UNIVERSIDAD MAYOR DE SAN ANDRES

CARRERA DE INFORMÀTICA

PROGRAMACION II

INF - 121

Practica 2

Integrante: Leon Limachi Yessica Belinda

CI: 8329058

Docente: Lic. Felipez

Paralelo: E

Fecha: 15 de septiembre

LA PAZ – BOLIVIA

2025

TEMA: INTRODUCCIÓN A LA PROGRAMACIÓN ORIENTADA A OBJETOS

Ejercicio 2.

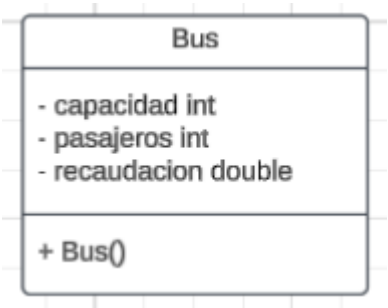
Realiza la abstracción de un Bus.

a) Al bus desean subir X cantidad de pasajeros, actualiza los datos del bus.

b) Crea un método para cobrar pasaje a los pasajeros. (Costo del pasaje: bs. 1.50)'

c) Muestra cuántos asientos quedan disponibles.

d) Crea una instancia del bus y utiliza los métodos de los incisos anteriores.




```java
public void subirPasajeros(int cantidad) {
    if (pasajeros + cantidad <= capacidad) {
        pasajeros += cantidad;
        System.out.println(cantidad + " pasajeros subieron al bus");
    } else {
        int disponibles = capacidad - pasajeros;
        System.out.println("Solo hay " + disponibles + " asientos disponibles");
    }
}

public void cobrarPasaje() {
    double total = pasajeros * 1.50;
    recaudacion += total;
    System.out.println("Se cobro Bs. " + total + " por " + pasajeros + " pasajeros");
}

public void mostrarAsientosDisponibles() {
    int disponibles = capacidad - pasajeros;
    System.out.println("Asientos disponibles: " + disponibles);
}
```

```
run:
30 pasajeros subieron al bus
Se cobro Bs. 45.0 por 30 pasajeros
Asientos disponibles: 20
Solo hay 20 asientos disponibles
Se cobro Bs. 45.0 por 30 pasajeros
Asientos disponibles: 20
BUILD SUCCESSFUL (total time: 0 seconds)
```

Ejercicio 3.

Diseña una clase Producto que tenga atributos nombre, precio y stock. Agrega métodos para vender productos y reabastecer el stock.

a) Define la clase y los atributos.

b) Agrega un método vender(cantidad: Int) que reste del stock

c) Agrega un método reabastecer(cantidad: Int) que sume al stock.


```java
public void vender(int cantidad) {
    if (cantidad <= stock) {
        stock -= cantidad;
        System.out.println("Vendidos: " + cantidad + " " + nombre);
    } else {
        System.out.println("Stock insuficiente. Solo hay: " + stock);
    }
}

public void reabastecer(int cantidad) {
    stock += cantidad;
    System.out.println("Reabastecidos: " + cantidad + " " + nombre);
}

public void mostrarStock() {
    System.out.println("Stock actual de " + nombre + ": " + stock);
}
```

```
Stock actual de Laptop: 10
Vendidos: 3 Laptop
Stock actual de Laptop: 7
Reabastecidos: 5 Laptop
Stock actual de Laptop: 12
Stock insuficiente. Solo hay: 12
BUILD SUCCESSFUL (total time: 1 second)
```

Ejercicio 5.

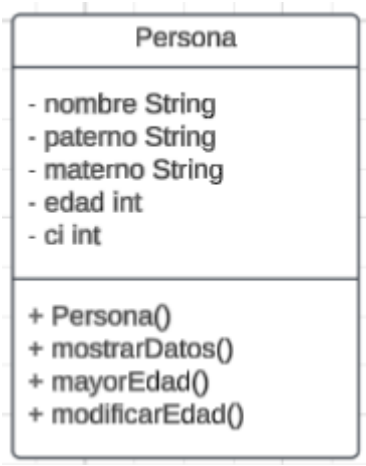
Define la clase Persona (nombre, paterno, materno, edad, ci)

a) Instanciar a dos personas.

