Erick Antillón

Introducción a la programación y computación 1

Clase 4- Introducción a POO

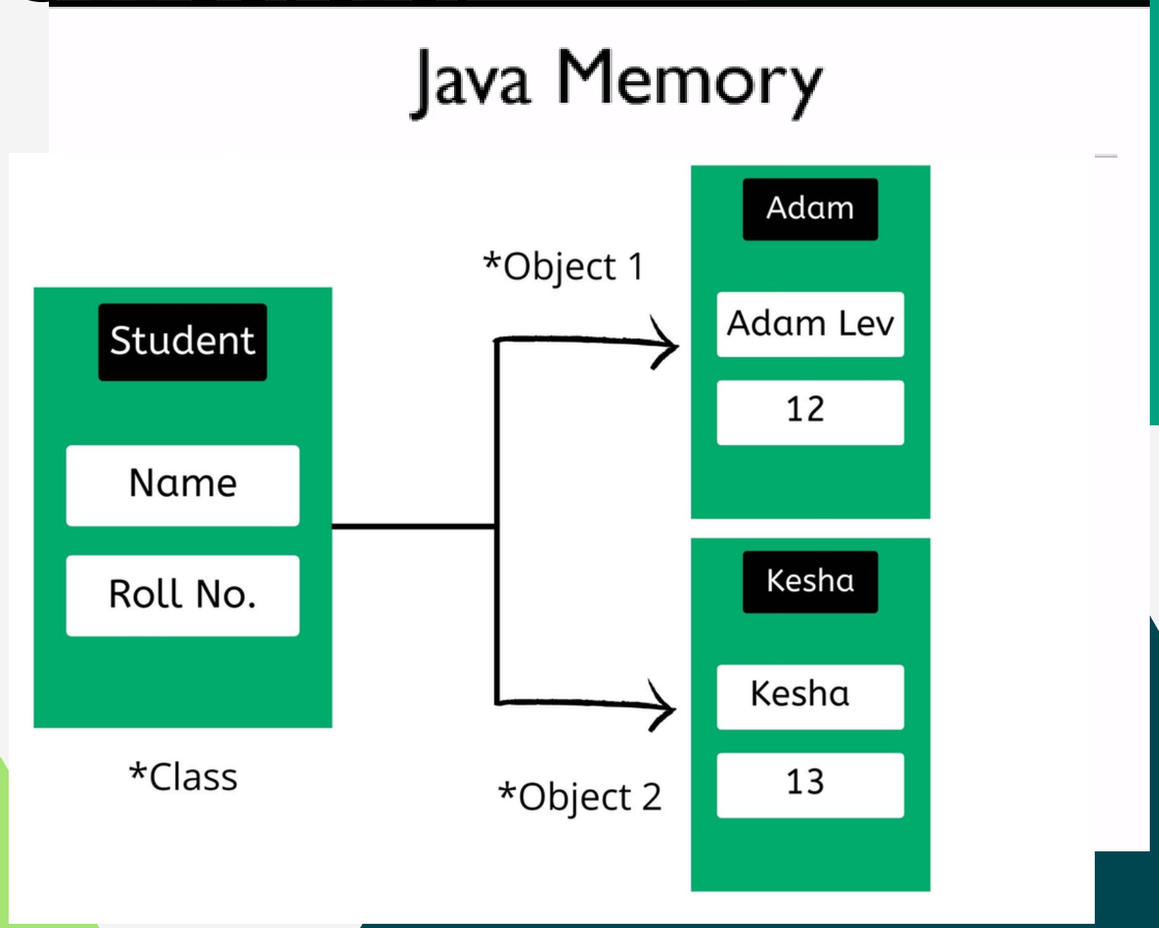
Agenda

- Introduccion a POO
- Atributos y métodos
- Conceptos básicos de POO
- Usando Git con herramientas visuales
- Ejemplo
- Proyecto 1
- Asistencia

Programación Orientada a Objetos

- Programación Orientada a Objetos (POO) es un paradigma de programación que se basa en la representación de objetos y la interacción entre ellos.

