CASO PRÁCTICO 7
PROGRAMACIÓN II

Profesor/es: Ramiro Huelpa
Alumno/a: Lauk, Karen

Herencia y Polimorfismo en Java

Objetivo General
Comprender y aplicar los conceptos de herencia y polimorfismo en la Programación Orientada a Objetos, reconociendo su importancia para la reutilización de código, la creación de jerarquías de clases y el diseño flexible de soluciones en Java.

Caso práctico 1
Desarrollar las siguientes Katas en Java aplicando herencia y polimorfismo. Se recomienda repetir cada kata para afianzar el concepto.

1. Vehículos y herencia básica
   - Clase base: Vehículo con atributos marca, modelo y método mostrarInfo()

```java
package caso7;
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);
    }
}
```

   - Subclase: Auto con atributo adicional cantidadPuertas, sobrescribe mostrarInfo()

```java
package caso7;
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);
    }
}
```

   - Tarea: Instanciar un auto y mostrar su información completa.

```java
package caso7;
public class Main {
    public static void main(String[] args) {
        Auto miAuto = new Auto("Toyota", "Corolla", 4);
        miAuto.mostrarInfo();
    }
}
```

Resultado:

```
run:
Marca: Toyota, Modelo: Corolla, Puertas: 4
BUILD SUCCESSFUL (total time: 0 seconds)
```

2. Figuras geométricas y métodos abstractos
   - Clase abstracta: Figura con método calcularArea() y atributo nombre.

```java
package ejercicio2;
public abstract class Figura {
    protected String nombre;
    public Figura(String nombre) {
        this.nombre = nombre;
    }
    public abstract double calcularArea(); // método abstracto
    public void mostrarNombre() {
        System.out.println("Figura: " + nombre);
    }
}
```

   - Subclases: Círculo y Rectángulo implementan el cálculo del área.

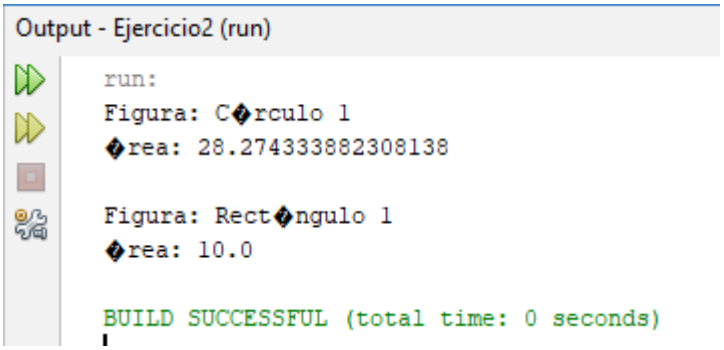
```java
package ejercicio2;
public class Circulo extends Figura {
    private double radio;

    public Circulo(String nombre, double radio) {
        super(nombre);
        this.radio = radio;
    }
    @Override
    public double calcularArea() {
        return Math.PI * radio * radio;
    }
}
```

```java
package ejercicio2;
public class Rectangulo extends Figura {
    private double base;
    private double altura;
    public Rectangulo(String nombre, double base, double altura) {
        super(nombre);
        this.base = base;
        this.altura = altura;
    }
    @Override
    public double calcularArea() {
        return base * altura;
    }
}
```

- Tarea: Crear un array de figuras y mostrar el área de cada una usando polimorfismo.

```java
package ejercicio2;
public class Ejercicio2 {
    public static void main(String[] args) {
        Figura[] figuras = {
            new Circulo("Círculo 1", 3.0),
            new Rectangulo("Rectángulo 1", 4.0, 2.5)
        };
        for (Figura f : figuras) {
            f.mostrarNombre();
            System.out.println("Área: " + f.calcularArea());
            System.out.println();
        }
    }
}
```

Resultado:



3. Empleados y polimorfismo
   - Clase abstracta: Empleado con método calcularSueldo()

```java
package ejercicio3;
public abstract class Empleado {
    protected String nombre;
    public Empleado(String nombre) {
        this.nombre = nombre;
    }
    public abstract double calcularSueldo();
}
```

