

TP 3 – Introducción a la Programación Orientada a Objetos

Objetivo:

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.

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. 2

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.

3. Encapsulamiento con la Clase Libro

- a. Crear una clase **Libro** con atributos privados: **título**, **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.

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.

5. Simulación de Nave Espacial

- a. 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.

1)

```
package tp3ejer1;  
import java.io.PrintStream;  
import java.nio.charset.StandardCharsets;
```

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

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

```
private void setCalificacion(double calificacion) {
    if (calificacion < 0) {
        this.calificacion = 0;
    } else if (calificacion > 10) {
        this.calificacion = 10;
    } else {
        this.calificacion = calificacion;
    }
}

public void mostrarInfo() {
    System.out.println(nombre + " " + apellido + " - Curso: " + curso + " -
Calificación: " + calificacion);
}

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

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

}

public class TP3Ejer1 {

    public static void main(String[] args) {
        System.setOut(new PrintStream(System.out, true,
StandardCharsets.UTF_8));

        Estudiante e1 = new Estudiante("Ana", "Pérez", "Matemáticas", 8.5);

        e1.mostrarInfo();
        e1.subirCalificacion(1.0);
        e1.mostrarInfo();
        e1.bajarCalificacion(0.5);
        e1.mostrarInfo();
    }
}
```

```
}  
  
}
```

OUTPUT:

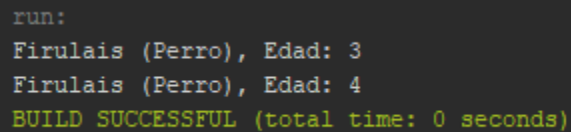
```
run:  
Ana Pérez - Curso: Matemáticas - Calificación: 8.5  
Ana Pérez - Curso: Matemáticas - Calificación: 9.5  
Ana Pérez - Curso: Matemáticas - Calificación: 9.0  
BUILD SUCCESSFUL (total time: 0 seconds)
```

2)

```
package tp3ej2;  
  
class Mascota {  
    String nombre;  
    String especie;  
    int edad;  
  
    public Mascota(String nombre, String especie, int edad) {  
        this.nombre = nombre;  
        this.especie = especie;  
        this.edad = edad;  
    }  
  
    public void mostrarInfo() {  
        System.out.println(nombre + " (" + especie + "), Edad: " + edad);  
    }  
  
    public void cumplirAnios() {  
        edad++;  
    }  
}  
  
public class TP3Ejer2 {  
  
    public static void main(String[] args) {  
        Mascota m1 = new Mascota("Firulais", "Perro", 3);  
        m1.mostrarInfo();  
        m1.cumplirAnios();  
    }  
}
```

```
        m1.mostrarInfo();  
    }  
  
}
```

OUTPUT:



```
run:  
Firulais (Perro), Edad: 3  
Firulais (Perro), Edad: 4  
BUILD SUCCESSFUL (total time: 0 seconds)
```

3)

```
package tp3ej3;  
  
import java.io.PrintStream;  
import java.nio.charset.StandardCharsets;  
  
class Libro {  
    private String titulo;  
    private String autor;  
    private int anioPublicacion;  
  
    public Libro(String titulo, String autor, int anioPublicacion) {  
        this.setTitulo(titulo);  
        this.setAutor(autor);  
        setAnioPublicacion(anioPublicacion);  
    }  
  
    public String getTitulo() {  
        return titulo;  
    }  
    public void setTitulo(String titulo) {  
        this.titulo = titulo;  
    }  
  
    public String getAutor() {  
        return autor;  
    }  
    public void setAutor(String autor) {  
        this.autor = autor;  
    }  
}
```

```
}

public int getAnioPublicacion() {
    return anioPublicacion;
}

public void setAnioPublicacion(int anioPublicacion) {
    if (anioPublicacion > 1400 && anioPublicacion <= 2025) {
        this.anioPublicacion = anioPublicacion;
    } else {
        System.out.println("Año inválido: " + anioPublicacion);
    }
}

public void mostrarInfo() {
    System.out.println("Libro: " + this.getTitulo() + " - Autor: " + this.getAutor() +
"- Año: " + this.getAnioPublicacion());
}

}

public class TP3Ejer3 {

    public static void main(String[] args) {
        System.setOut(new PrintStream(System.out, true,
StandardCharsets.UTF_8));
        Libro l1 = new Libro("Cien años de soledad", "Gabriel García Márquez",
1967);
        l1.mostrarInfo();

        l1.setAnioPublicacion(1200); // inválido
        l1.setAnioPublicacion(1980); // válido
        l1.mostrarInfo();
    }

}
```

OUTPUT:

```
run:
Libro: Cien años de soledad - Autor: Gabriel García Márquez - Año: 1967
Año inválido: 1200
Libro: Cien años de soledad - Autor: Gabriel García Márquez - Año: 1980
BUILD SUCCESSFUL (total time: 0 seconds)
```