- **Clases:** es una plantilla o un modelo que define las propiedades y comportamientos de un objeto
- **Objetos:** es una entidad única que se basa en esa clase y que tiene sus propios valores para esas propiedades.



Atributos y métodos

- **Definición de atributos**: Los atributos son variables que describen el estado o información asociada a un objeto. Se pueden definir dentro de una clase y estan disponibles para ser accedidos y modificados por los métodos de la clase.

```java
public class Animal
{
    // Atributos de la clase
    private String raza;
    private String nombre;
    private int edad;

}
```

Atributos y métodos

- **Definición de métodos**: Los métodos describen el comportamiento o las acciones que un objeto puede realizar. Pueden tomar argumentos y devolver un valor.

```java
public class Animal
{
  // Atributos de la clase
  // ...

  // Metodos

  // constructor
  public Animal(String nuevoNombre)
  {
      nombre = nuevoNombre; //Se le da un nombre al animal
  }


  //Método para obtener el nombre del animal
  public String getNombre()
  {
      return nombre;
  }

}
```

Modificadores de acceso

- Los modificadores de acceso controlan el nivel de acceso que tienen los métodos y atributos de una clase. Los tres modificadores básicos en Java son public, private y protected.

  - **public**: Un elemento marcado como public puede ser accedido desde cualquier parte del código.
  - **private**: Un elemento marcado como private solo puede ser accedido desde dentro de la clase donde se ha definido.
  - **protected**: Un elemento marcado como protected puede ser accedido desde dentro de la clase donde se ha definido y también desde sus clases hijas.

Palabra reservda this

- **this**: La palabra reservada **this** se utiliza para hacer referencia a un atributo o método específico dentro de una clase. Por ejemplo, se puede usar para acceder a un atributo privado desde dentro de la clase:

```
public class miClase
{
  // Atributo privade
  private int atributo;

  public void metodo() {
    // se accede usando la palabra this
    this.atributo = 10;
  }
}
```

Palabra reservda this

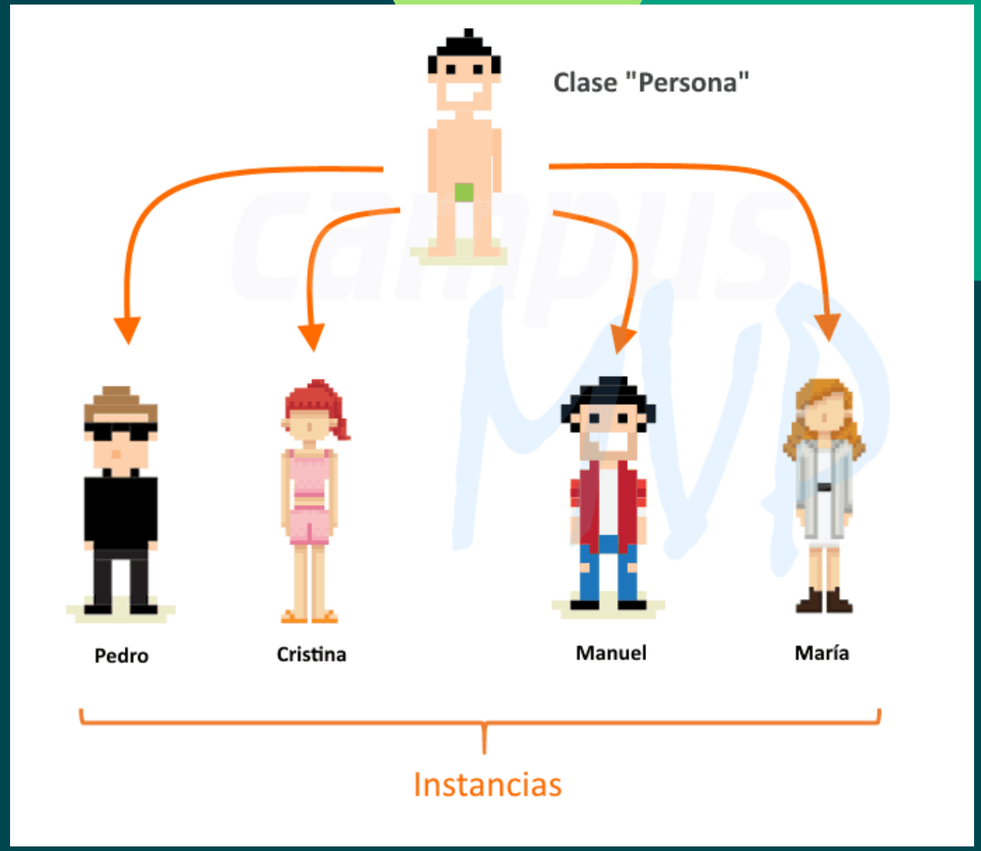
- **this**: La palabra reservada **this** se utiliza para hacer referencia a un atributo o método específico dentro de una clase. Por ejemplo, se puede usar para acceder a un atributo privado desde dentro de la clase:

```
public class miClase
{
  // Atributo privade
  private int atributo;

  public void metodo() {
    // se accede usando la palabra this
    this.atributo = 10;
  }
}
```

