PROGRAMACIÓN II

TRABAJO PRÁCTICO 3: Introducción a la POO

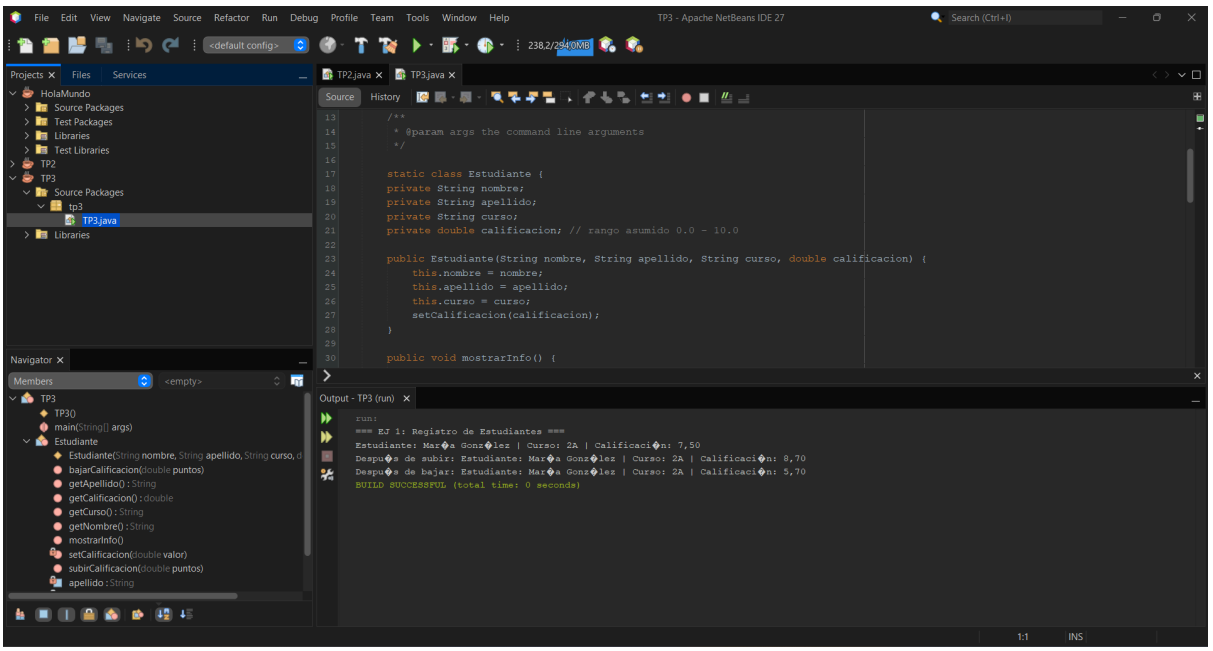
Alumna: Garcia Galfione, Fiorella

Enlace a repositorio en Github: https://github.com/FioreGG/UTN-TUPaD-P2.git

1. Resolución ejercicio 1

```
package tp3;

public class TP3 {

    static class Estudiante {
    private String nombre;
    private String apellido;
    private String curso;
    private double calificacion; // rango asumido 0.0 - 10.0

    public Estudiante(String nombre, String apellido, String curso, double calificacion) {
        this.nombre = nombre;
        this.apellido = apellido;
        this.curso = curso;
        setCalificacion(calificacion);
    }

    public void mostrarInfo() {
        System.out.printf("Estudiante: %s %s | Curso: %s | Calificación: %.2f%n",
                nombre, apellido, curso, calificacion);
    }

    public void subirCalificacion(double puntos) {
        setCalificacion(this.calificacion + puntos);
    }

    public void bajarCalificacion(double puntos) {
        setCalificacion(this.calificacion - puntos);
    }

    private void setCalificacion(double valor) {
        if (valor < 0.0) {
            this.calificacion = 0.0;
        } else if (valor > 10.0) {
            this.calificacion = 10.0;
        } else {
            this.calificacion = valor;
        }
    }
```

