Interface and ABSTRACT

CLASE ABSTRACTA

Definición:
Plantilla para otras clases. No se pueden instanciar.
Palabra clave: abstract
Propósito: Definir una estructura común y obligar a implementar métodos.
Características:

- Pueden tener métodos normales y abstractos.
- Las subclases deben implementar los métodos abstractos.

```
interface Volador {
    void volar();
}

interface Nadador {
    void nadar();
}

class Pato implements Volador, Nadador {
    public void volar() { System.out.println("Vuelo bajo"); }
    public void nadar() { System.out.println("Nado en el lago"); }
}
```

INTERFACES

Definición: Contrato que define métodos (sin implementación).
Palabra clave: implements
Propósito: Lograr polimorfismo y herencia múltiple.
Características:

- Solo tienen métodos abstractos (implícitamente).
- Una clase puede implementar varias interfaces.
- Obligan a implementar todos los métodos definidos.

```
interface Volador {
    void volar();
}

interface Nadador {
    void nadar();
}

class Pato implements Volador, Nadador {
    public void volar() { System.out.println("Vuelo bajo"); }
    public void nadar() { System.out.println("Nado en el lago"); }
}
```

POLIMORFISMO

Definición:
Capacidad de un objeto de tomar muchas formas.
Para qué sirve: Permite que un mismo método se comporte de manera diferente según el objeto que lo ejecute.

CUANDO USAR...

CLASES ABSTRACTAS:

- Cuando necesitas herencia múltiple
- Cuando defines un contrato para clases no relacionadas
- Para desacoplar código (principio de inversión de dependencias)

INTERFACES

- Cuando necesitas herencia múltiple
- Cuando defines un contrato para clases no relacionadas
- Para desacoplar código (principio de inversión de dependencias)

DEFAULT

Propósito:
Permiten agregar nuevos métodos a una interfaz **sin romper** las clases que ya la implementan.
Características:

- Tienen una implementación concreta dentro de la interfaz.
- Se definen con la palabra clave default.
- Pueden ser sobrescritos (@Override) en las clases que implementan la interfaz.

```
    default void detener() {
        System.out.println("El vehiculo se ha detenido.");
    }
}

public class Coche implements Vehiculo {
    @Override
    public void arrancar() {
        System.out.println("Coche arrancado.");
    }
```

Interface and ABSTRACT

STATIC

Propósito:
Permiten definir métodos de utilidad asociados a la interfaz.
Características:

- Se definen con static.
- No pueden ser sobrescritos por las clases implementadoras.
- Se invocan directamente desde la interfaz: NombreInterfaz.metodoEstatico()

PROGRAM TO AN INTERFACE, NOT AN IMPLEMENTATION

¿Qué significa?

- Programar contra interfaces (abstracciones) en lugar de contra clases concretas.
- Ejemplo: En lugar de usar ArrayList directamente, usas la interfaz List.

¿Cómo aplicarlo en Java?

```
public interface Vehiculo {
    void acelerar();
    void frenar();
}

public class Coche implements Vehiculo {
    public void acelerar() {
        System.out.println("Coche acelerando");
    }
    public void frenar() {
        System.out.println("Coche frenando");
    }
}

public class Conductor {
    private Vehiculo vehiculo;

    public Conductor(Vehiculo vehiculo) {
        this.vehiculo = vehiculo; // Inyección de dependencia
    }

    public void manejar() {
        vehiculo.acelerar();
    }
```