16TIN2054 – Teknik Pemrograman Praktek

Week 6

Dikerjakan oleh:

Muhammad Azhar Alauddin – 201524013

1AD4 Jurusan Teknik Komputer dan Informatika

Tugas ini dikumpulkan untuk memenuhi sebagian persyaratan kelulusan mata kuliah Teknik Pemrograman Praktek

Program Studi D4 Teknik Informatika

Jurusan Teknik Komputer dan Informatika

Politeknik Negeri Bandung

2020/2021

Task 1.1

Modify class Circle
Modify class Circle, add :
1. variable color : string
2. Constructor Circle(radius : double, color : string)
3. Getter and setter for color

Circle.java

```
package Task1;

public class Circle {
	private double radius;
	private String color;

	// Constructor(overloaded)
	public Circle() {
		radius = 1.0;
		color = "red";
	}

	public Circle(double r) {
		this.radius = r;
		color = "red";
	}

	public Circle(double r, String color) {
		this.radius = r;
		this.color = "red";
	}

	// Setter|Mutator
	public void setColor(String color) {
		this.color = color;
	}

	//  Getter|Accessor
	public String getColor() {
		return color;
	}

	// Returns the radius
	public double getRadius() {
		return radius;
	}

	// Returns the area
	public double getArea() {
		return radius*radius*Math.PI;
	}

	public String toString() {
		return "Circle[radius=" + radius + " color=" + color + "]";
	}
}
```

Task 1.2

Modify Overriding the getArea() method
Method Overriding and "Super": The subclass Cylinder inherits getArea() method from its superclass Circle. Try overriding the getArea() method in the subclass Cylinder to compute the surface area (=2π×radius×height + 2×base-area) of the cylinder instead of base area. That is, if getArea() is called by a Circle instance, it returns the area. If getArea() is called by a Cylinder instance, it returns the surface area of the cylinder. If you override the getArea() in the subclass Cylinder, the getVolume() no longer works. This is because the getVolume() uses the overridden getArea() method found in the same class. (Java runtime will search the superclass only if it cannot locate the method in this class). Fix the getVolume().

Cylinder.java

```java
package Task1;

public class Cylinder extends Circle{
	private double height;

	// Constructor
	public Cylinder() {
		super();
		height = 1.0;
	}

	public Cylinder(double height) {
		super();
		this.height = height;
	}

	public Cylinder(double radius, double height) {
		super(radius);
		this.height = height;
	}

	// Inherits getArea()
	public double getArea() {
		return ((2*(super.getArea())) + (2*Math.PI*getRadius()*height));
	}

	// Retrieving the height
	public double getHeight() {
		return height;
	}

	// Superclass method getArea()
	public double getVolume() {
		return (super.getArea()*height);
	}
}
```

Task 1.3

Provide a toString() method
Provide a toString() method to the Cylinder class, which overrides the toString() inherited from the superclass Circle

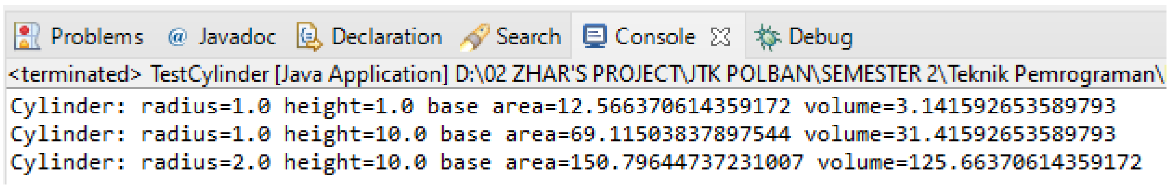
Cylinder.java

```
package Task1;

public class Cylinder extends Circle{
	private double height;

	// Constructor
	public Cylinder() {
		super();
		height = 1.0;
	}

	// Constructor
	public Cylinder(double height) {
		super();
		this.height = height;
	}

	// Constructor
	public Cylinder(double radius, double height) {
		super(radius);
		this.height = height;
	}

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

	// Retrieving the height
	public double getHeight() {
		return height;
	}

	// Superclass method getArea()
	public double getVolume() {
		return (super.getArea()*height);
	}

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

TestCylinder.java

