

## PROGRAMACIÓN II

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

**Alumno :**

- González Federico

**Profesor :**

- Enferrel Ariel

**Tutor :**

- Ferro Tomás

**GitHub:**

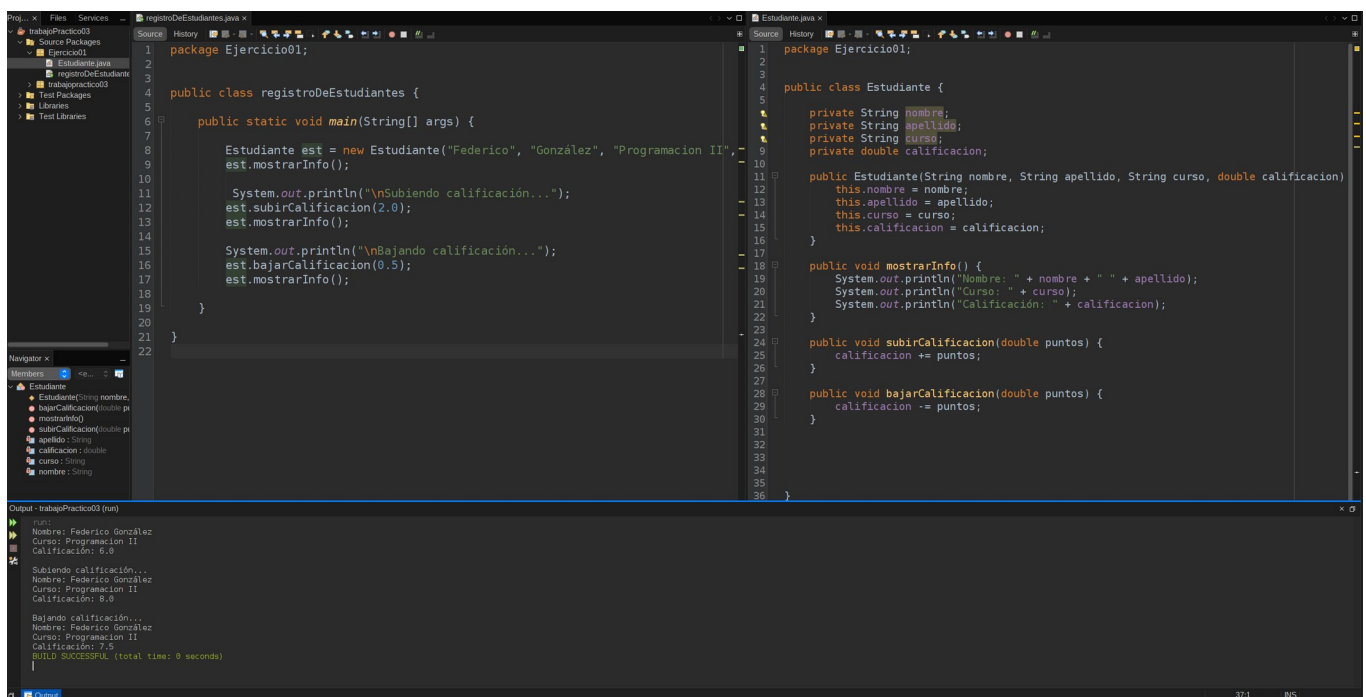
- <https://github.com/fedeglz/UTN-TUPaD-P2.git>

## TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN A DISTANCIA

### Ejercicio 1: Registro de Estudiantes

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

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



```

package Ejercicio01;

public class registroDeEstudiantes {

    public static void main(String[] args) {

        Estudiante est = new Estudiante("Federico", "González", "Programacion II", 8.0);
        est.mostrarInfo();

        System.out.println("\nSubiendo calificación...");
        est.subirCalificacion(2.0);
        est.mostrarInfo();

        System.out.println("\nBajando calificación...");
        est.bajarCalificacion(0.5);
        est.mostrarInfo();

    }

}

```

```

package Ejercicio01;

public class Estudiante {

    private String nombre;
    private String apellido;
    private String curso;
    private double calificacion;

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

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

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

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

}

