ESTRUCTURA DE DATOS

Nombre: joel reynaldo condori tumiri

- **1. ¿A que se refiere cuando se habla de ESTRUCTURA DE DATOS?**
- **R.-** las estructuras de datos son aquellas que nos permiten, como desarrolladores, organizar la información de manera eficiente, y en definitiva diseñar la solución correcta para un determinado problema.
- **2¿Cuáles son los tipos de estructura que existe?**
- **R.-** en programación estructurada se utilizan tres tipos de estructuras:
- -Secuenciales, aquellas que se ejecutan una después de otra siguiendo el orden en que se han escrito.
- -Decisión, que permiten omitir parte del código o seleccionar el flujo de ejecución de entre dos o mas alternativas.
- -Iterativas, que se utilizan para repetir la ejecución de cierta parte del programa
- **3¿apoyándose en el link adjunto, explique, porque son útiles?**
- **R.- L**as estructuras de datos son una forma de organizar los datos en la computadora, de tal manera que nos permitan realizar unas operaciones con ellas de forma muy eficiente.
- **4¿Qué es una pila?**
- **R-** En java, una stack es una estructura de datos que sigue el principio lifo, lo que significa que el ultimo elemnto agregado a la pila es el primero en ser eliminado. En otras palabras, los elementos se agregan y eliminan desde la parte superior de la pila.

- **5¿Qué es Stack en java, una STACK será lo mismo que una PILA?**
- **R-** Un objeto de la clase Stack es una pila. Permite almacenar objetos y luego recuperarlos en el orden inverso en el cual se insertaron.
- **6¿Que es TOPE en una PILA?**
- **R-** Numero de elementos que tiene la pila
- **7¿Qué es MAX en una PILA?**
- **R-** Numero máximo de elementos que soporta la pila
- **8¿A que se refiere los métodos esVacia() y esLLena() en una pila?**
- **R-esVacia() se utiliza para comprobar si la pila está vacía, mientras que esLLena() se utiliza para comprobar si la pila está llena, dependiendo de la implementación de la pila**
- **9¿Qué son los métodos estáticos en JAVA?**
- **R-**Un método static en Java es un método que pertenece a la clase y no al objeto. Un método static solo puede acceder a variables o tipos de datos declarados como static. Un método static sólo puede acceder a datos static.

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

R-

```
13 usages
public class PilaCliente
{
    9 usages
    private int tope;
    4 usages
    private int max;
    3 usages
    private Cliente[] pilita;

    4 usages
    public PilaCliente(int max)
    {
        this.max = max;
        this.tope = 0;
        this.pilita = new Cliente[this.max+1];

    }
    //tope es la cantidad de elementos que tiene una pila
    //max  es la cantidad maxima que tiene una pila
    10 usages
    public boolean estaVacia()
    {
        if(this.tope == 0)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
```

```
public boolean estaLleno()
{

    if(this.tope == this.max)
    {
        return true;
    }
    else
    {
        return false;
    }
}

public int numeroElementos( )
{
    return this.tope;
}

10 usages
public void insertar(Cliente cliente)
{
    if(!estaLleno())
    {
        this.tope=this.tope+1;
        this.pilita[this.tope] = cliente;
    }
    else
    {
        System.out.println("La pila esta llena");
    }
}
```

