

TAREA – HITO 3 PILAS

ESTRUCTURA DE DATOS

ESTUDIANTE: CARLOS DANIEL FLORES PAUCARA

DOCENTE: WILLIAM BARRA PAREDEZ

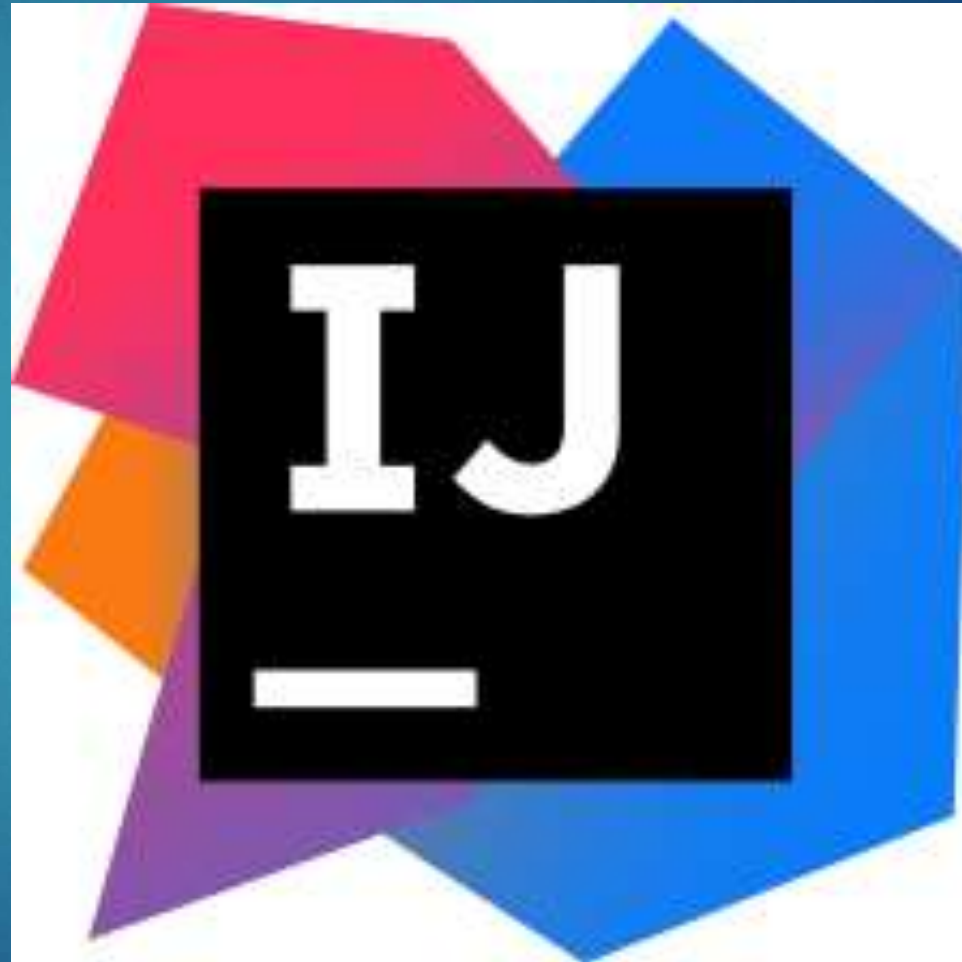
CARRERA: INGENIERÍA DE SISTEMAS

SEMESTRE: 3RO

Manejo De Conceptos

1.- ¿A que se refiere cuando se habla de ESTRUCTURA DE DATOS?

La estructura de datos se refiere a la organización y almacenamiento de datos en un sistema informático de manera que pueden ser utilizados de manera eficiente y efectiva por el software. En programación, las estructuras de datos son herramientas que permiten a los desarrolladores organizar y manipular datos de diferentes tipos y tamaños. La elección de una estructura de datos adecuada depende de los requisitos del programa y del tipo de datos que se están manejando. Algunas de las estructuras de datos más comunes incluyen matrices, listas enlazadas, árboles, grafos, pilas y colas, entre otras. Las estructuras de datos son fundamentales en la programación y son esenciales para la creación de algoritmos y aplicaciones eficientes y escalables.



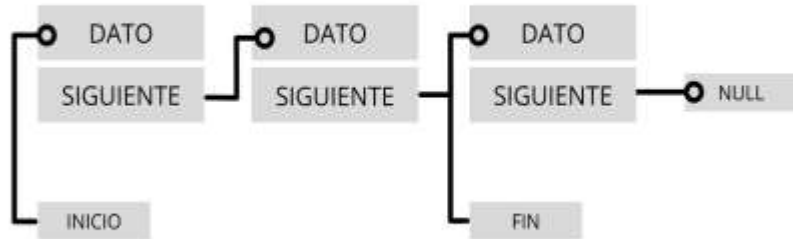
2.-¿Cuáles son los TIPOS DE ESTRUCTURA QUE EXISTE ?