```

```

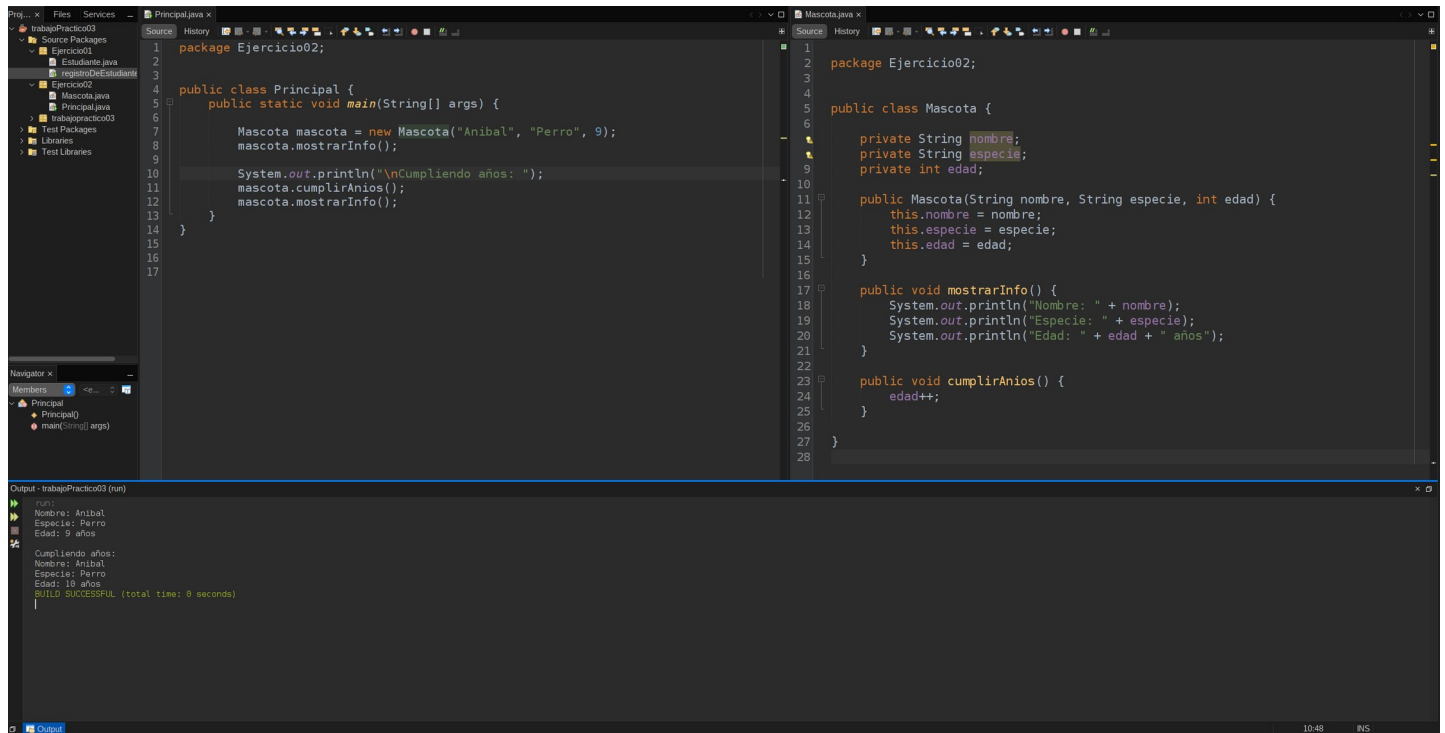
Output - trabaipractic03 (new)
run:
Nombre: Federico González
Curso: Programacion II
Calificación: 8.0
Subiendo calificación...
Nombre: Federico González
Curso: Programacion II
Calificación: 8.0
Bajando calificación...
Nombre: Federico González
Curso: Programacion II
Calificación: 7.5
BUILD SUCCESSFUL (total time: 0 seconds)

```

## Ejercicio 2: Registro de Mascotas

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

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



```

package Ejercicio02;

public class Principal {
    public static void main(String[] args) {
        Mascota mascota = new Mascota("Anibal", "Perro", 9);
        mascota.mostrarInfo();

        System.out.println("\nCumpliendo años: ");
        mascota.cumplirAños();
        mascota.mostrarInfo();
    }
}

