

## PROGRAMACIÓN II

### Trabajo Práctico 3: Introducción a la Programación Orientada a Objetos

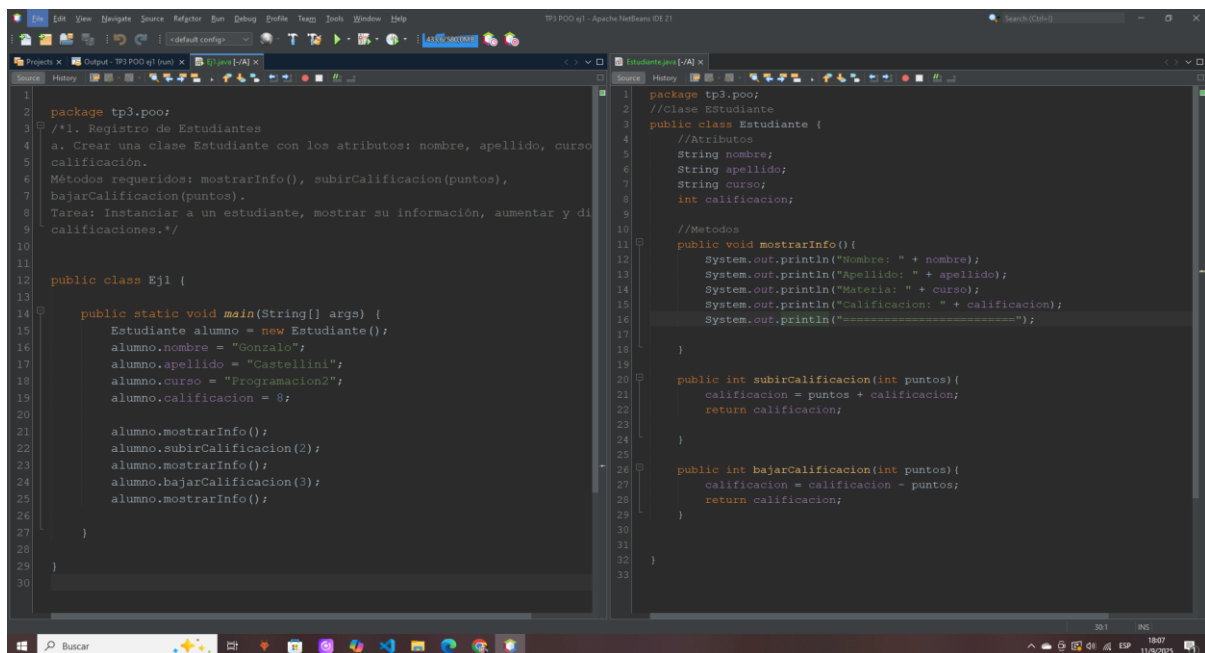
Alumno Castellini Gonzalo

**OBJETIVO GENERAL** Comprender los fundamentos de la Programación Orientada a Objetos, incluyendo clases, objetos, atributos y métodos, para estructurar programas de manera modular y reutilizable en Java.

#### 1. Registro de Estudiantes

a. Crear una clase Estudiante con los atributos: nombre, apellido, curso, calificación. Métodos requeridos: mostrarInfo(), subirCalificacion(puntos), bajarCalificacion(puntos).