Existen tres tipos de estructura de datos :

Listas enlazadas

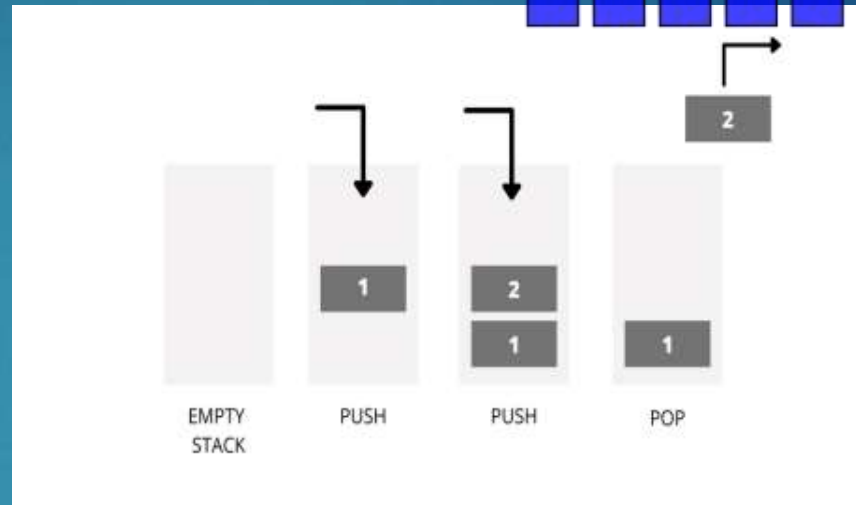
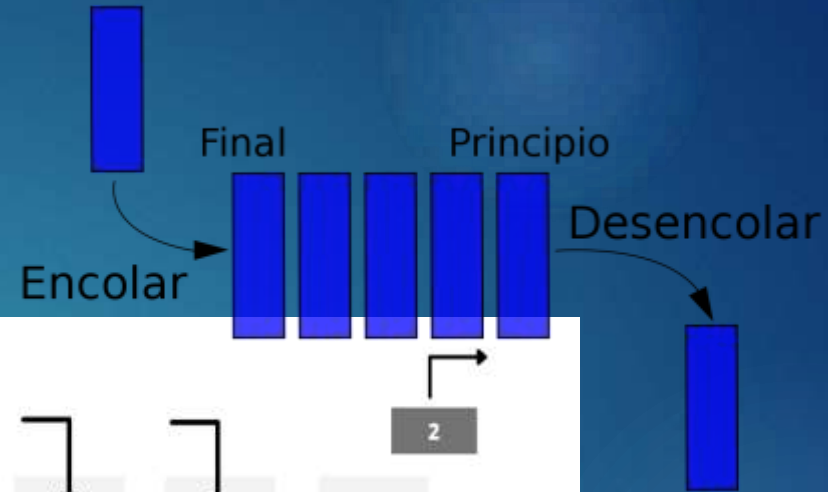
Pilas

Colas



ESTRUCTURAS DE DATOS LINEALES

1. Arreglos (Arrays)
2. Listas (Linked lists)
3. Pilas (Stacks)
4. Colas (Queues)



ESTRUCTURAS DE DATOS NO LINEALES

1. Arboles(Trees)
2. Grafos (Graphs)
3. Tablas hash (Hash tables)
4. Heaps

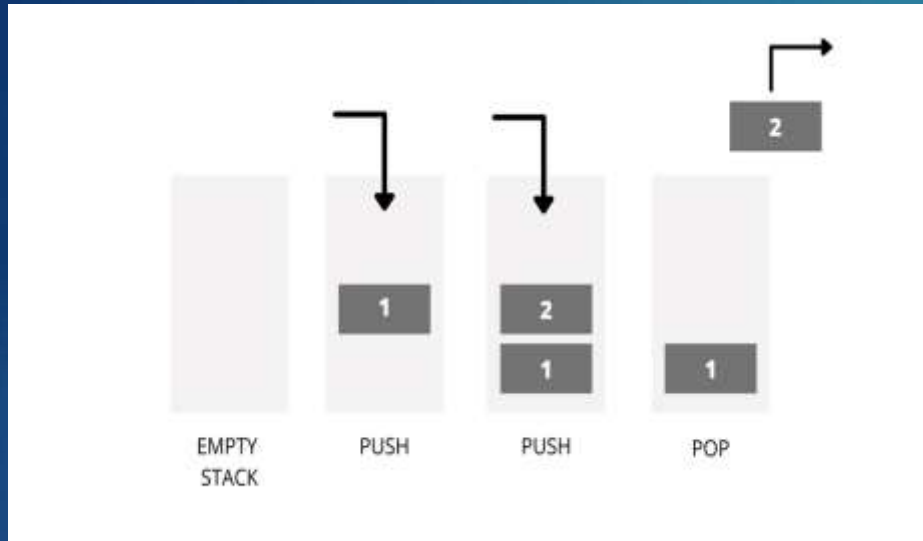
3.- ¿Apoyándose en el link adjunto, explique, **por qué son útiles las estructuras de datos?**