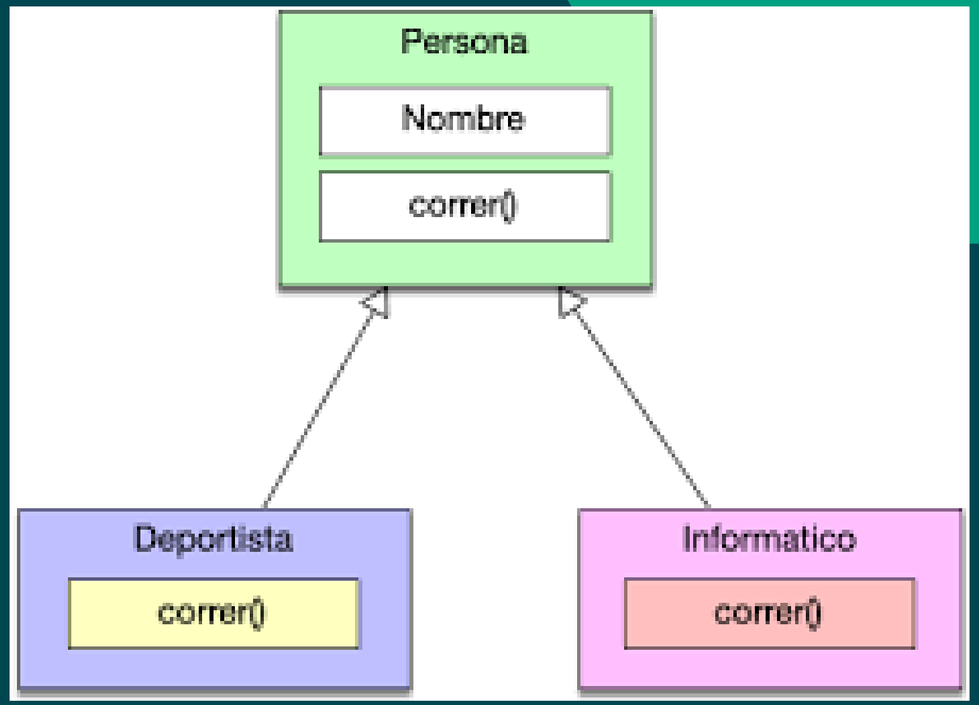
Herencia en Java

- La herencia permite a los desarrolladores reutilizar código existente. La herencia se logra al crear una nueva clase que extiende una clase existente. La clase existente se conoce como superclase y la nueva clase se conoce como subclase. La subclase hereda todos los atributos y métodos de la superclase y puede agregar o modificar la funcionalidad. Se utiliza la palabra clave **extends** para indicar que una clase es una subclase de otra.



