DEFENSA HITO 2

ESTRUCTURA DE DATOS

ESTUDIANTE:
JOSIAS JONATHAN LEON LUIS

INGENIERIA DE SISTEMAS

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

La Programación Orientada a Objetos (POO) es un paradigma de programación, es decir, un modelo o un estilo de programación que nos da unas guías sobre cómo trabajar con él. Se basa en el concepto de clases y objetos.

2. ¿Cuáles son los 4 componen que componen POO?

1. Clase.
2. Propiedad.
3. Métodos.
4. Objetos.

3.¿Cuáles son los pilares de POO?

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

4. ¿Que es Encapsulamiento y muestre un ejemplo?

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

```java
public class MiClase{

    private int tipo;

        public void setTipo(int t) {

            tipo = t;

        }

    public int getTipo() {

        return tipo;

    }

}

class AccesoIndirecto {

    public static void main(String[] args) {

        MiClase mc = new MiClase();
        mc.setTipo(5);
        System.out.println("El tipo es:" + mc.getTipo());

    }

}
```

Contains Here

You can simply impress your audience and add a unique zing

5. ¿Que es Abstracción y muestra un ejemplo?

La **abstracción** consiste en seleccionar datos de un conjunto más 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 más importantes .

```java
public class UniversityStudent {
    int id;
    String name;
    String gender;
    String university;
    String career;
    int numSubjects;
    public UniversityStudent(int id, String name, String gender,
        String university, String  career, int numSubjects) {
        this.id = id;
        this.name = name;
        this.gender = gender;
        this.university = university;
        this.career = career;
        this.numSubjects = numSubjects;
    }
    void inscribeSubjects() {
      // TODO: implement
    }
    void cancelSubjects() {
       // TODO: implement
    }
    void consultRatings() {
       // TODO: implement
    }
}
```

6. ¿Que es Herencia y muestre un ejemplo?

La **herencia permite** que se puedan 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, métodos o redefinir los heredados.

```
class Vehiculo
{
    public decimal VelocidadMaxima { get; set; }
    public int NumeroRuedas { get; set; }
    public string Marca { get; set; }
    public string Modelo { get; set; }

}

class Moto : Vehiculo
{
    public int Cilindrada { get; set; }
}

class Coche : Vehiculo
{
    public string Traccion { get; set; }
}
```

Content Here

You can simply impress your audience and add a unique zing

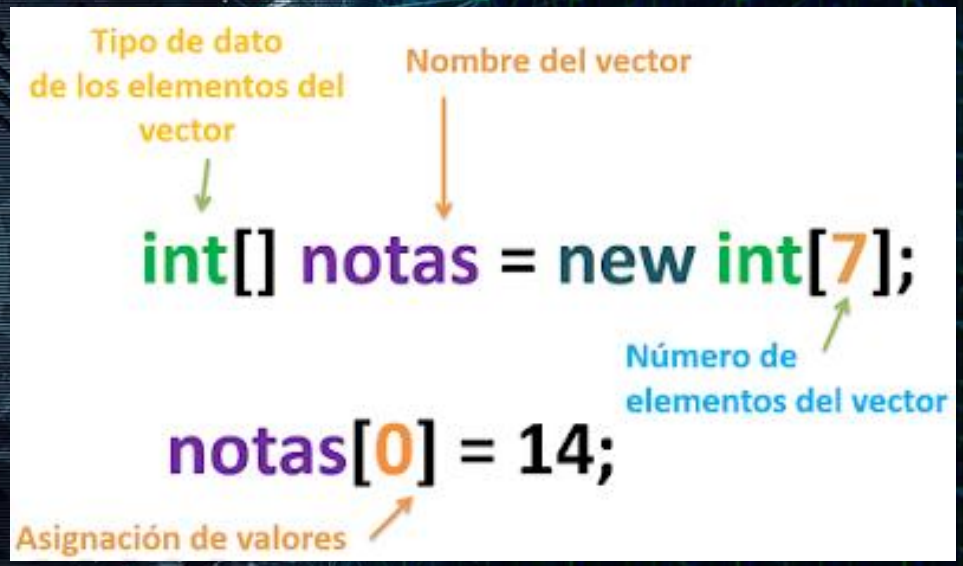
7. ¿Que es Polimorfismo y muestra un ejemplo?

Polimorfismo (en POO) es la capacidad que tienen ciertos lenguajes para hacer que, al enviar el mismo mensaje (o, en otras palabras, invocar al mismo método) desde distintos objetos, cada uno de esos objetos pueda responder a ese mensaje (o a esa invocación) de forma distinta.

```
// No se puede crear un objeto de una clase abstracta
SeleccionFutbol casillas = new SeleccionFutbol();
Cannot instantiate the type SeleccionFutbol
SeleccionFutbol delBosque = new Entrenador();

SeleccionFutbol iniesta = new Futbolista();

SeleccionFutbol raulMartinez = new Masajista();
```

8. ¿Qué es un ARRAY?