Las estructuras de datos son útiles porque permiten organizar y almacenar información de manera eficiente y accesible. Al utilizar una estructura de datos adecuada, se puede procesar y recuperar información de manera más rápida y fácil que si se utiliza una estructura de datos inapropiada o no se utiliza ninguna estructura de datos.

Por ejemplo, una lista enlazada es una estructura de datos que se utiliza para almacenar una secuencia de elementos en orden. Cada elemento de la lista contiene un puntero que apunta al siguiente elemento en la lista. Si se desea insertar un nuevo elemento en la lista, se puede hacer simplemente modificando los punteros apropiados, lo que es mucho más eficiente que mover todos los elementos en una matriz.



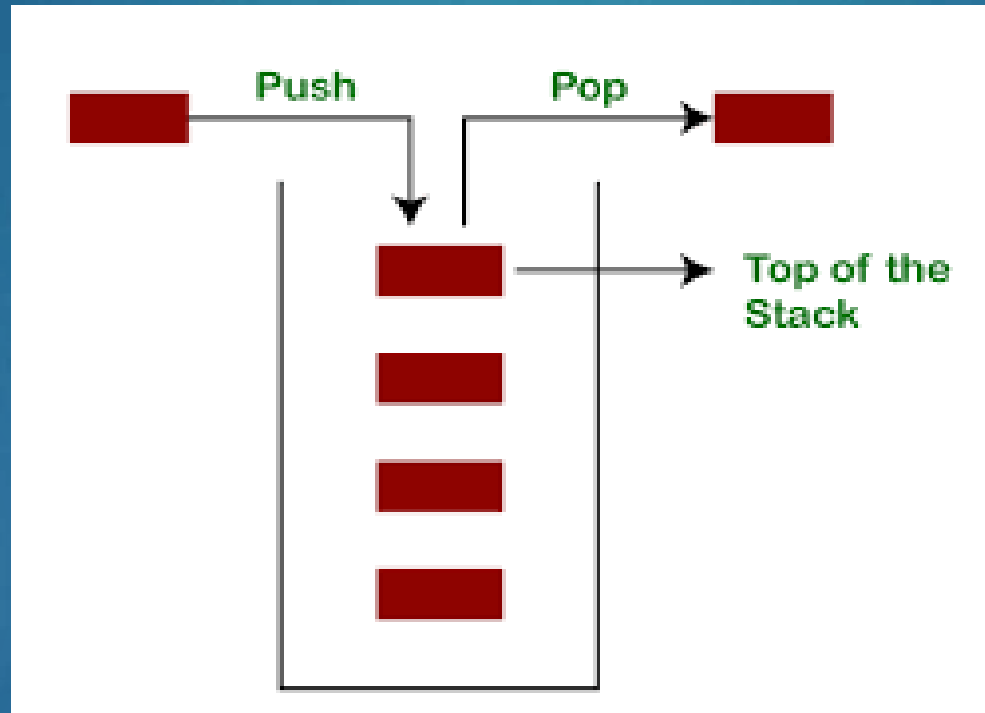
4.-¿Qué es una PILA?



La pila es un tipo especial de lista lineal dentro de las estructuras de datos dinámicas que permite almacenar y recuperar datos, siendo el modo de acceso a sus elementos de tipo LIFO (del inglés Last In, First Out, es decir, último en entrar, primero en salir). ¿Cómo funciona? A través de dos operaciones básicas: apilar (push), que coloca un objeto en la pila, y su operación inversa, desapilar (pop), que retira el último elemento apilado.

5.-¿Qué es **STACK en JAVA**, una STACK sera lo mismo que una PILA?

- En Java, una "stack" es una estructura de datos que funciona como una pila, por lo que se puede considerar que "stack" y "pila" son términos equivalentes en este contexto.



