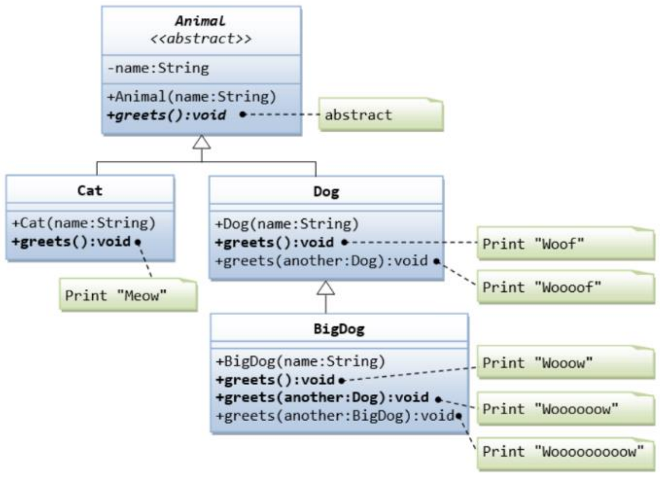
Nama : La Ode Muhammad Gazali
NIM : 222212696
Kelas : 2KS2

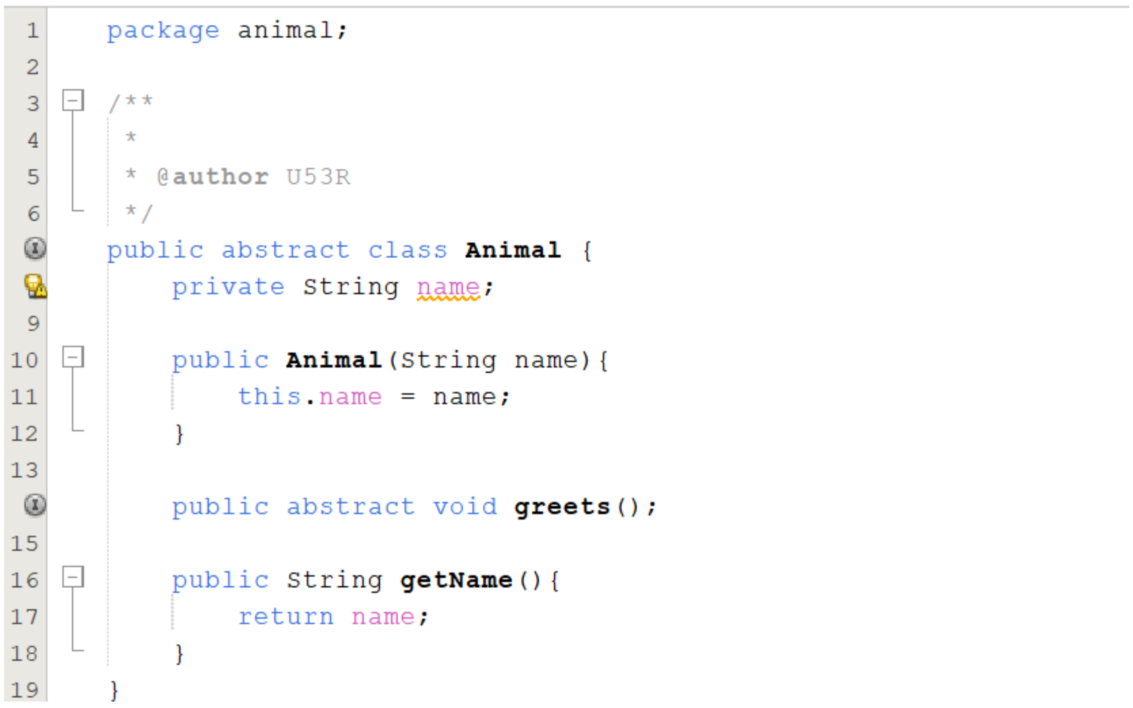
PRA PERTEMUAN 4

PEMROGRAMAN BERBASIS OBJEK

1. Tuliskan kode dari class diagram berikut ini, dan buat test class untuk mengetesnya

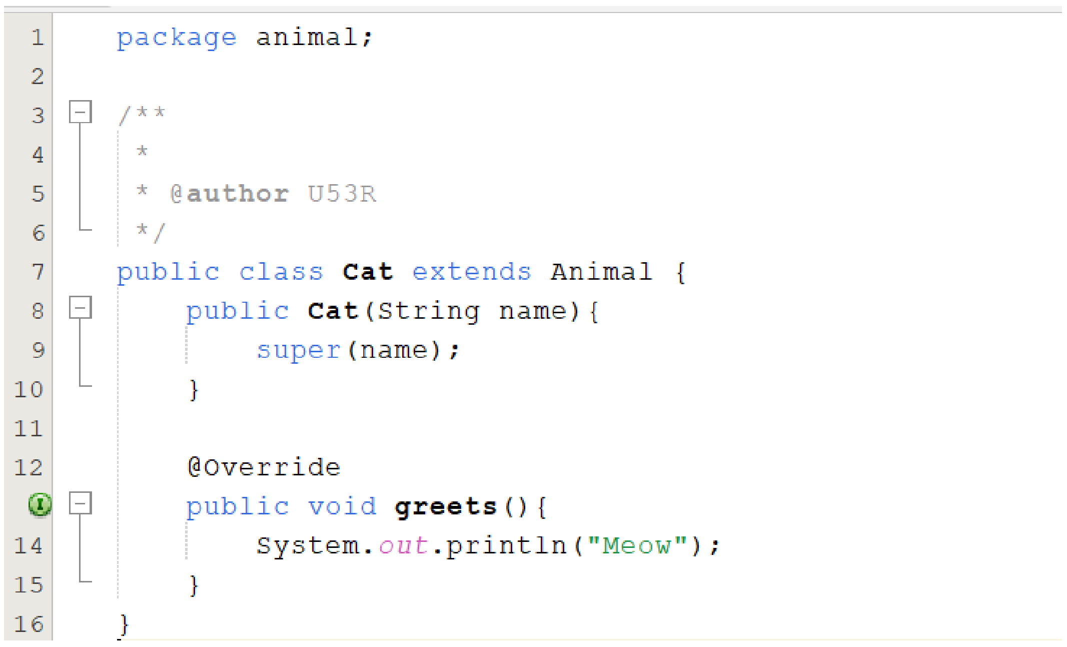


- Program Animal.java



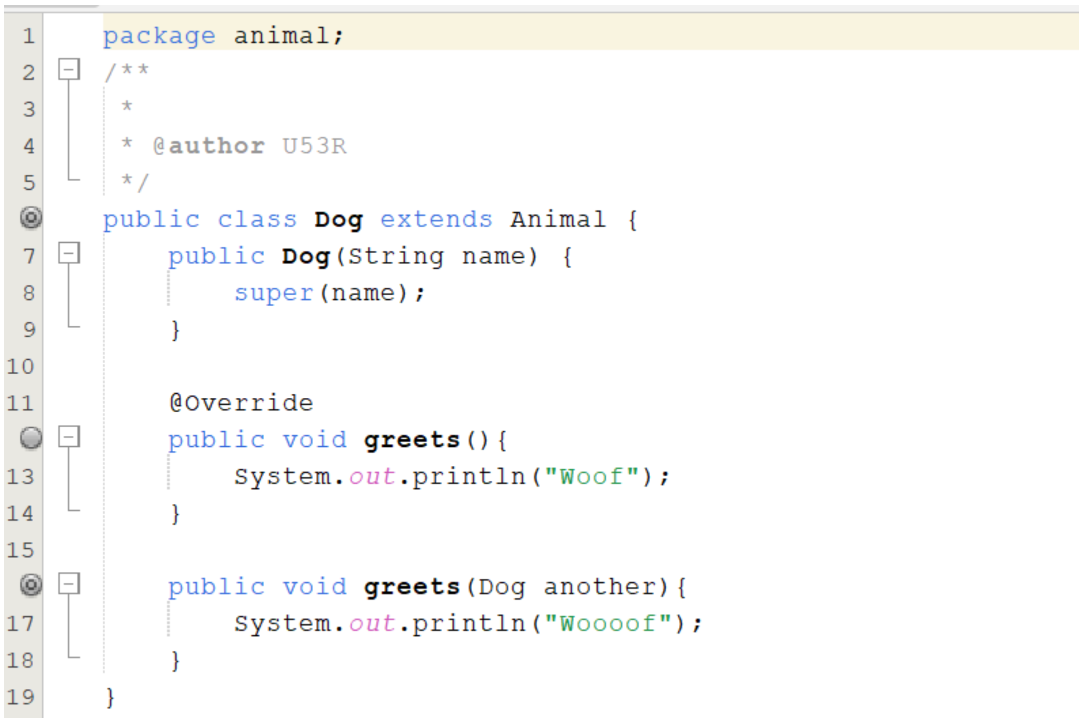
```java
package animal;

/**
 *
 * @author U53R
 */
public abstract class Animal {
    private String name;

    public Animal(String name){
        this.name = name;
    }

    public abstract void greets();

    public String getName(){
        return name;
    }
}
```