b) Implementar método mostrarDatos().

c) Implementar el metodo mayorDeEdad() que determina si la persona es mayor de edad o no.
d) Implementar método modificarEdad(nuevo) que modifica la edad de la persona.

e) Verificar si las dos personas tienen el mismo apellido paterno.




```java
public void mostrarDatos() {
    System.out.println("Nombre: " + nombre);
    System.out.println("Apellidos: " + paterno + " " + materno);
    System.out.println("Edad: " + edad);
    System.out.println("CI: " + ci);
}

public boolean mayorDeEdad() {
    return edad >= 18;
}

public void modificarEdad(int nuevaEdad) {
    edad = nuevaEdad;
    System.out.println("Edad modificada a: " + edad);
}

public String getPaterno() {
    return paterno;
}
```

```
=== Persona 1 ===
Nombre: Juan
Apellidos: Perez Gomez
Edad: 25
CI: 1234567
Es mayor de edad: true

=== Persona 2 ===
Nombre: Maria
Apellidos: Perez Lopez
Edad: 17
CI: 7654321
Es mayor de edad: false
Edad modificada a: 20
Es mayor de edad: true

Tienen el mismo apellido paterno: true
BUILD SUCCESSFUL (total time: 1 second)
```

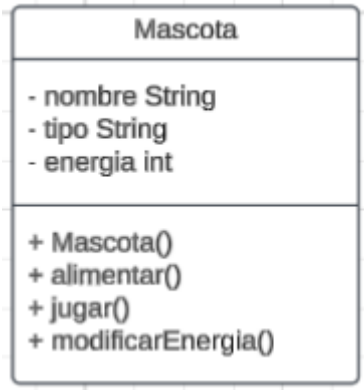
Ejercicio 7.

Dada la clase Mascota.

a) Agrega un método para alimentar (+20 de energía, máximo 100).

b) Agrega un método para jugar (-15 de energía, mínimo 0).

c) Crea dos mascotas, aliméntalas y hazlas jugar, mostrando su energía en cada paso.




```java
public void alimentar() {
    energia += 20;
    if (energia > 100) {
        energia = 100;
    }
    System.out.println(nombre + " fue alimentado. Energia: " + energia);
}

public void jugar() {
    energia -= 15;
    if (energia < 0) {
        energia = 0;
    }
    System.out.println(nombre + " jugo. Energia: " + energia);
}

public void mostrarEnergia() {
    System.out.println(nombre + " tiene " + energia + " de energia");
}
```

```
Firulais tiene 50 de energia
Michi tiene 30 de energia
Firulais fue alimentado. Energia: 70
Michi fue alimentado. Energia: 50
Firulais jugo. Energia: 55
Michi jugo. Energia: 35
Firulais fue alimentado. Energia: 75
Michi jugo. Energia: 20
Firulais tiene 75 de energia
Michi tiene 20 de energia
BUILD SUCCESSFUL (total time: 0 seconds)
```

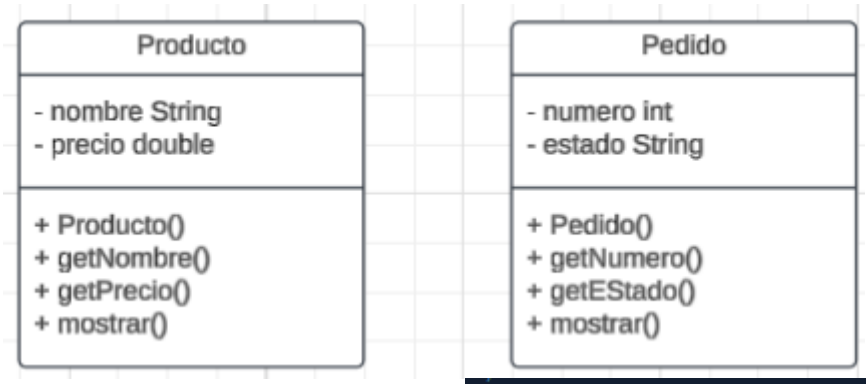
Ejercicio 11.

