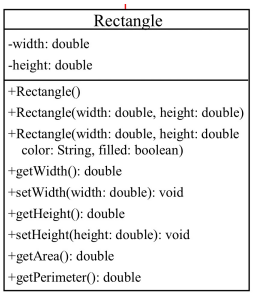
```
	public void setRadius(double radius) {
		this.radius = radius;
	}

	/**
	 * Returns the area of the current circle
	 * Implemention of abstract method in GeometricObject
	 * @return area of the circle
	 */
	public double findArea() {
		return radius * radius * Math.PI;
	}

	/**
	 * Returns the perimeter of the current circle
	 * Implemention of abstract method in GeometricObject
	 * @return perimeter of the circle
	 */
	public double findPerimeter() {
		return 2 * radius * Math.PI;
	}

	/**
	 * Provide a string representation of the object
	 */
	public String toString() {
		return "Circle: radius = " + radius;
	}
}
```

1.1 **ให้อธิบายลำดับการเรียกใช้** constructor **เมื่อสร้างวัตถุจากคลาส** Circle

1. เรียก GeometricObject
2. เรียก Circle(double radius, String color, boolean filled)
3. เรียก Circle(double radius)
4. เรียก Circle();

1.2 **ให้สร้างคลาส** Rectangle **ตาม** UML Class diagram **ด้านบน**




```
Public class Rectangle extends GeometricObject{
    Private double width;
    Private double height;

    Public Rectangle() {...}
    Public Rectangle(double width, double height) {...}
    Public Rectangle(double width, double height, String color, boolean filled) {...}

    Public double getWidth() {...}
    Public void setWidth(double width) {...}
    Public double getHeight() {...}
    Public Void setWidth(double height) {...}

    Public double getArea() {...}
    Public double getPerimeter() {...}
}
```

2. จงหาข้อผิดพลาดของส่วนของโปรแกรมต่อไปนี้

```
public class Circle{
    private double radius;

    public Circle(double radius){
        radius=radius;
    }
    public double getRadius(){
        return radius;
    }
    public double findArea(){
        return radius*radius*Math.PI;
    }
}
class Cylinder extends Circle{
    private double length;

    Cylinder(double radius, double length){
        Circle(radius);
        length=length;
    }
}
```

ผิดตรง radius = radius (ชื่อซ้ำ)

length = length ก

3. จงหาผลลัพธ์การรันโปรแกรมต่อไปนี้

3.1

```
class A{
   public A(){
      System.out.println("The no-arg constructor of A is
invoked");
   }
}
class B extends A{
   public B(){

   }
}
public class C{
   public static void main(String[] args){
     B b =new B();
   }
}
```

ผลลัพธ์การรันโปรแกรม

A ไม่มี constructre

3.2

Animal.java:

```
01     public class Animal {
02         public Animal() {
03             System.out.println("A new
animal has been created!");
04         }
05
06         public void sleep() {
07             System.out.println("An animal
sleeps...");
08         }
09
10         public void eat() {
11             System.out.println("An animal
eats...");
12         }
13     }
```

Bird.java:

```
01     public class Bird extends Animal {
02         public Bird() {
03             super();
04             System.out.println("A new
bird has been created!");
05         }
06
07         @Override
08         public void sleep() {
09             System.out.println("A bird
sleeps...");
10         }
11
12         @Override
13         public void eat() {
14             System.out.println("A bird
eats...");
15         }
16     }
```

Dog.java:

```
01     public class Dog extends Animal {
02         public Dog() {
03             super();
04             System.out.println("A new dog
has been created!");
05         }
06
07         @Override
08         public void sleep() {
09             System.out.println("A dog
sleeps...");
10         }
11
12         @Override
13         public void eat() {
14             System.out.println("A dog
eats...");
15         }
16     }
```

MainClass.java:

```
01     public class MainClass {
02         public static void main(String[]
args) {
03             Animal animal = new Animal();
04             Bird bird = new Bird();
05             Dog dog = new Dog();
06
07             System.out.println();
08
09             animal.sleep();
10             animal.eat();
11
12             bird.sleep();
13             bird.eat();
14
15             dog.sleep();
16             dog.eat();
17         }
18     }
```

ผลลัพธ์การรันโปรแกรม

A new animal has been created!
A new animal has been created!
A new bird has been created!
A new animal has been created!
A new dog has been created!

An animal sleeps...
An animal eats...
A bird sleeps...
A bird eats...
A dog sleeps...
A dog eats...

4. จากโปรแกรมด้านล่างต่อไปนี้ ข้อใดคือ Overriding ที่สามารถนำมาเติมลงในบรรทัดที่ 9 ของโปรแกรมได้โดยไม่ทำให้เกิดข้อผิดพลาดขึ้นกับการทำงานของโปรแกรม อธิบายเหตุผลประกอบ

```
1  class SuperClass{
2     private int num=1;
3     protected int getNumber(){
4        return num;
5     }
6  }
7  class Subclass extends SuperClass{
8     private int num=10;
9     //overiding method
10    public static void main(String[] args){
11      Subclass s= new Subclass();
12      System.out.println(s.getNumber());
13    }
14 }
```

โปรแกรม	สามารถนำมาเติมลงในบรรทัดที่ 9 ของโปรแกรมได้โดยไม่ทำให้เกิดข้อผิดพลาด**หรือไม่**	อธิบายเหตุผลประกอบ
`protected int getNumbers(){ return num+5; }`	ผิดพลาด	ชื่อไม่ตรง
`protected long getNumber(){ return num+5; }`	ผิดพลาด	return type ผิด
`protected int getNumber(){ return num+5; }`	ไม่ ผิด พลาด	
`public int getNumber(){ return num+5; }`	ผิดพลาด	access modifier ผิด
`protected int getNumber(int num){ return num+5; }`	ผิดพลาด	มีการรับ Parameter
`int getNumber(){ return num+5; }`	ผิดพลาด	ไม่มี access modifier
`private int getNumber(){ return num+5; }`	ผิดพลาด	access modifier ผิด