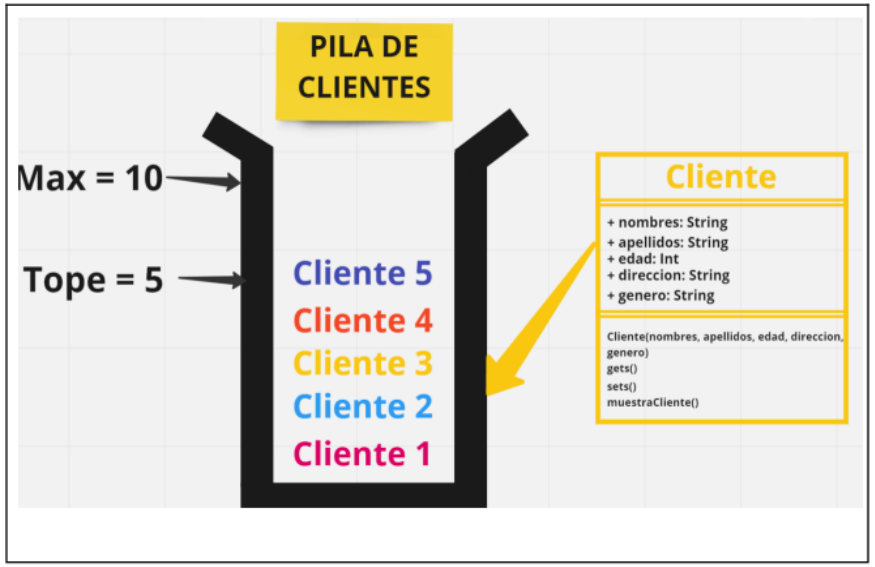
```
public Cliente eliminar()
{
    Cliente elementoEliminado= null;
    if(!estaVacia())
    {
        elementoEliminado = this.pilita[this.tope];
        this.tope--;
    }
    else
    {
        System.out.println("La pila esta vacia");
    }
    return elementoEliminado;
}

3 usages
public void mostrar(){
    Cliente elem=null;
    if(estaVacia()){
        System.out.println("La pila esta vacia");
    } else {
        System.out.println("Elementos de la pila");
        PilaCliente aux = new PilaCliente(this.max);
        while(!estaVacia()){
            elem = this.eliminar();
            aux.insertar(elem);
            elem.mostrarDatos();
        }
        vaciar(aux);
    }
}

3 usages
public void vaciar(PilaCliente a){
    while (!a.estaVacia()) {
        insertar(a.eliminar());
    }
}
```

```java
21 usages
public class Cliente
{
    4 usages
    private String nombre;
    4 usages
    private String apellido;
    4 usages
    private String direccion;
    4 usages
    private int edad;
    4 usages
    private String genero;

    5 usages
    public Cliente(String nombre, String apellido, String direccion, int edad, String genero)
    {
        this.nombre = nombre;
        this.apellido = apellido;
        this.direccion = direccion;
        this.edad = edad;
        this.genero = genero;
    }

    public String getNombre() { return nombre; }

    public String getApellido() { return apellido; }

    public String getDireccion() { return direccion; }

    1 usage
    public int getEdad() { return edad; }

    2 usages
    public String getGenero() { return genero; }

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

    public void setApellido(String apellido) { this.apellido = apellido; }
```