Analiza el siguiente Texto: En una cafetería del centro de la ciudad, cada día llegan personas a disfrutar de sus bebidas y comidas favoritas. Al ingresar, los clientes observan el menú que contiene una variedad de opciones, desde cafés calientes hasta pasteles recién horneados. Una vez que eligen lo que desean, hacen su pedido al personal que se encuentra detrás del mostrador. Los empleados, organizados en distintos roles según su turno, se encargan de registrar el pedido, prepararlo y entregarlo en la mesa correspondiente. Cada pedido contiene uno o más productos, y puede cambiar de estado según avance el proceso: primero es registrado, luego preparado y finalmente entregado. Los productos disponibles cambian ocasionalmente, ya sea por disponibilidad o por cambios en el menú. Algunos empleados se especializan en tomar pedidos, mientras que otros se enfocan en la preparación. Al finalizar, los clientes pueden pagar y recibir un resumen de lo que consumieron. El sistema de la cafetería guarda todos los pedidos realizados para mantener un historial y mejorar el servicio. Realizar lo siguiente:

a) Encontrar 2 clases con 2 atributos significativos.

b) Implementar las dos clases encontradas con constructor, destructor, getters/setters.

c) Crea un objeto por cada clase y muéstralo




```java
public Producto(String nombre, double precio) {
    this.nombre = nombre;
    this.precio = precio;
}

public String getNombre() {
    return nombre;
}

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

public double getPrecio() {
    return precio;
}

public void setPrecio(double precio) {
    this.precio = precio;
}

public void mostrar() {
    System.out.println("Producto: " + nombre + " - Precio: Bs. " + precio)
}
```

```java
public class Pedido {
    private int numero;
    private String estado;

    public Pedido(int numero, String estado) {
        this.numero = numero;
        this.estado = estado;
    }

    public int getNumero() {
        return numero;
    }

    public void setNumero(int numero) {
        this.numero = numero;
    }

    public String getEstado() {
        return estado;
    }

    public void setEstado(String estado) {
        this.estado = estado;
    }

    public void mostrar() {
        System.out.println("Pedido " + numero + " - Estado: " + e
    }
}
```

```java
public class Main11 {
    public static void main(String[] args) {
        Producto producto1 = new Producto("Cafe Latte", 15.50);
        Pedido pedido1 = new Pedido(101, "Registrado");

        producto1.mostrar();
        pedido1.mostrar();

        producto1.setPrecio(16.00);
        pedido1.setEstado("Preparando");

        producto1.mostrar();
        pedido1.mostrar();
    }
}
```

```
run:
Producto: Cafe Latte - Precio: Bs. 15.5
Pedido 101 - Estado: Registrado
Producto: Cafe Latte - Precio: Bs. 16.0
Pedido 101 - Estado: Preparando
BUILD SUCCESSFUL (total time: 0 seconds)
```

Ejercicio 13.

Considere el siguiente diagrama de clases:

Ejemplo:

nombre : kiwi

tipo : subtropical

nroVitaminas : 3

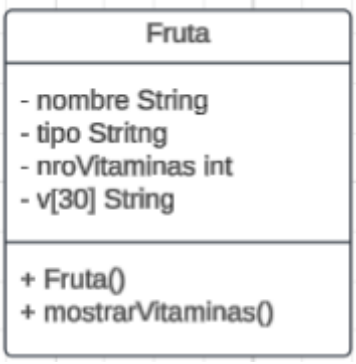
a) Instanciar de 3 maneras diferentes 3 frutas con al menos 2 vitaminas cada una.

b) Verificar cuál es la fruta con más vitaminas.

c) Mostrar las vitaminas que tiene la fruta x.

d) Cuantas vitaminas son cítricas.

