

ESTRUCTURA DE DATOS

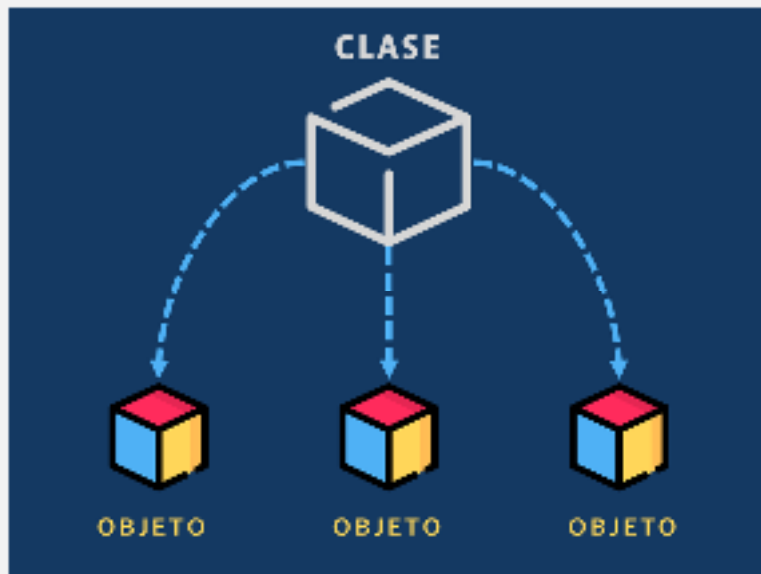
Estudiante : Elvin Braxail Cussi Aranibar

Semestre : 3er

Año : 2023

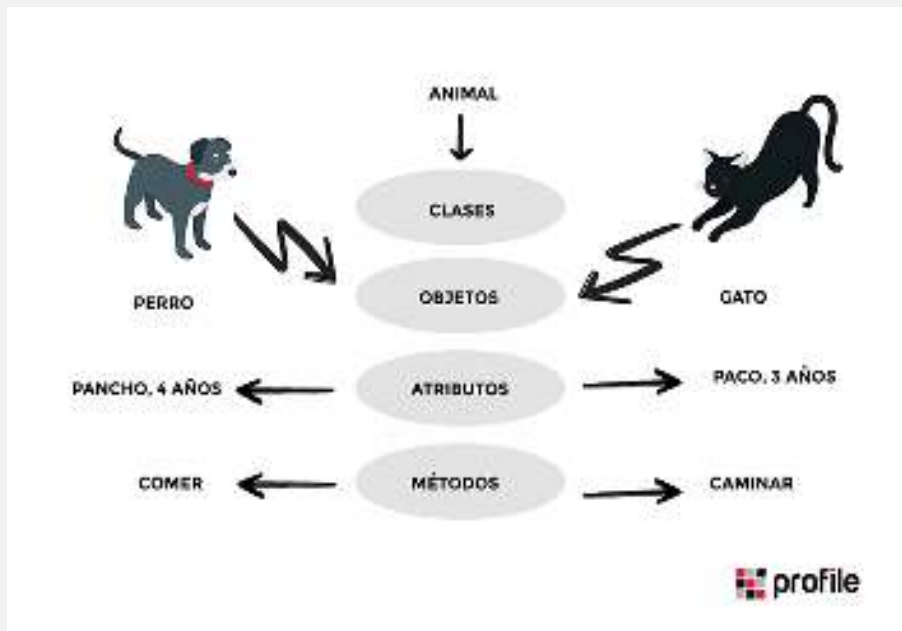
MANEJO DE CONCEPTOS

I. ¿A QUE SE REFIERE CUANDO SE HABLA DE POO?



En la programación orientada a objetos, se modela el problema a resolver como una colección de objetos que representan entidades o conceptos del mundo real. Cada objeto tiene propiedades (datos) y métodos (funciones) que lo definen y le permiten interactuar con otros objetos del sistema.

2. ¿CUÁLES SON LOS 4 COMPONENTES QUE COMPONEN POO?



Abstracción: Es el proceso de identificar las características esenciales de un objeto

Encapsulamiento: Es el proceso de ocultar la complejidad interna de un objeto y exponer solo su interfaz pública.

