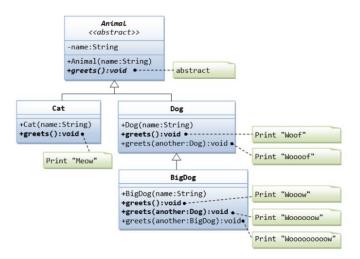
Nama : La Ode Muhammad Gazali

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

```
package animal;
 2
 3
 4
 5
         @author U53R
 6
 1
       public abstract class Animal {
 <u>Q.</u>
           private String name;
 9
10
   _
           public Animal(String name) {
               this.name = name;
11
12
13
 1
           public abstract void greets();
15
   Ţ
           public String getName() {
16
17
               return name;
18
19
```

• Program Cat.java

```
package animal;
 2
   _ / * *
 3
 4
 5
       * @author U53R
 6
 7
      public class Cat extends Animal {
   public Cat(String name) {
 8
              super(name);
 9
10
11
          @Override
12
public void greets() {
              System.out.println("Meow");
14
15
16
```

• Program Dog.java

```
package animal;
2 - /**
3
 4
      * @author U53R
5
0
     public class Dog extends Animal {
  _
7
          public Dog(String name) {
              super(name);
8
9
10
          @Override
11
public void greets() {
              System.out.println("Woof");
13
14
15
public void greets(Dog another) {
              System.out.println("Woooof");
17
18
19
      }
```

Program BigDog.java

```
package animal;
2
3 - /**
 4
 5
       * @author U53R
 6
 7
      public class BigDog extends Dog {
   8
          public BigDog(String name) {
 9
              super(name);
10
11
12
          @Override
0
          public void greets() {
14
              System.out.println("wooow");
15
16
17
          @override
public void greets(Dog another) {
19
              System.out.println("wooooow");
20
21
22
  public void greets(BigDog another) {
23
              System.out.println("wooooooow");
24
25
```

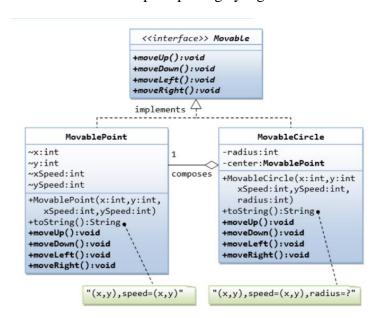
Program AnimalTest.java

```
1
2
3 🖵 /**
 4
       * @author U53R
 5
 7
     public class AnimalTest {
   8
          public static void main(String args[]) {
              Cat cat = new Cat("Cepot");
9
10
              System.out.print(cat.getName()+" says: ");
11
              cat.greets();
12
13
              Dog Doggy = new Dog("Dero");
              System.out.print(Doggy.getName()+" says: ");
14
15
              Doggy.greets();
16
17
              BigDog BigDoggy = new BigDog("Dimas");
              System.out.print(BigDoggy.getName()+" says: ");
18
19
              BigDoggy.greets();
20
21
              System.out.print(Doggy.getName()+" reply: ");
22
              Doggy.greets(BigDoggy);
23
              System.out.print(BigDoggy.getName()+" reply again: ");
24
25
              BigDoggy.greets(Doggy);
26
```

• Hasil Running

```
run:
Cepot says: Meow
Dero says: Woof
Dimas says: wooow
Dero reply: Woooof
Dimas reply again: woooooow
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

```
package movable;
 2
   - / * *
 3
 4
 5
       * @author U53R
 6
 1
      public interface Movable {
 1
          public void moveUp();
 (1)
          public void moveDown();
 1
          public void moveLeft();
 1
          public void moveRight();
12
```

• Program MovablePoint.java

```
package movable;
2
3 🖵 /**
 4
 5
      * @author U53R
 6
 7
     public class MovablePoint implements Movable{
 8
         int xSpeed, ySpeed;
10
11 📮
         public MovablePoint(int x, int y, int xSpeed, int ySpeed) {
12
            this.x = x;
             this.y = y;
13
             this.xSpeed = xSpeed;
14
             this.ySpeed = ySpeed;
15
16
17
18
         @Override

