# CLASES ABSTRACTAS

## Ejercicio U5_B4_E1:

*Solución:* el método esAlquilable() es abstracto por lo que la subclase libro está obligada a sobreescribirlo. Por el contrario el método getPrecio() no es abstracto con lo que las subclases pueden sobreescribirlo o no.

## Ejercicio U5_B4_E2:

```
abstract class Figura{
   protected String color;
   Figura(String color){
      this.color=color;
   }
   abstract double area();
}

class Triangulo extends Figura{
   private double base;
   private double altura;

   public Triangulo(double base, double altura, String color) {
      super(color);
      this.base = base;
      this.altura = altura;
   }

   @Override
   double area() {
      return base*altura/2;
   }

}

class Circulo extends Figura{
   private double radio;

   public Circulo(double radio, String color) {
      super(color);
      this.radio = radio;
   }



   @Override
   double area() {
      return Math.PI*radio*radio;
   }

}
class App{
   public static void main (String[] args) {
      //Figura f= new Figura("Verde"); error
      Circulo c= new Circulo(3,"rojo");
      System.out.println("área del círculo: "+ c.area());
      Triangulo t = new Triangulo(2,3,"Verde");
```

```
        System.out.println("área triángulo: "+ t.area());
    }
}
```

Recuerda las ventajas de que area() sea abstracto en Figura:

1. Se obliga por cuestiones de diseño que los programadores que escriben el código de Figuras concretas obligatoriamente implemente area(). Es decir "se obedece al diseñador".

2. Al estudiar polimorfismo le veremos ventajas adicionales a las clases abstractas.