Cuando hablamos de programación orientada a objetos, **una array se considera un objeto**. Eso quiere decir que si la declaramos tal y como hemos hecho, no estamos creando el objeto, sino que crea una referencia para poder utilizarlo. Para inicializar una array solemos utilizar la palabra reservada new .



9. ¿Qué son los paquetes de Java?

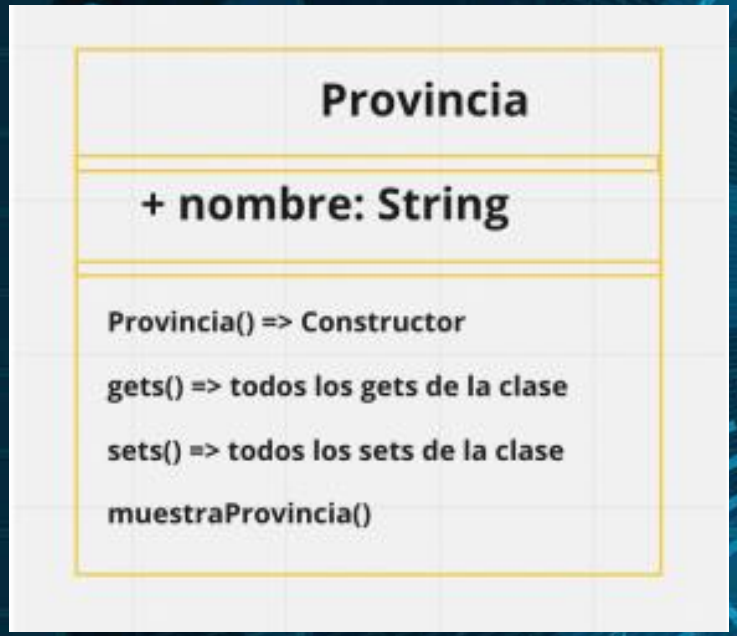
Es un contenedor de clases que permite agrupar las distintas partes de un programa y que por lo general tiene una funcionalidad y elementos comunes

10. ¿Cómo se define una clase main en JAVA y muestre un ejemplo?

El método main es el punto de entrada de un programa ejecutable, es donde se inicia y finaliza el control del programa.

Práctica

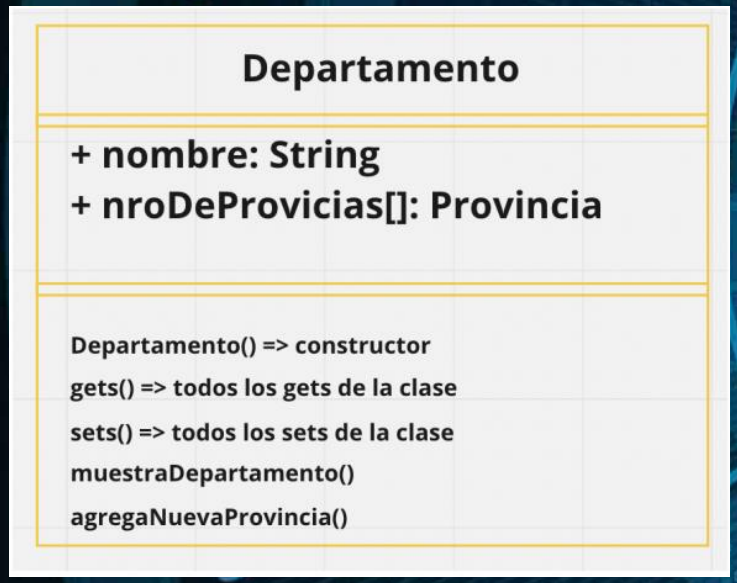
11. Generar la clase Provincia




```java
public class Provincia {
    4 usages
    private String NombreProvincia;

    public Provincia (String NombreProvincia) {
        this.NombreProvincia = NombreProvincia;
    }
    2 usages
    public Provincia() {
        this.NombreProvincia = "";
    }
    2 usages
    public void setNombreProvincia(String NuevoNombre) {
        NombreProvincia = NuevoNombre;
    }
    1 usage
    public String getNombreProvincia() {
        return this.NombreProvincia;
    }
    1 usage
    public void mostrarProvincia() {
        System.out.println("Mostrando datos de la provincia");
        System.out.println("Nombre Provincia: " + this.getNombreProvincia());
        System.out.println("\n");
    }

}
```