e) Ordenar las frutas alfabéticamente según el nombre de sus vitaminas.



```java
public class Fruta {

    String nombre;
    String tipo;
    int nroVitaminas;
    String[] v;

    public Fruta(String nombre, String tipo, String[] vitaminas) {
        this.nombre = nombre;
        this.tipo = tipo;
        this.v = vitaminas;
        this.nroVitaminas = vitaminas.length;
    }

    public void mostrarVitaminas() {
        System.out.print("Vitaminas de " + nombre + ": ");
        for (int i = 0; i < nroVitaminas; i++) {
            System.out.print(v[i] + " ");
        }
        System.out.println();
    }
}
```

```java
public static void main(String[] args) {
    Fruta frutal = new Fruta("Kiwi", "Subtropical", new String[]{"K", "C", "E"});
    Fruta fruta2 = new Fruta("Naranja", "Citrica", new String[]{"C", "A"});
    Fruta fruta3 = new Fruta("Manzana", "Templada", new String[]{"C", "K"});

    System.out.println("Fruta con mas vitaminas:");
    if (frutal.nroVitaminas > fruta2.nroVitaminas && frutal.nroVitaminas > fruta3.nroVitaminas) {
        System.out.println(frutal.nombre);
    } else if (fruta2.nroVitaminas > fruta3.nroVitaminas) {
        System.out.println(fruta2.nombre);
    } else {
        System.out.println(fruta3.nombre);
    }

    System.out.println("\nVitaminas de la Naranja:");
    fruta2.mostrarVitaminas();

    int contadorC = 0;
    for (int i = 0; i < frutal.v.length; i++) {
        if (frutal.v[i].equals("C")) contadorC++;
    }
    for (int i = 0; i < fruta2.v.length; i++) {
        if (fruta2.v[i].equals("C")) contadorC++;
    }
    for (int i = 0; i < fruta3.v.length; i++) {
        if (fruta3.v[i].equals("C")) contadorC++;
    }
    System.out.println("\nTotal de vitaminas C: " + contadorC);
}
```

```
Fruta con mas vitaminas:
Kiwi

Vitaminas de la Naranja:
Vitaminas de Naranja: C A

Total de vitaminas C: 3
BUILD SUCCESSFUL (total time: 0 seconds)
```

TEMA: POLIMORFISMO

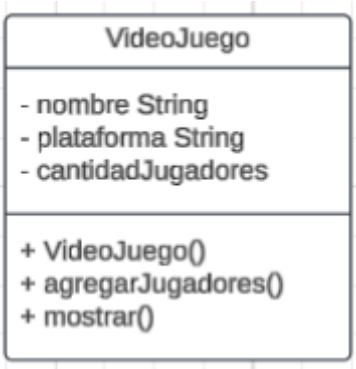
Ejercicio 2.

Sea la clase Videojuego que tenga los atributos nombre, plataforma, cantidad de jugadores. Se pide:

a) Instanciar al menos 2 videojuegos

b) Sobrecargar el constructor 2 veces

c) Sobrecargar el método agregarJugadores() para: agregar solo un jugador, y agregar numero de jugadores según una cantidad ingresada por teclado.




```java
public VideoJuego() {
    this.nombre = "Sin nombre";
    this.plataforma = "Sin plataforma";
    this.cantidadJugadores = 0;
}

public VideoJuego(String nombre, String plataforma) {
    this.nombre = nombre;
    this.plataforma = plataforma;
    this.cantidadJugadores = 0;
}

public VideoJuego(String nombre, String plataforma, int cantidadJugadores) {
    this.nombre = nombre;
    this.plataforma = plataforma;
    this.cantidadJugadores = cantidadJugadores;
}

public void agregarJugadores() {
    cantidadJugadores++;
    System.out.println("Se agrego 1 jugador. Total: " + cantidadJugadores);
}

public void agregarJugadores(int cantidad) {
    cantidadJugadores += cantidad;
    System.out.println("Se agregaron " + cantidad + " jugadores. Total: " + cantidadJugadores);
}

public void mostrarDatos() {
    System.out.println(nombre + " - " + plataforma + " - " + cantidadJugadores + " jugadores");
}
```

```java
public static void main(String[] args) {
    Scanner leer = new Scanner(System.in);

    VideoJuego j1 = new VideoJuego();
    VideoJuego j2 = new VideoJuego("FIFA", "PlayStation");
    VideoJuego j3 = new VideoJuego("Mario Kart", "Nintendo", 4);

    j1.mostrarDatos();
    j2.mostrarDatos();
    j3.mostrarDatos();

    j1.agregarJugadores();
    j2.agregarJugadores();

    System.out.print("Cuantos jugadores agregar a FIFA? ");
    int cantidad = leer.nextInt();
    j2.agregarJugadores(cantidad);

    System.out.print("Cuantos jugadores agregar a Mario Kart? ");
    cantidad = leer.nextInt();
    j3.agregarJugadores(cantidad);

    j1.mostrarDatos();
    j2.mostrarDatos();
    j3.mostrarDatos();

}
```

```
Sin nombre - Sin plataforma - 0 jugadores
FIFA - PlayStation - 0 jugadores
Mario Kart - Nintendo - 4 jugadores
Se agrego 1 jugador. Total: 1
Se agrego 1 jugador. Total: 1
Cuantos jugadores agregar a FIFA? 1
Se agregaron 1 jugadores. Total: 2
Cuantos jugadores agregar a Mario Kart? 1
Se agregaron 1 jugadores. Total: 5
Sin nombre - Sin plataforma - 1 jugadores
FIFA - PlayStation - 2 jugadores
Mario Kart - Nintendo - 5 jugadores
BUILD SUCCESSFUL (total time: 19 seconds)
```