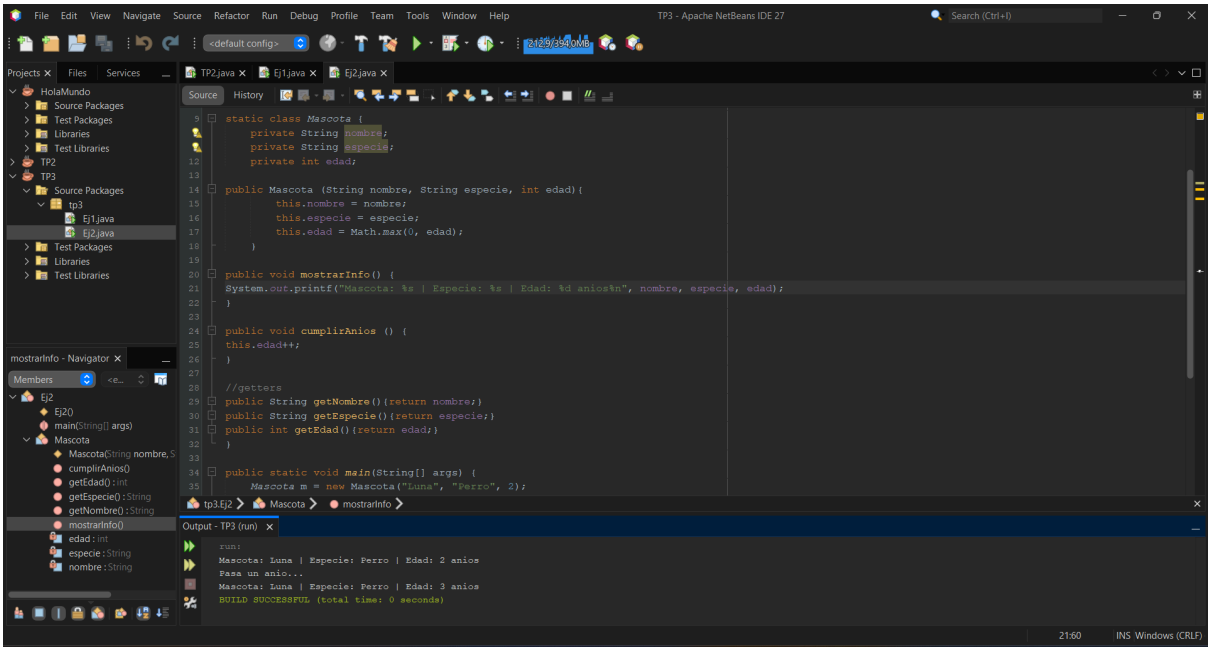
```
    // getters (si se necesitan los datos)
    public String getNombre() { return nombre; }
    public String getApellido() { return apellido; }
    public String getCurso() { return curso; }
    public double getCalificacion() { return calificacion; }
}

    public static void main(String[] args) {
        System.out.println("=== EJ 1: Registro de Estudiantes ===");
        Estudiante e = new Estudiante("María", "González", "2A", 7.5);
        e.mostrarInfo();
        e.subirCalificacion(1.2);
        System.out.print("Después de subir: ");
        e.mostrarInfo();
        e.bajarCalificacion(3.0);
        System.out.print("Después de bajar: ");
        e.mostrarInfo();


    }
}
```



2. Resolución ejercicio 2

```
package tp3;

public class Ej2 {

static class Mascota {
```

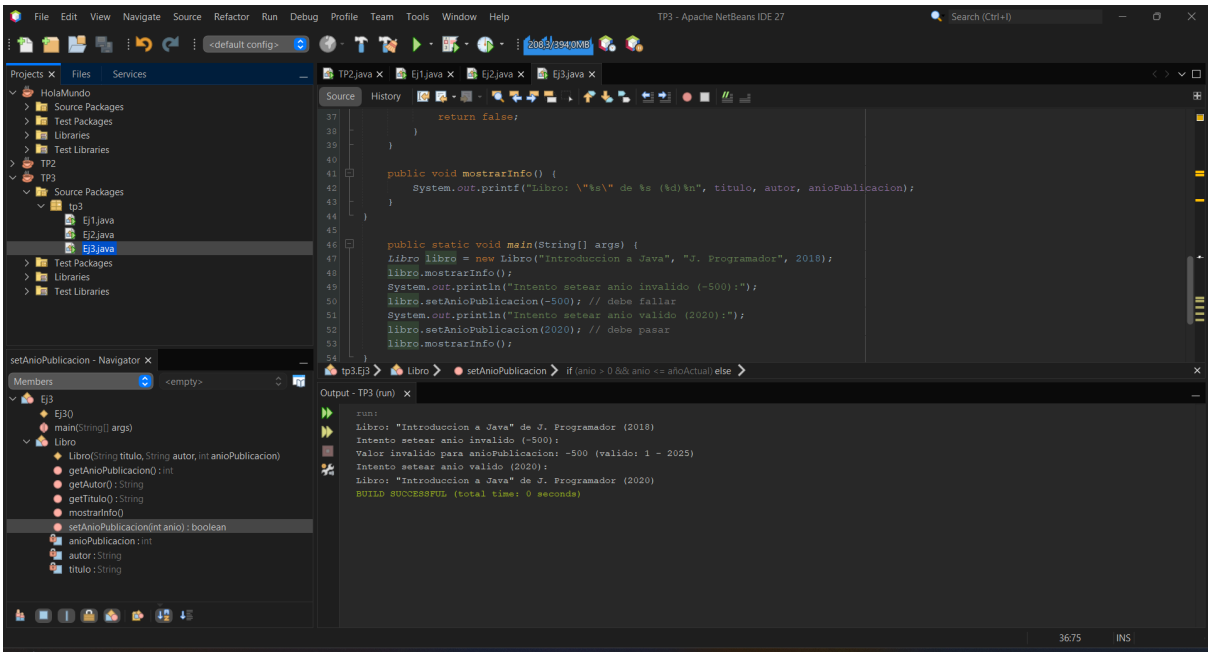
```
    private String nombre;
    private String especie;
    private int edad;

public Mascota (String nombre, String especie, int edad){
        this.nombre = nombre;
        this.especie = especie;
        this.edad = Math.max(0, edad);
    }

public void mostrarInfo() {
System.out.printf("Mascota: %s | Especie: %s | Edad: %d anios%n", nombre, especie,
edad);
}

public void cumplirAnios () {
this.edad++;
}

//getters
public String getNombre(){return nombre;}
public String getEspecie(){return especie;}
public int getEdad(){return edad;}
}

public static void main(String[] args) {
    Mascota m = new Mascota("Luna", "Perro", 2);
    m.mostrarInfo();
    System.out.println("Pasa un anio...");
    m.cumplirAnios();
    m.mostrarInfo();
}
}
```