Practica

12. Generar la clase Departamento

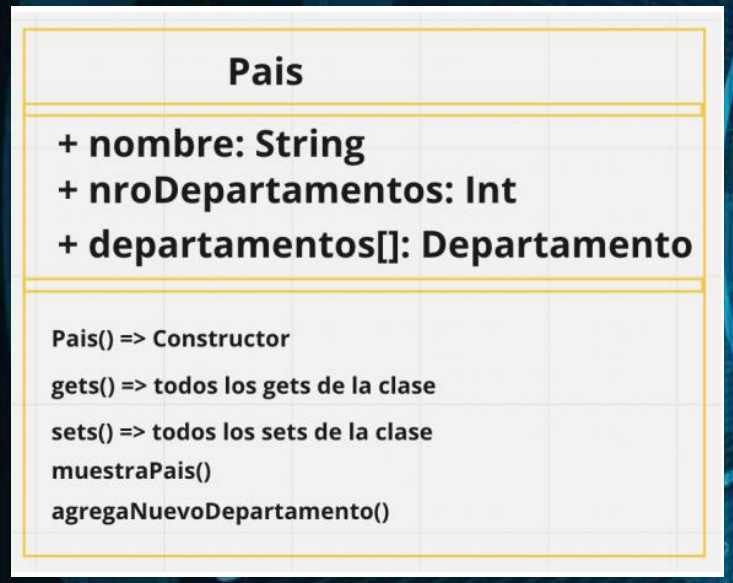



```java
public class Departamento {
    3 usages
    private String nombreDepartamento;
    5 usages
    private int noProvincias;
    3 usages
    private Provincia[] provincias;

    public Departamento(String nombreDepartamento, int noProvincias, Provincia[] provincias) {
        this.nombreDepartamento = nombreDepartamento;
        this.noProvincias = noProvincias;
        this.provincias = provincias;
    }
    2 usages
    public Departamento () {}
    1 usage
    public String getNombreDepartamento() {
        return this.nombreDepartamento;
    }
    1 usage
    public Provincia[] getProvincias() {
        return this.provincias;
    }

    public int getNoProvincias() {
        return noProvincias;
    }

    1 usage
    public void setNoProvincias(int noProvincias) {
        this.noProvincias = noProvincias;
    }

    2 usages
    public void setNombreDepartamento(String nombreDepartamento) {
        this.nombreDepartamento = nombreDepartamento;
    }
```

```java
    public Provincia[] getProvincias() {
        return this.provincias;
    }

    public int getNoProvincias() {
        return noProvincias;
    }

    1 usage
    public void setNoProvincias(int noProvincias) {
        this.noProvincias = noProvincias;
    }

    2 usages
    public void setNombreDepartamento(String nombreDepartamento) {
        this.nombreDepartamento = nombreDepartamento;
    }

    1 usage
    public void setProvincias(Provincia[] provincias) {
        this.provincias = provincias;
    }

    1 usage
    public void mostrarDepartamento() {
        System.out.println("\nMOSTRANDO DATOS DEL DEPARTAMENTO");
        System.out.println("Nombre Departamento: " + this.getNombreDepartamento());
        System.out.println("No Provincias: " + this.noProvincias);
        for (int i=0; i < this.noProvincias; i++) {
            this.getProvincias()[i].mostrarProvincia();
        }
    }
}
```

