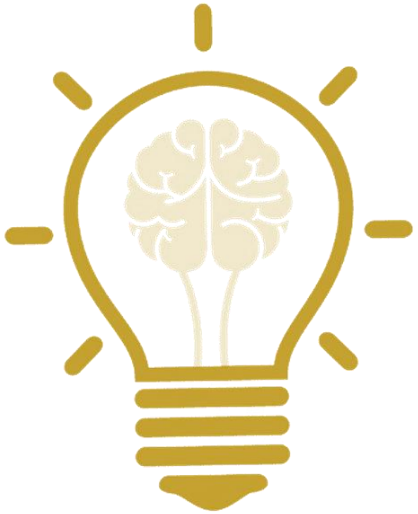
PROGRAMACIÓN EN JAVA

Materia PJA-001 material creado por el Ing. Jim Requena

Programación Orientada a Objetos en JAVA (Primara Parte – Introducción OOP)



Materia PJA-001 material creado por el Ing. Jim Requena

Objetivos de la sesión

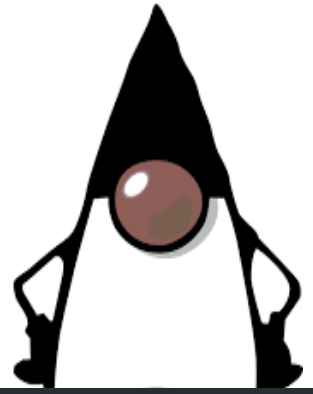
- Comprender los conceptos básicos de la POO
- Aprender a crear clases y objetos en Java
- Entender y aplicar los conceptos de atributos y métodos
- Practicar con ejercicios de POO

¿Qué es la POO?

- Paradigma de programación basado en "objetos“
- Los objetos contienen datos y código
- Promueve la organización y reutilización del código
- Conceptos clave: clases, objetos, atributos, métodos

Clases y Objetos

- Clase: plantilla para crear objetos
- Objeto: instancia de una clase



Ejemplo:

```java
// Definición de una clase
public class Coche {
    // Atributos y métodos aquí
}

// Creación de un objeto
Coche miCoche = new Coche();
```

Atributos

- Variables que pertenecen a una clase
- Representan las características de un objeto

```java
public class Coche {
    String marca;
    String modelo;
    int año;
    String color;
}
```

Métodos

- Funciones que pertenecen a una clase
- Representan el comportamiento de un objeto

```java
public class Coche {
    // ... atributos ...

    void arrancar() {
        System.out.println("El coche está arrancando");
    }

    void acelerar(int velocidad) {
        System.out.println("Acelerando a " + velocidad + " km/h");
    }
}
```

Creación y uso de objetos

```java
public class PruebaCoche {
    public static void main(String[] args) {
        Coche miCoche = new Coche();
        miCoche.marca = "Toyota";
        miCoche.modelo = "Corolla";
        miCoche.año = 2022;
        miCoche.color = "Rojo";

        miCoche.arrancar();
        miCoche.acelerar(60);
    }
}
```

Encapsulamiento

```java
public class Coche {
    private String marca;
    private int velocidad;

    public void setMarca(String marca) {
        this.marca = marca;
    }

    public String getMarca() {
        return marca;
    }

    public void acelerar() {
        if (velocidad < 120) {
            velocidad += 10;
        }
    }
}
```

- **Ocultar** los **detalles** internos de una clase
- Usar **modificadores de acceso**: public, private, protected

El encapsulamiento es como poner tus pertenencias en una caja con cerradura. Tú decides qué puede entrar y salir de la caja, y cómo.

En el contexto de la clase Coche:
La clase Coche es como una caja que contiene toda la información y funcionalidad de un coche.

Las variables marca y velocidad son como objetos dentro de esta caja.

Acceso controlado:
private String marca; y private int velocidad;
// son como poner estos "objetos" bajo llave dentro de la caja.
Solo la clase Coche puede acceder directamente a estos valores.

Métodos como interfaz:setMarca(), getMarca(), y acelerar() son como botones o palancas en el exterior de la caja.

Estos métodos permiten interactuar con lo que está dentro de la caja de manera controlada.

El encapsulamiento ayuda a crear código más seguro, fácil de mantener y que imita mejor cómo funcionan las cosas en el mundo real.

Ejercicio Práctico

- Crear una clase Estudiante con Atributos:
  - nombre
  - edad
  - grado
- Implementa el constructor para inicializar estos atributos
- Desarrolla los métodos para:
  - Mostrar información del estudiante
  - Pasar al siguiente grado
  - Crear varios objetos Estudiante y usar sus métodos

```java
public class Estudiante {
    private String nombre;
    private int edad;
    private int grado;

    public Estudiante(String nombre, int edad, int grado) {
        this.nombre = nombre;
        this.edad = edad;
        this.grado = grado;
    }
    public void mostrarInfo() {
        System.out.println("Nombre: " + nombre + ", Edad: " + edad + ", Grado: " + grado);
    }

    public void pasarDeGrado() {
        grado++;
        System.out.println(nombre + " ha pasado al grado " + grado);
    }

    public static void main(String[] args) {
        Estudiante est1 = new Estudiante("Ana", 15, 9);
        Estudiante est2 = new Estudiante("Juan", 16, 10);

        est1.mostrarInfo();
        est2.mostrarInfo();

        est1.pasarDeGrado();
        est2.pasarDeGrado();

        est1.mostrarInfo();
        est2.mostrarInfo();
    }
}
```

Tarea

- Amplía la clase Estudiante para incluir un array de calificaciones

- Agrega métodos para:
  - Añadir una nueva calificación
  - Calcular el promedio de calificaciones
  - Determinar si el estudiante aprueba (promedio >= 6)

- Crea una clase Curso que pueda contener varios estudiantes

- Implementa métodos en Curso para:
  - Agregar estudiantes
  - Mostrar la información de todos los estudiantes
  - Calcular el promedio general del curso
  - Sube tu solución al repositorio del curso

Recursos adicionales

- Tutorial de Oracle sobre POO en Java:
  docs.oracle.com/javase/tutorial/java/concepts/

- Libro "Thinking in Java" de Bruce Eckel
  (está disponible gratuitamente en línea)

Gracias!