- Subclases: EmpleadoPlanta, EmpleadoTemporal

```java
package ejercicio3;
public class EmpleadoPlanta extends Empleado {
    private double sueldoBase;
    public EmpleadoPlanta(String nombre, double sueldoBase) {
        super(nombre);
        this.sueldoBase = sueldoBase;
    }
    @Override
    public double calcularSueldo() {
        return sueldoBase;
    }
}
```

```java
package ejercicio3;
public class EmpleadoTemporal extends Empleado {
    private double horas;
    private double pagoPorHoras;
    public EmpleadoTemporal(String nombre, double horas, double pagoPorHoras){
        super(nombre);
        this.horas=horas;
        this.pagoPorHoras=pagoPorHoras;
    }
    @Override
    public double calcularSueldo(){
        return horas * pagoPorHoras;
    }
}
```

- Tarea: Crear lista de empleados, invocar calcularSueldo() polimórficamente, usar instanceof para clasificar

```java
package ejercicio3;
import java.util.ArrayList;
public class Ejercicio3 {
    public static void main(String[] args) {
        ArrayList<Empleado> empleados = new ArrayList<>();
        empleados.add(new EmpleadoPlanta("María", 150000));
        empleados.add(new EmpleadoTemporal("Juan", 80, 1500));
        for (Empleado e : empleados) {
            System.out.println(e.nombre + " cobra: " + e.calcularSueldo());
            if (e instanceof EmpleadoPlanta) {
                System.out.println(" -> Es empleado de planta.");
            } else {
                System.out.println(" ->  Es empleado temporal.");
            }
        }
    }
}
```

Resultados:

```
run:
Mar?a cobra: 150000.0
 -> Es empleado de planta.
Juan cobra: 120000.0
 ->  Es empleado temporal.
BUILD SUCCESSFUL (total time: 0 seconds)
```

4. Animales y comportamiento sobrescrito
   - Clase: Animal con método hacerSonido() y describirAnimal()

```java
package ejercicio4;
public class Animal {
    public void describirAnimal() {
        System.out.println("Este es un animal.");
    }
    public void hacerSonido() {
        System.out.println("Sonido generico de animal");
    }
}
```

   - Subclases: Perro, Gato, Vaca sobrescriben hacerSonido() con @Override

```java
package ejercicio4;
public class Perro extends Animal {
    @Override
    public void describirAnimal() {
        System.out.println("Soy un mamifero domestico de la familia de los canidos");
    }
    @Override
    public void hacerSonido() {
        System.out.println("Y hago: ¡Guau guau!");
    }
}
```

```java
package ejercicio4;
public class Gato extends Animal {
    @Override
    public void describirAnimal(){
        System.out.println("Soy un mamifero carnivoro, domestico y depredador.");
    }
    @Override
    public void hacerSonido() {
        System.out.println("Y hago: Miau!");
    }
}
```

```java
package ejercicio4;
public class Vaca extends Animal {
    @Override
    public void describirAnimal(){
        System.out.println("Soy un mamifero herbivoro grande de la familia de los bovidos");
    }
    @Override
    public void hacerSonido() {
        System.out.println("Muuu!");
    }
}
```

- Tarea: Crear lista de animales y mostrar sus sonidos con polimorfismo

```java
package ejercicio4;
public class Ejercicio4 {
    public static void main(String[] args) {
        Animal[] animales= {new Perro(), new Gato(), new Vaca() };
        for (Animal a : animales) {
            a.describirAnimal();
            a.hacerSonido();
            System.out.println();
        }
    }
}
```

Resultado:

```
Output - Ejercicio4 (run)
run:
Soy un mamifero domestico de la familia de los canidos
Y hago: Guau guau!

Soy un mamifero carnivoro, domestico y depredador.
Y hago: Miau!

Soy un mamifero herbivoro grande de la familia de los bovidos
Muuu!

BUILD SUCCESSFUL (total time: 0 seconds)
```

Link de repositorio GitHub:
https://github.com/karenlauk/Programaci-n2