UNIFRANZ
Internacionalízate

UNIFRANZ
Internacionalízate
INNOVACIÓN
EN EDUCACIÓN
Blended Learning

1. ¿ A que se refiere cuando se habla de POO?

R.- es un paradigma de programación, es decir un modelo o un estilo de programación que nos de unas guías sobre como trabajar con el. Se basa en el concepto de clases y objetos

2. ¿Cuales son los 4 componentes que componen POO?

R.- CLASE, PROPIEDAD, METODOS, OBJETOS

3. ¿Cuales son Los pilares de POO?

R.- abstracción, encapsulamiento, herencia, polimorfismo

4. ¿Qué es Encapsulamiento y muestre un ejemplo?

R.- Es el proceso de almacenar en una misma sección los elementos de una abstracción que comportamiento; sirve para separar el interfaz contractual de una abstracción y su implantación.

. ¿Qué es Abstracción y muestra un ejemplo?

R.- La abstracción consiste en seleccionar datos de un conjunto mas grande para mostrar solo los detalles relevantes del objeto ayuda a reducir la complejidad y el esfuerzo de programación. En java, la abstracción se logra usando clases e interfaces abstractas. Es uno de los conceptos mas importantes

```java
package Hito_2_Practica;

no usages
public class Provincia {
    3 usages
    private String nombre;
     no usages
     public Provincia(){
         this.nombre="";
     }

    1 usage
    public String getNombre() {
        return nombre;
    }

    no usages
    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
     no usages
     public void mostrarNombre(){
         System.out.println("Mostrar nombre de provincia: ");
         System.out.println("Nombre de provincia: " + getNombre());
     }
}
```

- 6. ¿Qué es Herencia y muestre un ejemplo?

- R.- la herencia permite que se pueda definir nuevas clases basadas de unas ya existentes a fin de reutilizar el código, generando así una jerarquía de clases dentro de una aplicación si una clase deriva de otra, esta hereda sus atributos y métodos y puede añadir nuevos atributos metodos o redefinir los heredados

```
class DosDimensiones{
double base;
double altura;
void mostrarDimension(){
System.out.println("La base y altura es:
"+base+" y "+altura);
  }
}
```

7. ¿Que es Polimorfismo y muestra un ejemplo?

- R.- es la capacidad que tienen ciertos lenguajes para hacer que, al enviar el mismo mensaje desde distintos objetos, cada uno de esos objetos pueda responder a ese mensaje de forma distinta.

```
class Animal {
    public void makeSound() {
        System.out.println("Grr...");
    }
};
class Cat extends Animal {
    public void makeSound() {
        System.out.println("Meow");
    }
}
class Dog extends Animal {
    public void makeSound() {
        System.out.println("Woof");
    }
}
```

- ## 8. ¿Que es un ARRAY?

R.- los arrays se utilizan para agrupar objetos del mismo tipo. De esta manera, Podemos referirnos a este grupo con el mismo nombre. Pero te lo puedes encontrar con muchos nombres:

- Arreglos
- Vectores
- Matrices

9. ¿Que son los paquetes en java?

R.- Los paquetes son el mecanismo que usa Java para facilitar la modularidad del código. Un paquete puede contener una o más definiciones de interfaces y clases, distribuyéndose habitualmente como un archivo. Para utilizar los elementos de un paquete es necesario importar este en el módulo de código en curso, usando para ello la sentencia IMPORT

- **10. ¿Que son los paquetes en java?**

R.- El método main() acepta un parámetro (y solo uno): una matriz de tipo String. Esta matriz recoge los valores que introduzcas a la hora de ejecutar tu aplicación desde la línea de comandos.

Da igual el valor que introduzcas; el JRE lo transformará a String.

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

        Scanner lectura=new Scanner(System.in);
        int nPais =1;
        Pais[] pais = new Pais[nPais];
```

- Generar la clase Provincia Crear una clase MAIN ■ Crear todos los gets y sets de la clase. ■ El constructor no recibe parámetros. ■ Crear una instancia de la clase Provincia. ■ Mostrar los datos de una provincia.

```java
package Hito_2_Practica;

no usages
public class Provincia {
    3 usages
    private String nombre;
     no usages
     public Provincia(){
         this.nombre="";
     }

    1 usage
    public String getNombre() {
        return nombre;
    }

    no usages
    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
     no usages
     public void mostrarNombre(){
         System.out.println("Mostrar nombre de provincia: ");
         System.out.println("Nombre de provincia: " + getNombre());
```

- Generar la clase Departamento. Crear una clase MAIN (Utilizar el MAIN del anterior ejercicio) ■ Crear todos los gets y sets de la clase. ■ El constructor no recibe parámetros. ■ Crear una instancia de la clase Departamento. ■ Omitir el método agregaNuevaProvincia () ■ Mostrar los datos de los departamentos.