- Program Cat.java



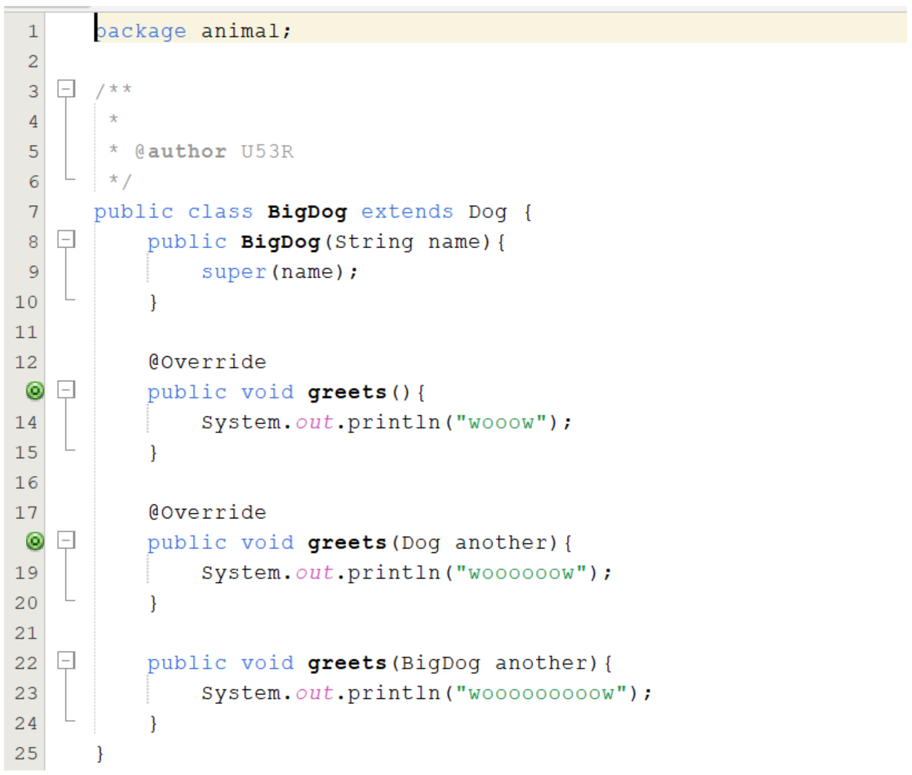
```
package animal;

/**
 *
 * @author U53R
 */
public class Cat extends Animal {
    public Cat(String name){
        super(name);
    }

    @Override
    public void greets(){
        System.out.println("Meow");
    }
}
```

- Program Dog.java



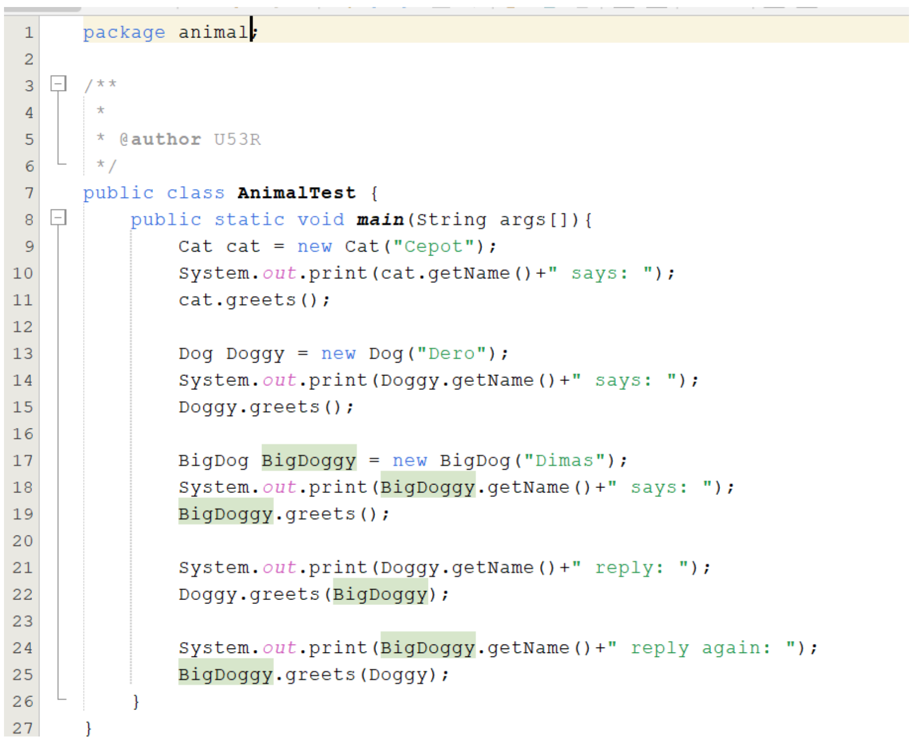
```
package animal;
/**
 *
 * @author U53R
 */
public class Dog extends Animal {
    public Dog(String name) {
        super(name);
    }

    @Override
    public void greets(){
        System.out.println("Woof");
    }

    public void greets(Dog another){
        System.out.println("Woooof");
    }
}
```