```java
package Task1;

public class TestCylinder {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Cylinder c1 = new Cylinder();
		System.out.println("Cylinder:"
					+ " radius=" + c1.getRadius()
					+ " height=" + c1.getHeight()
					+ " base area=" + c1.getArea()
					+ " volume=" + c1.getVolume());

		Cylinder c2 = new Cylinder(10.0);
		System.out.println("Cylinder:"
					+ " radius=" + c2.getRadius()
					+ " height=" + c2.getHeight()
					+ " base area=" + c2.getArea()
					+ " volume=" + c2.getVolume());

		Cylinder c3 = new Cylinder(2.0,10.0);
		System.out.println("Cylinder:"
					+ " radius=" + c3.getRadius()
					+ " height=" + c3.getHeight()
					+ " base area=" + c3.getArea()
					+ " volume=" + c3.getVolume());
	}
}
```

Screenshoot hasil akhir program :



```
<terminated> TestCylinder [Java Application] D:\02 ZHAR'S PROJECT\JTK POLBAN\SEMESTER 2\Teknik Pemrograman\
Cylinder: radius=1.0 height=1.0 base area=12.566370614359172 volume=3.141592653589793
Cylinder: radius=1.0 height=10.0 base area=69.11503837897544 volume=31.41592653589793
Cylinder: radius=2.0 height=10.0 base area=150.79644737231007 volume=125.66370614359172
```

Task 2.1

Write a superclass called Shape (as shown in the class diagram), which contains:

- Two instance variables color (String) and filled (boolean).
- Two constructors: a no-arg (no-argument) constructor that initializes the color to "green" and filled to true, and a constructor that initializes the color and filled to the given values.
- Getter and setter for all the instance variables. By convention, the getter for a boolean variable xxx is called isXXX() (instead of getXxx() for all the other types).

- A toString() method that returns "A Shape with color of xxx and filled/Not filled".

Write a test program to test all the methods defined in Shape.

Write two subclasses of Shape called Circle and Rectangle, as shown in the class diagram.

The Circle class contains:

- An instance variable radius (double).
- Three constructors as shown. The no-arg constructor initializes the radius to 1.0.
- Getter and setter for the instance variable radius.
- Methods getArea() and getPerimeter().
- Override the toString() method inherited, to return "A Circle with radius=xxx, which is a subclass of yyy", where yyy is the output of the toString() method from the superclass.

The Rectangle class contains:

- Two instance variables width (double) and length (double).
- Three constructors as shown. The no-arg constructor initializes the width and length to 1.0.
- Getter and setter for all the instance variables.
- Methods getArea() and getPerimeter().
- Override the toString() method inherited, to return "A Rectangle with width=xxx and length=zzz, which is a subclass of yyy", where yyy is the output of the toString() method from the superclass.

Write a class called Square, as a subclass of Rectangle. Convince yourself that Square can be modeled as a subclass of Rectangle. Square has no instance variable, but inherits the instance variables width and length from its superclass Rectangle.

- Provide the appropriate constructors (as shown in the class diagram).
- Override the toString() method to return "A Square with side=xxx, which is a subclass of yyy", where yyy is the output of the toString() method from the superclass.
- Do you need to override the getArea() and getPerimeter()? Try them out.
- Override the setLength() and setWidth() to change both the width and length, so as to maintain the square geometry.

Shape.java

```java
package Task2;

public class Shape {
	private String color;
	private boolean filled;

	public Shape() {
		color = "green";
		filled = true;
	}

	public Shape(String c, boolean f) {
		color = c;
		filled = f;
	}

	// Getter|Accessor
	public String getColor() {
		return color;
	}

	public boolean isXXX() {
		return filled;
	}

	// Setter|Mutator
	public void setColor(String color) {
		this.color = color;
	}

	public void setFilled(boolean filled) {
		this.filled = filled;
	}

	public String toString() {
		return "A Shape with color of "+color+" and "+
			(filled? "filled":"not filled");
	}
}
```

Circle.java

```
package Task2;

public class Circle extends Shape{
	private double radius;

