

# **EVALUACIÓN PROCESUAL HITO 2**

**ESTRUCTURA DE DATOS  
LUIS ALVAREZ MEDINA**

---



# MANEJO DE CONCEPTOS





01

## ¿ A que se refiere cuando se habla de POO?

Cuando hablamos de POO se refiere a la programación orientada a objetos, donde esta se enfoca en la creación de objetos los cuales generan instancias de clases las cuales están conectadas entre si para realizar una tarea.

02

## ¿ Cuales son los 4 componentes que compone POO?


Los cuatro componente son:

1. Clases.
2. Objetos.
3. Herencia.
4. Abstracción.

03

## ¿ Cuales son los pilares de POO?

Los pilares de POO son:

1. Encapsulamiento.
  2. Herencia.
  3. Polimorfismo.
  4. Abstracción.
- 

# 04

## ¿ Que es Encapsulamiento y muestre un ejemplo?

Encapsulamiento es la protección de información dentro de un objeto, donde se permite el acceso a ella mediante métodos públicos .

```
public class Ejemplo {  
  
    private String paraleo;  
    private String[] nombres;  
  
    /* CONSTRUCTOR*/  
    public Ejemplo (String paraleo, String[] nombres){  
  
        this.paraleo = paraleo;  
        this.nombres = nombres;  
    }  
  
    public void mostraNombres(){  
        for (int i=0;i<this.nombres.length;i++){  
            System.out.println(nombres[i]+"");  
        }  
        System.out.println();  
    }  
}
```

Por ejemplo en este caso estamos encapsulando los tipos de datos (String paraleo, String[] nombres), protegiendo información, donde estos datos son alterados mediante el método mostrarNombres () y donde el usuario únicamente podrá alterar dichos será mediante este método.

## 05

## ¿ Que es abstracción y muestre un ejemplo?

Es la capacidad de ocultar los detalles de implementación de una clase o método, donde solo se muestra la interfaz necesaria para utilizarlas.

```
public class Ejemplo {  
  
    private String paraleo;  
    private String[] nombres;  
  
    public Ejemplo (String paraleo, String[] nombres){  
  
        this.paraleo = paraleo;  
        this.nombres = nombres;  
    }  
  
    public void mostraNombres(){  
        for (int i=0;i<this.nombres.length;i++){  
            System.out.println(nombres[i]+"");  
        }  
        System.out.println();  
    }  
}
```

```
public class Main {  
    public static void main(String[] args) {  
  
        String [] NombresEDD = new String[5];  
        NombresEDD[0] = "Ana";  
        NombresEDD[1] = "Juan";  
        NombresEDD[2] = "Pepito";  
        NombresEDD[3] = "Carla";  
        NombresEDD[4] = "Freddy";  
  
        Ejemplo edd =new Ejemplo("EDD",NombresEDD);  
        edd.mostraNombres();  
    }  
}
```

En este ejemplo podemos ver como se usa la abstracción, podemos ver que en la clase ejemplos, el método mostrarNombres() tiene procesos, pero esos procesos no se muestran al utilizarlos en la clase main donde solo ingresamos el método mostrarNombres() sin ningún proceso, y como resultado mostrara 5 nombres.

## 06

## ¿ Que es herencia y muestre un ejemplo?

La herencia en la POO es fundamental ya que esta permite a las clases compartir sus atributos o sus métodos ya existentes.

```
public class Animales {  
    public void sonidoAnimales(){  
        System.out.println("Los animales hacen sonidos");  
    }  
}
```

```
public class Gato extends Animales {  
    public void sonidoAnimales2(){  
        sonidoAnimales();  
    }  
}
```

```
public class Perro extends Animales {  
    public void sonidoAnimales3(){  
        sonidoAnimales();  
    }  
}
```

```
public class Main {  
    public static void main(String[] args) {  
  
        Animales an1 = new Animales();  
        Gato an2 = new Gato();  
        Perro an3 = new Perro();  
  
        an1.sonidoAnimales();  
        an2.sonidoAnimales2();  
        an3.sonidoAnimales3();  
    }  
}
```

En este ejemplo podemos ver como el padre (Animales) hereda a sus hijos (Gato, Perro), una método donde al utilizarlos en main y crear sus instancias podemos ver que si heredo el mismo método a sus hijos.

```
C:\Users\USUARIO\.jdk\openjdk-11.0.15\bin\java.exe  
Los animales hacen sonidos  
Los animales hacen sonidos  
Los animales hacen sonidos
```