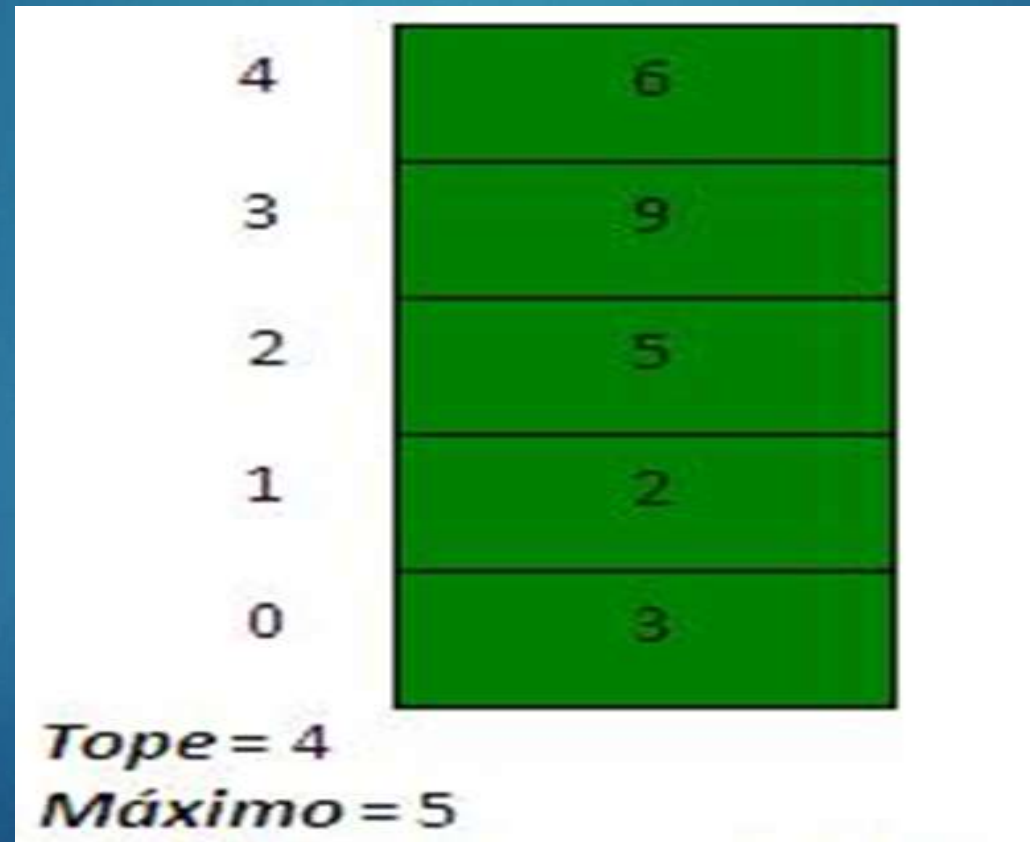
6.-¿Qué es **TOPE** en una PILA?

- En una pila, el "tope" (también conocido como "cima" o "top" en inglés) se refiere al elemento que se encuentra en la posición más alta de la pila. Es decir, el tope es el elemento más recientemente agregado a la pila y es el primer elemento que se retiraría si se realizara una operación de eliminación en la pila.



7.-¿Qué es **MAX** en una PILA?

- En una pila, "MAX" se refiere al elemento máximo actual de la pila. En otras palabras, el elemento más grande que se encuentra actualmente en la pila.



8.-¿A que se refiere los métodos **esVacia()** y **esLLena()** en una PILA?

- ▶ El método "esVacia()" se utiliza para determinar si la pila está vacía, es decir, si no hay ningún elemento en la pila. Este método devuelve "true" si la pila está vacía y "false" si hay uno o más elementos en la pila.
- ▶ Método "esLLena()" se utiliza para determinar si la pila está llena, es decir, si se ha alcanzado el tamaño máximo de la pila y no se pueden agregar más elementos.

```
public boolean esVacio(){  
    if (tope==0){  
        return true;  
    }  
    else{  
        return false;  
    }  
}
```

```
public boolean esllena(){  
    if (tope==max){  
        return true;  
    }  
    else{  
        return false;  
    }  
}
```

9.-¿Qué son los **métodos estáticos** en JAVA?