- Program BigDog.java



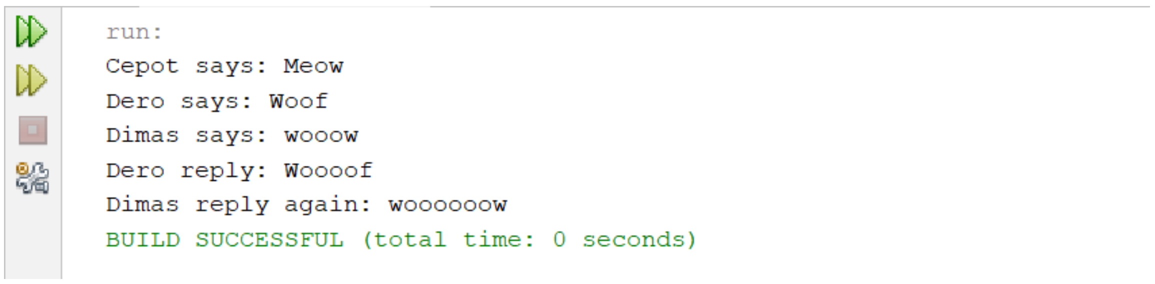
```
package animal;

/**
 *
 * @author U53R
 */
public class BigDog extends Dog {
    public BigDog(String name){
        super(name);
    }

    @Override
    public void greets(){
        System.out.println("wooow");
    }

    @Override
    public void greets(Dog another){
        System.out.println("woooooow");
    }

    public void greets(BigDog another){
        System.out.println("wooooooooow");
    }
}
```

- Program AnimalTest.java



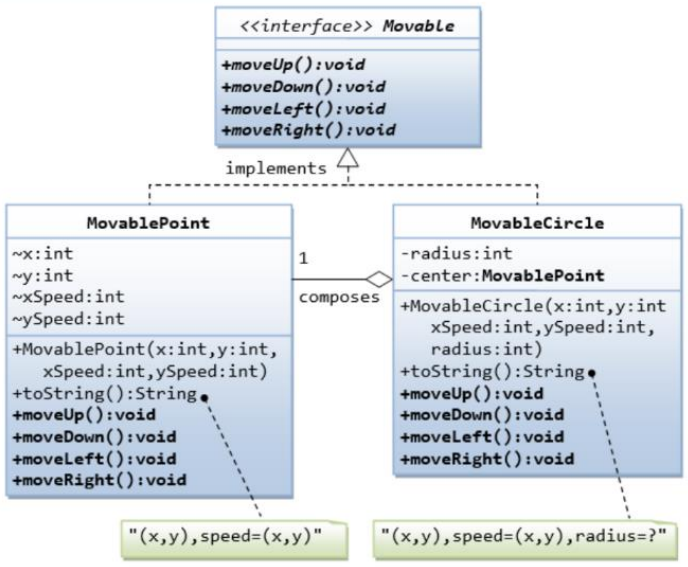
```
package animal;

/**
 *
 * @author U53R
 */
public class AnimalTest {
    public static void main(String args[]){
        Cat cat = new Cat("Cepot");
        System.out.print(cat.getName()+" says: ");
        cat.greets();

        Dog Doggy = new Dog("Dero");
        System.out.print(Doggy.getName()+" says: ");
        Doggy.greets();

        BigDog BigDoggy = new BigDog("Dimas");
        System.out.print(BigDoggy.getName()+" says: ");
        BigDoggy.greets();

        System.out.print(Doggy.getName()+" reply: ");
        Doggy.greets(BigDoggy);

        System.out.print(BigDoggy.getName()+" reply again: ");
        BigDoggy.greets(Doggy);
    }
}
```

- Hasil Running

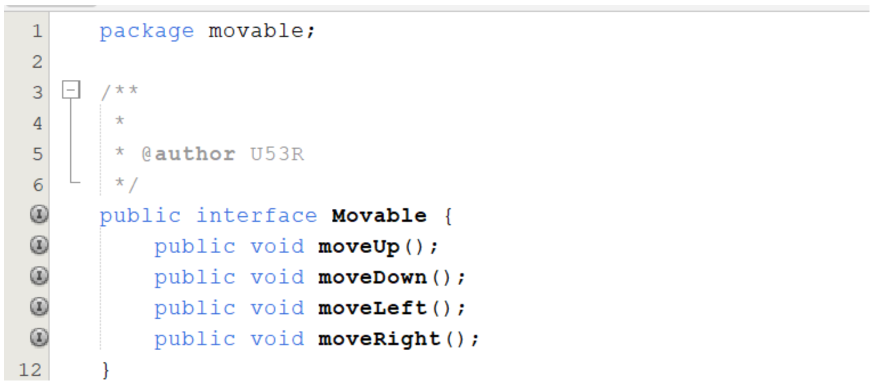


```
run:
Cepot says: Meow
Dero says: Woof
Dimas says: wooow
Dero reply: Woooof
Dimas reply again: wooooooow
BUILD SUCCESSFUL (total time: 0 seconds)
```

2. Tuliskan kode dari class diagram berikut ini, buat test class untuk mengetesnya. Tanda ~ di depan property/variable menandakan package access modifier atau bersifat default yang hanya bisa diakses oleh kelas lain pada package yang sama.