## 07

## ¿ Que es polimorfismo y muestre un ejemplo?

El polimorfismo permite que un objeto de una clase pueda tomar diferentes formas y comportarse de maneras diferentes según el contexto en el que se utilice.

```
public class Animales {  
    public void sonidoAnimales(){  
        System.out.println("Los animales hacen sonidos");  
    }  
}
```

```
public class Gato extends Animales {  
    public void sonidoAnimales(){  
        System.out.println("El gato hace miau");  
    }  
}
```

```
public class Perro extends Animales {  
    public void sonidoAnimales(){  
        System.out.println("El Perro hace guaf ");  
    }  
}
```

```
public class Main {  
    public static void main(String[] args) {  
  
        Animales an1 = new Animales();  
        Animales an2 = new Gato();  
        Animales an3 = new Perro();  
  
        an1.sonidoAnimales();  
        an2.sonidoAnimales();  
        an3.sonidoAnimales();  
  
    }  
}
```

En este ejemplo podemos ver como el método sonidoAnimales() que el padre heredo a sus hijos es el mismo, pero están siendo utilizadas de diferentes formas.

```
C:\Users\USUARIO\.jdk\openjdk-19.0.2\bin\java.exe "-javaa  
Los animales hacen sonidos  
El gato hace miau  
El Perro hace guaf  
  
Process finished with exit code 0
```



08

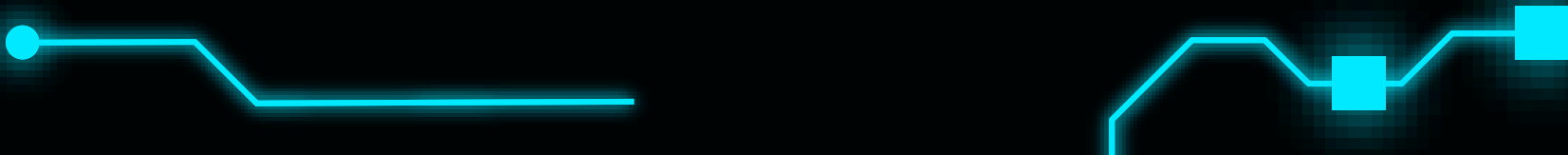
## ¿ Que es un ARRAY?

Un ARRAY es una estructura de datos que permite almacenar conjunto de elementos del mismo tipo en una única variable, donde este se compone en un conjunto de posiciones de forma consecutiva, empezando desde el 0, en donde cada posición del ARRAY puede contener un valor del mismo tipo que los demás ARRAYS.

09

## ¿ Que son los paquetes en java?

Los paquetes son una herramienta importante para organizar y estructurar el código en Java, facilitando su mantenimiento y evitando conflictos de nombre entre clases.





# 10

## ¿ Como se define la clase main en java y muestre un ejemplo?

Esta se define como el punto de entrada de una aplicación java. Es el punto donde comienza a ejecutarse la aplicación y por eso esta debe de ser PUBLIC (accesible fuera de la clase), STATIC (puede ejecutar sin una instancia de la clase).

```
public class Main {  
    public static void main(String[] args) {  
  
        System.out.println("HOLA MUNDO");  
    }  
}
```

```
C:\Users\USUARIO\.jdk\op  
HOLA MUNDO
```



# Parte Pratica

## 11

## Generar la clase Provincia

```
package ProcesualHlto2;

public class Provincia {

    private String nombre;
    public Provincia(){
        this.nombre=" ";
    }
    public String getNombre(){
        return nombre;
    }
    public void setNombre(String nombre){
        this.nombre = nombre;
    }
    public void muestraProvincia(){
        System.out.println(getNombre());
    }
}
```

```
public class Main {
    public static void main(String[] args) {

        Provincia prv = new Provincia();
        prv.setNombre("Pacajes");
        prv.muestraProvincia();
    }
}
```



## 12

## Generar la clase Departamento.

```
public class Departamento {

    private String nombre;
    private Provincia[] nroProvincia;
    public Departamento() {

        Provincia[] nroProvincia = new Provincia[0];
        this.nombre = " ";
        this.nroProvincia = nroProvincia;
    }
    public String getNombre1(){

        return nombre;
    }
    public Provincia[] getNroDeProvincias(){
        return nroProvincia;
    }
    public void setNombre(String nombre){
        this.nombre = nombre;
    }
}
```

```
public void setNroDeProvincias(Provincia[]
nroDeProvincias)
{
    this.nroProvincia = nroDeProvincias;
}
public void muestraDepartamento(int
especifico){

    System.out.println(getNombre1());

    for(int i=especifico;
i<this.getNroDeProvincias().length;i++){
        System.out.print(" Provincia: ");

        this.getNroDeProvincias()[i].muestraProvincia(
);
    }
}
```

```
public class Main {
    public static void main(String[] args) {
        Departamento dep = new Departamento();
```

```
        dep.setNombre("La Paz");
```

```
        Provincia[] nroDEProvincias = new Provincia[2];
```

```
        nroDEProvincias[0] = new Provincia();
        nroDEProvincias[1] = new Provincia();
```

```
        nroDEProvincias[0].setNombre("Pacajes");
        nroDEProvincias[1].setNombre("Los Andes");
```

```
        dep.setNroDeProvincias(nroDEProvincias);
        dep.muestraDepartamento(0);
    }
}
```

