;
        this.filled = filled;
    }
    public String getColor() {
        return color;
    }
    public void setColor(String color) {
        this.color = color;
    }
    public boolean isFilled() {
        return filled;
    }
    public void setFilled(boolean filled) {
        this.filled = filled;
    }
    @Override
    public String toString(){
        String ket_filled = this.filled?"Filled":"Not Filled";
        return "A shape with color of "+this.color+" and "+ ket_filled+"]";
    }
}
```

In Circle.java



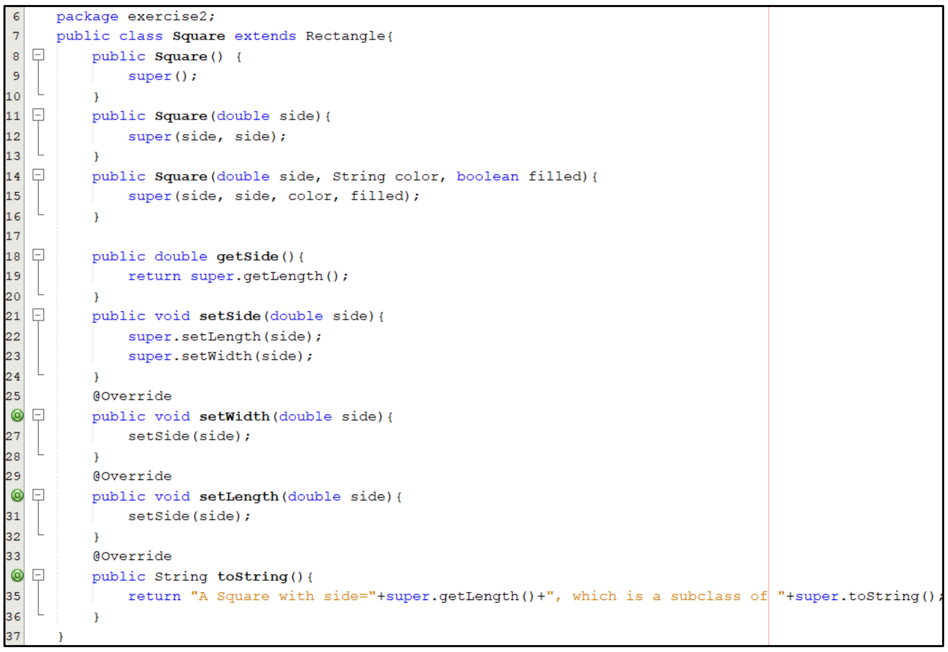
```java
package exercise2;
public class Circle extends Shape{
    private double radius;
    public Circle(){
        radius = 1.0;
    }
    public Circle(double radius) {
        this.radius = radius;
    }
    public Circle(double radius, String color, boolean filled){
        this.radius = radius;
        super.setFilled(filled);
        super.setColor(color);
    }
    public double getRadius() {
        return radius;
    }
    public void setRadius(double radius) {
        this.radius = radius;
    }
    public double getArea(){
        return Math.PI*radius*radius;
    }
    public double getPerimeter(){
        return Math.PI*(2*radius);
    }
    @Override
    public String toString(){
        return "A Circle with radius="+this.radius+" which is a subclass of "+super.toString()+"";
    }
}
```

In Rectangle.java



```java
public class Rectangle extends Shape{
    private double width;
    private double length;
    public Rectangle(){
        this.width = 1.0;
        this.length = 1.0;
    }
    public Rectangle(double width, double length) {
        this.width = width;
        this.length = length;
    }
    public Rectangle(double width, double length, String color, boolean filled) {
        super(color, filled);
        this.width = width;
        this.length = length;
    }
    public double getWidth() {
        return width;
    }
    public void setWidth(double width) {
        this.width = width;
    }
    public double getLength() {
        return length;
    }
    public void setLength(double length) {
        this.length = length;
    }
    public double getArea(){
        return this.width*this.length;
    }
    public double getPerimeter(){
        return (2*this.width)+(2*this.length);
    }
    @Override
    public String toString(){
        return "A Rectangle with width="+this.width+" and length="+this.length+", which is a subclass of "+super.toString();
```

In Square.java

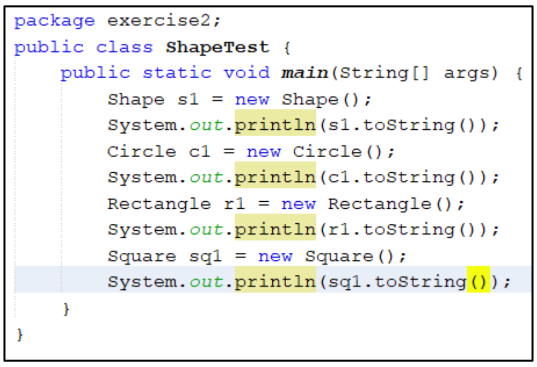