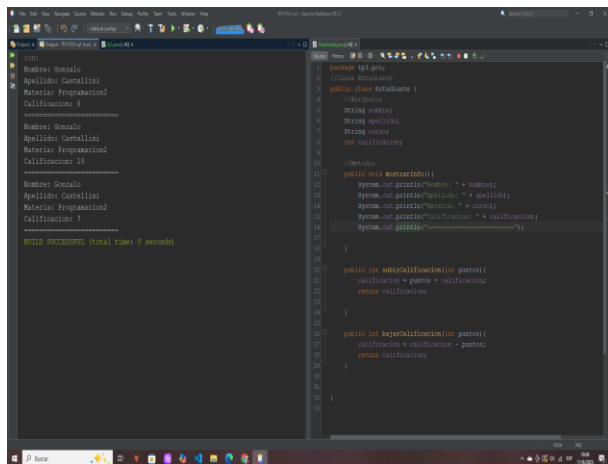
Tarea: Instanciar a un estudiante, mostrar su información, aumentar y disminuir calificaciones.



```
1 package tp3.poo;
2
3 /*1. Registro de Estudiantes
4 a. Crear una clase Estudiante con los atributos: nombre, apellido, curso
5 calificación.
6 Métodos requeridos: mostrarInfo(), subirCalificacion(puntos),
7 bajarCalificacion(puntos).
8 Tarea: Instanciar a un estudiante, mostrar su información, aumentar y di
9 calificaciones.*/
10
11
12 public class Ej1 {
13
14     public static void main(String[] args) {
15         Estudiante alumno = new Estudiante();
16         alumno.nombre = "Gonzalo";
17         alumno.apellido = "Castellini";
18         alumno.curso = "Programacion2";
19         alumno.calificacion = 8;
20
21         alumno.mostrarInfo();
22         alumno.subirCalificacion(2);
23         alumno.mostrarInfo();
24         alumno.bajarCalificacion(3);
25         alumno.mostrarInfo();
26     }
27 }
28
29
30
```

```
1 package tp3.poo;
2 //Clase ESTudiante
3 public class Estudiante {
4     //Atributos
5     String nombre;
6     String apellido;
7     String curso;
8     int calificacion;
9
10    //Metodos
11    public void mostrarInfo(){
12        System.out.println("Nombre: " + nombre);
13        System.out.println("Apellido: " + apellido);
14        System.out.println("Materia: " + curso);
15        System.out.println("Calificacion: " + calificacion);
16        System.out.println("=====");
17    }
18
19
20    public int subirCalificacion(int puntos){
21        calificacion = puntos + calificacion;
22        return calificacion;
23    }
24
25
26    public int bajarCalificacion(int puntos){
27        calificacion = calificacion - puntos;
28        return calificacion;
29    }
30
31
32 }
33
```

Resultados por consola

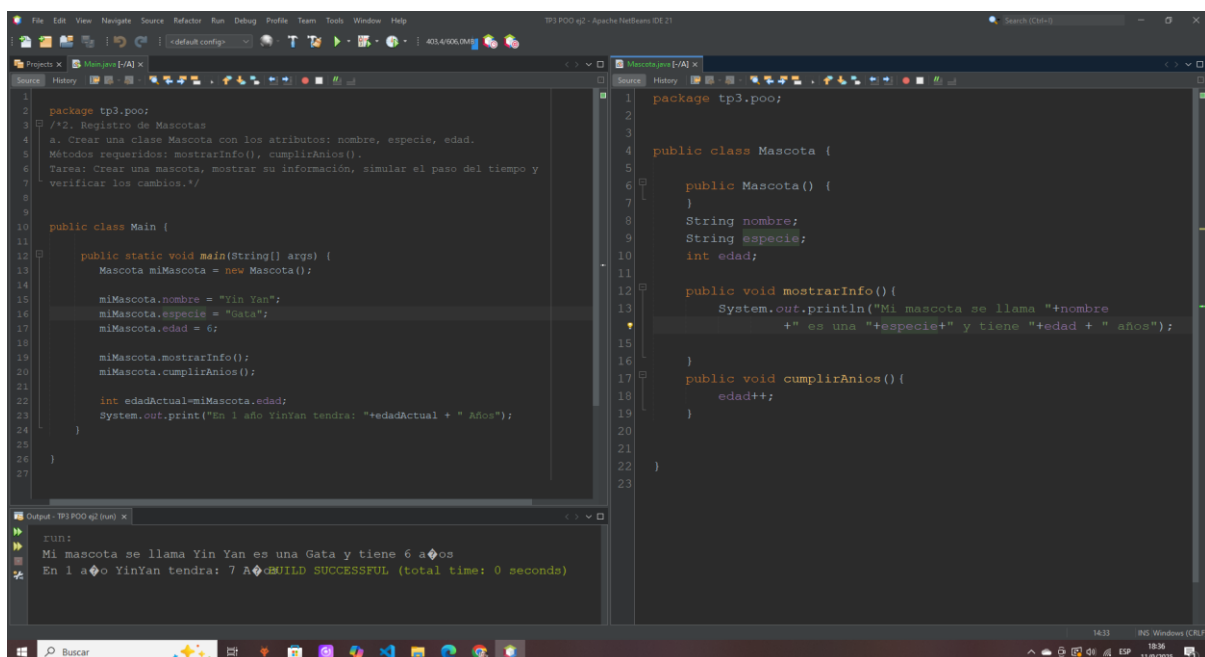


## 2. Registro de Mascotas

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

Métodos requeridos: mostrarInfo(), cumplirAños().