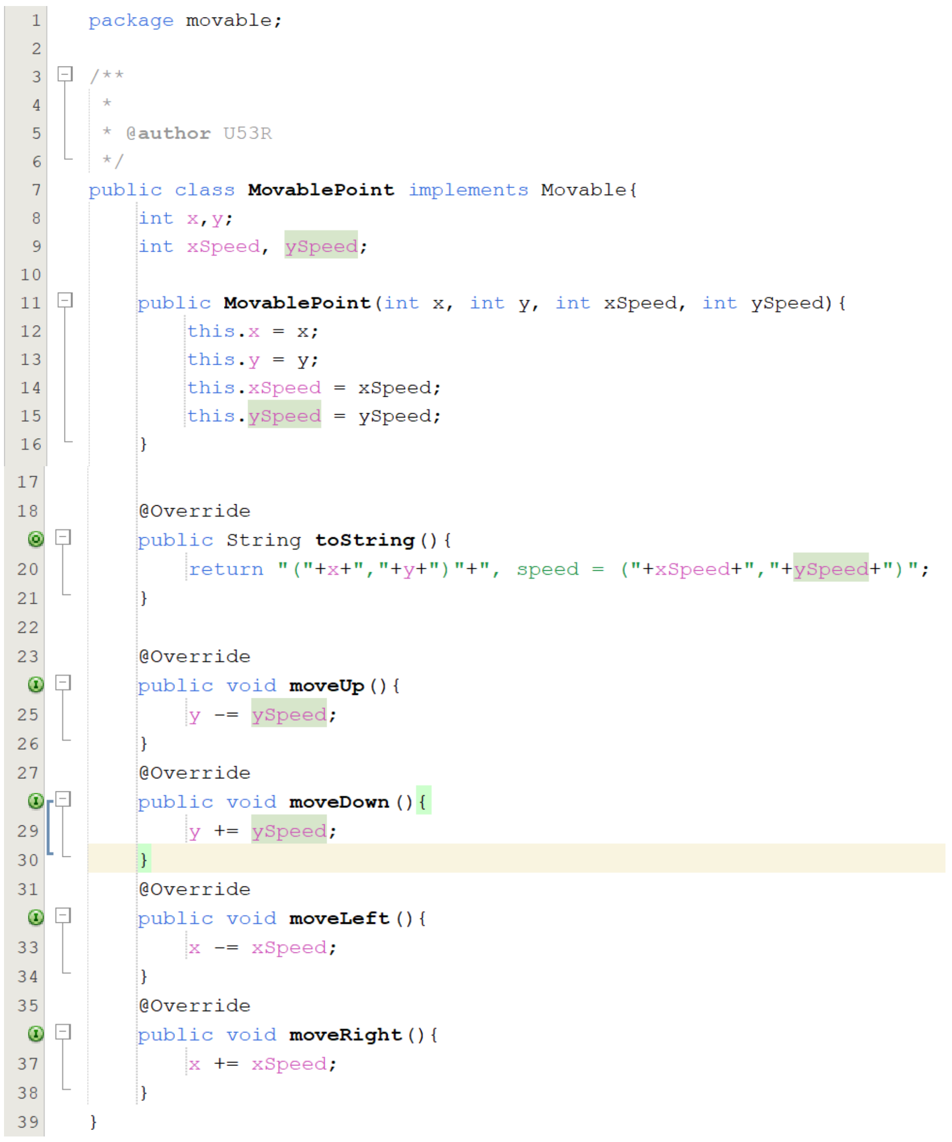


- Program Movable.java



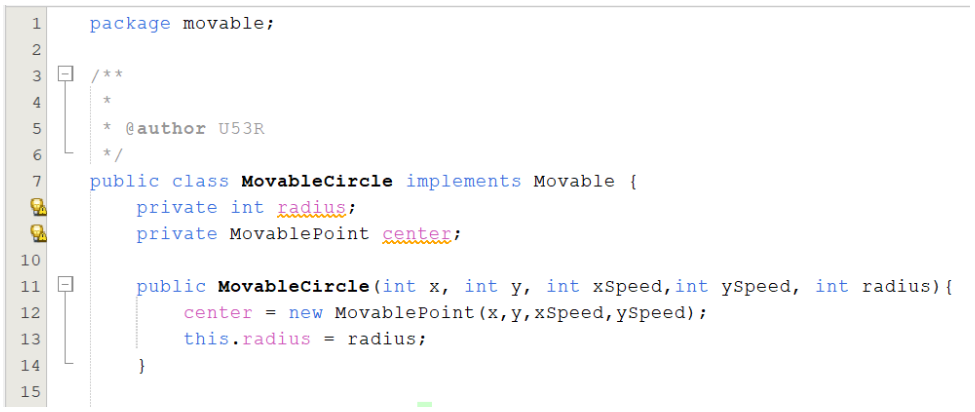
```java
package movable;

/**
 *
 * @author U53R
 */
public interface Movable {
    public void moveUp();
    public void moveDown();
    public void moveLeft();
    public void moveRight();
}
```

- Program MovablePoint.java

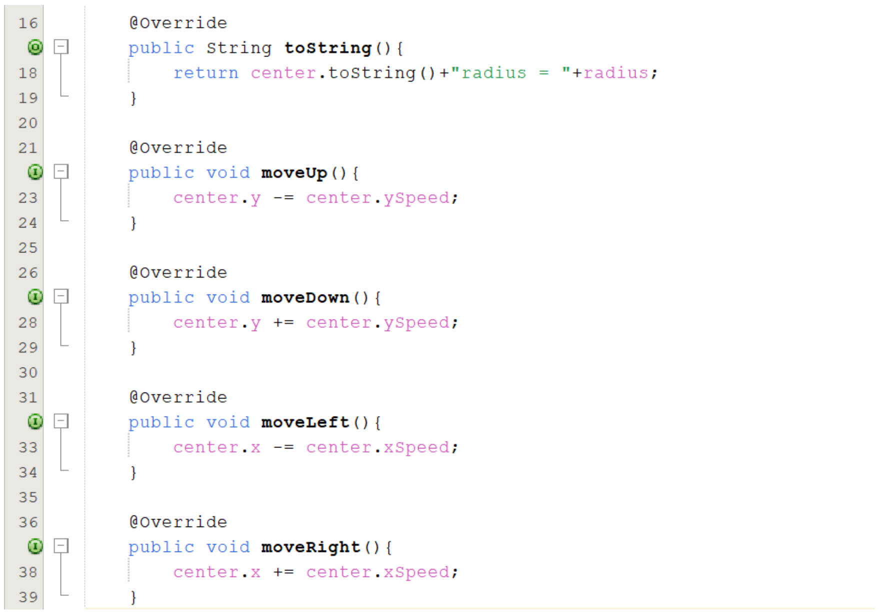


```java
package movable;

/**
 *
 * @author U53R
 */
public class MovablePoint implements Movable{
    int x,y;
    int xSpeed, ySpeed;

    public MovablePoint(int x, int y, int xSpeed, int ySpeed){
        this.x = x;
        this.y = y;
        this.xSpeed = xSpeed;
        this.ySpeed = ySpeed;
    }

    @Override
    public String toString(){
        return "("+x+","+y+")"+", speed = ("+xSpeed+","+ySpeed+")";
    }

    @Override
    public void moveUp(){
        y -= ySpeed;
    }
    @Override
    public void moveDown(){
        y += ySpeed;
    }
    @Override
    public void moveLeft(){
        x -= xSpeed;
    }
    @Override
    public void moveRight(){
        x += xSpeed;
    }
}
```

- Program MovableCircle.java

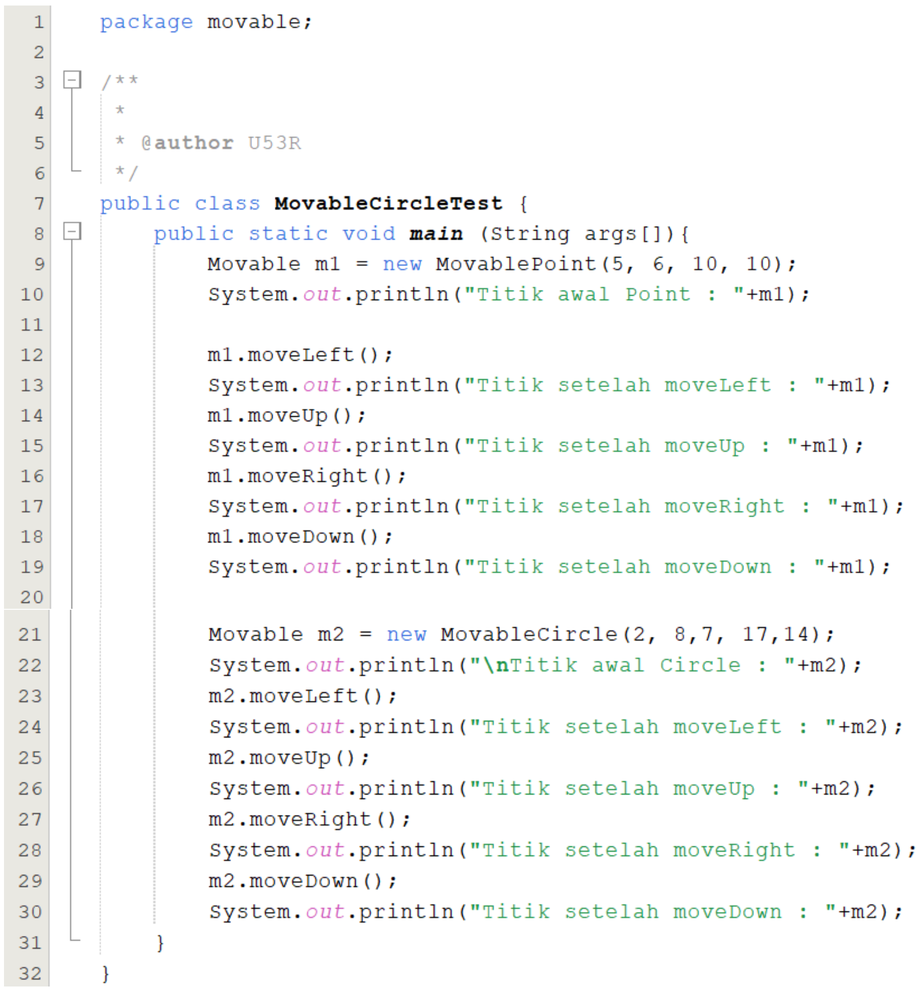


```java
package movable;

/**
 *
 * @author U53R
 */
public class MovableCircle implements Movable {
    private int radius;
    private MovablePoint center;

    public MovableCircle(int x, int y, int xSpeed,int ySpeed, int radius){
        center = new MovablePoint(x,y,xSpeed,ySpeed);
        this.radius = radius;
    }