```java
package exercise2;
public class Square extends Rectangle{
    public Square() {
        super();
    }
    public Square(double side){
        super(side, side);
    }
    public Square(double side, String color, boolean filled){
        super(side, side, color, filled);
    }

    public double getSide(){
        return super.getLength();
    }
    public void setSide(double side){
        super.setLength(side);
        super.setWidth(side);
    }
    @Override
    public void setWidth(double side){
        setSide(side);
    }
    @Override
    public void setLength(double side){
        setSide(side);
    }
    @Override
    public String toString(){
        return "A Square with side="+super.getLength()+", which is a subclass of "+super.toString();
    }
}
```

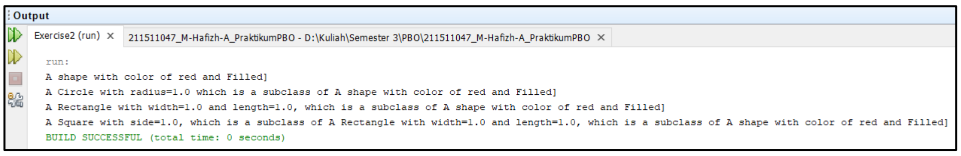
Hasil Akhir :

Create ShapeTest.java



```java
package exercise2;
public class ShapeTest {
    public static void main(String[] args) {
        Shape s1 = new Shape();
        System.out.println(s1.toString());
        Circle c1 = new Circle();
        System.out.println(c1.toString());
        Rectangle r1 = new Rectangle();
        System.out.println(r1.toString());
        Square sq1 = new Square();
        System.out.println(sq1.toString());
    }
}
```

Output :



```
Output
Exercise2 (run) ×   211511047_M-Hafizh-A_PraktikumPBO - D:\Kuliah\Semester 3\PBO\211511047_M-Hafizh-A_PraktikumPBO ×
run:
A shape with color of red and Filled]
A Circle with radius=1.0 which is a subclass of A shape with color of red and Filled]
A Rectangle with width=1.0 and length=1.0, which is a subclass of A shape with color of red and Filled]
A Square with side=1.0, which is a subclass of A Rectangle with width=1.0 and length=1.0, which is a subclass of A shape with color of red and Filled]
BUILD SUCCESSFUL (total time: 0 seconds)
```

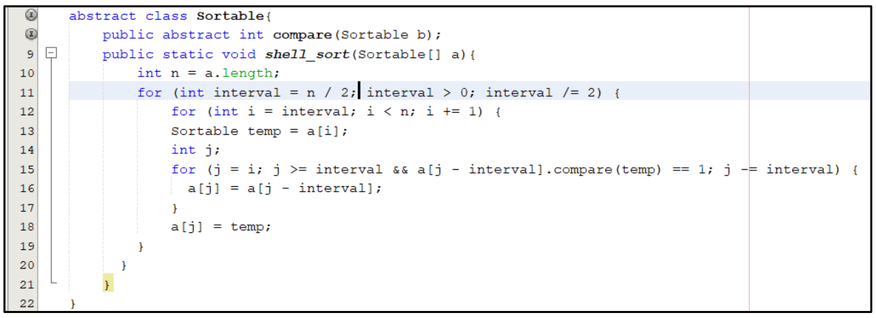
Permsalahan : Bingung cara menyesuaikan length dan width saat setWidth dan setLength supaya sama
Solusi : Cukup dengan memanggil setSide, maka lenth dan width akans selalu sama
Teman yang membantu : M Rizki Halomoan

1.3. Exercise 3

Jawaban Soal :

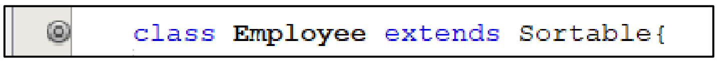
- Case 1

  In Sortable.java

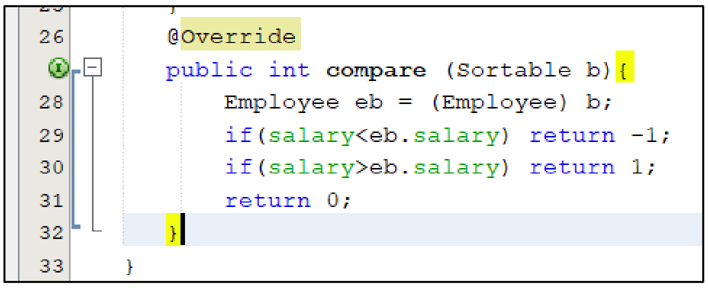