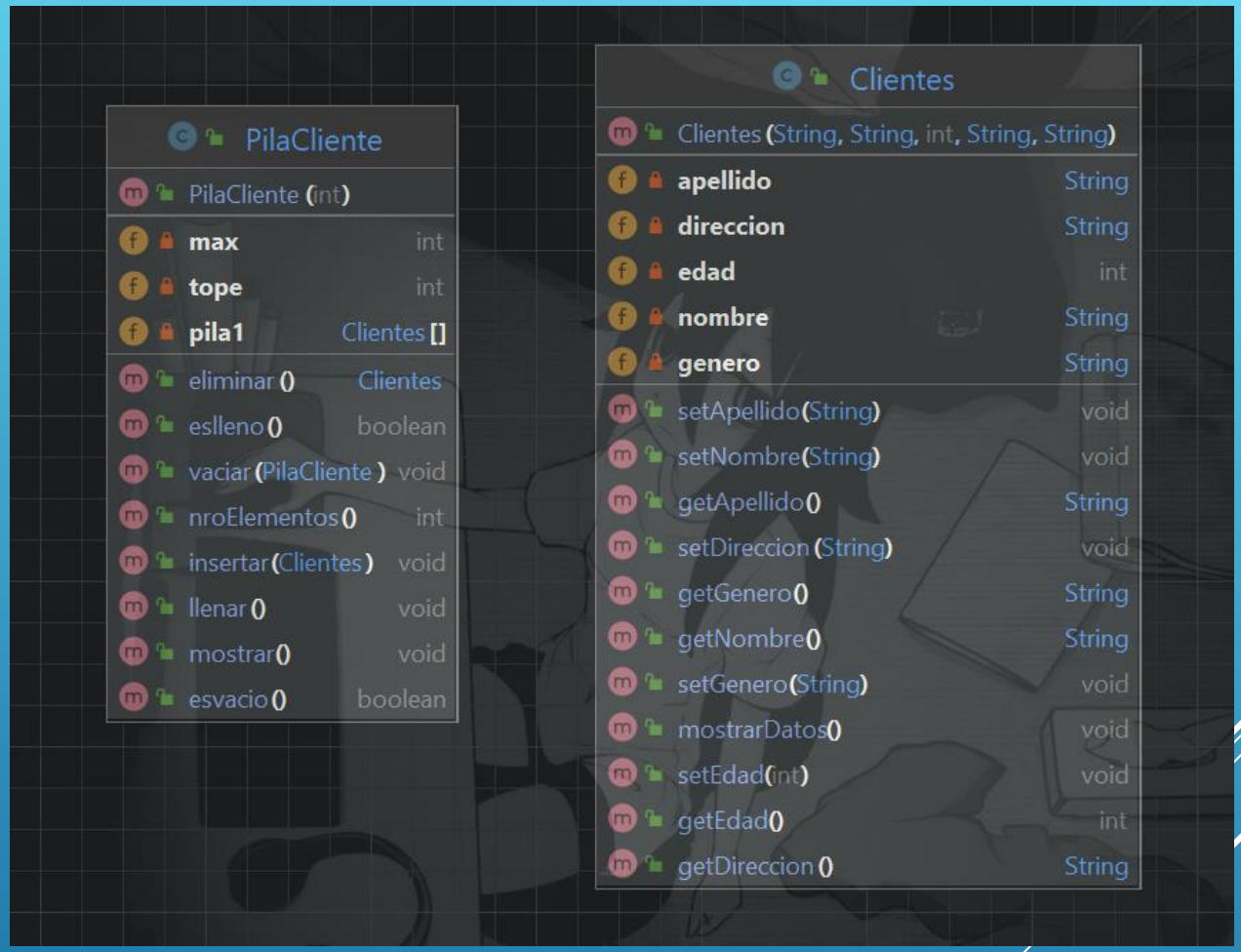
```java
    1 usage
    public void setDireccion(String direccion) { this.direccion = direccion; }

    public void setEdad(int edad) { this.edad = edad; }

    public void setGenero(String genero) { this.genero = genero; }

    3 usages
    public void mostrarDatos()
    {
        System.out.println("Nombre: " + this.nombre);
        System.out.println("Apellido: " + this.apellido);
        System.out.println("Direccion: " + this.direccion);
        System.out.println("Edad: " + this.edad);
        System.out.println("Genero: " + this.genero);
    }

}
```

11Crear las clases necesarias para la PILA DE CLIENTES

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



- Crear la clase **Cliente**
- Crear la clase **PilaCliente**
- Crear la clase **Main**.
- Crear un **paquete** de nombre **PilaDeClientes** (todas las clases deberán de estar dentro de este paquete)



