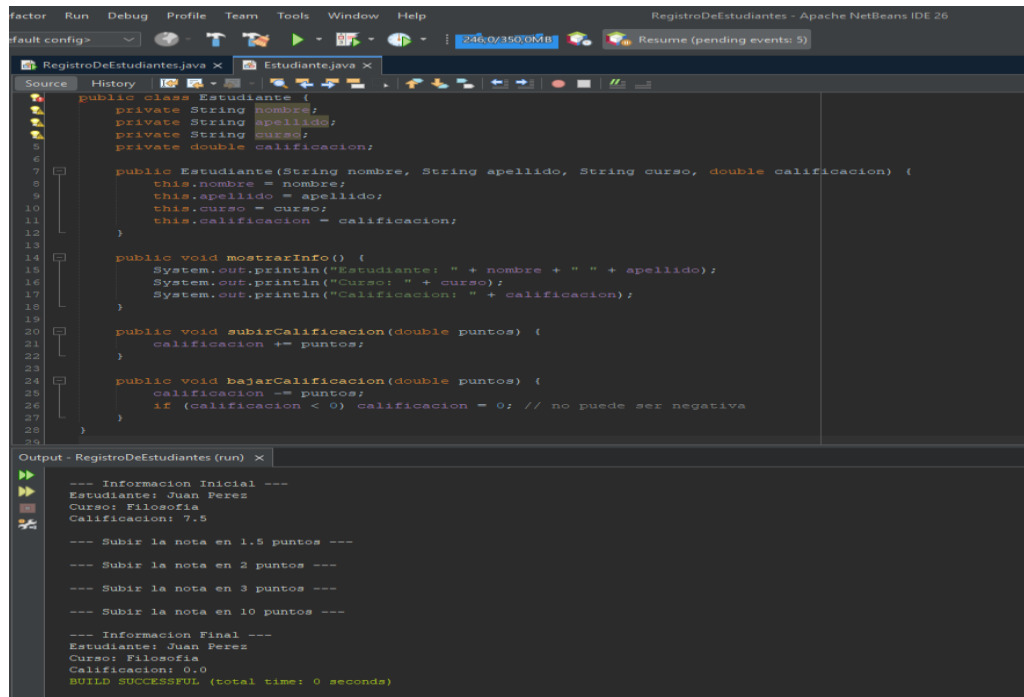


Trabajo Práctico 3: Introducción a la Programación Orientada a Objetos

1)



