

## **PROGRAMACIÓN II**

### **Trabajo Práctico 7: Herencia y**

#### **Polimorfismo en Java**

##### **1. vehículo y herencia básica**

###### **Vehiculo.java**

```
public class Vehiculo {  
    protected String marca;  
    protected String modelo;  
  
    public Vehiculo(String marca, String modelo) {  
        this.marca = marca;  
        this.modelo = modelo;  
    }  
  
    public void mostrarInfo() {  
        System.out.println("Marca: " + marca + ", Modelo: " + modelo);  
    }  
}
```

###### **Auto.java**

```
public class Auto extends Vehiculo {  
    private int cantidadPuertas;  
  
    public Auto(String marca, String modelo, int cantidadPuertas) {  
        super(marca, modelo);  
    }  
}
```

```

        this.cantidadPuertas = cantidadPuertas;
    }

    @Override
    public void mostrarInfo() {
        System.out.println("Marca: " + marca + ", Modelo: " + modelo +
            ", Puertas: " + cantidadPuertas);
    }
}

```

### **Main.java**

```

public class MainVehiculo {
    public static void main(String[] args) {
        Auto auto = new Auto("Toyota", "Corolla", 4);
        auto.mostrarInfo();
    }
}

```

## **2. Figuras geométricas y métodos abstractos**

### **Figura.java**

```

public abstract class Figura {
    protected String nombre;

    public Figura(String nombre) {
        this.nombre = nombre;
    }

    public abstract double calcularArea();
}

```

### **Circulo.java**

```
public class Circulo extends Figura {  
    private double radio;  
  
    public Circulo(double radio) {  
        super("Círculo");  
        this.radio = radio;  
    }  
  
    @Override  
    public double calcularArea() {  
        return Math.PI * radio * radio;  
    }  
}
```

### **Rectangulo.java**

```
public class Rectangulo extends Figura {  
    private double ancho;  
    private double alto;  
  
    public Rectangulo(double ancho, double alto) {  
        super("Rectángulo");  
        this.ancho = ancho;  
        this.alto = alto;  
    }  
  
    @Override  
    public double calcularArea() {
```

```
        return ancho * alto;
    }
}
```

### **Main.figuras**

```
public class MainFiguras {
    public static void main(String[] args) {
        Figura[] figuras = {
            new Circulo(5),
            new Rectangulo(4, 6)
        };

        for (Figura f : figuras) {
            System.out.println(f.nombre + " - Área: " + f.calcularArea());
        }
    }
}
```

## **3. Empleados y polimorfismos**

### **Empleado.java**

```
public abstract class Empleado {
    protected String nombre;

    public Empleado(String nombre) {
        this.nombre = nombre;
    }

    public abstract double calcularSueldo();
}
```

```
}
```

### **EmpleadoPlanta.java**

```
public class EmpleadoPlanta extends Empleado {  
    private double sueldoFijo;  
  
    public EmpleadoPlanta(String nombre, double sueldoFijo) {  
        super(nombre);  
        this.sueldoFijo = sueldoFijo;  
    }  
  
    @Override  
    public double calcularSueldo() {  
        return sueldoFijo;  
    }  
}
```

### **EmpleadoTemporal.java**

```
public class EmpleadoTemporal extends Empleado {  
    private int horasTrabajadas;  
    private double valorHora;  
  
    public EmpleadoTemporal(String nombre, int horas, double valorHora) {  
        super(nombre);  
        this.horasTrabajadas = horas;  
        this.valorHora = valorHora;  
    }  
}
```

```

    }

    @Override

    public double calcularSueldo() {

        return horasTrabajadas * valorHora;

    }

}

```

### **MainEmpleados.java**

```

import java.util.ArrayList;

public class MainEmpleados {

    public static void main(String[] args) {

        ArrayList<Empleado> empleados = new ArrayList<>();

        empleados.add(new EmpleadoPlanta("Laura", 200000));

        empleados.add(new EmpleadoTemporal("Carlos", 80, 1500));

        for (Empleado e : empleados) {

            System.out.println(e.nombre + " - Sueldo: " + e.calcularSueldo());

            if (e instanceof EmpleadoPlanta)

                System.out.println("→ Es empleado de planta");

            else if (e instanceof EmpleadoTemporal)

                System.out.println("→ Es empleado temporal");

            System.out.println();

        }

    }

}

```

#### **4. Animales y comportamiento**

```
public class Animal {  
    public void hacerSonido() {  
        System.out.println("Sonido genérico de animal");  
    }  
  
    public void describirAnimal() {  
        System.out.println("Soy un animal.");  
    }  
}
```

##### **Perro.java**

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

##### **Gato.java**

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

### **Vaca.java**

```
public class Vaca extends Animal {  
    @Override  
    public void hacerSonido() {  
        System.out.println("Muuu");  
    }  
}
```

### **MainAnimales.java**

```
import java.util.ArrayList;  
  
public class MainAnimales {  
    public static void main(String[] args) {  
        ArrayList<Animal> animales = new ArrayList<>();  
  
        animales.add(new Perro());  
        animales.add(new Gato());  
        animales.add(new Vaca());  
  
        for (Animal a : animales) {  
            a.describirAnimal();  
            a.hacerSonido();  
            System.out.println();  
        }  
    }  
}
```