```

```

package Ejercicio02;

public 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 = edad;
    }

    public void mostrarInfo() {
        System.out.println("Nombre: " + nombre);
        System.out.println("Especie: " + especie);
        System.out.println("Edad: " + edad + " años");
    }

    public void cumplirAños() {
        edad++;
    }
}

```

Output - trabajoPractico03 (run)

```

run:
Nombre: Anibal
Especie: Perro
Edad: 9 años

Cumpliendo años:
Nombre: Anibal
Especie: Perro
Edad: 10 años
BUILD SUCCESSFUL (total time: 0 seconds)

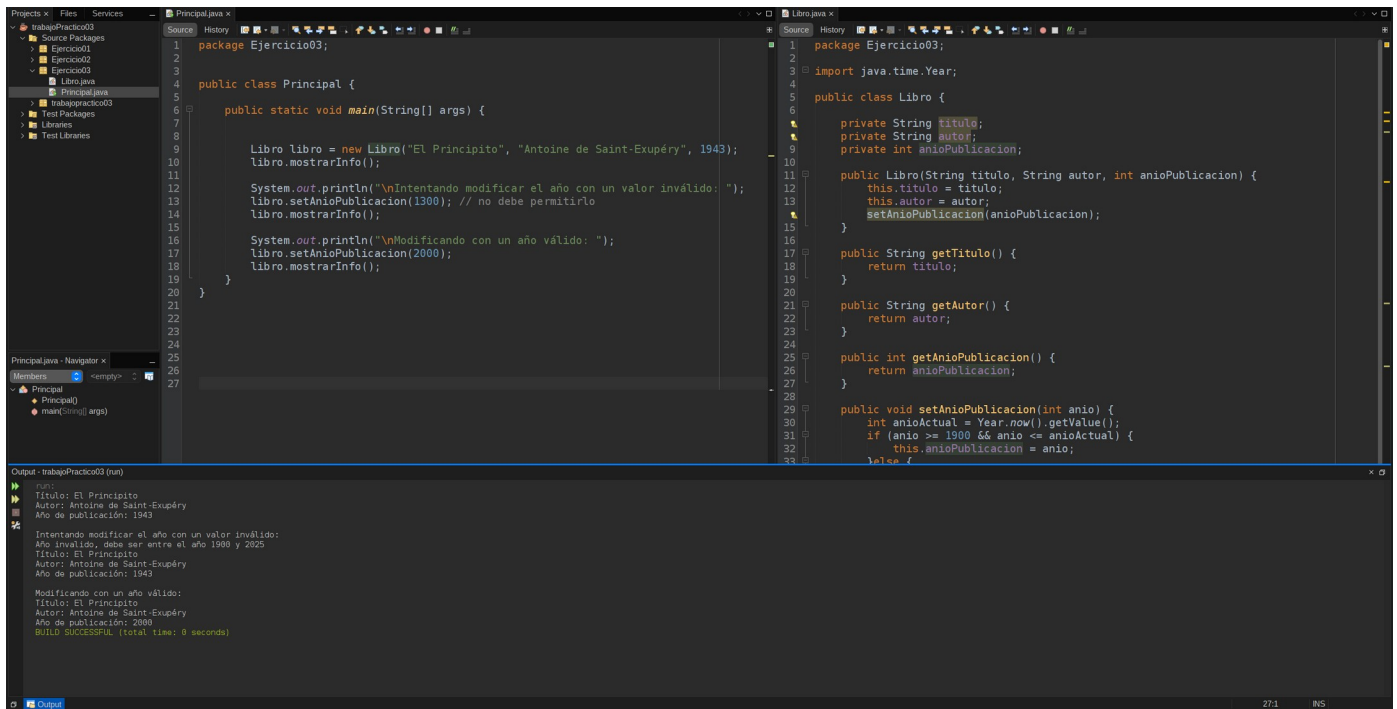
```

## TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN A DISTANCIA

### Ejercicio 3: Encapsulamiento con la Clase Libro

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

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



```

package Ejercicio03;

public class Principal {

    public static void main(String[] args) {

        Libro libro = new Libro("El Principito", "Antoine de Saint-Exupéry", 1943);
        libro.mostrarInfo();

        System.out.println("\nIntentando modificar el año con un valor inválido: ");
        libro.setAñoPublicacion(1300); // no debe permitirlo
        libro.mostrarInfo();

        System.out.println("\nModificando con un año válido: ");
        libro.setAñoPublicacion(2000);
        libro.mostrarInfo();
    }
}

```

```

package Ejercicio03;

import java.time.Year;

public class Libro {

    private String titulo;
    private String autor;
    private int añoPublicacion;

    public Libro(String titulo, String autor, int añoPublicacion) {
        this.titulo = titulo;
        this.autor = autor;
        setAñoPublicacion(añoPublicacion);
    }

    public String getTitulo() {
        return titulo;
    }

    public String getAutor() {
        return autor;
    }

    public int getAñoPublicacion() {
        return añoPublicacion;
    }

    public void setAñoPublicacion(int año) {
        int añoActual = Year.now().getValue();
        if (año >= 1900 && año <= añoActual) {
            this.añoPublicacion = año;
        }
    }
}

```

```

run:
Titulo: El Principito
Autor: Antoine de Saint-Exupéry
Año de publicación: 1943

Intentando modificar el año con un valor inválido:
Año inválido, debe ser entre el año 1900 y 2025
Titulo: El Principito
Autor: Antoine de Saint-Exupéry
Año de publicación: 1943

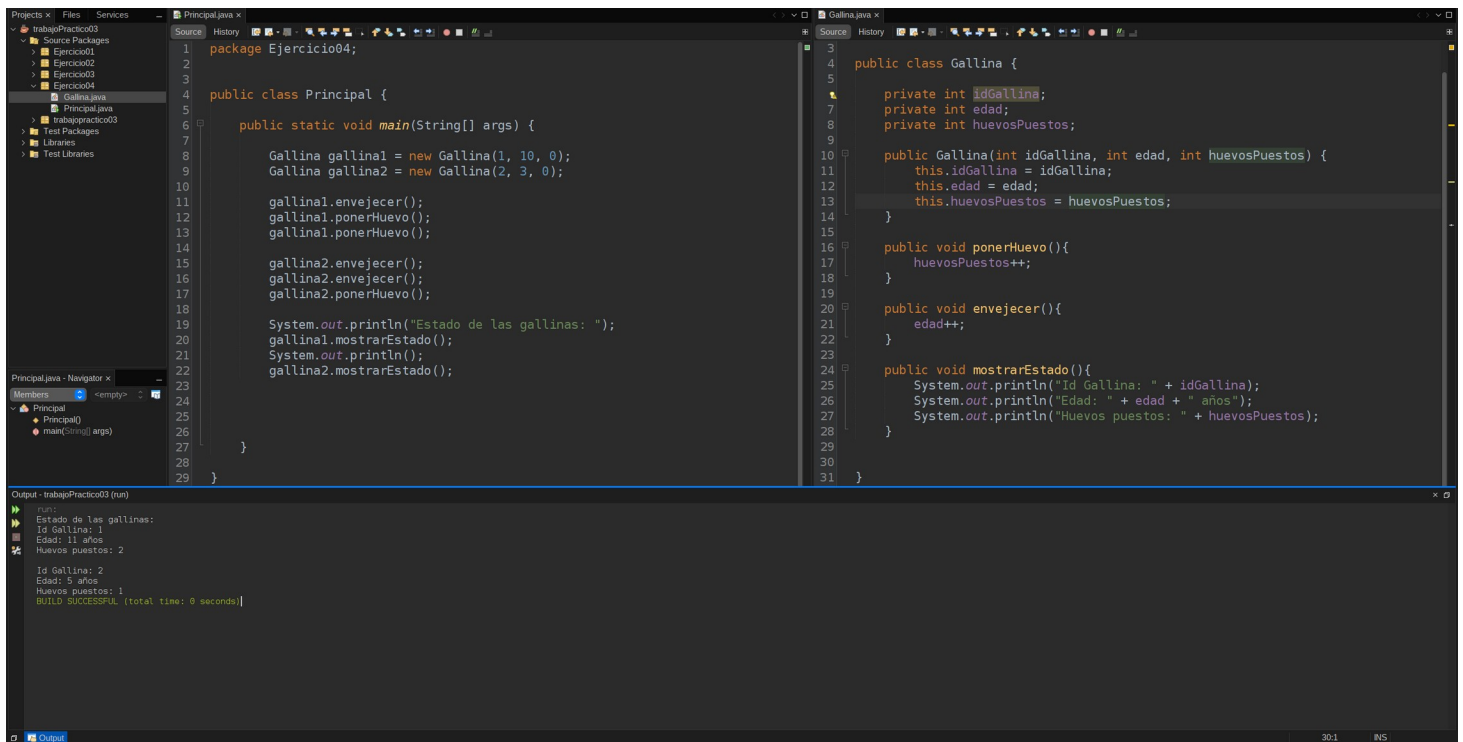
Modificando con un año válido:
Titulo: El Principito
Autor: Antoine de Saint-Exupéry
Año de publicación: 2000
BUILD SUCCESSFUL (total time: 0 seconds)

