UNIVERSIDAD TECNOLO GICA DE PANAMA

FACULTAD DE INGENIERIA EN SISTEMAS COMPUTACIONALES

LICENCIATURA EN CIBERSEGURIDAD

MATERIA:

PROGRAMACIÓN I

INVESTIGACIÓN 1

NOMBRE DEL ESTUDIANTE:

IRVIN MARTINEZ

PROFESOR: NAPOLEON IBARRA

FECHA DE ENTREGA: 20 DE AGOSTO DE 2025

1. Construcción de clase (Java)

- **Concepto**
  Una clase en Java es una plantilla o molde que permite crear objetos dentro de la programación orientada a objetos. Define las características (atributos) y comportamientos (métodos) que tendrán los objetos.

- **Importancia y/o relevancia**

Las clases son fundamentales en Java porque:

- Permiten aplicar los principios de la **programación orientada a objetos** (POO).
- Favorecen la **organización y modularidad** del código.
- Facilitan el **re-uso y mantenimiento**, ya que una misma clase puede instanciarse múltiples veces en diferentes programas.
- Promueven la **seguridad y encapsulamiento**, protegiendo la información sensible.

- **¿Ventajas y Desventajas? Mínimo 3, Máximo 5 item**
  Ventajas:
  1. Reutilización del código mediante objetos creados a partir de una misma clase.
  2. Encapsulamiento, lo que permite proteger datos sensibles.
  3. Facilidad para mantener y escalar proyectos grandes, ya que cada clase organiza su propia lógica.

  Desventajas:

  1. Puede ser difícil de comprender para principiantes.
  2. El mal uso de clases y objetos puede generar código poco eficiente.
  3. En proyectos pequeños, puede sentirse como una estructura "excesiva" comparado con la programación estructurada.

- **Ejemplos de cada subpunto**

1.1. Miembros de una clase

Son los componentes internos de una clase, es decir, la definición de todos los elementos de que está hecho un objeto.

- Atributos: variables que guardan el estado.
- Métodos: funciones que definen el comportamiento.
- Constructores: inicializan los objetos.

Los atributos se declaran de la siguiente forma:

[=];

Los métodos de una clase se declaran de la siguiente forma:

() { }

Ejemplo:

```
class Persona {
    String nombre;   // Atributo
    int edad;        // Atributo

    Persona(String n, int e) {  // Constructor
        nombre = n;
        edad = e;
    }

    void mostrarInfo() {  // Método
        System.out.println("Nombre: " + nombre + ", Edad: " + edad);
    }
}
```

1.2. Modificadores de acceso

Es la forma en cómo clasificamos nuestras propiedades o métodos para hacerlos visibles o invisibles a quienes usen un objeto de una clase que creamos. Los modificadores de acceso son: **Public**, **Private**, **Protected**.

Ejemplo:

```
class MiOtraClase {
    private int miVariablePrivada;
    private void miMetodoPrivado() {
        // ...
    }
}
```

Ejemplo de **private**

```java
public class MiClase {
    public int miVariablePublica;
    public void miMetodoPublico() {
        // ...
    }
}
```

Ejemplo de **public**

```java
class ClasePadre {
    protected int miVariableProtegida;
}

class ClaseHija extends ClasePadre {
    public void metodo() {
        miVariableProtegida = 10; //  Acceso permitido
    }
}
```

Ejemplo de protected

1.3. Otras opciones

Clases Regulares o de Programador

Este tipo de clases, son las clases estándar que los programadores crean para modelar objetos y comportamientos en sus aplicaciones. No hay ninguna distinción especial en las diferentes versiones de Java.

```java
public class Usuario {
    private final String nombre;
    private final String apellido;
    private final String email;

    public Usuario(String nombre, String apellido, String email) {
        this.nombre = nombre;
        this.apellido = apellido;
        this.email = email;
    }

    public String getNombreCompleto() {
        return String.format("%s %s", nombre, apellido);
    }

    public String getNombre() {
        return nombre;
    }

    public String getApellido() {
        return apellido;
    }

    public String getEmail() {
        return email;
    }
}
```

Clases enumeradas (Enums)

Las clases enumeradas son un tipo especial de clase que representa un conjunto fijo de valores, como días de la semana o tipos de cartas en una baraja. A partir de versiones anteriores de Java, las clases Enum se han mejorado en términos de concurrencia y desempeño.

```
public enum DiaSemana {
    LUNES, MARTES, MIERCOLES, JUEVES, VIERNES, SABADO, DOMINGO
}

// Uso de la clase enum
    var algunDia = DiaSemana.LUNES;

    switch (algunDia) {
        case LUNES -> System.out.println("¡Ve al gimnasio!");
    }
```

Clases Anidadas o Internas (Nested/Inner Classes)

Las clases anidadas son clases definidas dentro de otras clases. Este tipo de clases pueden ser estáticas o no estáticas. La anidación de clases se ha permitido en Java desde sus primeras versiones.

```
public class ClaseExterna {
    // Miembros de la clase externa

    public class ClaseAnidada {
        // Miembros de la clase anidada
    }
}
```

```java
public class ClaseExterna {
    private static int x = 10;

    public static class ClaseAnidadaEstatica {
        public void mostrarValorExterno() {
            System.out.println("Valor de x en la clase externa: " + x);
        }
    }
}
```

Clases Anónimas

Las clases anónimas son clases que se crean sin un nombre y se utilizan para implementar interfaces o clases abstractas en el lugar donde se necesitan.

```java
InterfazEjemplo miObjeto = new InterfazEjemplo() {
    public void metodoEjemplo() {
        System.out.println("Implementación de método en clase anónima.");
    }
};
```

Referencias

Blasco, J. L. (2023, octubre 27). Introducción a POO en Java: Objetos y clases. *Openwebinars.net*. https://openwebinars.net/blog/introduccion-a-poo-en-java-objetos-y-clases/

Rodriguez, G. J. (2012, febrero 18). *Programación Orientada a Objetos*. Northware. https://www.northware.mx/blog/programacion-orientada-a-objetos/

Tipos de Clases en Java. (2023, noviembre 1). Programandojava.com. https://www.programandojava.com/blog/tipos-de-clases