```java
abstract class Sortable{
    public abstract int compare(Sortable b);
    public static void shell_sort(Sortable[] a){
        int n = a.length;
        for (int interval = n / 2; interval > 0; interval /= 2) {
            for (int i = interval; i < n; i += 1) {
            Sortable temp = a[i];
            int j;
            for (j = i; j >= interval && a[j - interval].compare(temp) == 1; j -= interval) {
              a[j] = a[j - interval];
            }
            a[j] = temp;
        }
      }
    }
}
```

  In Employee.java

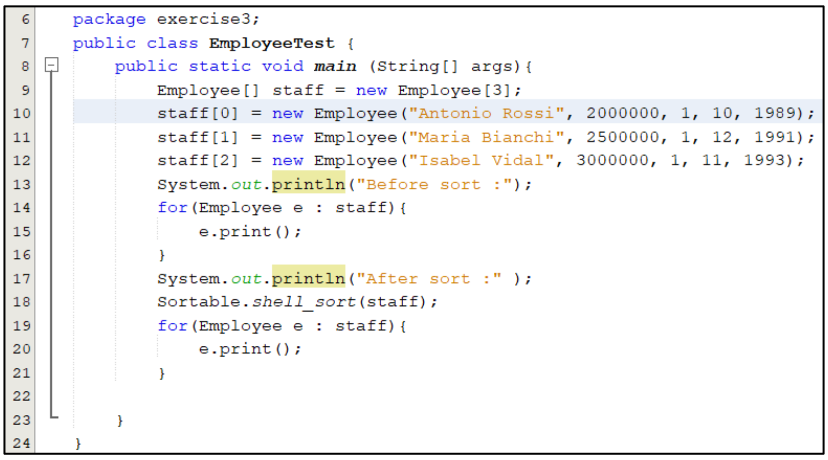


```java
class Employee extends Sortable{
```



```java
@Override
public int compare (Sortable b){
    Employee eb = (Employee) b;
    if(salary<eb.salary) return -1;
    if(salary>eb.salary) return 1;
    return 0;
}
}
```

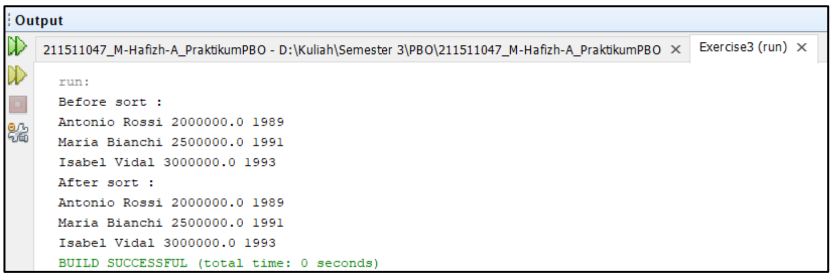
In EmployeeTest.java



```
package exercise3;
public class EmployeeTest {
    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("Maria Bianchi", 2500000, 1, 12, 1991);
        staff[2] = new Employee("Isabel Vidal", 3000000, 1, 11, 1993);
        System.out.println("Before sort :");
        for(Employee e : staff){
            e.print();
        }
        System.out.println("After sort :" );
        Sortable.shell_sort(staff);
        for(Employee e : staff){
            e.print();
        }

    }
}
```

Output :



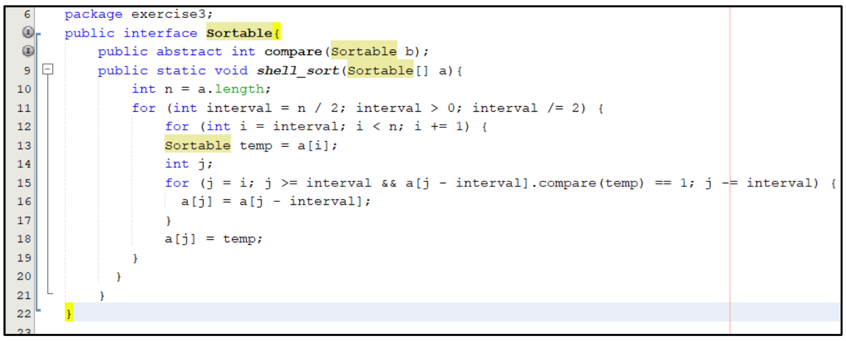
Output

211511047_M-Hafizh-A_PraktikumPBO - D:\Kuliah\Semester 3\PBO\211511047_M-Hafizh-A_PraktikumPBO × | Exercise3 (run) ×