	// Constructor
	public Circle() {
		super();
		radius = 1.0;
	}

	public Circle(double radius, String color, boolean filled) {
		super(color, filled);
		this.radius=radius;
	}
	// Getter|Accessor
	public double getRadius() {
		return radius;
	}

	// Setter|Mutator
	public void setRadius(double radius) {
		this.radius = radius;
	}

	// Method getArea() and getPerimeter()
	public double getArea() {
		double Area = 2*Math.PI*Math.pow(radius,2);
		return Area;
```

```java
	}

	public double getPerimeter() {
		double Perimeter = 2*Math.PI*radius;
		return Perimeter;
	}

	// Overriding
	public String toString() {
		return "A Circle with radius="+radius+",which is a subclass of "+
			super.toString();
	}
}
```

Rectangle.java

```java
package Task2;

public class Rectangle extends Shape{
	private double width;
	private double length;

	// Constructor
	public Rectangle() {
		super();
		this.width = 1.0;
		this.length = 1.0;
	}

	public Rectangle(double width, double length) {
		super();
		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;
	}
	// Getter|Accessor
	public double getWidth() {
		return width;
	}

	public double getLength() {
		return length;
	}

	// Setter|Mutator
	public void setWidth(double width) {
		this.width = width;
	}

	public void setLength(double length) {
		this.length = length;
	}
	// getArea() & getPerimeter()
	public double getArea() {
		double Area= length*width;
		return Area;
	}

	public double getPerimeter() {
		double Perimeter = (2*length)+(2*width);
		return Perimeter;
	}

	// ToString
	public String toString() {
		return "A Rectangle with width="+width+" and length="+
			length+",which is a subclass of"+super.toString();
	}
}
```

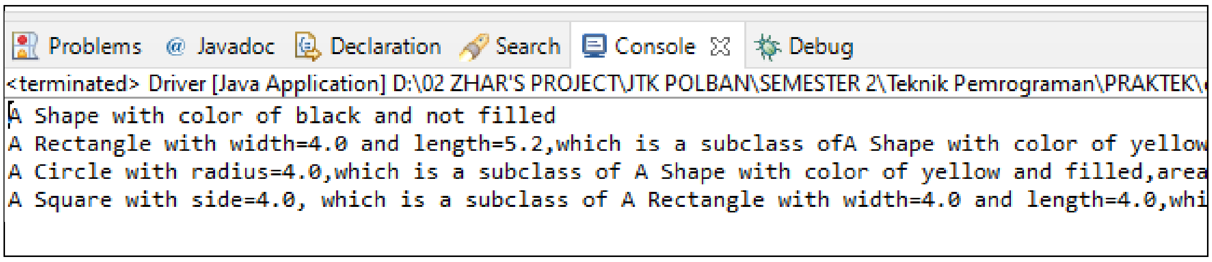
Square.java

```java
package Task2;

public class Square extends Rectangle{
	// Constructor
	public Square() {
		super();
	}

	public Square(double side) {
		super(side, side);
	}

	public Square(double side, String color, boolean filled) {
		super(side, side, color,  filled);
	}

	// Setter | Mutator
	@Override
	public void setWidth(double side) {
		super.setWidth(side);
		setLength(side);
	}

	@Override
	public void setLength(double side) {
		super.setLength(side);
		setWidth(side);
	}

	// Getter | Accessor
	@Override
	public double getArea() {
		double Area;
		Area = super.getLength() * super.getWidth();
		return Area;
	}

	@Override
	public double getPerimeter() {
		double Perimeter;
		Perimeter = (4*super.getLength());
		return Perimeter;
	}