The screenshot shows the Apache NetBeans IDE with the `Estudiante.java` file open. The code defines a `Estudiante` class with private attributes `nombre`, `apellido`, `curso`, and `calificacion`. It includes a constructor, a `mostrarInfo()` method, and two methods to update the grade: `subirCalificacion()` and `bajarCalificacion()`. The output window shows the execution results, including the initial information and the changes made to the grade.

```
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("Estudiante: " + nombre + " " + apellido);
        System.out.println("Curso: " + curso);
        System.out.println("Calificacion: " + calificacion);
    }

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

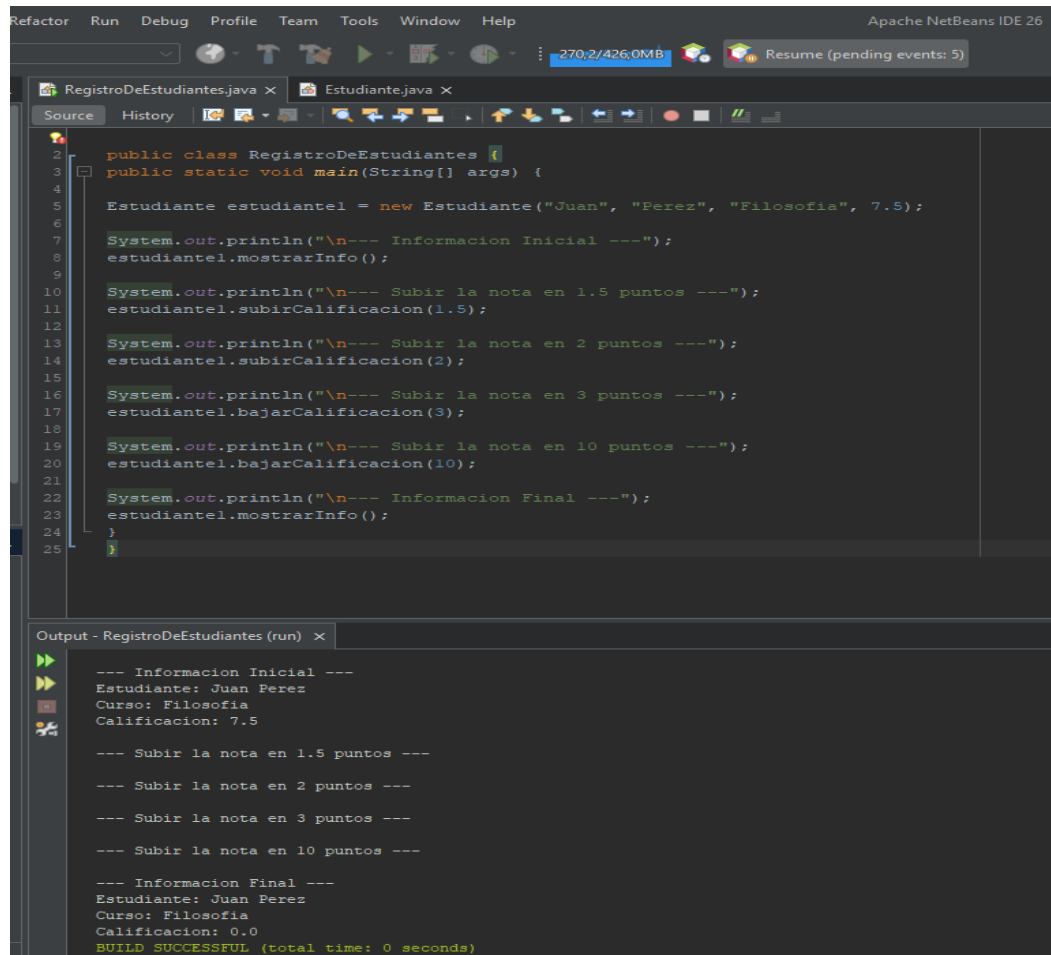
    public void bajarCalificacion(double puntos) {
        calificacion -= puntos;
        if (calificacion < 0) calificacion = 0; // no puede ser negativa
    }
}
```

Output - RegistroDeEstudiantes (run) ×

```
--- Informacion Inicial ---
Estudiante: Juan Perez
Curso: Filosofia
Calificacion: 7.5

--- Subir la nota en 1.5 puntos ---
--- Subir la nota en 2 puntos ---
--- Subir la nota en 3 puntos ---
--- Subir la nota en 10 puntos ---

--- Informacion Final ---
Estudiante: Juan Perez
Curso: Filosofia
Calificacion: 0.0
BUILD SUCCESSFUL (total time: 0 seconds)
```



The screenshot shows the Apache NetBeans IDE with the `RegistroDeEstudiantes.java` file open. The code defines a `RegistroDeEstudiantes` class with a `main` method that creates an `Estudiante` object and calls its methods to update the grade. The output window shows the execution results, including the initial information and the changes made to the grade.

```
public class RegistroDeEstudiantes {
    public static void main(String[] args) {
        Estudiante estudiante1 = new Estudiante("Juan", "Perez", "Filosofia", 7.5);

        System.out.println("\n--- Informacion Inicial ---");
        estudiante1.mostrarInfo();

        System.out.println("\n--- Subir la nota en 1.5 puntos ---");
        estudiante1.subirCalificacion(1.5);

        System.out.println("\n--- Subir la nota en 2 puntos ---");
        estudiante1.subirCalificacion(2);

        System.out.println("\n--- Subir la nota en 3 puntos ---");
        estudiante1.bajarCalificacion(3);

        System.out.println("\n--- Subir la nota en 10 puntos ---");
        estudiante1.bajarCalificacion(10);

        System.out.println("\n--- Informacion Final ---");
        estudiante1.mostrarInfo();
    }
}
```

Output - RegistroDeEstudiantes (run) ×

```
--- Informacion Inicial ---
Estudiante: Juan Perez
Curso: Filosofia
Calificacion: 7.5

--- Subir la nota en 1.5 puntos ---
--- Subir la nota en 2 puntos ---
--- Subir la nota en 3 puntos ---
--- Subir la nota en 10 puntos ---

--- Informacion Final ---
Estudiante: Juan Perez
Curso: Filosofia
Calificacion: 0.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

2)