```java
public class Departamento {
    4 usages
    private String nombre;
    5 usages
    private Provincia[] nroProvincias;

    no usages
    public Departamento(){
        this.nombre="";
        this.nroProvincias = new Provincia[0];
    }
    no usages
    public Departamento(String nombre, Provincia[] nroProvincias){
        this.nombre= nombre;
        this.nroProvincias =nroProvincias;
    }
    // Cree un metodo que inrese provicias
    no usages
    public void agregarNuebaProvincia(Provincia[] nuevoNroProvincias){
        this.nroProvincias = nuevoNroProvincias;
    }

    1 usage
    public String getNombre() {
        return nombre;
    }

    no usages
    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    2 usages
    public Provincia[] getNroProvincias() {
        return nroProvincias;
    }
    no usages
    public void setNroProvincias(Provincia[] nroProvincias) {
        this.nroProvincias = nroProvincias;
    }

    //mostrar
    no usages
    public void mostrarDepartamento(){
        System.out.println("---DATOS DE DEPARTAMENTO---");
        System.out.println("Nombre de departamento: "+ getNombre());

        for (int i =0; i< this.getNroProvincias().length;i++){
            this.getNroProvincias()[i].mostrarProvinvia();
        }
    }
```

- Generar la clase País. Crear una clase MAIN (Utilizar el MAIN del anterior ejercicio) ■ Crear una instancia de la clase País ■ El constructor no recibe parámetros. ■ Crear una instancia de la clase Departamento. ■ Omitir el método agregaNuevoDepartamento() ■ Mostrar los datos del País.

```java
public class Pais {
    4 usages
    private String nombre;
    4 usages
    private int nroDeDepartamentos;
    5 usages
    private Departamento[] departamentos1;

    no usages
    public Pais(){
        this.nombre="";
        this.nroDeDepartamentos=0;
        this.departamentos1 = new Departamento[0];
    }
    no usages
    public Pais(String nombre, int nroDeDepartamentos, Departamento[] departamentos1){
        this.nombre = nombre;
        this.nroDeDepartamentos = nroDeDepartamentos;
        this.departamentos1 = departamentos1;
    }
    no usages
    public void agregarNuevoDepartamento(Departamento[] nuevoDepartamentos1){
        this.departamentos1 = nuevoDepartamentos1;
    }

    1 usage
    public String getNombre() {
        return nombre;
    }

    no usages
    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    1 usage
    public int getNroDeDepartamentos() {
        return nroDeDepartamentos;
    }

    no usages
    public void setNroDeDepartamentos(int nroDeDepartamentos) {
        this.nroDeDepartamentos = nroDeDepartamentos;
    }

    2 usages
    public Departamento[] getDepartamentos1() {
        return departamentos1;
    }

    no usages
    public void setDepartamentos1(Departamento[] departamentos1) {
        this.departamentos1 = departamentos1;
    }
    no usages
    public void mostrarPais(){
        System.out.println("---MOSTRAR DATOS DE PAIS---");
        System.out.println("Nombre de departamentos: "+ getNombre());
        System.out.println("Numero de departamentos: "+ getNroDeDepartamentos());

        for (int i =0; i<this.getDepartamentos1().length;i++){
            this.getDepartamentos1()[i].mostrarDepartamento();
        }
    }
```

- Crear el diseño completo de las clases.
- Crear todos gets y sets de cada clase.
- ○ Implementar los métodos agregarNuevoDepartamento(), agregarNuevaProvincia(), es decir todos los métodos.
- ○ El método agregarNuevoDepartamento permite ingresar un nuevo departamento a un país.
- ○ El método agregarNuevaProvincia permite ingresar una nueva provincia a un departamento. ○ La clase Main debe mostrar lo siguiente:
- ■ Crear el PAÍS Bolivia
- ■ Al país Bolivia agregarle 3 departamentos.
- ■ Cada departamento deberá tener 2 provincias.

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

    Scanner lectura =new Scanner(System.in);
    int nPais =1;
    Pais[] pais = new Pais[nPais];

    for(int i=0; i<nPais;i++){
        System.out.println("Ingresar pais "+ (i+1)+": ");
        String nombrePais = lectura.nextLine();

        int nDepartamento =3;
        Departamento[] departamentos = new Departamento[nDepartamento];

        for (int  j=0;j<nDepartamento;j++){
            System.out.println("Ingresar departamento "+ (j+1)+": ");
            String nombreDepartamento = lectura.nextLine();

            int nProvincia=2;
            Provincia[] provincias = new Provincia[nProvincia];

            for(int k=0;k<nProvincia;k++){
                System.out.println("Ingresar Provincia "+ (k+1)+": ");
                String nombreProvinvia = lectura.nextLine();

                Provincia pr1 = new Provincia();
                pr1.setNombre(nombreProvinvia);
                provincias[k] = pr1;
            }
            Departamento dep1 = new Departamento();
            dep1.setNombre(nombreDepartamento);
            dep1.setNroProvincias(provincias);
            departamentos[j]=dep1;

            dep1.mostrarDepartamento();
        }
    }
    Pais pais1 = new Pais();
    pais1.setNombre(nombrePais);
    pais1.setNroDeDepartamentos(nDepartamento);
    pais1.setDepartamentos1(departamentos);
    pais[i] = pais1;

    pais1.mostrarPais();
}
```