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

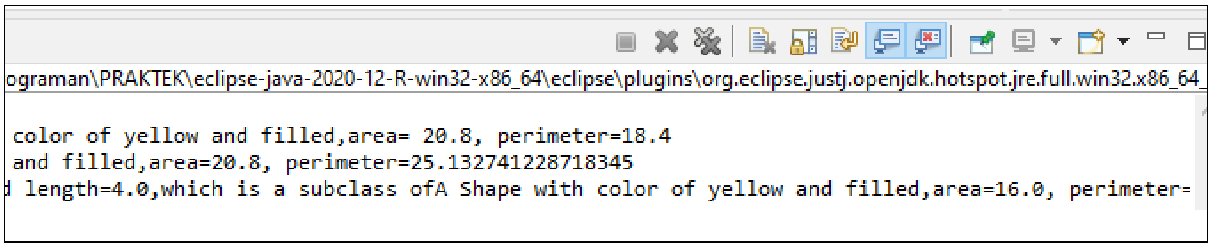
Main.java

```java
package Task2;

public class Main {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Shape s = new Shape("black",false);
		System.out.println(s.toString());

		Rectangle r = new Rectangle(4.0,5.2,"yellow",true);
		System.out.println(r.toString()+",area= "+r.getArea()
				+", perimeter="+r.getPerimeter());

		Circle c = new Circle(4.0, "yellow", true);
		System.out.println(c.toString()+",area="+r.getArea()
				+", perimeter="+c.getPerimeter());

		Square sq = new Square(4.0,"yellow",true);
		System.out.println(sq.toString()+",area="+sq.getArea()
				+", perimeter="+sq.getPerimeter());
	}
}
```

Screenshoot hasil akhir program :



```
<terminated> Driver [Java Application] D:\02 ZHAR'S PROJECT\JTK POLBAN\SEMESTER 2\Teknik Pemrograman\PRAKTEK\
A Shape with color of black and not filled
A Rectangle with width=4.0 and length=5.2,which is a subclass ofA Shape with color of yellow
A Circle with radius=4.0,which is a subclass of A Shape with color of yellow and filled,area
A Square with side=4.0, which is a subclass of A Rectangle with width=4.0 and length=4.0,whi
```


```
ograman\PRAKTEK\eclipse-java-2020-12-R-win32-x86_64\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_

 color of yellow and filled,area= 20.8, perimeter=18.4
 and filled,area=20.8, perimeter=25.132741228718345
d length=4.0,which is a subclass ofA Shape with color of yellow and filled,area=16.0, perimeter=
```

Task 3.1

Extending the Sortable abstract class

Write code above, and analyzed how it work.

[Case 1]
There is an abstract class named Sortable.

```
abstract class Sortable{
        public abstract int compare(Sortable b);
        public static void shell_sort(Sortable[] a){
 //Shell sort body
        }
}
```

When Sortable extended to Employee class, the method compare will be implemented.

```
class Employee extends Sortable{
        /* another methods */
        public int compare(Sortable b){
          Employee eb = (Employee) b;
          if (salary < eb.salary) return -1;
          if (salary > eb.salary) return +1;
          return 0;
        }
}
```

[Try] Please try the codes above. Call the method compare, in EmployeeTest class

```
        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);
        Sortable.shell_sort(staff);
```

[Case 2] Imagine that we want to order the Managers in a similar way :
class Managers extends Employee extends Sortable

It will be work?
What is your solution?

[CASE 1]

Sortable.java

```java
package Task3;

public abstract class Sortable{
	public abstract int compare(Sortable b);
	//source : https://www.geeksforgeeks.org/shellsort/
	public static void shell_sort(Sortable[] a){
		int n = a.length;
		     // Start with a big gap, then reduce the gap
		for (int gap = n/2; gap > 0; gap /= 2)
		{
		     // Do a gapped insertion sort for this gap size.
		    // The first gap elements a[0..gap-1] are already
		    // in gapped order keep adding one more element
		    // until the entire array is gap sorted
		    for (int i = gap; i < n; i += 1)
		    {
		        Sortable temp = a[i];
		        // shift earlier gap-sorted elements up until
		        // the correct location for a[i] is found
		        int j;
		        for (j = i; j >= gap && a[j - gap].compare(temp) < 0; j -
= gap)
		            a[j] = a[j - gap];
		        // put temp (the original a[i]) in its correct
		        // location
		        a[j] = temp;
		    }
		}
	}
}
```