```

MascotaMain.java x Mascota.java x
Source History
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++;
        System.out.println("Ahora " + nombre + " tiene " + edad + " años.");
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getEspecie() {
        return especie;
    }

    public void setEspecie(String especie) {
        this.especie = especie;
    }

    public int getEdad() {
        return edad;
    }

    public void setEdad(int edad) {
        if (edad >= 0) {
            this.edad = edad;
        } else {
            System.out.println("ERROR: La edad no puede ser negativa.");
        }
    }
}

```

```

MascotaMain.java x Mascota.java x
Source History
public class MascotaMain {
    public static void main(String[] args) {
        Mascota miMascota = new Mascota("Hachiko", "Perro", 7);

        System.out.println("Informacion inicial:");
        miMascota.mostrarInfo();

        miMascota.cumplirAños();
        System.out.println("\nInformacion actualizada:");
        miMascota.mostrarInfo();
    }
}

```

Output - MascotaMain (run) x

```

run:
Informacion inicial:
Nombre: Hachiko
Especie: Perro
Edad: 7 años
Ahora Hachiko tiene 8 años.

Informacion actualizada:
Nombre: Hachiko
Especie: Perro
Edad: 8 años
BUILD SUCCESSFUL (total time: 0 seconds)

```

3)

The screenshot shows the Apache NetBeans IDE with the 'LibroMain.java' file open. The code defines a 'LibroMain' class with a 'main' method that creates a 'Libro' object and calls its 'mostrarInfo()' method. It then demonstrates changing the publication year to 2000, 1800, and 2100, with error handling for years outside the 1900-2025 range.

```

1 public class LibroMain {
2
3     public static void main(String[] args) {
4
5         Libro miLibro = new Libro("Recetas de Cocina", "Gordon Ramsey", 2009);
6
7         System.out.println("Informacion Inicial del Libro:");
8         miLibro.mostrarInfo();
9
10        System.out.println("\nCambiar el año al 2000");
11        miLibro.setAñoPublicacion(2000);
12        miLibro.mostrarInfo();
13
14        System.out.println("\nCambiar el año al 1800");
15        miLibro.setAñoPublicacion(1800);
16        miLibro.mostrarInfo();
17        System.out.println("\nCambiar el año al futuro");
18        miLibro.setAñoPublicacion(2100);
19        miLibro.mostrarInfo();
20    }
21 }
22

```

The 'Output - LibroMain (run)' window shows the following execution results:

```

run:
Informacion Inicial del Libro:
Titulo: Recetas de Cocina
Autor: Gordon Ramsey
Año de Publicacion: 2009

Cambiar el año al 2000
Año de publicacion actualizado correctamente.
Titulo: Recetas de Cocina
Autor: Gordon Ramsey
Año de Publicacion: 2000

Cambiar el año al 1800
Error: El año debe estar entre 1900 y 2025
Titulo: Recetas de Cocina
Autor: Gordon Ramsey
Año de Publicacion: 2000