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



## 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.

The screenshot shows an IDE with two Java files. The left file, `tp3.poo.java`, contains a `Libro` class with attributes `titulo`, `autor`, and `anioPublicacion`, and a `Main` class that tests the `Libro` class. The right file, `Libro.java`, contains the `Libro` class with getters, setters, and a `setAnioPublicacion` method that validates the year. The output window at the bottom shows the results of running the `Main` class, including the creation of a `Libro` object and the display of its attributes.

```
package tp3.poo;

/** Encapsulamiento con la Clase Libro
 * 1. Crear una clase Libro con atributos privados: titulo, autor,
 * 2. anioPublicacion.
 * 3. Metodos requeridos: Getters para todos los atributos. Setter con validación
 * 4. para anioPublicacion.
 * 5. Tarea: Crear un libro, intentar modificar el año con un valor inválido y luego con
 * 6. uno válido, mostrar la información final.*/

public class Main {

    public static void main(String[] args) {
        Libro libro = new Libro();

        // Seteo inicial
        libro.setTitulo("El Angel Gris");
        libro.setAutor("Alejandro Dolina");

        // Intento con año inválido
        libro.setAnioPublicacion(-100); // debería mostrar mensaje de error

        // Intento con año válido
        libro.setAnioPublicacion(1988);

        // Mostrar información final
        System.out.println("Información del libro:");
        System.out.println("Titulo: " + libro.getTitulo());
        System.out.println("Autor: " + libro.getAutor());
        System.out.println("Año de publicación: " + libro.getAnioPublicacion());
    }
}

Output - TP3.POO.g3 (run) x
RUN:
Año de publicación inválido: -100
Información del libro:
Titulo: El Angel Gris
Autor: Alejandro Dolina
Año de publicación: 1988
BUILD SUCCESSFUL (total time: 1 second)
```