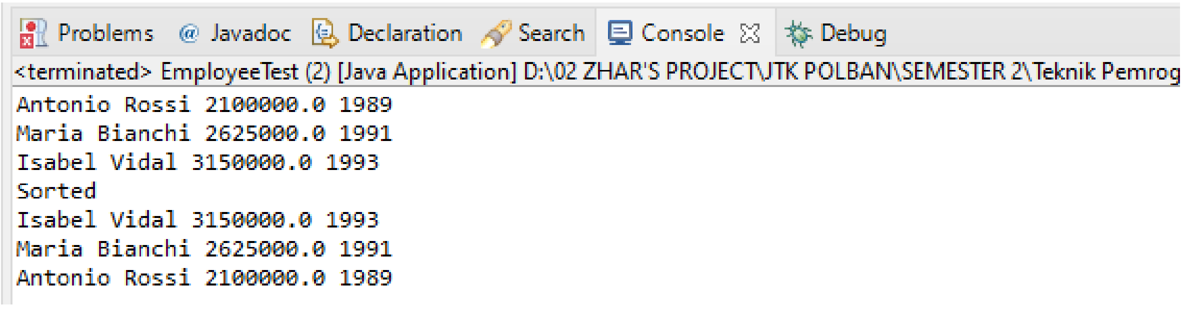
Employee.java

```java
package Task3;

public class Employee extends Sortable{
	private String name;
	private double salary;
	private int hireday;
	private int hiremonth;
	private int hireyear;

	public Employee(String n, double s, int day, int month, int year) {
		this.name = n;
		this.salary = s;
		this.hireday = day;
		this.hiremonth = month;
		this.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;
	}
	
	public int compare(Sortable b) {
		Employee eb = (Employee) b;
		if(salary < eb.salary) return -1;
		if(salary < eb.salary) return 1;
		return 0;
	}
}
```

EmployeeTest.java

```java
package Task3;

public class EmployeeTest {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		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);
		int i;
		for(i = 0; i < 3; i++) staff[i].raiseSalary(5);
		for(i = 0; i < 3; i++) staff[i].print();
		
		Sortable.shell_sort(staff);
		System.out.println("Sorted");
		for(i = 0; i < 3; i++)
			staff[i].print();
	}
}
```

Screenshoot hasil akhir program :



```
<terminated> EmployeeTest (2) [Java Application] D:\02 ZHAR'S PROJECT\JTK POLBAN\SEMESTER 2\Teknik Pemrog
Antonio Rossi 2100000.0 1989
Maria Bianchi 2625000.0 1991
Isabel Vidal 3150000.0 1993
Sorted
Isabel Vidal 3150000.0 1993
Maria Bianchi 2625000.0 1991
Antonio Rossi 2100000.0 1989
```

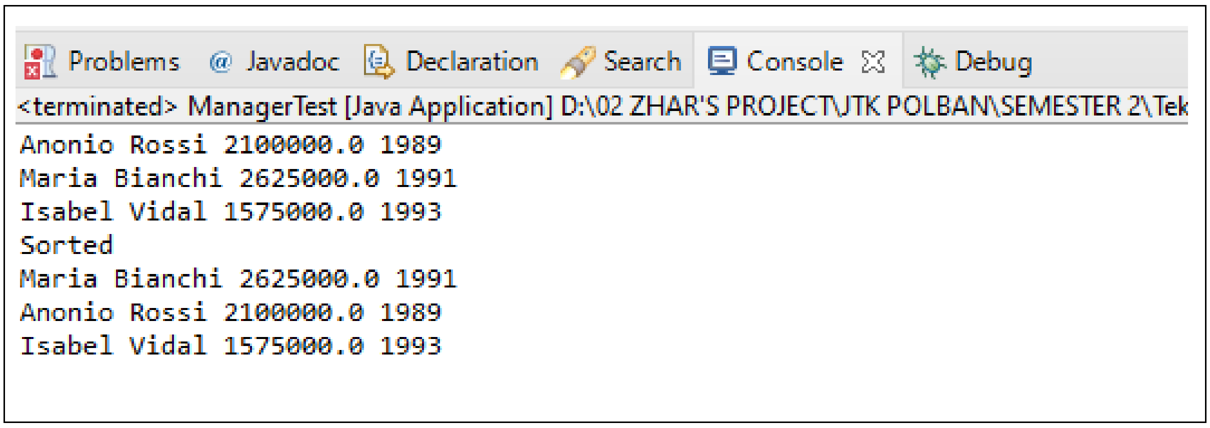
Manager.java

```java
package Task3;

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

public class Manager extends Employee{
	private String secretaryName;

	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;
	}
}
```