Cambiar el año al futuro
Error: El año debe estar entre 1900 y 2025
Titulo: Recetas de Cocina
Autor: Gordon Ramsey
Año de Publicacion: 2000
BUILD SUCCESSFUL (total time: 0 seconds)

```

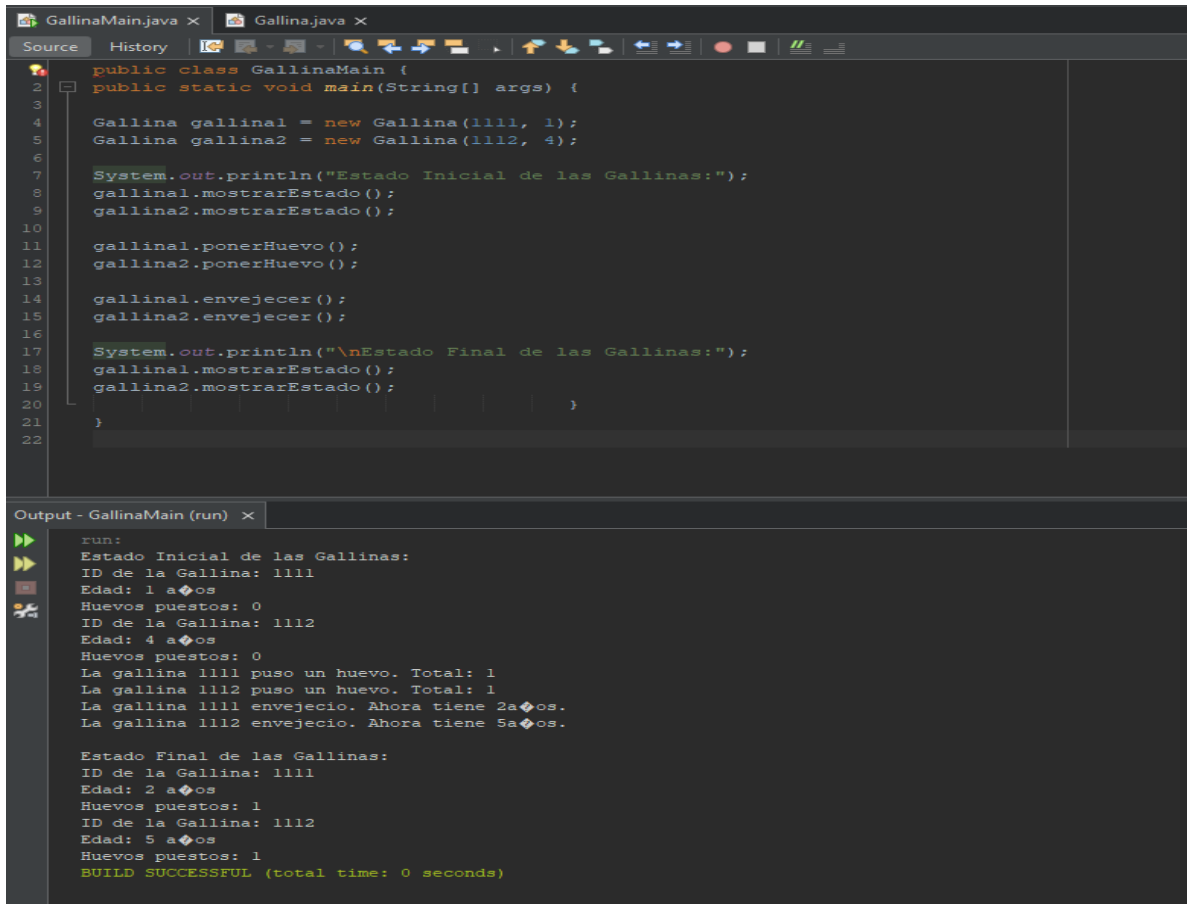
The screenshot shows the 'Libro.java' file in the Apache NetBeans IDE. It defines a 'Libro' class with private attributes for title, author, and publication year, and public methods for getting and setting these attributes. It includes validation logic for the publication year, ensuring it falls within the range of 1900 to the current year.

```

1 import java.time.Year;
2
3 public class Libro {
4
5     private String titulo;
6     private String autor;
7     private int añoPublicacion;
8
9     public Libro(String titulo, String autor, int añoPublicacion) {
10        this.titulo = titulo;
11        this.autor = autor;
12        this.añoPublicacion = añoPublicacion;
13    }
14
15    public String getTitulo() {
16        return titulo;
17    }
18
19    public String getAutor() {
20        return autor;
21    }
22
23    public int getAñoPublicacion() {
24        return añoPublicacion;
25    }
26
27    public void setAñoPublicacion(int nuevoAño) {
28        int añoActual = Year.now().getValue();
29        if (nuevoAño >= 1900 && nuevoAño <= añoActual) {
30            this.añoPublicacion = nuevoAño;
31            System.out.println("Año de publicacion actualizado correctamente.");
32        } else {
33            System.out.println("Error: El año debe estar entre 1900 y " + añoActual);
34        }
35    }
36
37    public void mostrarInfo() {
38        System.out.println("Titulo: " + titulo);
39        System.out.println("Autor: " + autor);
40        System.out.println("Año de Publicacion: " + añoPublicacion);
41    }
42 }
43

```

4)



The screenshot shows an IDE with two tabs: `GallinaMain.java` and `Gallina.java`. The `GallinaMain.java` tab is active, displaying the following code:

```

1 public class GallinaMain {
2     public static void main(String[] args) {
3
4         Gallina gallina1 = new Gallina(1111, 1);
5         Gallina gallina2 = new Gallina(1112, 4);
6
7         System.out.println("Estado Inicial de las Gallinas:");
8         gallina1.mostrarEstado();
9         gallina2.mostrarEstado();
10
11        gallina1.ponerHuevo();
12        gallina2.ponerHuevo();
13
14        gallina1.envejecer();
15        gallina2.envejecer();
16
17        System.out.println("\nEstado Final de las Gallinas:");
18        gallina1.mostrarEstado();
19        gallina2.mostrarEstado();
20    }
21 }
22

```

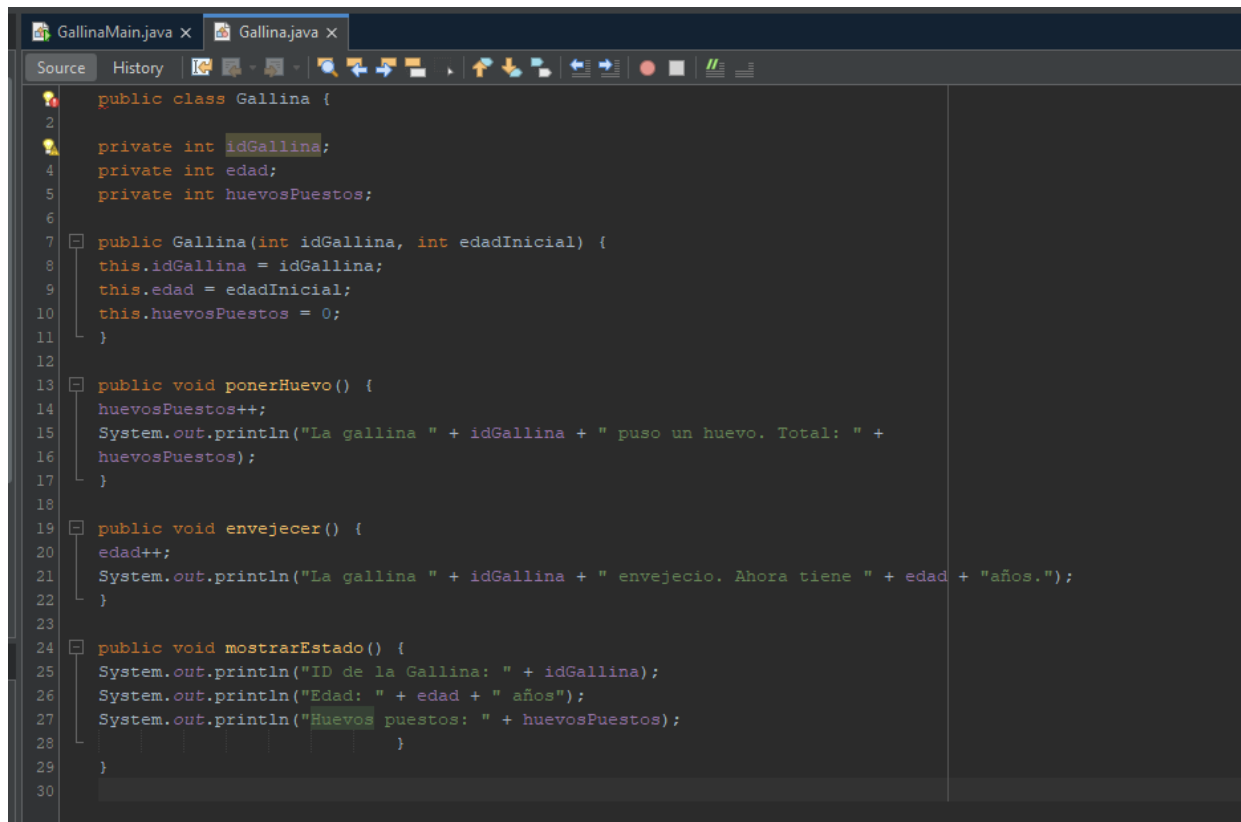
The `Output - GallinaMain (run)` tab shows the execution results:

```

run:
Estado Inicial de las Gallinas:
ID de la Gallina: 1111
Edad: 1 años
Huevos puestos: 0
ID de la Gallina: 1112
Edad: 4 años
Huevos puestos: 0
La gallina 1111 puso un huevo. Total: 1
La gallina 1112 puso un huevo. Total: 1
La gallina 1111 envejecio. Ahora tiene 2años.
La gallina 1112 envejecio. Ahora tiene 5años.