Practica

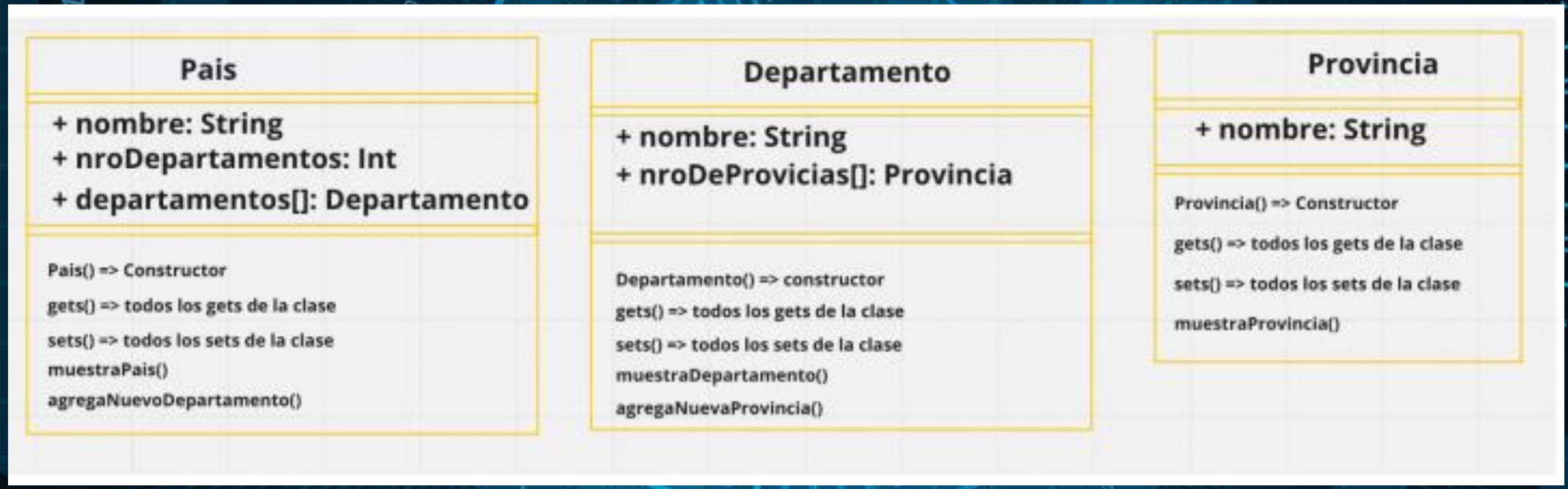
13. Generar la clase Pais




```java
public class Pais {
    3 usages
    private String nombrePais;
    4 usages
    private int noDepartamentos;
    3 usages
    private Departamento[] departamentos;

    1 usage
    public Pais(String nombrePais,int noDepartamentos, Departamento[] departamentos) {
        this.nombrePais = nombrePais;
        this.noDepartamentos = noDepartamentos;
        this.departamentos = departamentos;
    }
    public Pais() {}

    1 usage
    public String getNombrePais() {
        return this.nombrePais;
    }
    1 usage
    public Departamento[] getDepartamentos() {
        return this.departamentos;
    }
    public int getNoDepartamentos() {
        return noDepartamentos;
    }

    public void setNoDepartamentos(int noDepartamentos) {
        this.noDepartamentos = noDepartamentos;
    }

    public void setNombrePais(String nombrePais) {
        this.nombrePais = nombrePais;
    }

    public void setDepartamentos(Departamento[] departamentos) {
        this.departamentos = departamentos;
    }

    public void setNombrePais(String nombrePais) {
        this.nombrePais = nombrePais;
    }

    public void setDepartamentos(Departamento[] departamentos) {
        this.departamentos = departamentos;
    }

    1 usage
    public void mostrarPais() {
        System.out.println("\nMOSTRANDO DATOS DEL PAIS");
        System.out.println("Nombre Pais: " + this.getNombrePais());
        for (int i = 0; i < this.noDepartamentos; i++) {
            this.getDepartamentos()[i].mostrarDepartamento();
        }
    }
}
```

Practica

14. Crear el diseño completo de las clases



Practica

14. Crear el diseño completo de las clases

```java
import java.util.Scanner;

public class Main {
    public static void main(String [] args) {
        Scanner leer = new Scanner(System.in);

        //Pregunta 4

        String nombreProvincia;
        int i, nProvincias;
        nProvincias = 2;

        String nombreDepartamento;
        int j, nDepartamentos = 2;

        Departamento[] departamentos = new Departamento[100];

        for (j = 0; j < nDepartamentos; j = j + 1) {
            System.out.println("Ingrese el nombre del departamento " + (j + 1) + ": ");
            nombreDepartamento = leer.next();
            System.out.println("Ingrese el numero de provincias");

            Provincia[] provincias = new Provincia[100];

            for (i = 0; i < nProvincias; i = i + 1) {
                System.out.println("Ingrese el nombre de la provincia " + (i + 1) + ": ");
                nombreProvincia = leer.next();

                Provincia prov = new Provincia();
                prov.setNombreProvincia(nombreProvincia);

                provincias[i] = prov;
            }
            System.out.println("Ingrese el nombre de la nueva provincia: ");
            nombreProvincia = leer.next();

            Provincia prov = new Provincia();
            prov.setNombreProvincia(nombreProvincia);

            provincias[nProvincias] = prov;

            Departamento dep = new Departamento();
            dep.setNombreDepartamento(nombreDepartamento);
            dep.setProvincias(provincias);
            dep.setNoProvincias(nProvincias + 1);
            departamentos[j] = dep;
        }

        System.out.println("Ingrese el nombre del nuevo departamento: ");
        nombreDepartamento = leer.next();

        Departamento dep = new Departamento();
        dep.setNombreDepartamento(nombreDepartamento);

        departamentos[nDepartamentos] = dep;
        Pais pais = new Pais( nombrePais: "BOLIVIA", noDepartamentos: nDepartamentos + 1, departamentos);
        pais.mostrarPais();

    }
}
```