ManagerTest.java

```java
package Task3;

public class ManagerTest {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Employee[] staff = new Employee[3];
		staff[0] = new Employee("Anonio Rossi",2000000,1,10,1989);
		staff[1] = new Employee("Maria Bianchi",2500000,1,12,1991);
		staff[2] = new Employee("Isabel Vidal",1500000,1,11,1993);

		int i;
		for(i = 0; i < 3; i++) staff[i].raiseSalary(5);
		for(i = 0; i < 3; i++) staff[i].print();
		Sortable.shell_sort(staff);
		System.out.println("Sorted");
		for(i = 0; i < 3; i++) {
			staff[i].print();
		}
	}
}
```

Screenshoot hasil akhir program :



Problems @ Javadoc Declaration Search Console Debug

```
<terminated> ManagerTest [Java Application] D:\02 ZHAR'S PROJECT\JTK POLBAN\SEMESTER 2\Tek
Anonio Rossi 2100000.0 1989
Maria Bianchi 2625000.0 1991
Isabel Vidal 1575000.0 1993
Sorted
Maria Bianchi 2625000.0 1991
Anonio Rossi 2100000.0 1989
Isabel Vidal 1575000.0 1993
```

[CASE 2]

Sortable.java

```
package Task3_1;

public interface Sortable {
	public abstract int compare(Sortable b);
	public abstract void shell_sort(Sortable[] a);
}
```

Employee.java

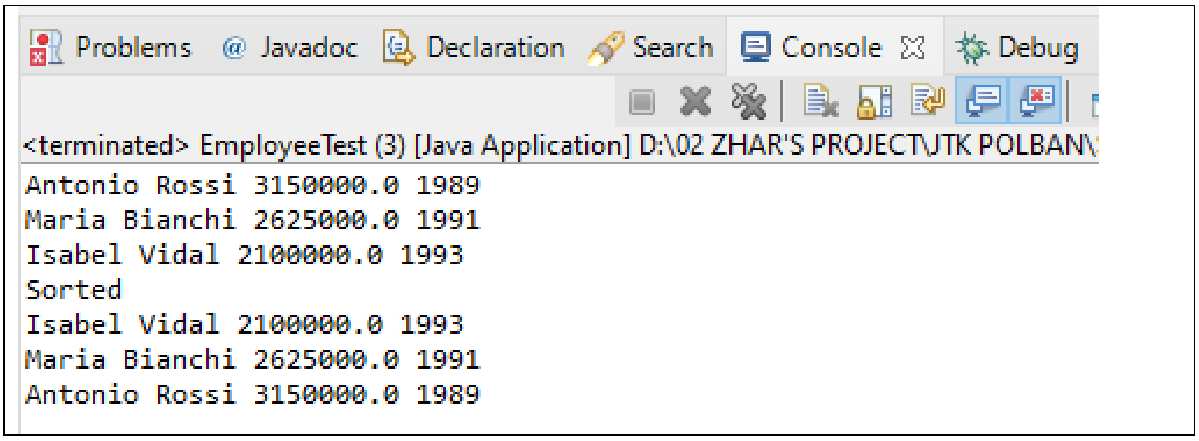
```
package Task3_1;

public class Employee implements Sortable{
	private String name;
	private double salary;
	private int hireday;
	private int hiremonth;
	private int hireyear;

	public Employee(String name, double salary, int hireday, int hiremonth,
int
	hireyear) {
		this.name = name;
		this.salary = salary;
		this.hireday = hireday;
		this.hiremonth = hiremonth;
		this.hireyear = hireyear;
	}

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

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

	public int hireyear(){
		return hireyear;
	}

	//Source : https://www.geeksforgeeks.org/shellsort/
	@Override
	public void shell_sort(Sortable[] a) {
		int n = a.length;
		for (int gap = n / 2; gap > 0; gap /= 2) {
			for (int i = gap; i < n; i += 1) {
				Sortable temp = a[i];
				int j;
				for (j = i; j >= gap && a[j - gap].compare(temp) == 
1; j 
							-= gap)
				a[j] = a[j - gap];
				a[j] = temp;
			}
		}
	}

	@Override
	public int compare(Sortable b) {
		// TODO Auto-generated method stub
		Employee eb = (Employee) b;
		if(salary < eb.salary) return -1;
		if(salary > eb.salary) return +1;
		return 0;
	}
}
```