```



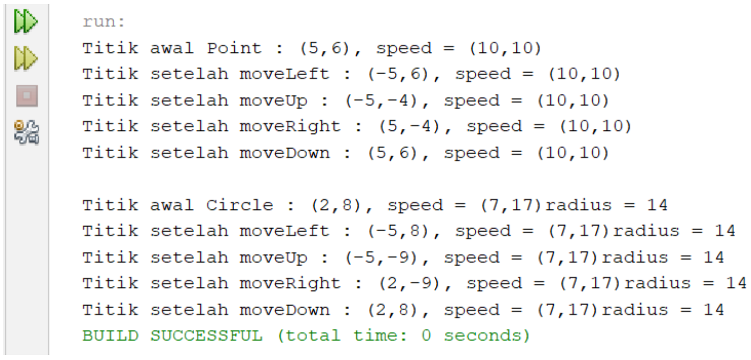
```java
    @Override
    public String toString(){
        return center.toString()+"radius = "+radius;
    }

    @Override
    public void moveUp(){
        center.y -= center.ySpeed;
    }

    @Override
    public void moveDown(){
        center.y += center.ySpeed;
    }

    @Override
    public void moveLeft(){
        center.x -= center.xSpeed;
    }

    @Override
    public void moveRight(){
        center.x += center.xSpeed;
    }
```

- Program MovableCircleTest.java



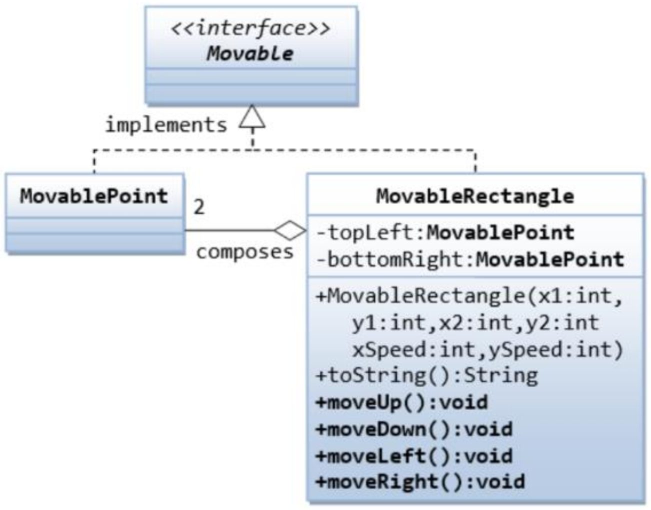
```java
package movable;

/**
 *
 * @author U53R
 */
public class MovableCircleTest {
    public static void main (String args[]){
        Movable m1 = new MovablePoint(5, 6, 10, 10);
        System.out.println("Titik awal Point : "+m1);

        m1.moveLeft();
        System.out.println("Titik setelah moveLeft : "+m1);
        m1.moveUp();
        System.out.println("Titik setelah moveUp : "+m1);
        m1.moveRight();
        System.out.println("Titik setelah moveRight : "+m1);
        m1.moveDown();
        System.out.println("Titik setelah moveDown : "+m1);

        Movable m2 = new MovableCircle(2, 8,7, 17,14);
        System.out.println("\nTitik awal Circle : "+m2);
        m2.moveLeft();
        System.out.println("Titik setelah moveLeft : "+m2);
        m2.moveUp();
        System.out.println("Titik setelah moveUp : "+m2);
        m2.moveRight();
        System.out.println("Titik setelah moveRight : "+m2);
        m2.moveDown();
        System.out.println("Titik setelah moveDown : "+m2);
    }
}
```

- Hasil Running

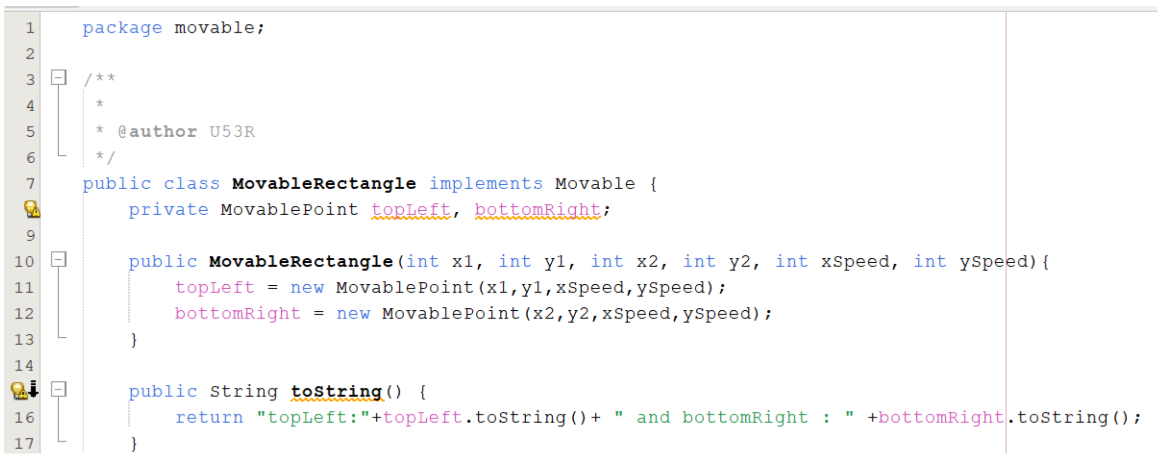


```
run:
Titik awal Point : (5,6), speed = (10,10)
Titik setelah moveLeft : (-5,6), speed = (10,10)
Titik setelah moveUp : (-5,-4), speed = (10,10)
Titik setelah moveRight : (5,-4), speed = (10,10)
Titik setelah moveDown : (5,6), speed = (10,10)

Titik awal Circle : (2,8), speed = (7,17)radius = 14
Titik setelah moveLeft : (-5,8), speed = (7,17)radius = 14
Titik setelah moveUp : (-5,-9), speed = (7,17)radius = 14
Titik setelah moveRight : (2,-9), speed = (7,17)radius = 14
Titik setelah moveDown : (2,8), speed = (7,17)radius = 14
BUILD SUCCESSFUL (total time: 0 seconds)
```