Herencia: Es el proceso de crear una nueva clase a partir de una clase existente, heredando sus atributos y métodos

Polimorfismo: Es la capacidad de un objeto de tomar varias formas o comportarse de diferentes maneras según el contexto en el que se utiliza

3. ¿CUÁLES SON LOS PILARES DE POO?

Arrays: Son estructuras de datos lineales que permiten almacenar un conjunto de elementos del mismo tipo

Listas enlazadas: Son estructuras de datos dinámicas que permiten almacenar un conjunto de elementos del mismo o diferente tipo.

Pilas: Son estructuras de datos lineales que permiten almacenar elementos siguiendo una política LIFO (Last In, First Out), es decir, el último elemento que se inserta es el primero que se elimina.

Colas: Son estructuras de datos lineales que permiten almacenar elementos siguiendo una política FIFO (First In, First Out), es decir, el primer elemento que se inserta es el primero que se elimina.



4. ¿QUÉ ES ENCAPSULAMIENTO Y MUESTRE UN EJEMPLO?

Nos permiten efectuar operaciones sobre un conjunto de resultados, pero devolviendo un único valor agregado para todos ellos



5. ¿QUÉ ES ABSTRACCIÓN Y MUESTRE UN EJEMPLO?

Un ejemplo de abstracción en estructuras de datos es la implementación de una lista enlazada. En una lista enlazada, cada elemento se representa por un nodo que contiene un valor y un puntero al siguiente nodo.

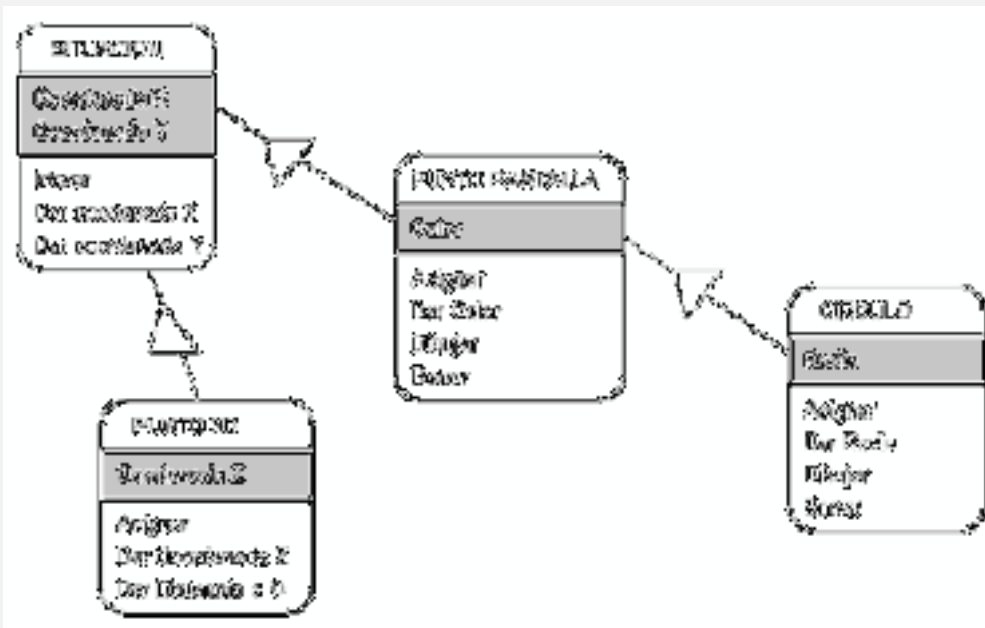
La abstracción se refiere a la capacidad de ocultar los detalles complejos de un objeto o sistema y presentar solo los aspectos relevantes y esenciales para el usuario. Esto permite simplificar el uso del objeto o sistema, reducir la complejidad y hacerlo más fácil de entender y manejar.