    □
         public String toString() {
           return "("+x+","+y+")"+", speed = ("+xSpeed+","+ySpeed+")";
20
21
22
23
         @Override
(1)
         public void moveUp(){
          y -= ySpeed;
25
26
27
         @Override
⊙r⊡
         public void moveDown(){
29
          y += ySpeed;
30
31
         @Override
public void moveLeft(){
33
          x -= xSpeed;
34
35
         @Override
② □
         public void moveRight(){
37
           x += xSpeed;
38
39
```

• Program MovableCircle.java

```
package movable;
2
3 - /**
4
 5
      * @author U53R
 7
     public class MovableCircle implements Movable {
 8
       private int radius;
         private MovablePoint center;
10
11 📮
         public MovableCircle(int x, int y, int xSpeed,int ySpeed, int radius){
           center = new MovablePoint(x, y, xSpeed, ySpeed);
12
13
             this.radius = radius;
14
15
```

```
16
           @Override

    □

          public String toString() {
18
              return center.toString()+"radius = "+radius;
19
20
21
          @Override
 ⓐ □
          public void moveUp(){
23
            center.y -= center.ySpeed;
24
25
26
           @override
 ⓐ □
          public void moveDown(){
28
              center.y += center.ySpeed;
29
30
31
          @override
 ■ □
          public void moveLeft() {
33
              center.x -= center.xSpeed;
34
35
36
          @Override
 ③ □
          public void moveRight() {
38
             center.x += center.xSpeed;
39
```

Program MovableCircleTest.java

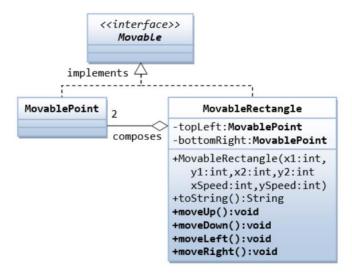
```
package movable;
 2
 3 - /**
 4
       * @author U53R
 6
 7
      public class MovableCircleTest {
 8
          public static void main (String args[]) {
 9
              Movable m1 = new MovablePoint(5, 6, 10, 10);
              System.out.println("Titik awal Point : "+m1);
10
11
12
              m1.moveLeft();
              System.out.println("Titik setelah moveLeft : "+m1);
13
14
              m1.moveUp();
15
              System.out.println("Titik setelah moveUp : "+m1);
              m1.moveRight();
16
              System.out.println("Titik setelah moveRight : "+m1);
17
              m1.moveDown();
18
              System.out.println("Titik setelah moveDown : "+m1);
19
20
21
              Movable m2 = new MovableCircle(2, 8,7, 17,14);
22
              System.out.println("\nTitik awal Circle : "+m2);
              m2.moveLeft();
23
              System.out.println("Titik setelah moveLeft: "+m2);
24
              m2.moveUp();
2.5
              System.out.println("Titik setelah moveUp : "+m2);
26
27
              m2.moveRight();
               System.out.println("Titik setelah moveRight : "+m2);
28
29
               m2.moveDown();
               System.out.println("Titik setelah moveDown : "+m2);
30
31
32
```

• 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
Titik setelah moveDown: (2,8), speed = (7,17) radius = 14
```

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

```
package movable;
 2
 3 🖵 /**
 4
 5
       * @author U53R
 6
 7
      public class MovableRectangle implements Movable {
 <u>Q.</u>
          private MovablePoint topLeft, bottomRight;
10 🖃
          public MovableRectangle(int x1, int y1, int x2, int y2, int xSpeed, int ySpeed) {
11
            topLeft = new MovablePoint(x1, y1, xSpeed, ySpeed);
              bottomRight = new MovablePoint(x2, y2, xSpeed, ySpeed);
12
13
14
₩. =
          public String toString() {
              return "topLeft:"+topLeft.toString()+ " and bottomRight: " +bottomRight.toString();
16
```

```
18
19
          private boolean sameSpeed() {
               return (topLeft.xSpeed == bottomRight.xSpeed)&&
(topLeft.ySpeed == bottomRight.xSpeed);
20
21
22
23
24
          @override
(1)
   public void moveUp(){
   阜
              if(! sameSpeed()){
27
              return;
28
              topLeft.y -= topLeft.ySpeed;
29
              bottomRight.y -= bottomRight.ySpeed;
30
31
32
33
          @override
0
   public void moveDown() {
   ₽
35
              if(! sameSpeed()){
36
                return;
37
              }
38
              topLeft.y += topLeft.ySpeed;
              bottomRight.y += bottomRight.ySpeed;
39
40
41
          @Override
⊕ □
          public void moveLeft() {
   if(! sameSpeed()){
43
44
                  return;
45
              topLeft.x -= topLeft.xSpeed;
46
47
              bottomRight.x -= bottomRight.xSpeed;
48
49
③ □
          public void moveRight() {
   51
              if(! sameSpeed()){
52
              return;
53
54
              topLeft.x += topLeft.xSpeed;
55
              bottomRight.x += bottomRight.xSpeed;
56
57
```