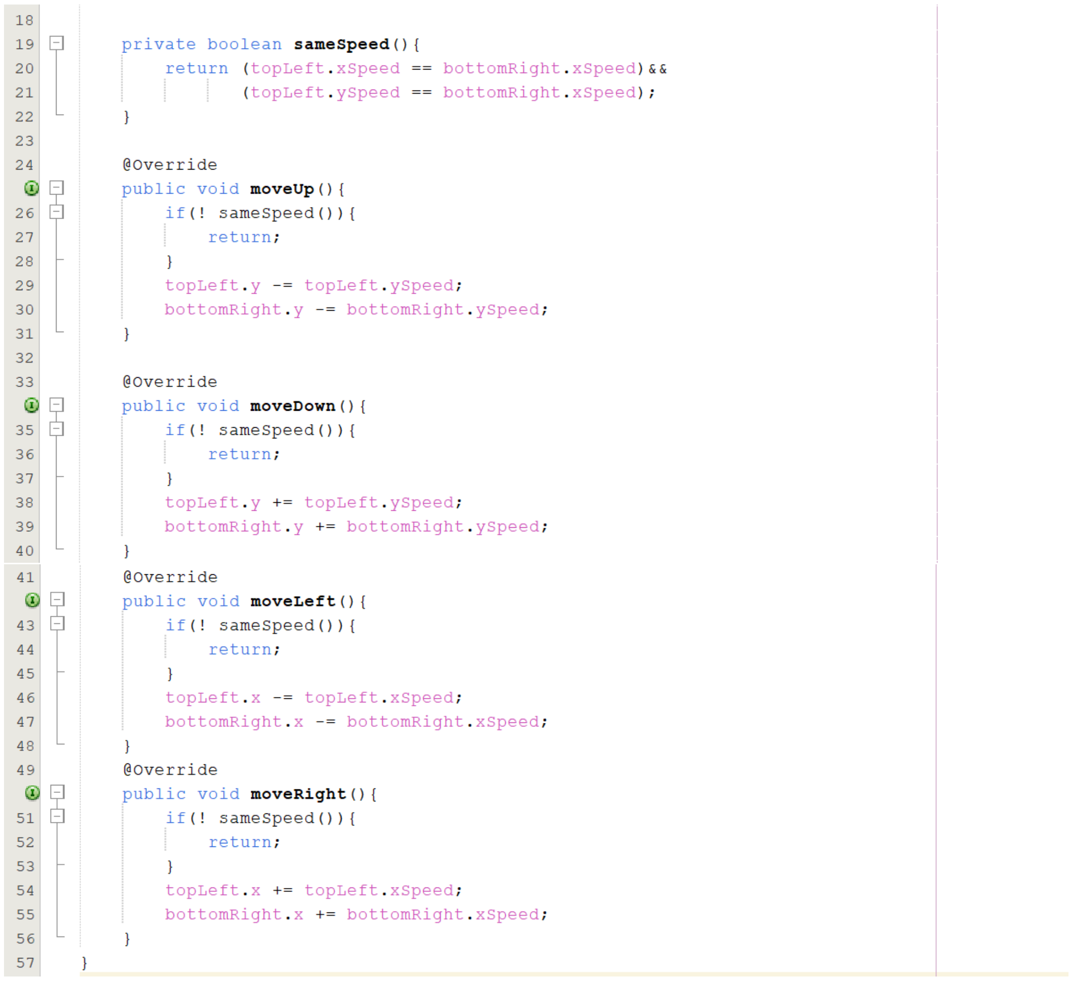
3. Mengembangkan dari class diagram pada soal nomor 2, Buatlah kelas baru bernama MovableRectangle, yang terdiri dari dua MovablePoint (mewakili titik pojok kiri-atas dan kanan-bawah) dan mengimplementasikan Interface Movable. Pastikan kedua point memiliki speed yang sama.



- Program MovableRectangle.java

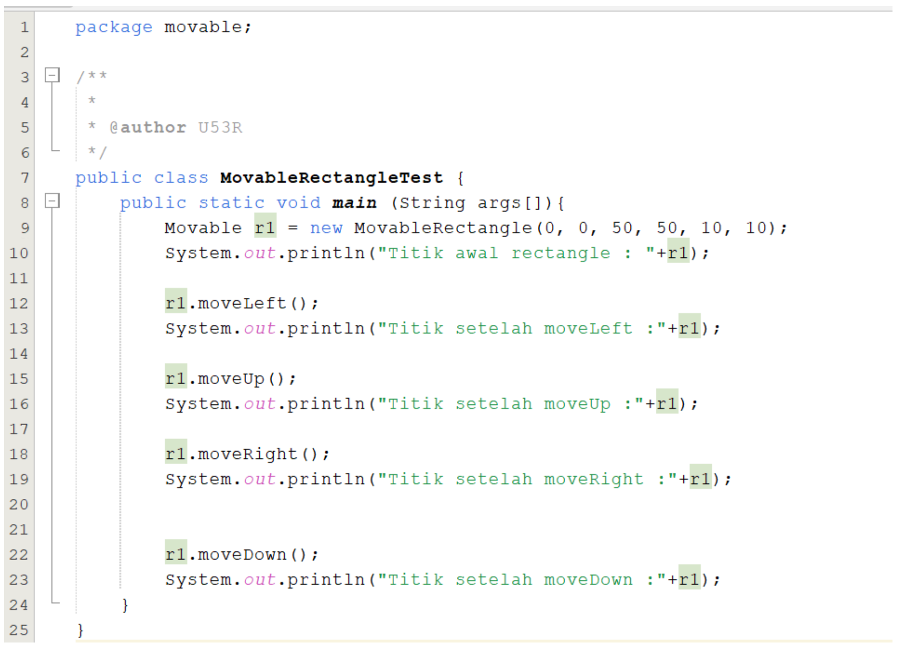


```java
package movable;

/**
 *
 * @author U53R
 */
public class MovableRectangle implements Movable {
    private MovablePoint topLeft, bottomRight;

    public MovableRectangle(int x1, int y1, int x2, int y2, int xSpeed, int ySpeed){
        topLeft = new MovablePoint(x1,y1,xSpeed,ySpeed);
        bottomRight = new MovablePoint(x2,y2,xSpeed,ySpeed);
    }

    public String toString() {
        return "topLeft:"+topLeft.toString()+ " and bottomRight : " +bottomRight.toString();
    }
```