Abstracción

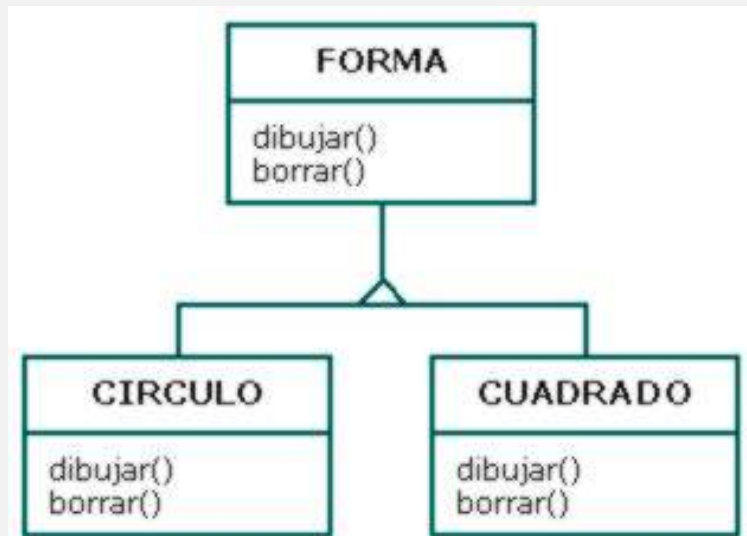
Programación orientada a objetos (POO)

6. ¿QUE ES HERENCIA Y MUESTRE UN EJEMPLO



La herencia es un mecanismo mediante el cual una clase puede heredar los atributos y métodos de otra clase. La clase que hereda se conoce como la subclase o clase derivada, mientras que la clase de la que se heredan los atributos y métodos se conoce como la superclase o clase base.

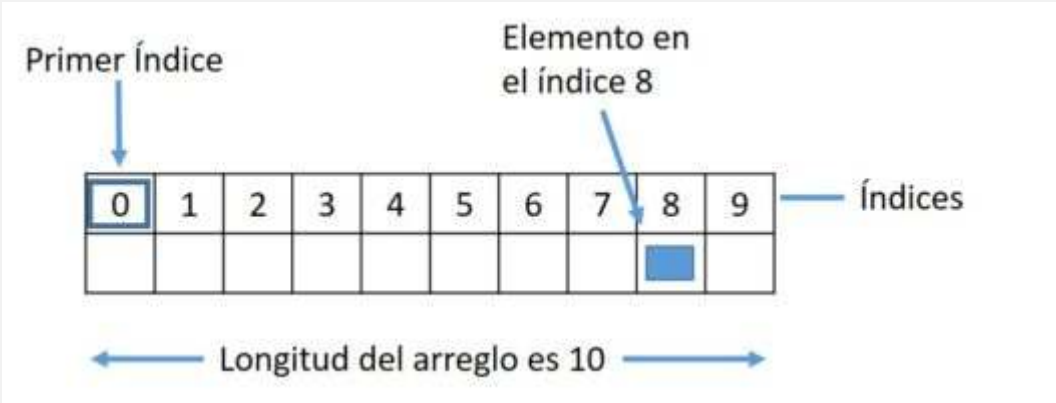
7. ¿QUÉ ES POLIMORFISMO Y MUESTRE UN EJEMPLO?



El polimorfismo se refiere a la capacidad de objetos de diferentes clases de responder al mismo mensaje o método de diferentes maneras. Esto significa que, aunque los objetos pueden tener implementaciones diferentes para un método específico, el programador puede utilizar el mismo nombre de método para interactuar con todos los objetos.

8. QUE ES UN ARRAY?

Un array (o arreglo) es una estructura de datos que almacena una colección de elementos del mismo tipo en una única variable. Los elementos de un array se organizan en una secuencia ordenada y se acceden mediante un índice o posición numérica.



9. ¿QUÉ SON LOS PAQUETES EN JAVA?

Un paquete es una forma de organizar clases relacionadas entre sí en una jerarquía de directorios. Un paquete puede contener clases, subpaquetes y archivos de recursos, y se utiliza para evitar conflictos de nombres entre clases de diferentes bibliotecas o proyectos.

10.¿CÓMO SE DEFINE UNA CLASE MAIN EN JAVA Y MUESTRA UN EJEMPLO?

En Java, la clase Main es la clase que contiene el método main, que es el punto de entrada de cualquier programa Java. El método main es el primer método que se ejecuta cuando se inicia un programa Java y es el lugar donde se inicia la ejecución del programa.

PARTE PRACTICA

II. GENERAR LA CLASE PROVINCIA.

```
17 public int getPoblacion() {
18     return poblacion;
19 }
20
21 // metodo
22 public void setPoblacion(int poblacion) {
23     this.poblacion = poblacion;
24 }
25
26 // metodo
27 public void mostrarInfo() {
28     System.out.println("Nombre: " + nombre);
29     System.out.println("Capital: " + capital);
30     System.out.println("Poblacion: " + poblacion);
31 }
```

Provincia

+ nombre: String

Provincia() => Constructor

gets() => todos los gets de la clase

sets() => todos los sets de la clase

muestraProvincia()

12.GENERAR LA CLASE DEPARTAMENTO.