## 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.

The screenshot shows an IDE with two Java files. The left file, `Gallinas.java`, contains a `Gallina` class with attributes `idGallina`, `edad`, and `huevosPuestos`, and methods `getIdGallina`, `setIdGallina`, `getEdad`, `setEdad`, `getHuevosPuestos`, `setHuevosPuestos`, `ponerHuevo`, `envejecer`, and `mostrarEstado`. The right file, `TP3POOe3.java`, contains a `Main` class that creates two `Gallina` objects, `gallina1` and `gallina2`, and simulates their actions (envejecer and ponerHuevo) and displays their state.

```
package tp3.pooe3;

public class Gallina {
    private int idGallina;
    private int edad;
    private int huevosPuestos;

    public int getIdGallina() {
        return idGallina;
    }

    public void setIdGallina(int idGallina) {
        this.idGallina = idGallina;
    }

    public int getEdad() {
        return edad;
    }

    public void setEdad(int edad) {
        this.edad = edad;
    }

    public int getHuevosPuestos() {
        return huevosPuestos;
    }

    public void setHuevosPuestos(int huevosPuestos) {
        this.huevosPuestos = huevosPuestos;
    }

    public int ponerHuevo() {
        return huevosPuestos++;
    }

    public int envejecer() {
        return edad++;
    }

    public void mostrarEstado() {
        System.out.println("La gallina: " + idGallina);
        System.out.println("Su edad es: " + edad);
        System.out.println("Puso " + huevosPuestos + " Huevos");
    }
}

package TP3POOe3;

public class TP3POOe3 {

    public static void main(String[] args) {
        Gallina gallina1 = new Gallina();
        Gallina gallina2 = new Gallina();

        System.out.println("==== GALLINA 1 ===");

        System.out.println("Huevos Puestos");
        gallina1.setHuevosPuestos(3);
        System.out.println(gallina1.getHuevosPuestos());
        System.out.println("La gallina pone un huevo");
        gallina1.ponerHuevo();
        System.out.println(gallina1.getHuevosPuestos());

        System.out.println("Edad Actual");
        gallina1.setEdad(3);
        System.out.println(gallina1.getEdad());
        System.out.println("Edad luego de 1 año");
        gallina1.envejecer();
        System.out.println(gallina1.getEdad());

        gallina1.mostrarEstado();

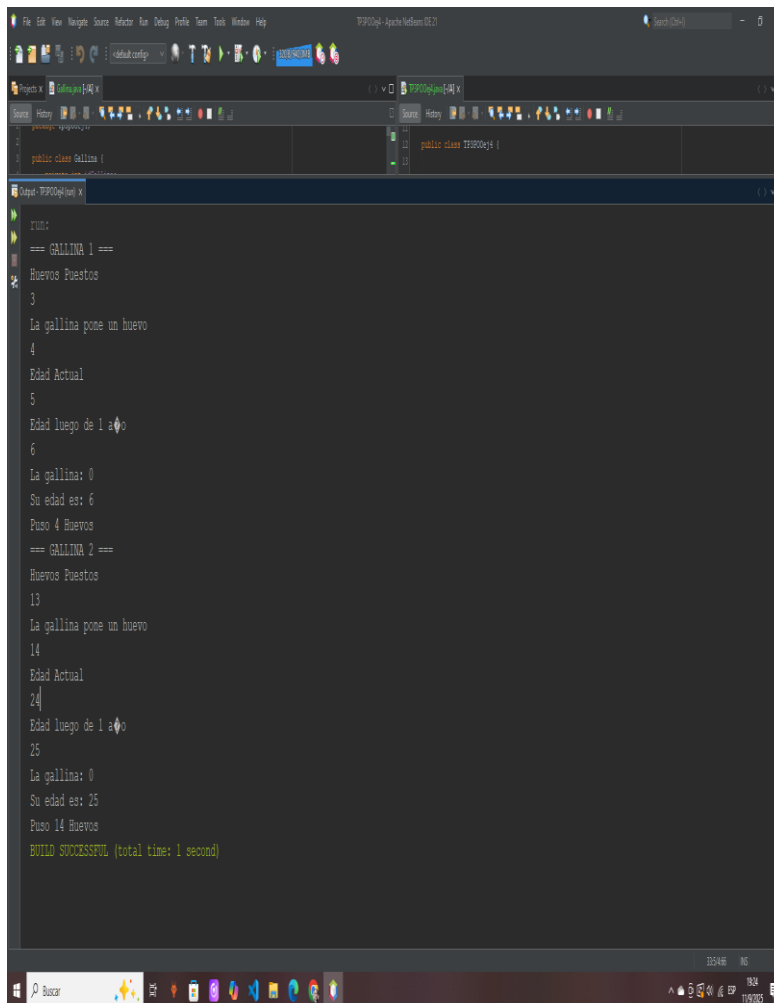
        System.out.println("==== GALLINA 2 ===");

        System.out.println("Huevos Puestos");
        gallina2.setHuevosPuestos(3);
        System.out.println(gallina2.getHuevosPuestos());
        System.out.println("La gallina pone un huevo");
        gallina2.ponerHuevo();
        System.out.println(gallina2.getHuevosPuestos());

        System.out.println("Edad Actual");
        gallina2.setEdad(3);
        System.out.println(gallina2.getEdad());
        System.out.println("Edad luego de 1 año");
        gallina2.envejecer();
        System.out.println(gallina2.getEdad());

        gallina2.mostrarEstado();
    }
}
```

## Resultado por consola



The screenshot shows an IDE with two tabs: 'Gallina.java' and 'TPPOrgenes.java'. The 'Gallina.java' tab is active, showing a Java class with methods for managing a chicken's state. The console output shows the execution of the program, which simulates a chicken's life cycle. The output includes the number of eggs laid, the current age, and the age after a year. The program ends with a 'BUILD SUCCESSFUL' message.

```
run:
=== GALLINA 1 ===
Huevos Puestos
3
La gallina pone un huevo
4
Edad Actual
5
Edad luego de 1 año
6
La gallina: 0
Su edad es: 6
Puso 4 Huevos
=== GALLINA 2 ===
Huevos Puestos
13
La gallina pone un huevo
14
Edad Actual
24
Edad luego de 1 año
25
La gallina: 0
Su edad es: 25
Puso 14 Huevos
BUILD SUCCESSFUL (total time: 1 second)
```

## 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.