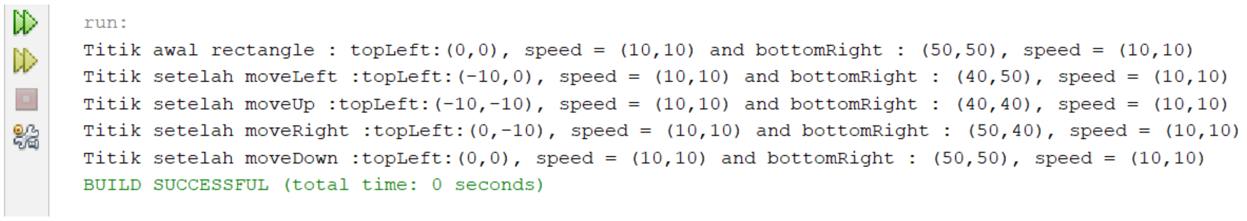
```java

    private boolean sameSpeed(){
        return (topLeft.xSpeed == bottomRight.xSpeed)&&
               (topLeft.ySpeed == bottomRight.xSpeed);
    }

    @Override
    public void moveUp(){
        if(! sameSpeed()){
            return;
        }
        topLeft.y -= topLeft.ySpeed;
        bottomRight.y -= bottomRight.ySpeed;
    }

    @Override
    public void moveDown(){
        if(! sameSpeed()){
            return;
        }
        topLeft.y += topLeft.ySpeed;
        bottomRight.y += bottomRight.ySpeed;
    }
    @Override
    public void moveLeft(){
        if(! sameSpeed()){
            return;
        }
        topLeft.x -= topLeft.xSpeed;
        bottomRight.x -= bottomRight.xSpeed;
    }
    @Override
    public void moveRight(){
        if(! sameSpeed()){
            return;
        }
        topLeft.x += topLeft.xSpeed;
        bottomRight.x += bottomRight.xSpeed;
    }
}
```

- Program MovableRectangleTest.java



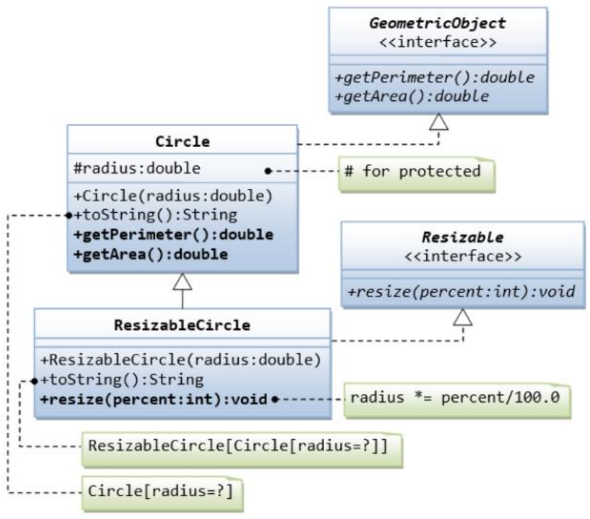
```java
package movable;

/**
 *
 * @author U53R
 */
public class MovableRectangleTest {
    public static void main (String args[]){
        Movable r1 = new MovableRectangle(0, 0, 50, 50, 10, 10);
        System.out.println("Titik awal rectangle : "+r1);

        r1.moveLeft();
        System.out.println("Titik setelah moveLeft :"+r1);

        r1.moveUp();
        System.out.println("Titik setelah moveUp :"+r1);

        r1.moveRight();
        System.out.println("Titik setelah moveRight :"+r1);


        r1.moveDown();
        System.out.println("Titik setelah moveDown :"+r1);
    }
}
```

- Hasil Running

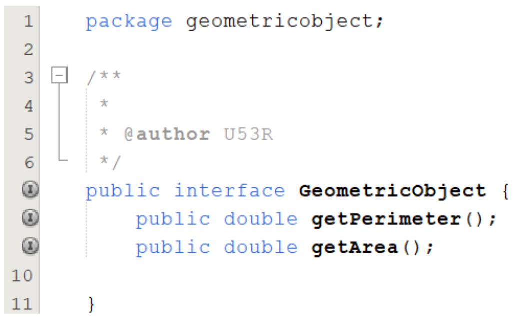


```
run:
Titik awal rectangle : topLeft:(0,0), speed = (10,10) and bottomRight : (50,50), speed = (10,10)
Titik setelah moveLeft :topLeft:(-10,0), speed = (10,10) and bottomRight : (40,50), speed = (10,10)
Titik setelah moveUp :topLeft:(-10,-10), speed = (10,10) and bottomRight : (40,40), speed = (10,10)
Titik setelah moveRight :topLeft:(0,-10), speed = (10,10) and bottomRight : (50,40), speed = (10,10)
Titik setelah moveDown :topLeft:(0,0), speed = (10,10) and bottomRight : (50,50), speed = (10,10)
BUILD SUCCESSFUL (total time: 0 seconds)
```

4. Tuliskan kode dari class diagram berikut ini, buat test class untuk mengetesnya.

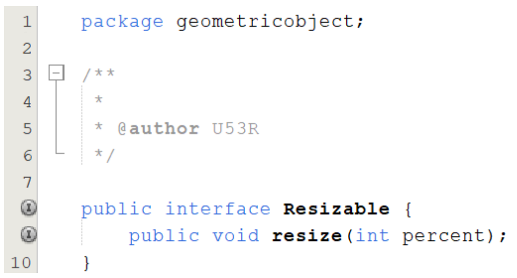


- Program GeometricObject.java



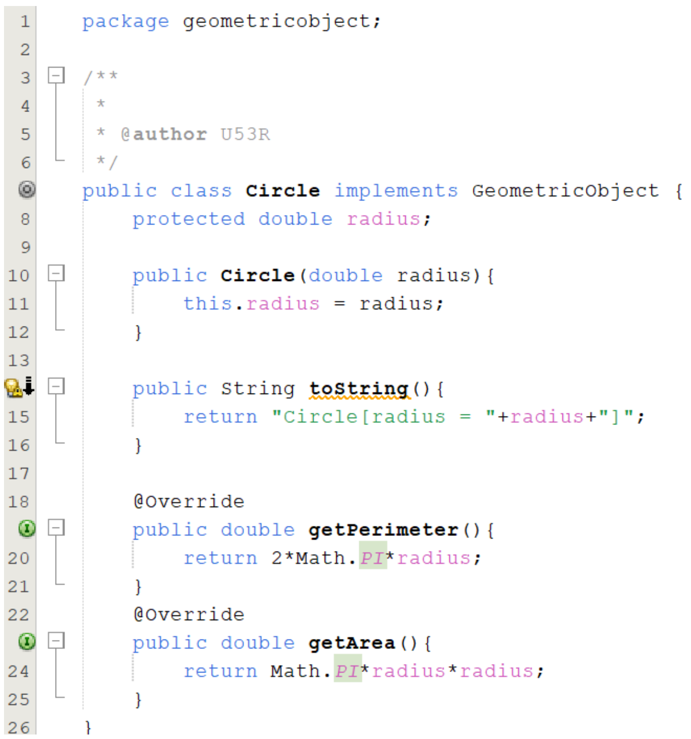
```java
package geometricobject;

/**
 *
 * @author U53R
 */
public interface GeometricObject {
    public double getPerimeter();
    public double getArea();

}
```