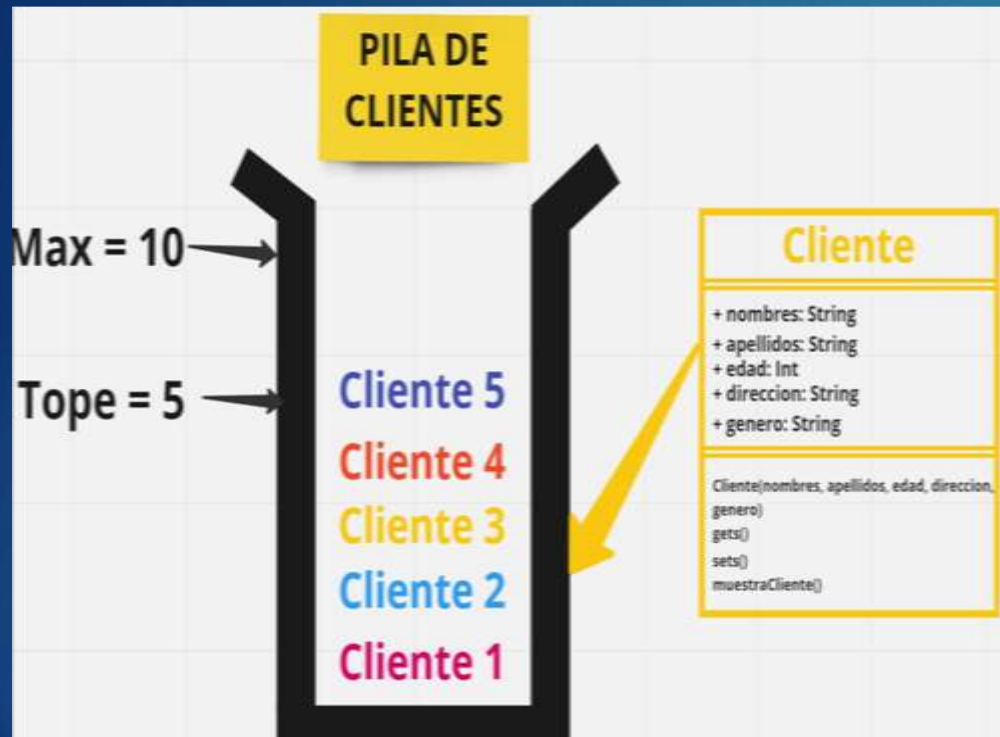
- ▶ En Java, los métodos estáticos son métodos que se definen en la clase en lugar de en una instancia específica de la clase. Es decir, un método estático puede ser llamado sin necesidad de crear una instancia de la clase en la que está definida.
- ▶ Para definir un método estático en Java, se utiliza la palabra clave "static" en la firma del método. Por ejemplo, el siguiente método "saludar" es un método estático:
- ▶ `public static void saludar() { System.out.println("¡Hola!"); }`

10.¿A través de un **gráfico**, muestre los **métodos mínimos** que debería de tener una PILA?

PilaLibros		
m	PilaLibros ()	
m	esVacio ()	boolean
m	nroElem ()	int
m	mostrar ()	void
m	vaciar (PilaLibros)	void
m	esllena ()	boolean
m	adicionar (Libro)	void
m	eliminar ()	Libro

