# Trabajo Practico N°3

# de

# Programación II

Estudiante: Tobias Leiva

Unidad: Introducción a POO

Universidad Tecnológica Nacional

## Caso Práctico

Desarrollar en Java los siguientes ejercicios aplicando los conceptos de programación orientada a objetos:

**1. Registro de Estudiantes**

a. Crear una clase Estudiante con los atributos: nombre, apellido, curso, calificación.

**Métodos requeridos**: mostrarInfo(), subirCalificacion(puntos), bajarCalificacion(puntos).

**Tarea:** Instanciar a un estudiante, mostrar su información, aumentar y disminuir calificaciones.

Clase Estudiante:

```
public class Estudiante {
    String nombre;
    String apellido;
    String curso;
    double calificacion;

    public void mostrarInfo(){
      System.out.println(nombre + " " + apellido + " " + curso + " "+
      calificacion);
    }
    public double bajarCalificacion(double puntos){
        return calificacion = calificacion - puntos;
    }
    public double subirCalificacion(double puntos){
        return calificacion = calificacion + puntos;
    }
}
```

Programa Main:

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Registro Estudiante");
        Estudiante a = new Estudiante();
        a.mostrarInfo();
        a.subirCalificacion(5.5);
        a.mostrarInfo();
```

```
        a.bajarCalificacion(2.5);
        a.mostrarInfo();
        a.subirCalificacion(5.5);
        a.mostrarInfo();
     }
}
```

2. Registro de Mascotas

a. Crear una clase Mascota con los atributos: nombre, especie, edad.

Métodos requeridos: mostrarInfo(), cumplirAnios().

Tarea: Crear una mascota, mostrar su información, simular el paso del tiempo y verificar los cambios.

Clase Mascota:

```
public class Mascota {
    String nombre;
    String especie;
    int edad;
    public void mostrarInfo(){
        System.out.println(nombre + " " + especie + " " + "Anios:" +edad);
    }
    public int cumplirAnios(){
        return edad = edad + 1;
    }
}
```

Programa Main:

```
public class Main {
    public static void main(String[] args) {
        Mascota m = new Mascota();
        m.mostrarInfo();
        m.cumplirAnios();
        m.mostrarInfo();
        m.cumplirAnios();
        m.mostrarInfo();
   }
}
```

3. Encapsulamiento con la Clase Libro

a. Crear una clase Libro con atributos privados: titulo, autor, añoPublicacion.

Métodos requeridos: Getters para todos los atributos. Setter con validación para añoPublicacion.

Tarea: Crear un libro, intentar modificar el año con un valor inválido y luego con uno válido, mostrar la información final.

Clase Libro:

```
public class Libro {

    private String titulo;

    private String autor;

    private int añoPublicacion;


    public String getTitulo() {

        return titulo;

    }

    public String getAutor() {

        return autor;

    }

    public int getAñoPublicacion() {

        return añoPublicacion;

    }

    public void setAñoPublicacion(int añoPublicacion) {

        if (añoPublicacion > 0 && añoPublicacion <= 2025) {

            this.añoPublicacion = añoPublicacion;

        } else {

            System.out.println("Anio de Publicacion invalido: " +
añoPublicacion);

        }

    }

}
```

Programa Main:

```
public class Main {
    public static void main(String[] args) {
       System.out.println("Encapsulamiento con la Clase Libro");
        Libro libro1 = new Libro();

        //Cambiar año valido
        libro1.setAñoPublicacion(2005);
        //Mostrar datos
        System.out.println("Anio: " + libro1.getAñoPublicacion());

        //Cambiar año invalido
        libro1.setAñoPublicacion(23432);
        //Mostrar datos
        System.out.println("Anio: " + libro1.getAñoPublicacion());
    }
}
```

4. Gestión de Gallinas en Granja Digital

a. Crear una clase Gallina con los atributos: idGallina, edad, huevosPuestos.

Métodos requeridos: ponerHuevo(), envejecer(), mostrarEstado().

Tarea: Crear dos gallinas, simular sus acciones (envejecer y poner huevos), y mostrar su estado.

Clase Gallina:

```
public class Gallina {
    String idGallina;
    int edad;
    int huevosPuestos;

    public int ponerHuevos(int huevos){
        return huevosPuestos = huevosPuestos + huevos;
    }
    public int envejecer(int anios){
        return edad = edad + anios;
    }
```

```
    public void mostrarEstado(){
        System.out.println("Gallina: "+idGallina + " " + "Edad: "+edad +
                                    " " + "Huevos Puestos: "+huevosPuestos);
    }
}
```

Programa Main:

```
public class Main {
    public static void main(String[] args) {
        //Primera gallina
        Gallina gallina1 = new Gallina();
        gallina1.idGallina = "gallina1";

        gallina1.envejecer(5);
        gallina1.ponerHuevos(2);
        gallina1.ponerHuevos(3);
        gallina1.ponerHuevos(1);
        gallina1.ponerHuevos(5);
        //Mostramos su estado
        gallina1.mostrarEstado();

        //Segunda gallina
        Gallina gallina2 = new Gallina();
        gallina2.idGallina = "gallina2";

        gallina2.envejecer(8);
        gallina2.ponerHuevos(10);
        gallina2.ponerHuevos(5);
        gallina2.ponerHuevos(5);
        gallina2.ponerHuevos(6);
        //Mostramos su estado
        gallina2.mostrarEstado();
   }
}
```

5. Simulación de Nave Espacial Crear una clase NaveEspacial con los atributos: nombre, combustible.

Métodos requeridos: despegar(), avanzar(distancia), recargarCombustible(cantidad), mostrarEstado().

Reglas: Validar que haya suficiente combustible antes de avanzar y evitar que se supere el límite al recargar.

Tarea: Crear una nave con 50 unidades de combustible, intentar avanzar sin recargar, luego recargar y avanzar correctamente. Mostrar el estado al final.

Clase Nave Espacial:

```
public class NaveEspacial {
    String nombre;
    double combustible;

    public void despegar(boolean despegar){
        if (despegar == true ) {
            System.out.println("La nave despego");
        } else{
            System.out.println("La nave no despego");
        }
    }
    public double recargarCombustible(double cantidad){
        return combustible = combustible + cantidad;
    }

    public void avanzar(double distancia){
        if (combustible <= 0) {
            System.out.println("Falta combustible");
        }else{
            combustible = combustible - distancia;
        }
    }
    public void mostrarEstado(){
        System.out.println("Combustible disponible: "+ combustible);
    }
}
```

Programa Main:

```
public class Main {
    public static void main(String[] args) {
        NaveEspacial nave = new NaveEspacial();
        nave.combustible = 50;

        nave.despegar(true);
        //Avanzamos sin recargar combustible
        nave.avanzar(20);
        nave.mostrarEstado();

        //Avanzamos hasta quedarnos sin combustible
        nave.avanzar(30);
        nave.mostrarEstado();

        //Si intentamos avanzar no podemos
        nave.avanzar(5);

        //Regarcamos combustible
        System.out.println();
        nave.recargarCombustible(100);
        nave.mostrarEstado();

        //Avanzamos nuevamente
        nave.avanzar(75);

        //Mostramos el estado final
        nave.mostrarEstado();
  }
}
```

**Repositorio: https://github.com/Tobias-L7/Trabajos-Practicos-P2.git**