Ejercicio 3.

Sea la clase Matriz(float matriz[10][10])

a) Implementar un constructor para instanciar un objeto con valores predeterminados(matriz identidad)

b) Implementar un constructor para Instanciar un objeto matriz

c) Implementar los metodos para sumar(Matriz matriz) y restar(Matriz matriz)

d) Implementar un método igual(Matriz matriz


```java
public Matriz() {
    matriz = new float[10][10];
    for(int i = 0; i < 10; i++) {
        for(int j = 0; j < 10; j++) {
            if(i == j) {
                matriz[i][j] = 1.0f;
            } else {
                matriz[i][j] = 0.0f;
            }
        }
    }
}

public Matriz(float[][] valores) {
    matriz = new float[10][10];
    for(int i = 0; i < 10; i++) {
        for(int j = 0; j < 10; j++) {
            matriz[i][j] = valores[i][j];
        }
    }
}
```

```java
public void mostrar() {
    for(int i = 0; i < 10; i++) {
        for(int j = 0; j < 10; j++) {
            System.out.print(matriz[i][j] + " ");
        }
        System.out.println();
    }
}
```

```java
public Matriz sumar(Matriz otra) {
    float[][] resultado = new float[10][10];
    for(int i = 0; i < 10; i++) {
        for(int j = 0; j < 10; j++) {
            resultado[i][j] = this.matriz[i][j] + otra.matriz[i][j];
        }
    }
    return new Matriz(resultado);
}

public Matriz restar(Matriz otra) {
    float[][] resultado = new float[10][10];
    for(int i = 0; i < 10; i++) {
        for(int j = 0; j < 10; j++) {
            resultado[i][j] = this.matriz[i][j] - otra.matriz[i][j];
        }
    }
    return new Matriz(resultado);
}

public boolean igual(Matriz otra) {
    for(int i = 0; i < 10; i++) {
        for(int j = 0; j < 10; j++) {
            if(this.matriz[i][j] != otra.matriz[i][j]) {
                return false;
            }
        }
    }
    return true;
}
```

```
Matriz identidad:
1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0

Matriz personalizada:
0.0 1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0
1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0 10.0
2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0 10.0 11.0
3.0 4.0 5.0 6.0 7.0 8.0 9.0 10.0 11.0 12.0
4.0 5.0 6.0 7.0 8.0 9.0 10.0 11.0 12.0 13.0
5.0 6.0 7.0 8.0 9.0 10.0 11.0 12.0 13.0 14.0
6.0 7.0 8.0 9.0 10.0 11.0 12.0 13.0 14.0 15.0
7.0 8.0 9.0 10.0 11.0 12.0 13.0 14.0 15.0 16.0
8.0 9.0 10.0 11.0 12.0 13.0 14.0 15.0 16.0 17.0
9.0 10.0 11.0 12.0 13.0 14.0 15.0 16.0 17.0 18.0
```

```
Suma:
1.0 1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0
1.0 3.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0 10.0
2.0 3.0 5.0 5.0 6.0 7.0 8.0 9.0 10.0 11.0
3.0 4.0 5.0 7.0 7.0 8.0 9.0 10.0 11.0 12.0
4.0 5.0 6.0 7.0 9.0 9.0 10.0 11.0 12.0 13.0
5.0 6.0 7.0 8.0 9.0 11.0 11.0 12.0 13.0 14.0
6.0 7.0 8.0 9.0 10.0 11.0 13.0 13.0 14.0 15.0
7.0 8.0 9.0 10.0 11.0 12.0 13.0 15.0 15.0 16.0
8.0 9.0 10.0 11.0 12.0 13.0 14.0 15.0 17.0 17.0
9.0 10.0 11.0 12.0 13.0 14.0 15.0 16.0 17.0 19.0

Resta:
-1.0 1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0
1.0 1.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0 10.0
2.0 3.0 3.0 5.0 6.0 7.0 8.0 9.0 10.0 11.0
3.0 4.0 5.0 5.0 7.0 8.0 9.0 10.0 11.0 12.0
4.0 5.0 6.0 7.0 7.0 9.0 10.0 11.0 12.0 13.0
5.0 6.0 7.0 8.0 9.0 9.0 11.0 12.0 13.0 14.0
6.0 7.0 8.0 9.0 10.0 11.0 11.0 13.0 14.0 15.0
7.0 8.0 9.0 10.0 11.0 12.0 13.0 13.0 15.0 16.0
8.0 9.0 10.0 11.0 12.0 13.0 14.0 15.0 15.0 17.0
9.0 10.0 11.0 12.0 13.0 14.0 15.0 16.0 17.0 17.0

Son iguales? false
```