4)

```
package tp3ej4;

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

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

    public void ponerHuevo() {
        huevosPuestos++;
        System.out.println("Gallina " + idGallina + " puso un huevo. Total: " +
huevosPuestos);
    }

    public void envejecer() {
        edad++;
        System.out.println("Gallina " + idGallina + " ha envejecido. Edad: " + edad);
    }

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

public class TP3Ejer4 {

    public static void main(String[] args) {
        Gallina g1 = new Gallina(1, 2);
        Gallina g2 = new Gallina(2, 1);

        g1.envejecer();
        g1.ponerHuevo();
        g2.ponerHuevo();
    }
}
```

```
        g2.ponerHuevo();

        g1.mostrarEstado();
        g2.mostrarEstado();
    }

}
```

OUTPUT:

```
run:
Gallina 1 ha envejecido. Edad: 3
Gallina 1 puso un huevo. Total: 1
Gallina 2 puso un huevo. Total: 1
Gallina 2 puso un huevo. Total: 2
Gallina 1 - Edad: 3 - Huevos: 1
Gallina 2 - Edad: 1 - Huevos: 2
BUILD SUCCESSFUL (total time: 0 seconds)
```

5)

```
package tp3ej5;

import java.io.PrintStream;
import java.nio.charset.StandardCharsets;

class NaveEspacial {
    String nombre;
    int combustible;
    final int CAPACIDAD_MAX = 100;
    private boolean haDespegado;

    public NaveEspacial(String nombre, int combustible) {
        this.nombre = nombre;
        this.combustible = combustible;
    }

    public void despegar() {
        if (haDespegado) {
            System.out.println("La nave ya esta en vuelo");
            return;
        }
        if (combustible >= 10 ){
```



```
        combustible -= 10;
        haDespegado = true;
        System.out.println(nombre + " ha despegado. Combustible restante: " +
combustible);
    } else {
        System.out.println("No hay suficiente combustible para despegar.");
    }
}

public void avanzar(int distancia) {
    if (!haDespegado) {
        System.out.println("La nave no puede avanzar porque aún no ha
despegado.");
        return;
    }
    int consumo = distancia * 2;
    if (combustible >= consumo) {
        combustible -= consumo;
        System.out.println(nombre + " avanzó " + distancia + " km. Combustible
restante: " + combustible);
    } else {
        System.out.println("No hay suficiente combustible para avanzar " +
distancia + " km.");
    }
}

public void recargarCombustible(int cantidad) {
    if (cantidad <= 0) {
        System.out.println("La cantidad de combustible debe ser mayor a
cero(0)");
        return;
    }
    if (combustible + cantidad <= CAPACIDAD_MAX) {
        combustible += cantidad;
        System.out.println("Se recargaron " + cantidad + " unidades. Combustible
actual: " + combustible);
    } else {
        combustible = CAPACIDAD_MAX;
        System.out.println("La nave está llena de combustible " + combustible +
"/" + CAPACIDAD_MAX);
    }
}
```

```
    }  
}  
  
public void mostrarEstado() {  
    String estadoVuelo = haDespegado ? "En vuelo" : "En tierra";  
    System.out.println("Nave: " + nombre +  
        " | Combustible: " + combustible + "/" + CAPACIDAD_MAX +  
        " | Estado: " + estadoVuelo);  
}  
}  
  
public class TP3Ejer5 {  
  
    public static void main(String[] args) {  
        System.setOut(new PrintStream(System.out, true,  
StandardCharsets.UTF_8));  
        NaveEspacial nave = new NaveEspacial("Apollo", 50);  
  
        nave.mostrarEstado();  
        // 1. Intentar avanzar sin despegar  
        nave.avanzar(10);  
  
        // 2. Despegar  
        nave.despegar();  
        nave.mostrarEstado();  
  
        // 3. Intentar avanzar demasiado (falla)  
        nave.avanzar(30);  
  
        // 4. Recargar  
        nave.recargarCombustible(70);  
  
        // 5. Avanzar correctamente  
        nave.avanzar(20);  
        nave.mostrarEstado();  
    }  
}
```

OUTPUT:

```
run:
Nave: Apollo | Combustible: 50/100 | Estado: En tierra
La nave no puede avanzar porque aún no ha despegado.
Apollo ha despegado. Combustible restante: 40
Nave: Apollo | Combustible: 40/100 | Estado: En vuelo
No hay suficiente combustible para avanzar 30 km.
La nave está llena de combustible 100/100
Apollo avanzó 20 km. Combustible restante: 60
Nave: Apollo | Combustible: 60/100 | Estado: En vuelo
BUILD SUCCESSFUL (total time: 0 seconds)
```