PRÁCTICA III: Herencia y Polimorfismo

1. Desarrollar en Java las clases Punto2D y Punto3D, de manera tal que Punto3D sea una clase derivada de Punto2D. Implemente en cada una el método equals() de manera tal de obtener por pantalla las salidas correctas a las siguientes instrucciones:

```
a.    Punto2D p2D = new Punto2D(0,0);
b.    Punto3D p3D = new Punto3D(0,0,1);
c.    Punto3D a3D = null;
d.    out.println(p2D.equals(p3D)); // false
e.    out.println(p3D.equals(p2D)); // false
f.    out.println(p2D.equals(a3D)); // false
g.    out.println(p3D.equals(a3D)); // false
h.    a3D = new Punto3D(0,0,0);
i.    out.println(p2D.equals(a3D)); // false
j.    out.println(a3D.equals(p2D)); // false
```

Importante: la salida no debe tener exceptions

2. Desarrollar la class PilaHL como una class derivada de la class Lista (ya implementada en la práctica II) y que implemente la interfaz Pila.

3. Desarrollar la class ColaHL como una class derivada de la class Lista (ya implementada en la práctica II) y que implemente la interfaz Cola.

4. Desarrollar la class PilaCL usando la relación contiene (composición) a la class Lista (ya implementada en la práctica II) y que implemente la interfaz Pila.

5. Desarrollar la class ColaCL usando la relación contiene (composición) a la class Lista (ya implementada en la práctica II) y que implemente la interfaz Cola.

6. Extraer conclusiones a partir de comparar las implementaciones 2 y 4.

7. Ídem para 3 y 5.

8. Desarrollar en Java la jerarquía de clases formada por la class Figura (abstracta) y dos clases derivadas Rectángulo y Círculo. Se deberá tener en cuenta lo siguiente:

a. La class Figura contiene en su declaración las posición x, y del centro de la figura.

b. La class **Rectangulo** tiene ancho y alto como variables miembro.
c. La class **Círculo** contiene los atributos centro y radio.
d. Ambas class implementan las interfaces **Rotable** y **Dibujable**

```
public interface Dibujable {
      public void dibujar();
}

public interface Rotable {
      public void rotar();
}
```

e. Implementar los métodos equals y área para la jerarquía de clases.

9. Desarrollar en Java la jerarquía de clases formada por la class Empleado y la class derivadas Jefe.
a. Atributos de Empleado: Legajo, Nombre y salario.
b. Atributo de Jefe: Sector.
c. Implemente los métodos equals

10. Indicar si son Verdaderas o Falsas las siguientes afirmaciones. Justifique:
a. Cuando una clase contiene un método abstracto tiene que declararse abstracta.
b. Todos los métodos de una clase abstracta tienen que ser abstractos.
c. Las clases abstractas no pueden tener métodos privados (no se podrían implementar) ni tampoco estáticos.
d. Una clase abstracta tiene que derivarse obligatoriamente.
e. Se puede hacer un new de una clase abstracta.
f. La principal diferencia entre interface y abstract es que un interface proporciona un mecanismo de encapsulación de los protocolos de los métodos sin forzar al usuario a utilizar la herencia.

11. Dado el siguiente código:

```
class BaseA {
      void mostrar() {
            out.println("clase BaseA");
      }
}
class DerivadaB extends BaseA {
      void mostrar() {
            out.println("clase DerivadaB");
      }
}
```

Indicar cuál es la salida si ejecutamos:

```
BaseA a1 = new BaseA();
a1.mostrar();
DerivadaB b1 = new DerivadaB();
b1.mostrar();
BaseA a2 = new DerivadaB();
a2.mostrar();
```