Departamento

+ nombre: String
+ nroDeProvincias[]: Provincia

Departamento() => constructor
gets() => todos los gets de la clase
sets() => todos los sets de la clase
muestraDepartamento()
agregaNuevaProvincia()

```
1 import java.util.ArrayList;
2 no usages
3 public class Departamento {
4
5     4 usages
6     private String nombre;
7     3 usages
8     private String [] nroDeProvincia;
9
10    no usages
11    public Departamento (String nombre, String[] nroDeProvincia ){
12
13        this.nombre = nombre;
14        this.nroDeProvincia = nroDeProvincia;
15    }
16
17    1 usage
18    public String getNombre (){
19        return nombre;
20    }
21 }
```

I 3.GENERAR LA CLASE PAÍS.

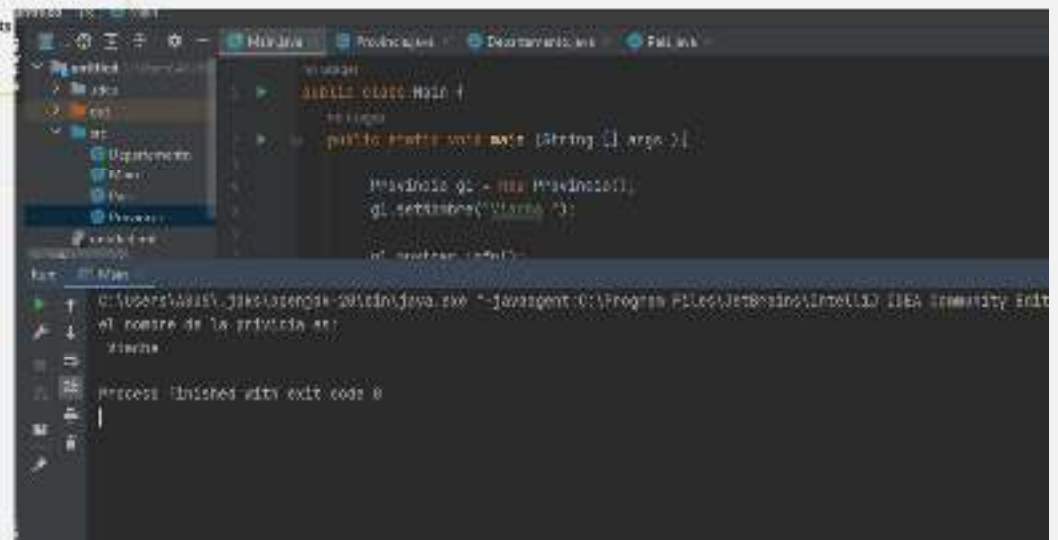
Pais
+ nombre: String + nroDepartamentos: Int + departamentos[]: Departamento
 Pais() => Constructor gets() => todos los gets de la clase sets() => todos los sets de la clase muestraPais() agregaNuevoDepartamento()

14. CREAR EL DISEÑO COMPLETO DE LAS CLASES.

Pais
+ nombre: String + nroDepartamentos: Int + departamentos[]: Departamento
Pais() => Constructor gets() => todos los gets de la clase sets() => todos los sets de la clase muestraPais() agregaNuevoDepartamento()

Departamento
+ nombre: String + nroDeProvincias[]: Provincia
Departamento() => constructor gets() => todos los gets de la clase sets() => todos los sets de la clase muestraDepartamento() agregaNuevaProvincia()

Provincia
+ nombre: String
Provincia() => Constructor gets() => todos los gets de la clase sets() => todos los sets muestraProvincia()



```
1 package main;
2
3 public class Main {
4     public static void main (String [] args) {
5
6         Provincia p1 = new Provincia();
7         p1.setNombre("Villarica");
8
9         System.out.println(p1);
10    }
11 }
12
13 Run - Main
14
15 C:\Users\Waisi_j\Desktop>java -cp .\bin\ Main
16 el nombre de la provincia es:
17 Villarica
18
19 Process finished with exit code 0
```