Departamento		
 	Departamento()	
 	nombre	String
 	nroProvincia	Provincia []
 	setNombre(String)	void
 	getNombre1()	String
 	getNroDeProvincias()	Provincia []
 	muestraDepartamento(int)	void
 	setNroDeProvincias(Provincia [])	void
 	agregaNuevaProvincia (String)	void

# 13

## Generar la clase País.

```
public class Pais {  
  
    private String nombre;  
    private int nroDEpartamentos;  
    private Departamento[] departamentos;  
  
    public Pais(){  
        Departamento[] nroDepartamentos = new Departamento[0];  
        this.nombre = " ";  
        this.nroDEpartamentos = 0;  
        this.departamentos = nroDepartamentos;  
    }  
    public String getNombre(){  
  
        return nombre;  
    }  
    public int getNroDEpartamentos(){  
        return nroDEpartamentos;  
    }  
    public Departamento[] getDepartamentos(){  
  
        return departamentos;  
    }  
}
```

```
public void setNombre(String nombre){  
    this.nombre= nombre;  
}  
public void setNroDEpartamentos(int nroDEpartamentos){  
    this.nroDEpartamentos=nroDEpartamentos;  
}  
public void setDepartamentos(Departamento[] departamentos){  
  
    this.departamentos=departamentos;  
}  
public void muestraPais(int especifico){  
  
    for(int i=especifico; i < this.getNroDEpartamentos() ; i++){  
        System.out.println(" DEPARTAMENTO:" +this.getDepartamentos()[i].getNombre1());  
    }  
}
```

# 13

## Generar la clase País.

```
public class Main {
    public static void main(String[] args) {

        Pais ps = new Pais();
        ps.setNombre("BOLIVIA");
        ps.setNroDEpartamentos(3);

        Departamento[] departamentos = new Departamento[3];
        departamentos[0] = new Departamento();
        departamentos[1] = new Departamento();
        departamentos[2] = new Departamento();

        departamentos[0].setNombre("La Paz");
        departamentos[1].setNombre("Cocha Bamba");
        departamentos[2].setNombre("Santa Cruz");

        ps.setDepartamentos(departamentos);
        ps.muestraPais(0);
    }
}
```

Pais		
m	Pais()	
f	nroDEpartamentos	int
f	nombre	String
f	departamentos	Departamento[]
m	agregarNuevoDepartamento(String)	void
m	setNroDEpartamentos(int)	void
m	muestraPais(int)	void
m	getNroDEpartamentos()	int
m	getDepartamentos()	Departamento[]
m	getNombre()	String
m	setNombre(String)	void
m	setDepartamentos(Departamento[])	void

## 14

# Crear el diseño completo de las clases.

```
public class Provincia {

    private String nombre;
    public Provincia(){
        this.nombre=" ";
    }
    public String getNombre(){
        return nombre;
    }
    public void setNombre(String nombre){
        this.nombre = nombre;
    }
    public void muestraProvincia(){
        System.out.println(getNombre());
    }
}
```

```
public class Departamento {

    private String nombre;
    private Provincia[] nroProvincia;
    public Departamento() {

        Provincia[] nroProvincia = new Provincia[0];
        this.nombre = " ";
        this.nroProvincia = nroProvincia;

    }
    public String getNombre1(){

        return nombre;
    }
    public Provincia[] getNroDeProvincias(){

        return nroProvincia;
    }
    public void setNombre(String nombre){

        this.nombre = nombre;
    }
}
```

```
public void setNroDeProvincias(Provincia[] nroDeProvincias)
{
    this.nroProvincia = nroDeProvincias;
}
public void muestraDepartamento(int especifico){

    System.out.println(getNombre1());

    for(int i=especifico; i<this.getNroDeProvincias().length;i++){
        System.out.print(" Provincia: ");
        this.getNroDeProvincias()[i].muestraProvincia();
    }
}
public void agregaNuevaProvincia(String newProvincia){

    Provincia[] oriProvincia = this.getNroDeProvincias();
    Provincia[] nuevaProvincia = new Provincia[ oriProvincia.length+1];

    nuevaProvincia[oriProvincia.length] = new Provincia();
    nuevaProvincia[oriProvincia.length].setNombre(newProvincia);

    for(int i=0; i<this.getNroDeProvincias().length;i++){

        nuevaProvincia[i] = new Provincia();
        nuevaProvincia[i] = oriProvincia[i];
    }
    setNroDeProvincias(nuevaProvincia);
}
}
```