3. Resolución ejercicio 3

```
package tp3;

import java.time.Year;


public class Ej3 {

    static class Libro {
    private String titulo;
    private String autor;
    private int anioPublicacion;


    public Libro(String titulo, String autor, int anioPublicacion){
        this.titulo = titulo;
        this.autor = autor;
        setAnioPublicacion(anioPublicacion);
    }

   // getters
    public String getTitulo(){return titulo;}
    public String getAutor(){return autor;}
    public int getAnioPublicacion(){return anioPublicacion;}

    // setter con validacion
    public boolean setAnioPublicacion (int anio){
     int añoActual = Year.now().getValue(); // usa el año actual del sistema
        if (anio > 0 && anio <= añoActual) {
```

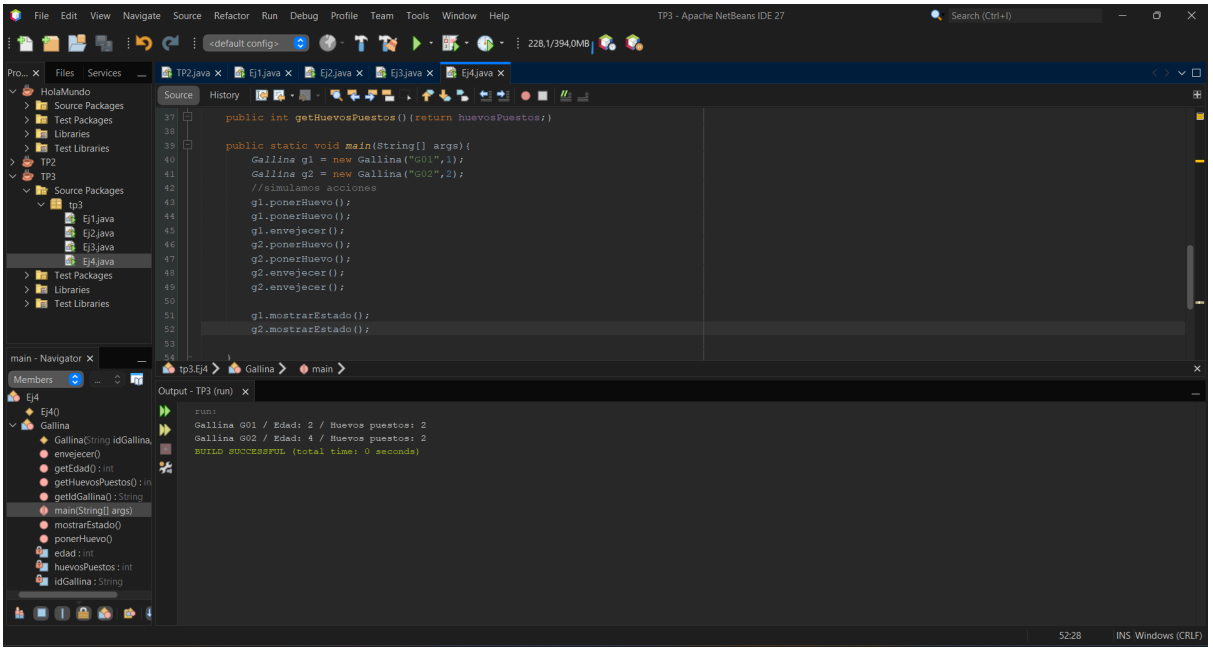
```
            this.anioPublicacion = anio;
            return true;
        } else {
             System.out.printf("Valor invalido para anioPublicacion: %d (valido: 1 - %d)%n", anio,
añoActual);
            return false;
        }
    }

    public void mostrarInfo() {
        System.out.printf("Libro: \"%s\" de %s (%d)%n", titulo, autor, anioPublicacion);
    }
}

    public static void main(String[] args) {
    Libro libro = new Libro("Introduccion a Java", "J. Programador", 2018);
    libro.mostrarInfo();
    System.out.println("Intento setear anio invalido (-500):");
    libro.setAnioPublicacion(-500); // debe fallar
    System.out.println("Intento setear anio valido (2020):");
    libro.setAnioPublicacion(2020); // debe pasar
    libro.mostrarInfo();
}

}
```