```
run:
Before sort :
Antonio Rossi 2000000.0 1989
Maria Bianchi 2500000.0 1991
Isabel Vidal 3000000.0 1993
After sort :
Antonio Rossi 2000000.0 1989
Maria Bianchi 2500000.0 1991
Isabel Vidal 3000000.0 1993
BUILD SUCCESSFUL (total time: 0 seconds)
```

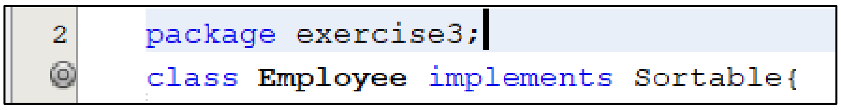
- Case 2

  Langkah 1 : *Mengubah Sortable dari abstract class menjadi interface (Sortable.java)*



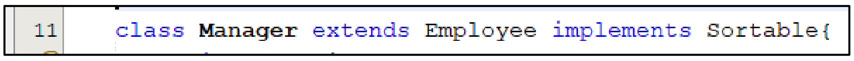
```
package exercise3;
public interface Sortable{
    public abstract int compare(Sortable b);
    public static void shell_sort(Sortable[] a){
        int n = a.length;
        for (int interval = n / 2; interval > 0; interval /= 2) {
            for (int i = interval; i < n; i += 1) {
            Sortable temp = a[i];
            int j;
            for (j = i; j >= interval && a[j - interval].compare(temp) == 1; j -= interval) {
              a[j] = a[j - interval];
            }
            a[j] = temp;
          }
        }
    }
}
```

  Langkah 2 : *Mengubah "extends Sortable" menjadi "implements Sortable" (Employee.java)*



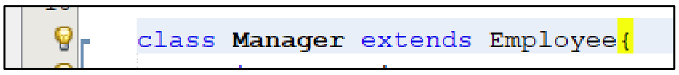
```
package exercise3;
class Employee implements Sortable{
```

Langkah 3 (Cara 1) : *Menambahkan "extends Employee implements Sortable" pada class Manager (Manager.java)*



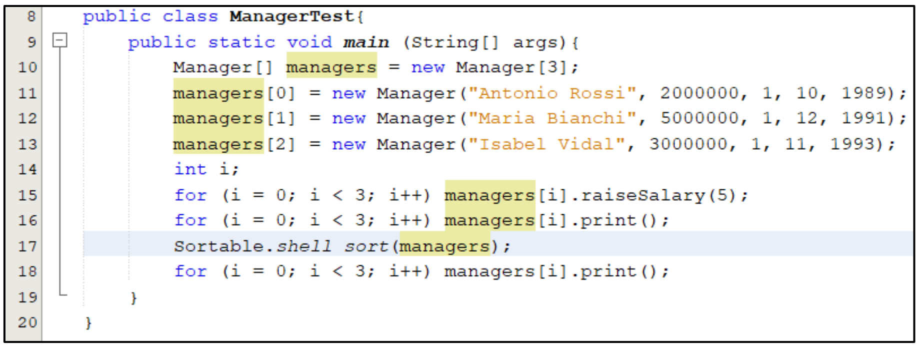
```java
    class Manager extends Employee implements Sortable{
```

Langkah 3 (Cara 2) : Jika mengikuti case 1, maka cukup tambahkan "extends Employee" maka Manager sudah implements Sortable

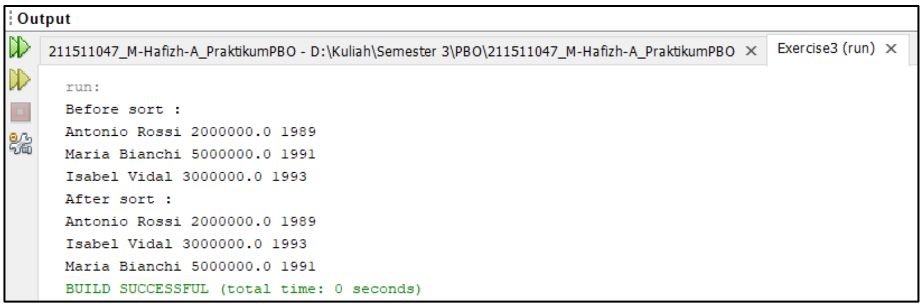