• Program MovableRectangleTest.java

```
package movable;
 3 - /**
 4
 5
       * @author U53R
 6
 7
      public class MovableRectangleTest {
 8 -
          public static void main (String args[]) {
              Movable r1 = new MovableRectangle(0, 0, 50, 50, 10, 10);
 9
10
              System.out.println("Titik awal rectangle : "+r1);
11
              r1.moveLeft();
12
              System.out.println("Titik setelah moveLeft :"+r1);
13
14
15
              r1.moveUp();
              System.out.println("Titik setelah moveUp :"+r1);
16
17
18
              r1.moveRight();
              System.out.println("Titik setelah moveRight :"+r1);
19
20
21
22
              r1.moveDown();
              System.out.println("Titik setelah moveDown :"+r1);
23
24
```

Hasil Running

```
run:

Titik awal rectangle: topLeft:(0,0), speed = (10,10) and bottomRight: (50,50), speed = (10,10)

Titik setelah moveLeft:(-10,0), speed = (10,10) and bottomRight: (40,50), speed = (10,10)

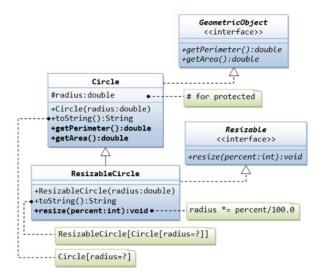
Titik setelah moveUp:topLeft:(-10,-10), speed = (10,10) and bottomRight: (40,40), speed = (10,10)

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

```
package geometricobject;
1
2
3
   - / * *
4
       * @author U53R
5
 6
(3)
      public interface GeometricObject {
(1)
       public double getPerimeter();
1
          public double getArea();
10
11
```

Program Resizable.java

```
package geometricobject;

/**

/**

* * @author U53R

/*/

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

Program Circle.java

```
package geometricobject;
2
3 - /**
 4
     * @author U53R
 5
 6
 0
    public class Circle implements GeometricObject {
 8
        protected double radius;
 9
10
         public Circle(double radius) {
11
         this.radius = radius;
12
13
<u>⊶</u> □
         public String toString() {
         return "Circle[radius = "+radius+"]";
15
16
17
18
         @override
(1)
         public double getPerimeter() {
         return 2*Math.PI*radius;
20
21
22
         @override
① 🖃
         public double getArea() {
24
         return Math. PI*radius*radius;
25
26
```

• Program ResizableCircle.java

```
package geometricobject;
2
3 - /**
4
5
      * @author U53R
6
7
     public class ResizableCircle extends Circle implements Resizable {
  8
          public ResizableCircle(double radius) {
9
         super(radius);
10
11
         @override

    □

         public String toString() {
            return "ResizableCircle: " + super.toString();
13
14
15
          @override
•
          public void resize(int percent) {
            radius *= (percent/100.0);
17
18
19
```

• Program GeometricObjectMain.java

```
package geometricobject;
 2
 3 🗐 /**
     * @author U53R
*/
 5
 6
 7
 8
      public class GeometricObjectTest {
9 =
         public static void main(String[] args) {
             ResizableCircle circle = new ResizableCircle(14.0);
10
11
             System.out.println(circle);
12
             System.out.format("Area = %.2f\n",circle.getArea());
             System.out.format("Perimeter = %.2f\n",circle.getPerimeter());
13
14
             circle.resize(50);
15
             System.out.println("Resize circle by 50%");
16
17
             System.out.println(circle);
18
             System.out.format("Area= %.5f\n", circle.getArea());
19
             System.out.format("Perimeter = %.2f\n", circle.getPerimeter());
20
21
22
```

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)
```