ESCUELA POLITÉCNICA NACIONAL
ESCUELA DE FORMACIÓN DE TECNÓLOGOS
DESARROLLO DE SOFTWARE

**PERÍODO ACADÉMICO:** 2025-A
**ASIGNATURA:** PROGRAMACIÓN ORIENTA A OBJETO
**DOCENTE:** MSc.
**PRUEBA 1 BIMESTRE**

**NOMBRE DEL ESTUDIANTE:** Zapata González Felipe Javier **NOTA:** /

### Instrucciones Generales:

- Resolver la presente EMULACIÓN evalución PRÁCTICA tiene una duración de 70 minutos.
- Cualquier intento de copia se retira el examen y la calificación es cero

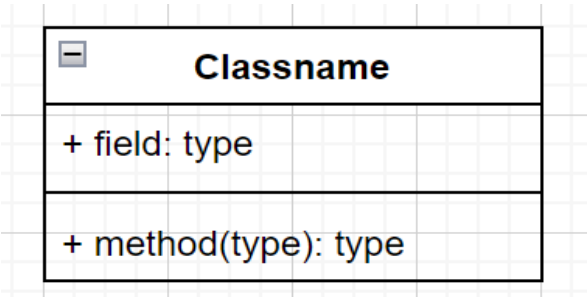
### Objetivo:

- La evaluación busca determinar la capacidad de los estudiantes para aplicar los principios fundamentales de la POO en la resolución de problemas prácticos.

## Problema a resolver

Utilizando los principios de programación orientada a objetos, implementar un sistema que me permita registrar hoteles de una ciudad.

1.- Cree un UML de la clase previo al desarrollo de la clase. De la clase con sus elementos correspondientes. (5 ptos.)




2.- Desde el main cree 5 instancias de la clase (queme los valores). (5 ptos.)

```
Ejemplo ejemplo1 = new Ejemplo("Felipe");
ejemplo1.setNombre("Felipe");
Ejemplo ejemplo2 = new Ejemplo("Javier");
ejemplo2.setNombre("Javier");
Ejemplo ejemplo3 = new Ejemplo("Isabella");
ejemplo3.setNombre("Isabella");
Ejemplo ejemplo4 = new Ejemplo("Virginia");
ejemplo4.setNombre("Virginia");
Ejemplo ejemplo5 = new Ejemplo("Eva");
ejemplo5.setNombre("Eva");
```

3.- Desde el main cree dos objetos con valores nulos y posteriormente setee sus valores. (5 pts.)

```
Ejemplo ejemplo1 = new Ejemplo("");
ejemplo1.setNombre("Felipe");
Ejemplo ejemplo2 = new Ejemplo("");
ejemplo2.setNombre("Isabella");
```

4.- Desarrolle métodos personalizados que permitan:

a.- Modificar los objetos previamente creados. (2 ptos.)

```
public void setNombreIndividual(String nombreIndividual) {
    this.nombreIndividual = nombreIndividual;
}
```

b.- Imprimir todos los objetos. (2 ptos.)

```
public static List<String> getNombres() {
    return new ArrayList<>(nombres);
}
```

c.- Modificar todos los atributos de al menos tres objetos con el valor de NULL y mostrar su nuevo valor. (2 ptos.)

```
System.out.println("\tCAMBIANDO NOMBRES...");
ejemplo1.setNombreIndividual("Feli");
ejemplo2.setNombreIndividual("Bella");
ejemplo3.setNombreIndividual("Evie");
System.out.println("Nuevo nombre 1. " +
ejemplo1.getNombreIndividual());
System.out.println("Nuevo nombre 2. " +
ejemplo2.getNombreIndividual());
System.out.println("Nuevo nombre 3. " +
ejemplo3.getNombreIndividual());
```

5.- Comente el código con explicación de lo más relevante del mismo, indicando su funcionalidad. (4 ptos).

```
import java.util.ArrayList;
import java.util.List;

class Ejemplo {

    private String nombreIndividual; // Atributo de instancia para
el nombre individual del objeto.
    private static List<String> nombres = new ArrayList<>(); //
Lista estática compartida por todas las instancias.

    public Ejemplo(String nombreIndividual) {
        this.nombreIndividual = nombreIndividual;
```

```
        agregarNombre(nombreIndividual); // Al crear una instancia,
se agrega el nombre a la lista compartida.
    }

    public String getNombreIndividual() {
        return nombreIndividual;
    }

    public void setNombreIndividual(String nombreIndividual) {
        this.nombreIndividual = nombreIndividual;
    }

    private static void agregarNombre(String nombre) {
        nombres.add(nombre); // Métod estático para añadir nombres a
la lista compartida.
    }

    public static List<String> getNombres() {
        return new ArrayList<>(nombres); // Métod estático para
obtener una copia de la lista compartida.
    }
}

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

        System.out.println("\tAGENDA DE NOMBRES: ");

        Ejemplo ejemplo1 = new Ejemplo("Felipe"); // Se crea una
instancia de Ejemplo, "Felipe" se añade a la lista 'nombres'.
        Ejemplo ejemplo2 = new Ejemplo("Isabella"); // Se crea otra
instancia, "Isabella" se añade a 'nombres'.
        Ejemplo ejemplo3 = new Ejemplo("Eva"); // Se crea una
tercera instancia, "Eva" se añade a 'nombres'.

        System.out.println("Su lista es:");
        List<String> listaNombres = Ejemplo.getNombres(); // Se
obtiene una copia de la lista de nombres compartida.
        for (int i = 0; i < listaNombres.size(); i++) {
            System.out.println((i + 1) + ". " +
listaNombres.get(i)); // Se imprime los nombres de la lista
        }

        System.out.println("\tCAMBIANDO NOMBRES...");
        ejemplo1.setNombreIndividual("Feli"); // Se modifica el
nombre individual de la instancia ejemplo1.
        ejemplo2.setNombreIndividual("Bella"); // Se modifica el
nombre individual de la instancia ejemplo2.
        ejemplo3.setNombreIndividual("Evie"); // Se modifica el
nombre individual de la instancia ejemplo3.
        System.out.println("Nuevo nombre 1. " +
ejemplo1.getNombreIndividual());
        System.out.println("Nuevo nombre 2. " +
ejemplo2.getNombreIndividual());
        System.out.println("Nuevo nombre 3. " +
ejemplo3.getNombreIndividual());

        System.out.println("\tFin");
    }
```

```
}

/*
**Funcionalidad del código:**

La clase `Ejemplo` gestiona objetos con un nombre individual
(`nombreIndividual`). Utiliza una lista estática (`nombres`) para
rastrear todos los nombres proporcionados al crear instancias de
`Ejemplo`.

- El constructor añade el nombre de la nueva instancia a la lista
estática.
- Los métodos `getNombreIndividual` y `setNombreIndividual` permiten
acceder y modificar el nombre específico de cada objeto `Ejemplo`.
- El métod estático `agregarNombre` es la forma de añadir nombres a
la lista compartida.
- El métod estático `getNombres` devuelve una copia de la lista de
nombres compartida.

La clase `Main` crea varias instancias de `Ejemplo`, lo que provoca
que sus nombres se añadan a la lista compartida. Luego, obtiene e
imprime esta lista. Finalmente, modifica los nombres individuales de
las instancias (sin afectar la lista compartida) y los imprime.

En esencia, la lista estática `nombres` actúa como un registro de
todos los nombres que han pasado por la creación de objetos
`Ejemplo`, mientras que `nombreIndividual` es específico de cada
objeto.
*/
```