Parte practica

11. Crear las clases necesarias para la **PILA DE CLIENTES**.



1. Crear la clase clientes.
2. Crear la clase PilaCliente.
3. Crear la clase main
4. Crear un paquete de nombre PilaDeClientes

```
package PilaDeClientes;

public class MainCliente {

    public static void main(String[] args) {

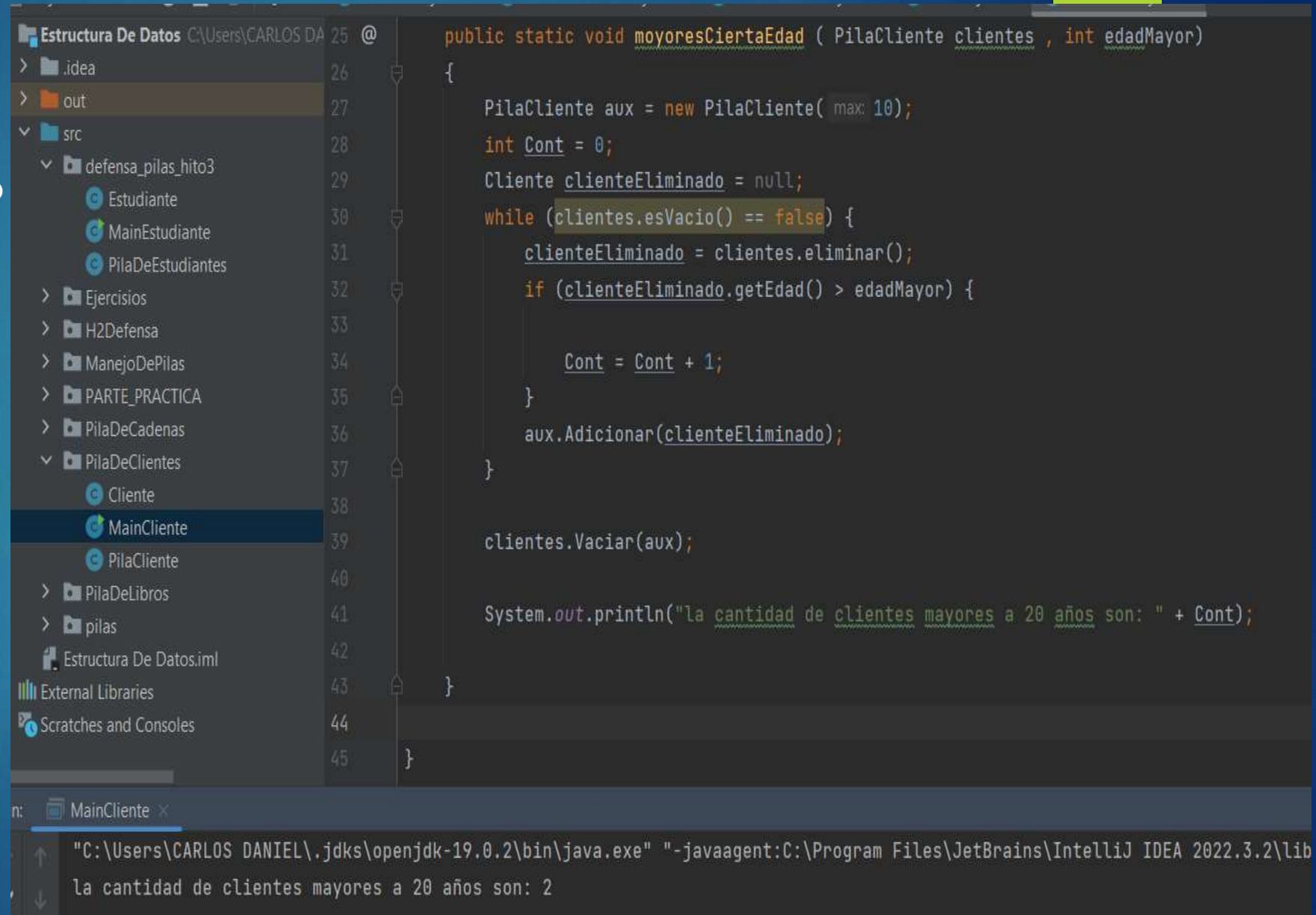
        Cliente Cli1 = new Cliente( nombres: " Jose ", apellidos: " Oblitas ", edad: 22 , direccion: "Achocalla calle 1 " , genero: " Masculino ");
        Cliente Cli2 = new Cliente( nombres: " Miguel ", apellidos: " Choque ", edad: 21 , direccion: "Achocalla calle 2 " , genero: " Masculino ");
        Cliente Cli3 = new Cliente( nombres: " Victor ", apellidos: " Quispe ", edad: 20 , direccion: "Achocalla calle 3 " , genero: " Masculino ");
        Cliente Cli4 = new Cliente( nombres: " Angela ", apellidos: " Mamani ", edad: 19 , direccion: "Achocalla calle 4 " , genero: " Femenino ");
        Cliente Cli5 = new Cliente( nombres: " Jima ", apellidos: " Mayta ", edad: 18 , direccion: "Achocalla calle 5 " , genero: " Femenino ");

        PilaCliente Pcliente = new PilaCliente( max: 10);

        Pcliente.Adicionar(Cli1);
        Pcliente.Adicionar(Cli2);
        Pcliente.Adicionar(Cli3);
        Pcliente.Adicionar(Cli4);
        Pcliente.Adicionar(Cli5);
    }
}
```


12.-DETERMINAR CUANTOS CLIENTES SON MAYORES DE 20 AÑOS

1. El método deberá llamarse mayoresCiertaEdad(Pila, edadMayor)
2. El método debe ser creado en la clase MAIN como un método estático.
3. El método recibe 2 parámetros
 - La Pila de Clientes
 - El valor de la edad



```
public static void mayoresCiertaEdad ( PilaCliente clientes , int edadMayor)
{
    PilaCliente aux = new PilaCliente( max: 10);
    int Cont = 0;
    Cliente clienteEliminado = null;
    while (clientes.esVacio() == false) {
        clienteEliminado = clientes.eliminar();
        if (clienteEliminado.getEdad() > edadMayor) {
            Cont = Cont + 1;
        }
        aux.Adicionar(clienteEliminado);
    }

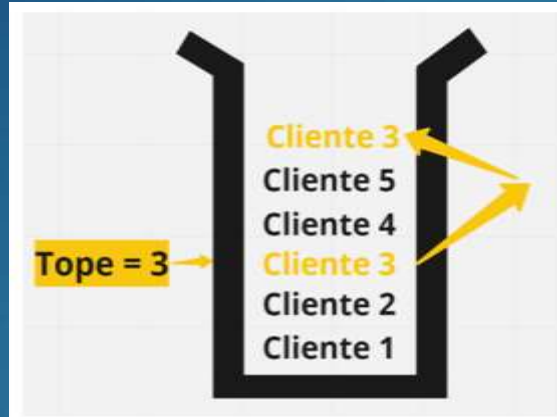
    clientes.Vaciar(aux);

    System.out.println("la cantidad de clientes mayores a 20 años son: " + Cont);
}
```

