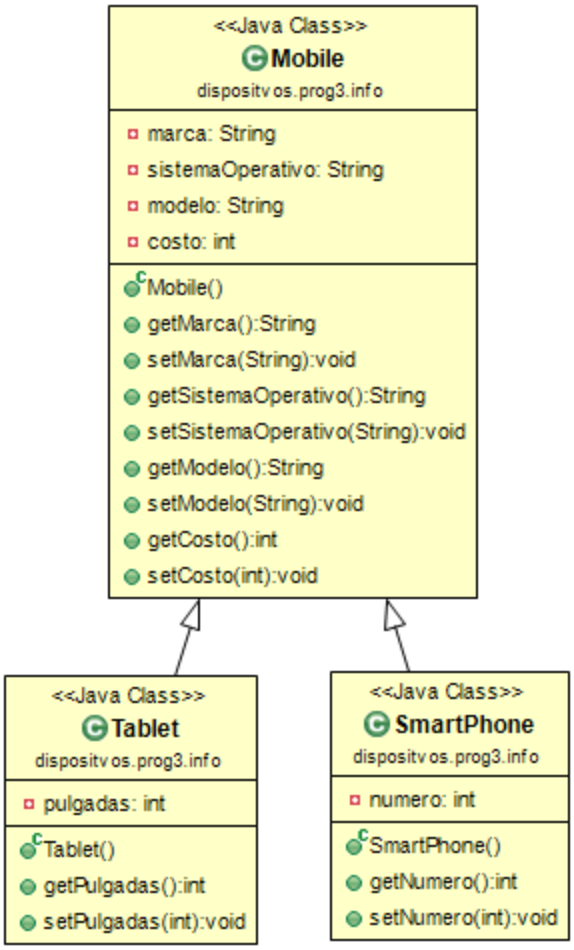
Programación III

TEMA 2: Conceptos Básicos - Herencia

Práctica nº 2 - B

1. Cree un proyecto llamado **DispositivosMoviles** y defina la siguiente jerarquía de clases en JAVA.



   a. Sobreescriba en las clases **Tablet** y **SmartPhone** el método **public boolean equals(Object)** de la clase Object de manera que las implementaciones sirvan para comparar dos instancias de estos tipos.
   b. Sobreescriba también en ambas clases el método public **String toString()** de la clase Object de manera que imprima los datos de esos objetos de manera **legible**.
   c. Escriba una clase dispositivos.prog3.info.EjercicioTestSobreescritura y pruebe los métodos sobreescritos (por ejemplo defina dos objetos de tipo **SmartPhone**, configúrele el mismo número y pruebe el método equals, imprima ambos objetos usando el toString()).

2. **Jerarquía de animales**

   a. Cree un proyecto llamado **Animales**.
   b. Escriba el siguiente código en Eclipse (cada clase debería ubicarse en su propio archivo y dentro del paquete animales.prog3.info) y luego responda las preguntas.

```
public abstract class Animal {
   public abstract void saludo();
}
```

```
public class Gato extends Animal {
   @Override
   public void saludo() {
      System.out.println("Miau!");
   }
}
```

```
public class Perro extends Animal {
   @Override
   public void saludo() {
      System.out.println("Guau!");
   }

   public void saludo(Perro otro) {
      System.out.println("Guau! Guau!");
   }
}
```

```
public class PerroGrande extends Perro {
   @Override
   public void saludo() {
      System.out.println("Guauuuuuu!");
   }

   @Override
   public void saludo(Perro otro) {
      System.out.println("Guauuuuuu!
Guauuuuuu!");
   }
}
```

```
public class TestAnimal1 {
   public static void main(String[] args) {
      Gato gato1 = new Gato();
      gato1.saludo();
      Perro perro1 = new Perro();
      perro1.saludo();
      PerroGrande perroGrande1 = new PerroGrande();
      perroGrande1.saludo();
       }

}
```

   c. **Indique** qué obtuvo como salida luego de la ejecución de TestAnimal1
   d. Escriba el siguiente código:

```
public class TestAnimal2 {
   public static void main(String[] args) {
```

```
            Animal animal1 = new Gato();
            animal1.saludo();
            Animal animal2 = new Perro();
            animal2.saludo();
            Animal animal3 = new PerroGrande();
            animal3.saludo();
         }
      }
```

e. Analice en el código la diferencia con TestAnimal1 e indique qué obtuvo como salida luego de la ejecución de TestAnimal2.
f. Agregue en la clase Gato el siguiente método:

```
         public void sonarCascabel(){
               System.out.println("clin!");
         }
```

g. ¿Es posible enviar el mensaje "**sonarCascabel**" a la instancia **animal1**? **JUSTIFIQUE** e indique ¿qué diferencia hay en declarar animal1, animal2 y animal3 de tipo Animal y no del tipo Gato, Perro o PerroGrande?
h. Escriba el siguiente código en eclipse:

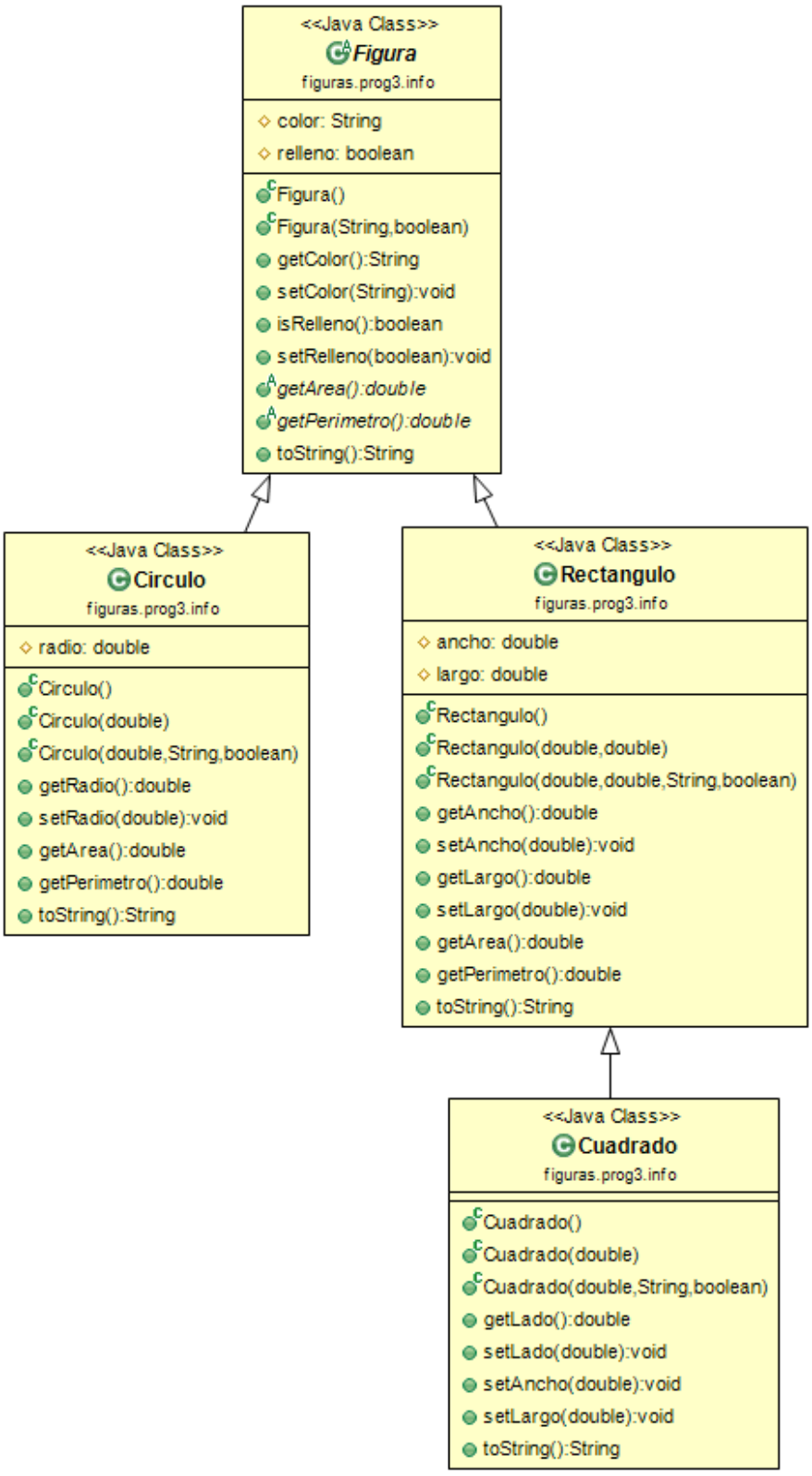
```
public class TestAnimal3 {
   public static void main(String[] args) {

      Gato gato1 = new Gato();
      gato1.saludo();
      Perro perro1 = new Perro();
      perro1.saludo();
      PerroGrande perroGrande1 = new PerroGrande();
      perroGrande1.saludo();

      Animal animal1 = new Gato();
      animal1.saludo();
      Animal animal2 = new Perro();
      animal2.saludo();
      Animal animal3 = new PerroGrande();
      animal3.saludo();

      Perro perro2 = animal2;
      PerroGrande perroGrande2 = animal3;
      Perro perro3 = animal3;
      Gato gato2 = animal2;
      perro2.saludo(perro3);
      perro3.saludo(perro2);
```

```
    perro2.saludo(perroGrande2);
    perroGrande2.saludo(perro2);
    perroGrande2.saludo(perroGrande1);
 }
}
```

i. Corrija los errores en compilación y **JUSTIFIQUE.** ¿cómo se llama el mecanismo aplicado?
j. **Responda:** ¿es posible crear una instancia de la clase Animal?

3. Jerarquía de formas

a. Implemente en Java la siguiente jerarquía de clases



Nota: En este ejercicio, **Figura** está definida como una clase abstracta, la cual contiene:

- Dos variables de instancia privadas : color(String) y relleno(boolean).
- Getter y setter para todas las variables de instancia y el método toString().
- Dos métodos abstractos getArea() y getPerimetro().

Las subclases **Circulo** y **Rectangulo** deben sobreescribir los métodos abstractos getArea() y getPerimetro() y proveer implementación propia. También sobreescriben el método toString().

b. Escriba una clase llamada TestDeFigurasGeometricas.
c. Defina en el método "main" de la clase TestDeFigurasGeometricas un arreglo de 3 posiciones, donde almacenará objetos de tipo "**Figura**".
d. Agregue al arreglo 1 Círculo, 1 Rectangulo y 1 Cuadrado.
e. Itere sobre el arreglo con una estructura de control de tipo "foreach" de modo que cada Figura imprima su información. **Responda:** ¿Qué método invocará?