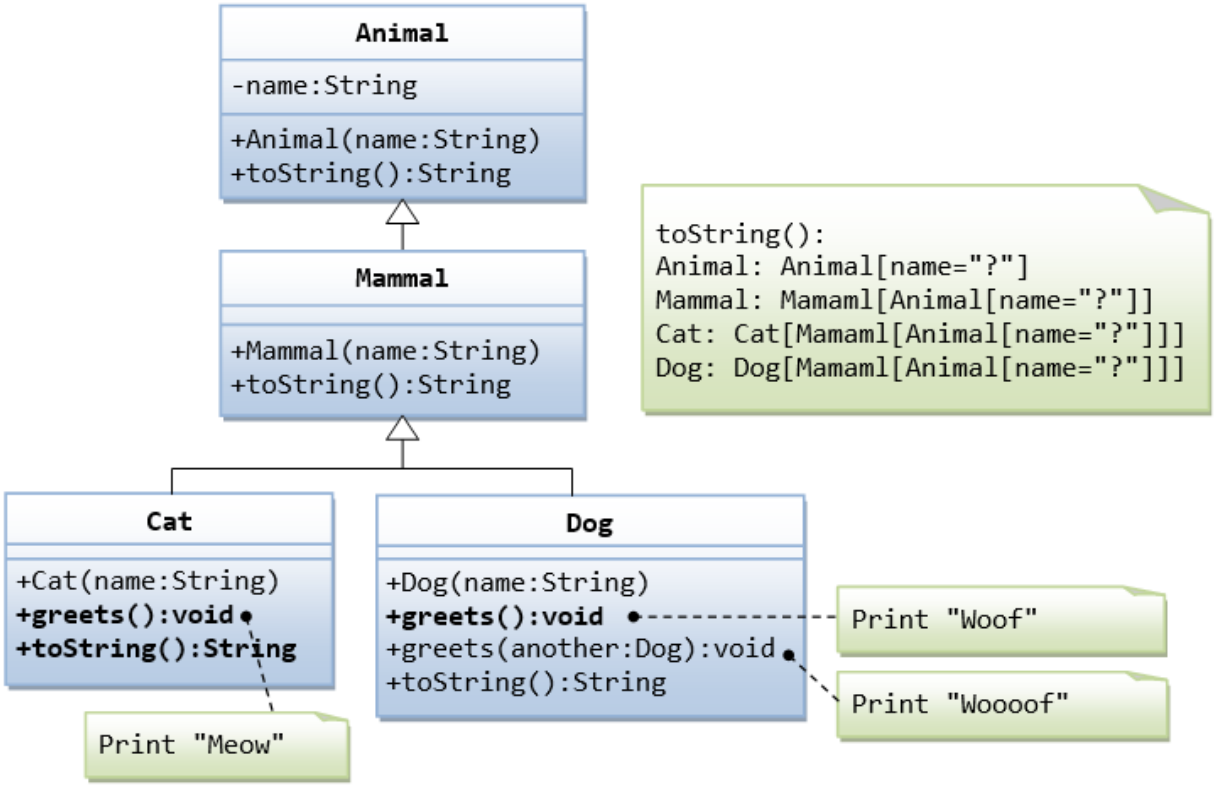
Practice Activities #2

A problem

Superclass `Animal` and its subclasses

Write the classes as shown in the following class diagram. Mark all the overridden methods with annotation `@Override`.



B problem

Superclass Shape and its subclasses Circle, Rectangle and Square

Write the classes as shown in the following class diagram. Mark all the overridden methods with annotation `@Override`.

Simple main:

```
public class Main {
    public static void main(String[] args) {
        Circle circle = new Circle(5.5, "red", false);
        System.out.println(circle);
        System.out.println(circle.getArea());
        System.out.println(circle.getPerimeter());
```

```
        System.out.println(circle.getColor());
        System.out.println(circle.isFilled());
        System.out.println(circle.getRadius());

        Rectangle rectangle = new Rectangle(3.8, 2.5, "green", false);
        System.out.println(rectangle);
        System.out.println(rectangle.getArea());
        System.out.println(rectangle.getPerimeter());
        System.out.println(rectangle.getColor());
        System.out.println(rectangle.getLength());


        Square square = new Square(6.6);
        System.out.println(square);
        System.out.println(square.getArea());
        System.out.println(square.getColor());
        System.out.println(square.getSide());
    }
}
```

Output:

```
Circle[Shape[color='red', filled=false],radius=5.5]
94.985
34.54
red
false
5.5
Rectangle[Shape[color='green', filled=false],width=3.8, length=2.5]
9.5
12.6
green
2.5
Square[Rectangle[Shape[color='black', filled=true],width=6.6, length=6.6]]
43.559999999999995
black
6.6
```

More detailed explanation:

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). Hint:

```
public Square(double side) {
   super(side, side);  // Call superclass Rectangle(double, double)
}
```

- 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.

C problem

Superclass `Person` and its subclasses

Write the classes as shown in the following class diagram. Mark all the overridden methods with annotation `@Override`.