la cantidad de clientes mayores a 20 años son: 2

13.-MOVER EL KESIMO ELEMENTO DE LA PILA

- ▶ 1. El método deberá llamarse `kEsimoPosicion(Pila,valorTope)`
- ▶ 2. El método debe ser creado en la clase MAIN como un método estático.
- ▶ 3. El método recibe 2 parámetros
 - ▶ - La Pila de Clientes
 - ▶ - El valor(int) de la posición que moverá al final de la pila



```
public static void kEsimoPosicion(PilaCliente pila, int valorTope){  
  
    PilaCliente aux = new PilaCliente();  
    Cliente clienteeliminado = null;  
    Cliente almacli = null;  
    while(!pila.esVacio()){  
  
        clienteeliminado = pila.eliminarCliente();  
  
        if(pila.nroClientes() +1 == valorTope){  
            almacli = clienteeliminado;  
        }  
        else{  
            aux.adicionarClientes(clienteeliminado);  
        }  
    }  
    pila.vaciar(aux);  
    pila.adicionarClientes(almacli);  
  
}
```

Mostrar a los Clientes

DATOS DEL CLIENTE
Nombre: Eddy
Apellido: Apaza
Edad: 25
Direccion: direccion5
Genero: Masculino

DATOS DEL CLIENTE
Nombre: Carolina
Apellido: Paucara
Edad: 22
Direccion: direccion4
Genero: Femenino

DATOS DEL CLIENTE
Nombre: Juan
Apellido: Mamani
Edad: 20
Direccion: direccion3
Genero: Masculino

DATOS DEL CLIENTE
Nombre: Yamiley
Apellido: Lipe
Edad: 18
Direccion: direccion2
Genero: Femenino

DATOS DEL CLIENTE
Nombre: Luis
Apellido: Alvarez
Edad: 19
Direccion: direccion1
Genero: Masculino

Mostrar a los Clientes

DATOS DEL CLIENTE
Nombre: Juan
Apellido: Mamani
Edad: 20
Direccion: direccion3
Genero: Masculino

DATOS DEL CLIENTE
Nombre: Eddy
Apellido: Apaza
Edad: 25
Direccion: direccion5
Genero: Masculino

DATOS DEL CLIENTE
Nombre: Carolina
Apellido: Paucara
Edad: 22
Direccion: direccion4
Genero: Femenino

DATOS DEL CLIENTE
Nombre: Yamiley
Apellido: Lipe
Edad: 18
Direccion: direccion2
Genero: Femenino

DATOS DEL CLIENTE
Nombre: Luis
Apellido: Alvarez
Edad: 19
Direccion: direccion1
Genero: Masculino

14.-CAMBIAR LA DIRECCION DE ALGUNOS CLIENTES DE LA PILA.

- ▶ 1. El método deberá llamarse `asignaDireccion(Pila,nuevaDireccion)`
- ▶ 2. El método debe ser creado en la clase MAIN como un método estático.
- ▶ 3. El método recibe 2 parámetros
 - ▶ - La Pila de Clientes
 - ▶ - El valor(String) de la nueva dirección.
- ▶ 5. Cambiar la dirección del cliente siempre y cuando el genero sea FEMENINO



```
public static void asignaDireccion(PilaCliente pila, String nuevaDireccion){

    PilaCliente aux = new PilaCliente();
    Cliente clienteeliminado = null;
    while(!pila.esVacio()){

        clienteeliminado = pila.eliminarCliente();

        if(clienteeliminado.getGenero().equals("Femenino")){
            clienteeliminado.setDireccion(nuevaDireccion);
        }
        aux.adicionarClientes(clienteeliminado);
    }
    pila.vaciar(aux);

}
```

Mostrar a los Clientes

DATOS DEL CLIENTE

Nombre: Eddy
Apellido: Apaza
Edad: 25
Direccion: direccion5
Genero: Masculino

DATOS DEL CLIENTE

Nombre: Carolina
Apellido: Paucara
Edad: 22
Direccion: direccion4
Genero: Femenino