- Program Resizable.java



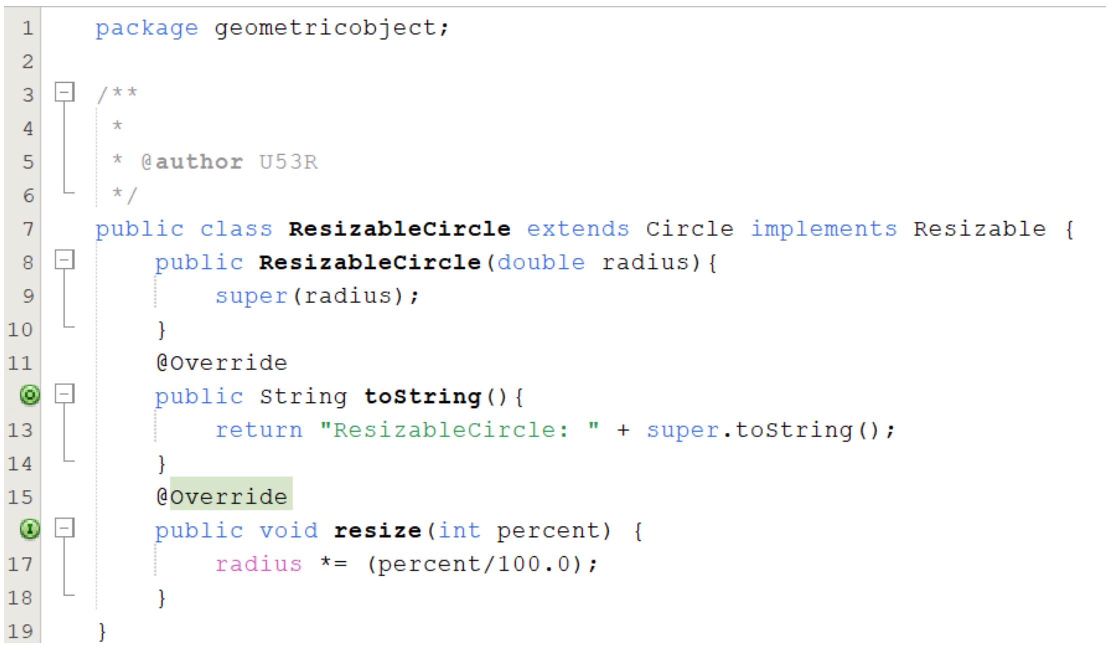
```java
package geometricobject;

/**
 *
 * @author U53R
 */

public interface Resizable {
    public void resize(int percent);
}
```

- Program Circle.java



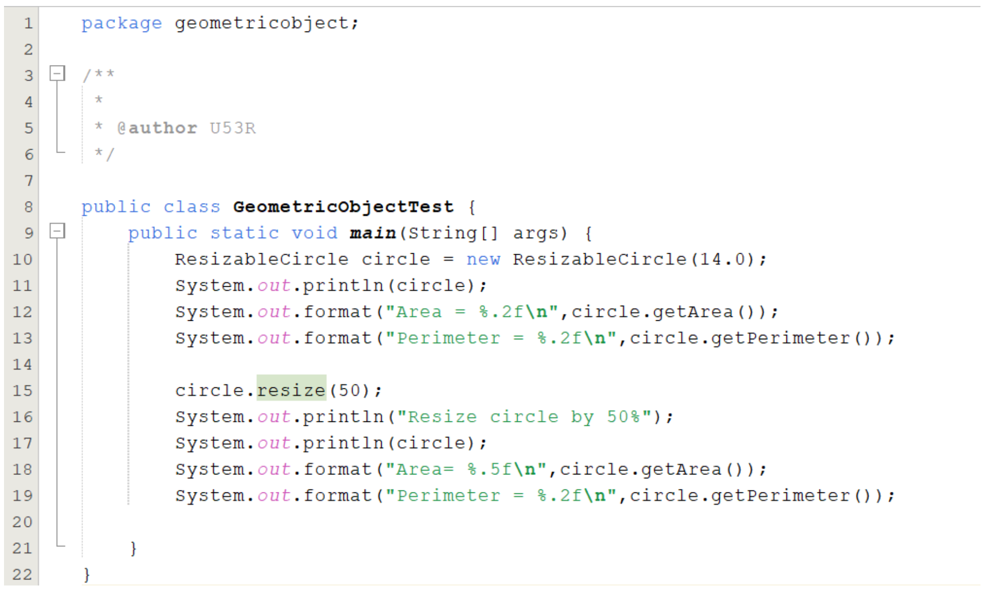
```java
package geometricobject;

/**
 *
 * @author U53R
 */
public class Circle implements GeometricObject {
    protected double radius;

    public Circle(double radius){
        this.radius = radius;
    }

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

    @Override
    public double getPerimeter(){
        return 2*Math.PI*radius;
    }
    @Override
    public double getArea(){
        return Math.PI*radius*radius;
    }
}
```

- Program ResizableCircle.java



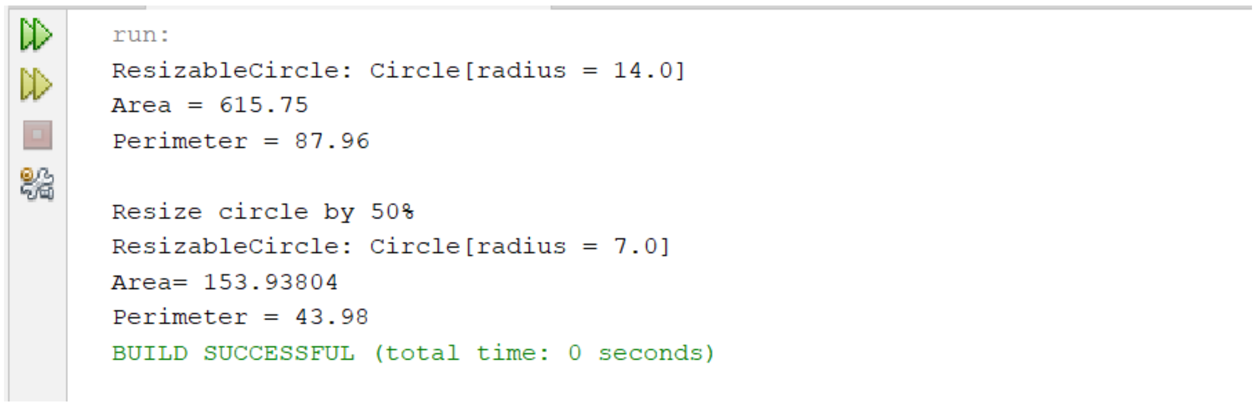
```java
package geometricobject;

/**
 *
 * @author U53R
 */
public class ResizableCircle extends Circle implements Resizable {
    public ResizableCircle(double radius){
        super(radius);
    }
    @Override
    public String toString(){
        return "ResizableCircle: " + super.toString();
    }
    @Override
    public void resize(int percent)  {
        radius  *= (percent/100.0);
    }
}
```

- Program GeometricObjectMain.java



```java
package geometricobject;

/**
 *
 * @author U53R
 */

public class GeometricObjectTest {
    public static void main(String[] args) {
        ResizableCircle circle = new ResizableCircle(14.0);
        System.out.println(circle);
        System.out.format("Area = %.2f\n",circle.getArea());
        System.out.format("Perimeter = %.2f\n",circle.getPerimeter());

        circle.resize(50);
        System.out.println("Resize circle by 50%");
        System.out.println(circle);
        System.out.format("Area= %.5f\n",circle.getArea());
        System.out.format("Perimeter = %.2f\n",circle.getPerimeter());

    }
}
```

- Hasil Running



```
run:
ResizableCircle: Circle[radius = 14.0]
Area = 615.75
Perimeter = 87.96

Resize circle by 50%
ResizableCircle: Circle[radius = 7.0]
Area= 153.93804
Perimeter = 43.98
BUILD SUCCESSFUL (total time: 0 seconds)
```