PROGRAMACIÓN ORIENTADA A OBJETOS

EDUARDO FELIPE POOT CHAIREZ

LOS 4 PARADIGMAS

- **Ensamblador:**
  - Se traduce a binario (1 y 0).
  - Funciona como una "lista de compras": instrucciones lineales, una tras otra.
  - Todavía se puede usar con lenguajes modernos para tareas extremadamente simples.
- **Programación procedural:** El código se divide en subrutinas o funciones (Modularidad).

- **Funcional:**
  - Se basa en funciones matemáticas y en la evaluación de expresiones.
  - Se enfoca en qué resultado se quiere obtener aplicando funciones a los datos, en vez de decirle al programa cómo hacer las cosas paso a paso.
- **Programación orientada a objetos:**
  - El programa gira en torno a objetos que imitan el mundo real.
  - Es el mas usado para sistemas complejos, cómo la gestión de inventarios en un almacén o productos en una tienda.

¿QUÉ ES UN OBJETO?

Es la representación en un programa de un concepto.
Contiene toda la información necesaria para abstraerlo:

- Datos que describen sus atributos.
- Operaciones que pueden realizarse sobre los mismos.

¿QUÉ ES UNA CLASE?

Es una plantilla o modelo que define cómo serán los objetos en la programación orientada a objetos.
En una clase se especifica qué datos tendrá un objeto y qué acciones podrá realizar, pero no es el objeto en sí, sino el molde para crearlo.

¿COMO SE DEFINE UNA CLASE?

- **Atributos:** Son las variables que representan el estado del objeto.
  Ejemplo: color, tamaño, precio.

- **Métodos:** Son las funciones que definen el comportamiento del objeto.
  Ejemplo: encender(), calcularTotal(), mover().

PRINCIPIOS DE POO

- Encapsulación.
- Herencia.
- Polimorfismo.
- Abstracción.

ENCAPSULACIÓN

Consiste en ocultar los detalles internos de un objeto y permitir el acceso solo a través de métodos definidos, protegiendo así sus datos.

```java
class Usuario {
    private String contraseña;

    public void cambiarContraseña(String nueva) {
        contraseña = nueva;
    }

    public String getContraseña() {
        return contraseña;
    }
}
```

HERENCIA

Permite que una clase herede atributos y métodos de otra clase, favoreciendo la reutilización de código.

```java
class Empleado {
    String nombre;
    double salario;
}

class Gerente extends Empleado {
    String departamento;
}
```

POLIMORFISMO

Permite que un mismo método tenga diferentes comportamientos según el objeto que lo utilice.

```java
class Animal {
    void sonido();
}

class Perro extends Animal {
    void sonido() {
        System.out.println("El perro ladra");
    }
}

class Gato extends Animal {
    void sonido() {
        System.out.println("El gato maúlla");
    }
}
```

ABSTRACCIÓN

Consiste en representar solo las características esenciales de un objeto, ocultando los detalles innecesarios.

```java
abstract class Figura {
    abstract double calcularArea();
}

class Rectangulo extends Figura {
    double base = 5;
    double altura = 3;

    double calcularArea() {
        return base * altura;
    }
}
```