11Crear las clases necesarias para la PILA DE CLIENTES

```java
public class Main
{
    public static void main(String[] args) {
        Cliente cliente1 = new Cliente( nombre: "Carlos", apellido: "Marcelo", direccion: "Calle 1", edad: 20, genero: "Masculino");
        Cliente cliente2 = new Cliente( nombre: "Lineth", apellido: "Santa", direccion: "Calle 2", edad: 25, genero: "Femenino");
        Cliente cliente3 = new Cliente( nombre: "Otavio", apellido: "Suarez", direccion: "Calle 3", edad: 30, genero: "Masculino");
        Cliente cliente4 = new Cliente( nombre: "Ana", apellido: "Marta", direccion: "Calle 4", edad: 35, genero: "Femenino");
        Cliente cliente5 = new Cliente( nombre: "Juan", apellido: "Gutierrez", direccion: "Calle 5", edad: 40, genero: "Masculino");


        PilaCliente pila = new PilaCliente( max: 5);
        pila.insertar(cliente1);
        pila.insertar(cliente2);
        pila.insertar(cliente3);
        pila.insertar(cliente4);
        pila.insertar(cliente5);
        pila.mostrar();


    }
```

```
"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe" -javaagent:C:\Users\PC\AppData\Local\JetBrains\Toolbox\apps\IDEA-U\ch-0\221.5921.22\lib\idea
Elementos de la pila
Nombre: Juan
Apellido: Gutierrez
Direccion: Calle 5
Edad: 40
Genero: Masculino
Nombre: Ana
Apellido: Marta
Direccion: Calle 4
Edad: 35
Genero: Femenino
```

12.Determinar cuantos CLIENTES son mayores de 20 años

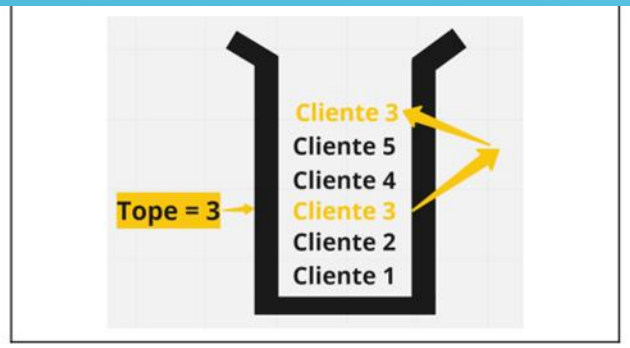
12.Determinar cuántos **CLIENTES** son mayores de 20 años.

- El método deberá llamarse **mayoresCiertaEdad(Pila, edadMayor)**
- El método debe ser creado en la clase **MAIN** como un método estático.
- El método recibe 2 parámetros
  - La Pila de Clientes
  - El valor de la edad.
- Adjuntar los siguientes
  - El **código** del método que resuelve el problema.
  - Una **imagen** de la salida de la consola.

```
public static void mayoresCiertaEdad(pilaCliente pila, int edadMayor){
    int contador = 0;
    Clientes elem = null;
    if(pila.esvacio()){
        System.out.println("La pila esta vacia");
    } else {
        while(!pila.esvacio()){
            elem = pila.eliminar();
            if(elem.getEdad() > edadMayor){
                contador++;
            }
        }
    }
    System.out.println("Hay " + contador + " clientes mayores de " + edadMayor + " años");
}
}
```

```
"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe" -javaagent:C:\Users\PC\AppDa
La cantidad de clientes con mas de 20 son: 4

Process finished with exit code 0
```

13.Mover el K-esimo elemento al final de la pila



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

```java
public static void moverK_esimo(pilaCliente pila, Clientes k){
    pilaCliente aux = new pilaCliente( max: 10);
    Clientes valorExtraidoPila=null;
    while(!pila.esvacio()){
        valorExtraidoPila = pila.eliminar();
        if(valorExtraidoPila!= k){
            aux.insertar(valorExtraidoPila);
        }
    }
    pila.vaciar(aux);
    pila.insertar(k);
    pila.mostrar();
}
```

```
Edad: 55
Genero: Femenino
Nombre: Otavio
Apellido: Suarez
Direccion: Calle 3
Edad: 30
Genero: Masculino
Nombre: Carlos
Apellido: Marcelo
Direccion: Calle 1
Edad: 20
Genero: Masculino
```