Ejercicio 5.

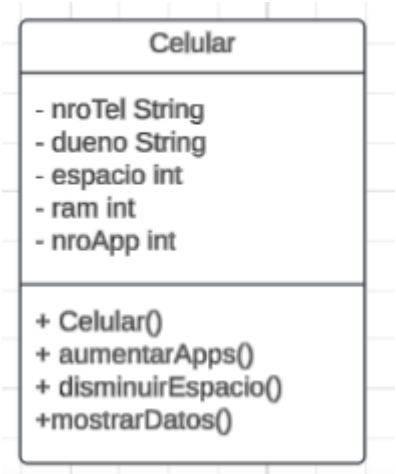
Dada la clase celular

a) realizar el diagrama de clases

b) realizar la sobrecarga del operador ++ para aumentar en 10 el nroApp.

c) realizar la sobrecarga del operador - - para disminuir el espacio en 5 gb.

d) mostrar los datos antes y después de los operadores.



```java
public class Celular {

    String nroTel;
    String dueño;
    int espacio;
    int ram;
    int nroApp;

    public Celular(String nroTel, String dueño, int espacio, int ram, int nroApp) {
        this.nroTel = nroTel;
        this.dueño = dueño;
        this.espacio = espacio;
        this.ram = ram;
        this.nroApp = nroApp;
    }

    public void aumentarApps() {
        nroApp += 10;
    }

    public void disminuirEspacio() {
        espacio -= 5;
    }

    public void mostrarDatos() {
        System.out.println("Nro: " + nroTel);
        System.out.println("Dueno: " + dueño);
        System.out.println("Espacio: " + espacio + " GB");
        System.out.println("RAM: " + ram + " GB");
        System.out.println("Apps: " + nroApp);
        System.out.println("------------------");
    }
}
```

```java
public class Main25 {
    public static void main(String[] args) {
        Celular celular = new Celular("77712345", "Juan", 64, 4, 20);

        System.out.println("ANTES:");
        celular.mostrarDatos();

        celular.aumentarApps();
        celular.disminuirEspacio();

        System.out.println("DESPUES:");
        celular.mostrarDatos();

        celular.aumentarApps();
        celular.disminuirEspacio();

        System.out.println("DESPUES DE SEGUNDA OPERACIEN:");
        celular.mostrarDatos();
    }
}
```

```
run:
ANTES:
Nro: 77712345
Dueno: Juan
Espacio: 64 GB
RAM: 4 GB
Apps: 20
------------------
DESPUES:
Nro: 77712345
Dueno: Juan
Espacio: 59 GB
RAM: 4 GB
Apps: 30
------------------
DESPUES DE SEGUNDA OPERACIEN:
Nro: 77712345
Dueno: Juan
Espacio: 54 GB
RAM: 4 GB
Apps: 40
------------------
BUILD SUCCESSFUL (total time: 2 seconds
```