DATOS DEL CLIENTE

Nombre: Juan
Apellido: Mamani
Edad: 20
Direccion: direccion3
Genero: Masculino

DATOS DEL CLIENTE

Nombre: Yamiley
Apellido: Lipe
Edad: 18
Direccion: direccion2
Genero: Femenino

DATOS DEL CLIENTE

Nombre: Luis
Apellido: Alvarez
Edad: 19
Direccion: direccion1
Genero: Masculino

Mostrar a los Clientes

DATOS DEL CLIENTE

Nombre: Eddy
Apellido: Apaza
Edad: 25
Direccion: direccion5
Genero: Masculino

DATOS DEL CLIENTE

Nombre: Carolina
Apellido: Paucara
Edad: 22
Direccion: extranca
Genero: Femenino

DATOS DEL CLIENTE

Nombre: Juan
Apellido: Mamani
Edad: 20
Direccion: direccion3
Genero: Masculino

DATOS DEL CLIENTE

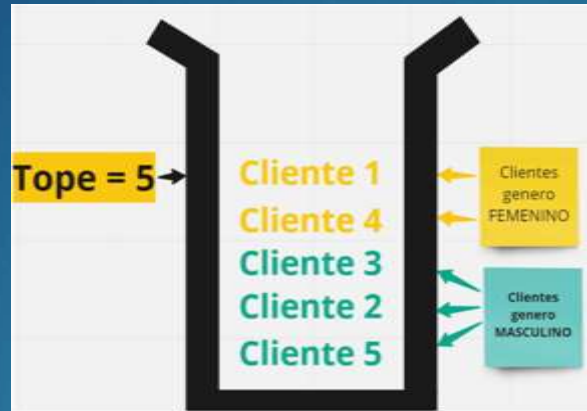
Nombre: Yamiley
Apellido: Lipe
Edad: 18
Direccion: extranca
Genero: Femenino

DATOS DEL CLIENTE

Nombre: Luis
Apellido: Alvarez
Edad: 19
Direccion: direccion1
Genero: Masculino

15.-MOVER ITEMS DE LA PILA

- ▶ 1. El método deberá llamarse `reordenaPila(Pila)`
- ▶ 2. El método debe ser creado en la clase MAIN como un método estático.
- ▶ 3. El método recibe 1 parámetros
 - La Pila de Clientes
- ▶ 4. Mover a la base todos los clientes del genero masculino y los genero femenino moverlos al final.



```
public static void reordenarPila (PilaCliente pila){  
  
    PilaCliente aux = new PilaCliente();  
    Cliente clienteeliminado = null;  
    PilaCliente almacenador = new PilaCliente();  
    while(!pila.esVacio()){  
  
        clienteeliminado = pila.eliminarCliente();  
  
        if(clienteeliminado.getGenero().equals("Femenino")){  
            almacenador.adicionarClientes(clienteeliminado);  
        }  
        else {  
            aux.adicionarClientes(clienteeliminado);  
        }  
    }  
    pila.vaciar(aux);  
    pila.vaciar(almacenador);  
}
```

Mostrar a los Clientes

DATOS DEL CLIENTE
Nombre: Eddy
Apellido: Apaza
Edad: 25
Direccion: direccion5
Genero: Masculino

DATOS DEL CLIENTE
Nombre: Carolina
Apellido: Paucara
Edad: 22
Direccion: direccion4
Genero: Femenino

DATOS DEL CLIENTE
Nombre: Juan
Apellido: Mamani
Edad: 20
Direccion: direccion3
Genero: Masculino

DATOS DEL CLIENTE
Nombre: Yamiley
Apellido: Lipe
Edad: 18
Direccion: direccion2
Genero: Femenino

DATOS DEL CLIENTE
Nombre: Luis
Apellido: Alvarez
Edad: 19
Direccion: direccion1
Genero: Masculino

Mostrar a los Clientes

DATOS DEL CLIENTE
Nombre: Carolina
Apellido: Paucara
Edad: 22
Direccion: direccion4
Genero: Femenino

DATOS DEL CLIENTE
Nombre: Yamiley
Apellido: Lipe
Edad: 18
Direccion: direccion2
Genero: Femenino

DATOS DEL CLIENTE
Nombre: Eddy
Apellido: Apaza
Edad: 25
Direccion: direccion5
Genero: Masculino

DATOS DEL CLIENTE
Nombre: Juan
Apellido: Mamani
Edad: 20
Direccion: direccion3
Genero: Masculino

DATOS DEL CLIENTE
Nombre: Luis
Apellido: Alvarez
Edad: 19
Direccion: direccion1
Genero: Masculino