EmployeeTest.java

```
package Task3_1;

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

```
                        staff[i].print();
        }
}
```

Screenshoot hasil akhir program :



```
<terminated> EmployeeTest (3) [Java Application] D:\02 ZHAR'S PROJECT\JTK POLBAN\
Antonio Rossi 3150000.0 1989
Maria Bianchi 2625000.0 1991
Isabel Vidal 2100000.0 1993
Sorted
Isabel Vidal 2100000.0 1993
Maria Bianchi 2625000.0 1991
Antonio Rossi 3150000.0 1989
```

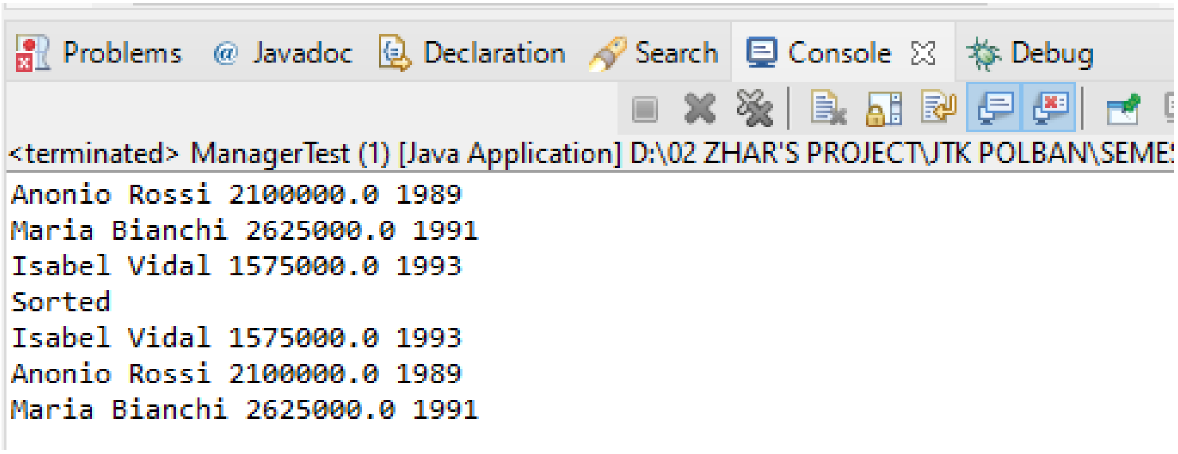
Manager.java

```
package Task3_1;

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

public class Manager extends Employee implements Sortable{
        private String secretaryName;

        public Manager(String name, double salary, int hireday, int hiremonth,
int
        hireyear) {
                super(name, salary, hireday, hiremonth, hireyear);
                secretaryName = "";
        }

        @Override
        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;
        }
}
```

ManagerTest.java

```
package Task3_1;

public class ManagerTest {
	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Employee[] staff = new Employee[3];
		staff[0] = new Employee("Anonio Rossi",2000000,1,10,1989);
		staff[1] = new Employee("Maria Bianchi",2500000,1,12,1991);
		staff[2] = new Employee("Isabel Vidal",1500000,1,11,1993);

		int i;
		for(i = 0; i < 3; i++) staff[i].raiseSalary(5);
		for(i = 0; i < 3; i++) staff[i].print();

		System.out.println("Sorted");
		for(i = 0; i < 3; i++) {
			staff[i].shell_sort(staff);
		}
		for(i = 0; i < 3; i++) {
			staff[i].print();
		}
	}
}
```

Screenshoot hasil akhir program :



```
<terminated> ManagerTest (1) [Java Application] D:\02 ZHAR'S PROJECT\JTK POLBAN\SEME
Anonio Rossi 2100000.0 1989
Maria Bianchi 2625000.0 1991
Isabel Vidal 1575000.0 1993
Sorted
Isabel Vidal 1575000.0 1993
Anonio Rossi 2100000.0 1989
Maria Bianchi 2625000.0 1991
```