```

## Ejercicio 4: Gestión de Gallinas en Granja Digital

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

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



```

package Ejercicio04;

public class Principal {

    public static void main(String[] args) {

        Gallina gallina1 = new Gallina(1, 10, 0);
        Gallina gallina2 = new Gallina(2, 3, 0);

        gallina1.envejecer();
        gallina1.ponerHuevo();
        gallina1.mostrarEstado();

        gallina2.envejecer();
        gallina2.ponerHuevo();
        gallina2.mostrarEstado();

        System.out.println("Estado de las gallinas: ");
        gallina1.mostrarEstado();
        System.out.println();
        gallina2.mostrarEstado();
    }
}
    
```

```

public class Gallina {

    private int idGallina;
    private int edad;
    private int huevosPuestos;

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

    public void ponerHuevo(){
        huevosPuestos++;
    }

    public void envejecer(){
        edad++;
    }

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

```

run:
Estado de las gallinas:
Id Gallina: 1
Edad: 11 años
Huevos puestos: 2

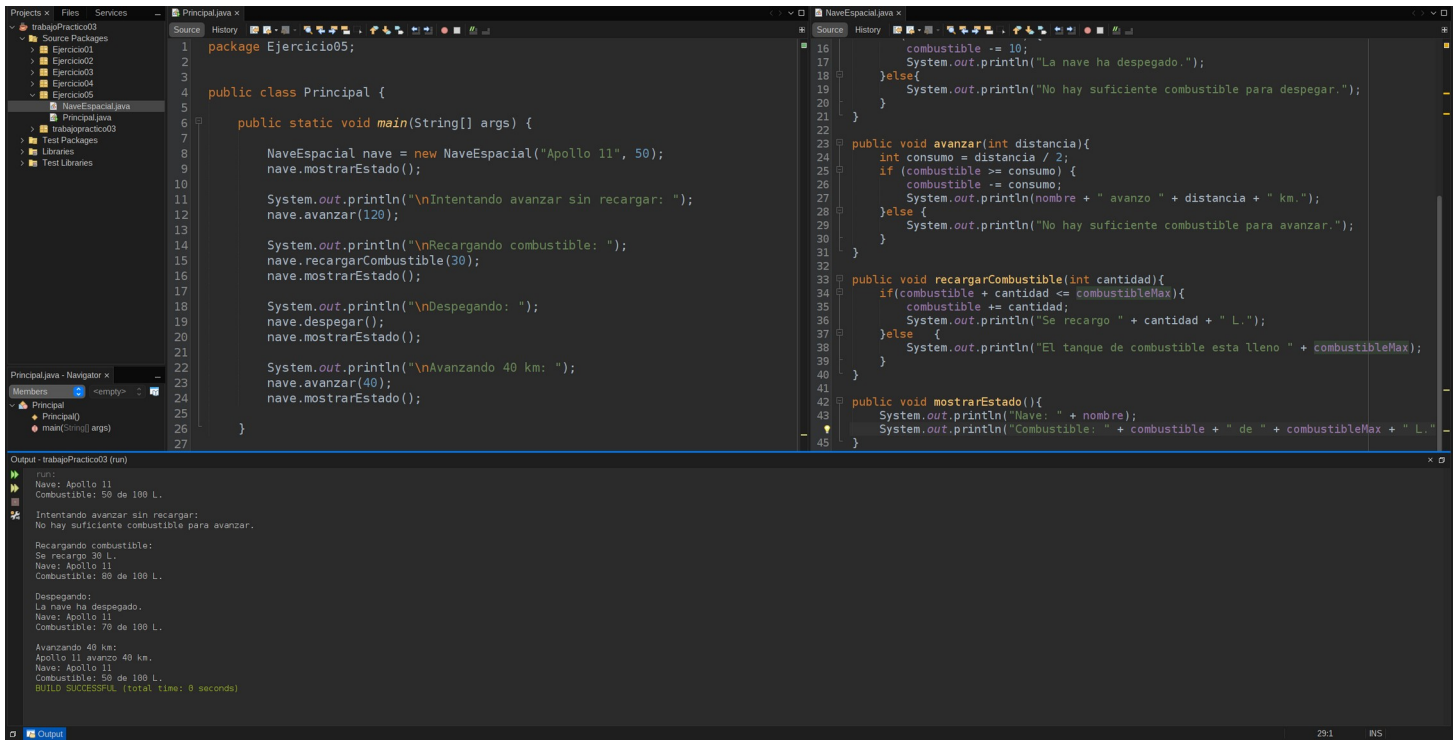
Id Gallina: 2
Edad: 5 años
Huevos puestos: 1
BUILD SUCCESSFUL (total time: 0 seconds)
    