```java
    class Manager extends Employee{
```

Langkah 4 : Pengujian (ManagerTest.java)



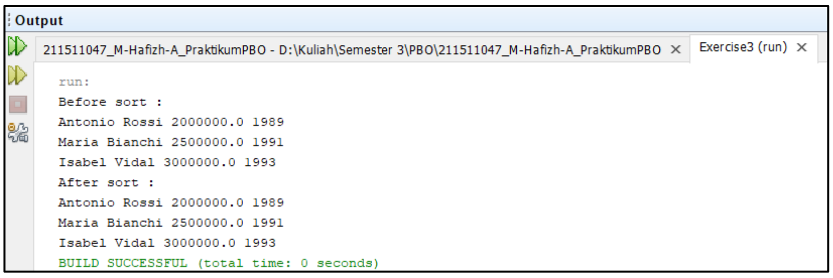
```java
public class ManagerTest{
    public static void main (String[] args){
        Manager[] managers = new Manager[3];
        managers[0] = new Manager("Antonio Rossi", 2000000, 1, 10, 1989);
        managers[1] = new Manager("Maria Bianchi", 5000000, 1, 12, 1991);
        managers[2] = new Manager("Isabel Vidal", 3000000, 1, 11, 1993);
        int i;
        for (i = 0; i < 3; i++) managers[i].raiseSalary(5);
        for (i = 0; i < 3; i++) managers[i].print();
        Sortable.shell_sort(managers);
        for (i = 0; i < 3; i++) managers[i].print();
    }
}
```


Output
211511047_M-Hafizh-A_PraktikumPBO - D:\Kuliah\Semester 3\PBO\211511047_M-Hafizh-A_PraktikumPBO
Exercise3 (run)

```
run:
Before sort :
Antonio Rossi 2000000.0 1989
Maria Bianchi 5000000.0 1991
Isabel Vidal 3000000.0 1993
After sort :
Antonio Rossi 2000000.0 1989
Isabel Vidal 3000000.0 1993
Maria Bianchi 5000000.0 1991
BUILD SUCCESSFUL (total time: 0 seconds)
```

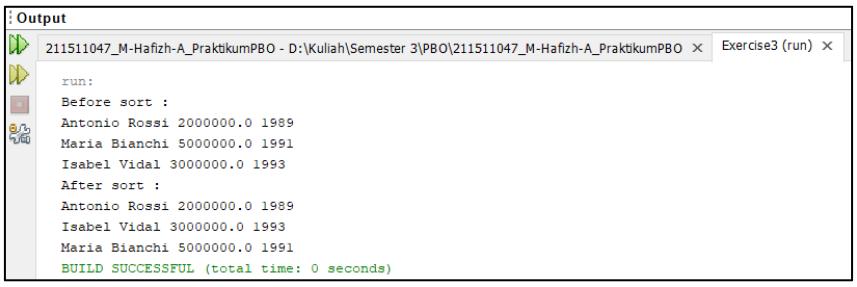
Hasil Akhir :

- Case 1


Output
211511047_M-Hafizh-A_PraktikumPBO - D:\Kuliah\Semester 3\PBO\211511047_M-Hafizh-A_PraktikumPBO
Exercise3 (run)

```
run:
Before sort :
Antonio Rossi 2000000.0 1989
Maria Bianchi 2500000.0 1991
Isabel Vidal 3000000.0 1993
After sort :
Antonio Rossi 2000000.0 1989
Maria Bianchi 2500000.0 1991
Isabel Vidal 3000000.0 1993
BUILD SUCCESSFUL (total time: 0 seconds)
```

- Case 2


Output
211511047_M-Hafizh-A_PraktikumPBO - D:\Kuliah\Semester 3\PBO\211511047_M-Hafizh-A_PraktikumPBO
Exercise3 (run)

```
run:
Before sort :
Antonio Rossi 2000000.0 1989
Maria Bianchi 5000000.0 1991
Isabel Vidal 3000000.0 1993
After sort :
Antonio Rossi 2000000.0 1989
Isabel Vidal 3000000.0 1993
Maria Bianchi 5000000.0 1991
BUILD SUCCESSFUL (total time: 0 seconds)
```

Permasalahan : Code shell sort yang didapat berdasarkan tipe data integer tidak sesuai kasus
Solusi : Memahami code dan mengubah kondisi komparasi dengan memanfaatkan fungsi Compare()
Teman yang membantu : -