4. Resolución ejercicio 4

```
package tp3;


public class Ej4 {

    static class Gallina {
    private String idGallina;
    private int edad;
    private int huevosPuestos;

    public Gallina(String idGallina, int edad){
        this.idGallina = idGallina;
        this.edad = Math.max(0, edad);
        this.huevosPuestos = 0;
    }

    public void ponerHuevo(){
        huevosPuestos++;
    }

    public void envejecer(){
        edad++;
    }

    public void mostrarEstado(){
        System.out.printf("Gallina %s / Edad: %d / Huevos puestos: %d%n",
                idGallina, edad, huevosPuestos);
    }

    //getters
    public String getIdGallina(){return idGallina;}
    public int getEdad(){return edad;}
    public int getHuevosPuestos(){return huevosPuestos;}

    public static void main(String[] args){
        Gallina g1 = new Gallina("G01",1);
        Gallina g2 = new Gallina("G02",2);
        //simulamos acciones
        g1.ponerHuevo();
        g1.ponerHuevo();
        g1.envejecer();
        g2.ponerHuevo();
        g2.ponerHuevo();
        g2.envejecer();
        g2.envejecer();

        g1.mostrarEstado();
        g2.mostrarEstado();
```

```
    }

    }

}
```



5. Resolución ejercicio 5

```
package tp3;

public class Ej5 {

    static class NaveEspacial{
    private String nombre;
    private double combustible;
    private final double capacidad_max = 100; //ejemplo de limite de la nave
    private final double consumo_por_distancia = 1;
    private final double consumo_despegue = 10;

    public NaveEspacial(String nombre, double combustibleInicial){
    this.nombre = nombre;
    this.combustible = Math.max(0,Math.min(combustibleInicial, capacidad_max));}


    public boolean despegar(){
        if (combustible >= consumo_despegue){
        combustible-= consumo_despegue;
```

```
            System.out.printf("La nave %s despego. Combustible restante: %.2f%n",nombre,
combustible);
        return true;
        } else {
           System.out.printf("No hay suficiente combustible para despegar (se necesitan %.2f).
Combustible actual: %.2f%n\",\n",
                    consumo_despegue, combustible);
           return false;
        }
        }

    public boolean avanzar (double distancia){
        double requerido = distancia * consumo_por_distancia;
        if (requerido <= combustible) {
            combustible -= requerido;
                 System.out.printf("La nave %s avanzo %.2f unidades. Combustible restante:
%.2f%n",
                    nombre, distancia, combustible);
            return true;
        } else {
                 System.out.printf("No hay suficiente combustible para avanzar %.2f unidades
(requiere %.2f, hay %.2f).%n",
                    distancia, requerido, combustible);
            return false;
        }
    }

     public boolean recargarCombustible(double cantidad) {
        if (cantidad <= 0) {
            System.out.println("Cantidad a recargar debe ser positiva.");
            return false;
        }
        if (combustible + cantidad > capacidad_max) {
             System.out.printf("No se puede recargar %.2f: excede la capacidad máxima (%.2f).
Combustible actual: %.2f%n",
                    cantidad, capacidad_max, combustible);
            return false;
        }
        combustible += cantidad;
          System.out.printf("Recargados %.2f unidades. Combustible actual: %.2f%n", cantidad,
combustible);
        return true;
    }

    public void mostrarEstado() {
            System.out.printf("Nave: %s | Combustible: %.2f/%.2f%n", nombre, combustible,
capacidad_max);
    }
```

```
    // getters
    public String getNombre() { return nombre; }
    public double getCombustible() { return combustible; }


    public static void main(String[] args){
        NaveEspacial nave = new NaveEspacial("Aurora", 50.0); // según enunciado
        nave.mostrarEstado();
        System.out.println("Intento avanzar 60 unidades sin recargar:");
        nave.avanzar(60); // debería fallar
        System.out.println("Intento despegar:");
        nave.despegar(); // si hay >= 10 unidades consumirá 10
        System.out.println("Recargo 40 unidades:");
        nave.recargarCombustible(40); // intentar recargar
        System.out.println("Ahora intento avanzar 60 unidades:");
        nave.avanzar(60); // ahora puede ser posible
        nave.mostrarEstado();
    }
}}
```