## 14 Crear el diseño completo de las clases.

```
public class Pais {

    private String nombre;
    private int nroDEpartamentos;
    private Departamento[] departamentos;

    public Pais(){
        Departamento[] nroDepartamentos = new Departamento[0];
        this.nombre = " ";
        this.nroDEpartamentos = 0;
        this.departamentos = nroDepartamentos;
    }
    public String getNombre(){

        return nombre;
    }
    public int getNroDEpartamentos(){
        return nroDEpartamentos;
    }
    public Departamento[] getDepartamentos(){

        return departamentos;
    }
}
```

```
public void setNombre(String nombre){
    this.nombre= nombre;
}
public void setNroDEpartamentos(int nroDEpartamentos){
    this.nroDEpartamentos=nroDEpartamentos;
}
public void setDepartamentos(Departamento[] departamentos){

    this.departamentos=departamentos;
}
public void muestraPais(int especifico){

    for(int i=especifico; i < this.getNroDEpartamentos() ; i++){
        System.out.println(" DEPARTAMENTO:" +this.getDepartamentos()[i].getNombre1());
    }
}
public void agregarNuevoDepartamento(String newDepartamento ){

    Departamento[] oriDepartamento = this.getDepartamentos();
    Departamento[] nuevoDepartamento = new Departamento[oriDepartamento.length+1];

    nuevoDepartamento[oriDepartamento.length] = new Departamento();
    nuevoDepartamento[oriDepartamento.length].setNombre(newDepartamento);

    setNroDEpartamentos(nuevoDepartamento.length);

    for(int i=0; i<oriDepartamento.length;i++){
        nuevoDepartamento[i] = new Departamento();
        nuevoDepartamento[i] = oriDepartamento[i];
    }
    setDepartamentos(nuevoDepartamento);
}
}
```



## 14

## Crear el diseño completo de las clases.

Pais		
m	Pais()	
f	nroDepartamentos	int
f	nombre	String
f	departamentos	Departamento []
m	agregarNuevoDepartamento (String)	void
m	setNroDepartamentos (int)	void
m	muestraPais (int)	void
m	getNroDepartamentos ()	int
m	getDepartamentos ()	Departamento []
m	getNombre ()	String
m	setNombre (String)	void
m	setDepartamentos (Departamento [])	void

Departamento		
m	Departamento ()	
f	nombre	String
f	nroProvincia	Provincia []
m	setNombre (String)	void
m	getNombre1 ()	String
m	getNroDeProvincias ()	Provincia []
m	muestraDepartamento (int)	void
m	setNroDeProvincias (Provincia [])	void
m	agregaNuevaProvincia (String)	void

Provincia		
m	Provincia ()	
f	nombre	String
m	muestraProvincia ()	void
m	getNombre ()	String
m	setNombre (String)	void

# 14

## Crear el diseño completo de las clases.

```
public class Main {
    public static void main(String[] args) {

int numPais = 1, numDepartamento=3,numProvincia = 2;
int c = 0;

Pais ps = new Pais();
Departamento dep = new Departamento();

ps.setNombre("BOLIVIA");
System.out.println(" PAIS: "+ ps.getNombre());
ps.setNroDEpartamentos(3);
System.out.println(" Numero de Departamentos: "+ ps.getNroDEpartamentos());
for(int i = 0; i<numPais; i++){

    for(int j=0; j< numDepartamento; j++) {

        switch (j) {

            case 0:
                ps.agregarNuevoDepartamento("La Paz");
                dep.agregaNuevaProvincia("Pacajes");
                dep.agregaNuevaProvincia("Los Andes");

                ps.muestraPais(j);
                dep.muestraDepartamento(c);
                break;
```

```
            case 1:
                ps.agregarNuevoDepartamento("Cochabamba");
                dep.agregaNuevaProvincia("Bolívar");
                dep.agregaNuevaProvincia("Carrasco");

                ps.muestraPais(j);
                dep.muestraDepartamento(c);
                break;

            case 2:
                ps.agregarNuevoDepartamento("Santa Cruz");
                dep.agregaNuevaProvincia("Chiquitos");
                dep.agregaNuevaProvincia("Cordillera");

                ps.muestraPais(j);
                dep.muestraDepartamento(c);
                break;

        }
        c = c+numProvincia;
    }
}
}
```

# 14

## Crear el diseño completo de las clases.

```
C:\Users\USUARIO\.jdk\openjdk-19.0.2\bin\java.exe "-javaagent:D:\Intelli
PAIS: BOLIVIA
Numero de Departamentos: 3
DEPARTAMENTO:La Paz
Provincia: Pacajes
Provincia: Los Andes
DEPARTAMENTO:Cochabamba
Provincia: Bolivar
Provincia: Carrasco
DEPARTAMENTO:Santa Cruz
Provincia: Chiquitos
Provincia: Cordillera
```



**GRACIAS POR SU  
ATENCION!!**