```

## TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN A DISTANCIA

### Ejercicio 5: Simulación de Nave Espacial

Crear una clase NaveEspacial con los atributos: nombre, combustible.

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



```

package Ejercicio05;

public class Principal {

    public static void main(String[] args) {

        NaveEspacial nave = new NaveEspacial("Apollo 11", 50);
        nave.mostrarEstado();

        System.out.println("\nIntentando avanzar sin recargar: ");
        nave.avanzar(120);

        System.out.println("\nRecargando combustible: ");
        nave.recargarCombustible(30);
        nave.mostrarEstado();

        System.out.println("\nDespegando: ");
        nave.despegar();
        nave.mostrarEstado();

        System.out.println("\nAvanzando 40 km: ");
        nave.avanzar(40);
        nave.mostrarEstado();
    }
}
    
```

```

        combustible -= 10;
        System.out.println("La nave ha despegado.");
    }else{
        System.out.println("No hay suficiente combustible para despegar.");
    }
}

    public void avanzar(int distancia){
        int consumo = distancia / 2;
        if (combustible >= consumo) {
            combustible -= consumo;
            System.out.println(nombre + " avanza " + distancia + " km.");
        }else {
            System.out.println("No hay suficiente combustible para avanzar.");
        }
    }

    public void recargarCombustible(int cantidad){
        if(combustible + cantidad <= combustibleMax){
            combustible += cantidad;
            System.out.println("Se recargo " + cantidad + " L.");
        }else {
            System.out.println("El tanque de combustible esta lleno " + combustibleMax);
        }
    }

    public void mostrarEstado(){
        System.out.println("Nave: " + nombre);
        System.out.println("Combustible: " + combustible + " de " + combustibleMax + " L.");
    }
    
```

```

run:
Nave: Apollo 11
Combustible: 50 de 100 L.

Intentando avanzar sin recargar:
No hay suficiente combustible para avanzar.

Recargando combustible:
Se recargo 30 L.
Nave: Apollo 11
Combustible: 80 de 100 L.

Despegando:
La nave ha despegado.
Nave: Apollo 11
Combustible: 70 de 100 L.

Avanzando 40 km:
Apollo 11 avanza 40 km.
Nave: Apollo 11
Combustible: 50 de 100 L.
BUILD SUCCESSFUL (total time: 0 seconds)
    
```