Estado Final de las Gallinas:
ID de la Gallina: 1111
Edad: 2 años
Huevos puestos: 1
ID de la Gallina: 1112
Edad: 5 años
Huevos puestos: 1
BUILD SUCCESSFUL (total time: 0 seconds)

```



The screenshot shows an IDE with two tabs: `GallinaMain.java` and `Gallina.java`. The `Gallina.java` tab is active, displaying the following code:

```

1 public class Gallina {
2
3     private int idGallina;
4     private int edad;
5     private int huevosPuestos;
6
7     public Gallina(int idGallina, int edadInicial) {
8         this.idGallina = idGallina;
9         this.edad = edadInicial;
10        this.huevosPuestos = 0;
11    }
12
13    public void ponerHuevo() {
14        huevosPuestos++;
15        System.out.println("La gallina " + idGallina + " puso un huevo. Total: " +
16        huevosPuestos);
17    }
18
19    public void envejecer() {
20        edad++;
21        System.out.println("La gallina " + idGallina + " envejecio. Ahora tiene " + edad + "años.");
22    }
23
24    public void mostrarEstado() {
25        System.out.println("ID de la Gallina: " + idGallina);
26        System.out.println("Edad: " + edad + " años");
27        System.out.println("Huevos puestos: " + huevosPuestos);
28    }
29 }
30

```

5)

```

NaveEspacialMain.java X  NaveEspacial.java X
Source  History
1 package naveespacialmain;
2 public class NaveEspacial {
3     private String nombre;
4     private int combustible;
5     private static final int combustible_maximo = 210;
6
7     public NaveEspacial(String nombre, int combustibleInicial) {
8         this.nombre = nombre;
9         this.combustible = Math.min(combustibleInicial, combustible_maximo);
10    }
11
12    public void despegar() {
13        if (combustible >= 10) {
14            combustible -= 10;
15            System.out.println(nombre + " ha despegado. Combustible restante: " +
16                combustible);
17        } else {
18            System.out.println("No hay suficiente combustible para despegar. Se requieren al menos 10 unidades.");
19        }
20    }
21
22    public void avanzar(int distancia) {
23        if (combustible >= distancia) {
24            combustible -= distancia;
25            System.out.println(nombre + " ha avanzado " + distancia + " unidades. Combustible restante: " + combustible);
26        } else {
27            System.out.println("No hay suficiente combustible para avanzar " + distancia + " unidades. Combustible actual: " + combustible);
28        }
29    }
30
31    public void recargarCombustible(int cantidad) {
32        if (cantidad <= 0) {
33            System.out.println("No es posible agregar cantidad negativa o cero de combustible.");
34            return;
35        }
36        if (combustible + cantidad > combustible_maximo) {
37            combustible = combustible_maximo;
38            System.out.println("Combustible recargado al máximo.");
39        } else {
40            combustible += cantidad;
41            System.out.println(nombre + " ha recargado " + cantidad + " unidades. Combustible actual: " + combustible);
42        }
43    }
44
45    public void mostrarEstado() {
46        System.out.println("\nNave: " + nombre);
47        System.out.println("Combustible disponible: " + combustible + " unidades");
48    }
49 }

```

```

NaveEspacialMain.java X  NaveEspacial.java X
Source  History
1 package naveespacialmain;
2
3 public class NaveEspacialMain {
4     public static void main(String[] args) {
5
6         NaveEspacial nave1 = new NaveEspacial("412T1B", 50);
7
8         System.out.println("Estado Inicial de la Nave:");
9         nave1.mostrarEstado();
10
11         System.out.println("\nAvanzando 60 unidades sin recargar...");
12         nave1.avanzar(60);
13
14         System.out.println("\nRecargando 40 unidades de combustible...");
15         nave1.recargarCombustible(40);
16
17         System.out.println("\nAvanzando 60 unidades nuevamente...");
18         nave1.avanzar(60);
19
20         System.out.println("\nEstado Final de la Nave:");
21         nave1.mostrarEstado();
22     }
23 }

```

Output - NaveEspacialMain (run) X

```

run:
Estado Inicial de la Nave:
Nave: 412T1B
Combustible disponible: 50 unidades

Avanzando 60 unidades sin recargar...
No hay suficiente combustible para avanzar 60 unidades. Combustible actual: 50

Recargando 40 unidades de combustible...
412T1B ha recargado 40 unidades. Combustible actual: 90

Avanzando 60 unidades nuevamente...
412T1B ha avanzado 60 unidades. Combustible restante: 30

Estado Final de la Nave:
Nave: 412T1B
Combustible disponible: 30 unidades
BUILD SUCCESSFUL (total time: 0 seconds)

```