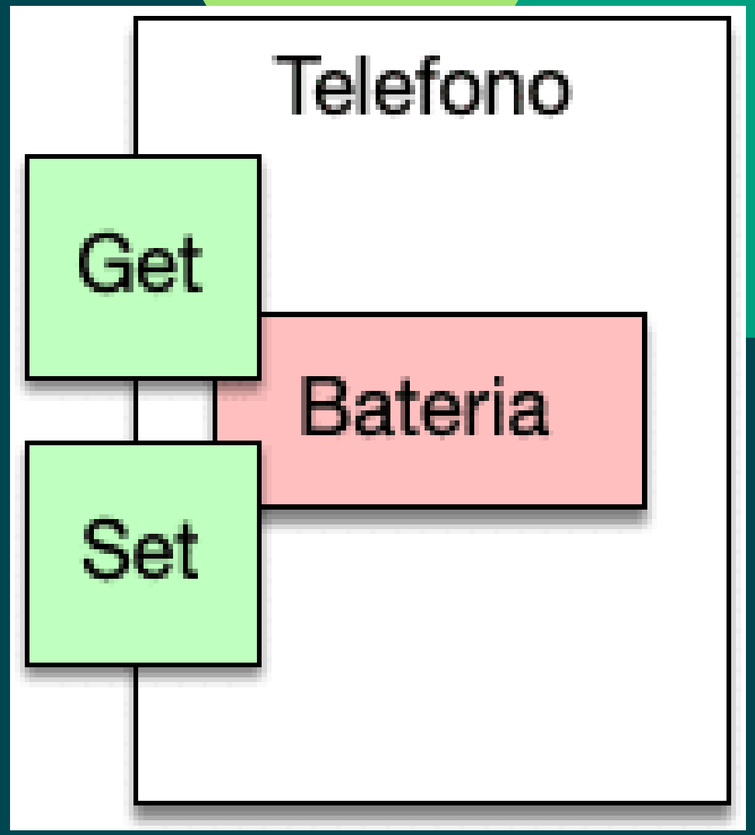
Polimorfismo en Java

- El polimorfismo permite que los objetos de diferentes clases respondan de manera diferente a los mismos mensajes. La sobrecarga de métodos se refiere a la capacidad de tener métodos con el mismo nombre en una clase pero con diferentes argumentos. La sobrescritura de métodos permite que una subclase redefina el comportamiento de un método en la superclase. Se utiliza la palabra clave override para indicar que un método en una subclase está redefiniendo un método en la superclase.



Encapsulamiento

- El término encapsulamiento en Java, consiste en ocultar atributos de un objeto de manera que solo se pueda cambiar mediante operaciones definidas en ese objeto. Está estrechamente relacionado con la visibilidad.
- Para definir la visibilidad en Java, se dispone de palabras reservadas: **public, private, protected**



Herramientas visuales para git

Existen múltiples herramientas visuales que pueden ayudar a trabajar con Git de manera más intuitiva y fácil de entender.

- https://git-scm.com/downloads/guis

Recomendación:

- https://www.gitkraken.com/git-client

Ejemplo Práctico

Ejemplo de Clases, Objetos, Herencia, Polimorfismo y encapsulamiento. Y uso de Gitkraken

```java
no usages
public class Main {

    7 usages
    public static String[] empleados = new String[5];
    1 usage
    public static int[] horasTrabajadas = new int[5];

    no usages
    public static String[] nombreBonos = new String[5];
    no usages
    public static int[] porcentajesBonos = new int[5];

    1 usage
    public static void crearEmpleado(){
        Scanner input = new Scanner(System.in);
        System.out.println("Ingrese el nombre del nuevo empleado: ");
        String nombre = input.nextLine();
        System.out.println("Ingrese las horas trabajadas: ");
        int horas = input.nextInt();

        // validar que no existe ese empleado
```

¿Dudas o preguntas?

Asistencia

- Formulario de Asistencia:
- https://forms.gle/N7i9bgT47r2r7RHB7