



ESCUELA POLITÉCNICA NACIONAL

INGENIERÍA DE SOFTWARE

**Estructura de Datos y Algoritmos I
ICCD343**

LISTAS SIMPLES-OPERACIONES

**Alumnos: Juan Mateo Quisilema, Sebastián
Ramos, Fernando Huilca.**

PROFESORA: Dra. Mayra CARRION

FECHA DE ENTREGA: 10-06-2024

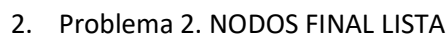


A. Desarrollo de la práctica:

a. Algoritmos Creación Nodos al inicio Lista Simple

b. Resultados ejecución





```

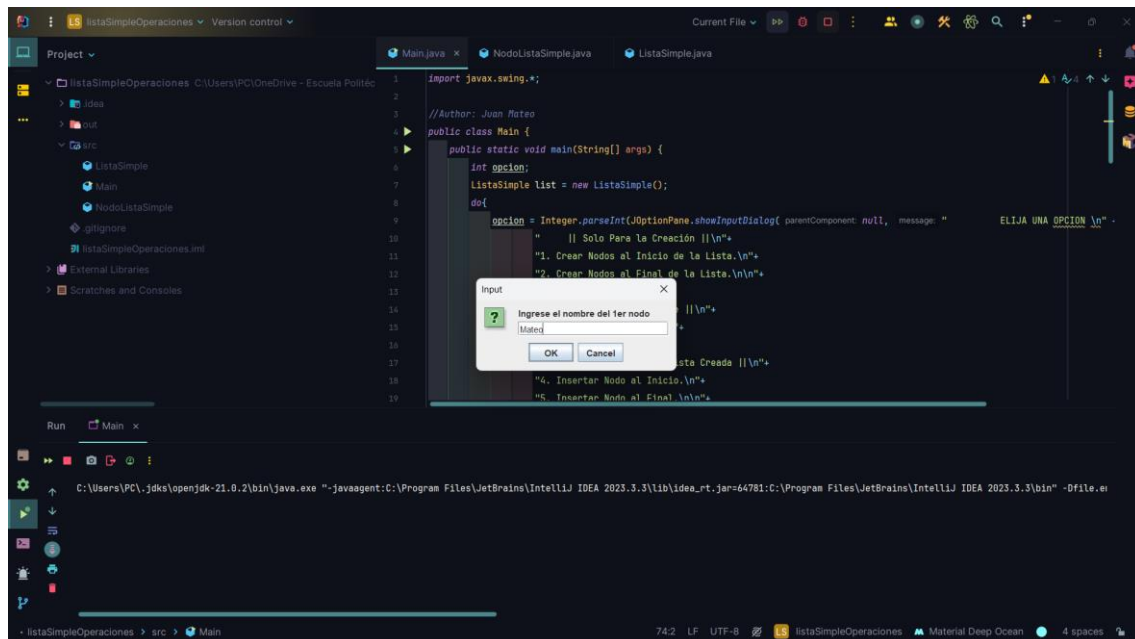
CrearFinalLista()
P,Q,T <- apuntadores
I,L <- camposNodo
O<- entero
Crear(P)
Leer P'.I
Hacer P'.L<- null
Hacer T<- P
Escribir ("Desea más nodos? 1:Si, 2:No.")
Leer O
Mientras (O=1)
    Crear Q
    Leer Q'.I
    Hacer Q'.L <- null
    Hacer T'.L <- Q
    Hacer T <- Q
Escribir ("Desea más nodos? 1:Si, 2:No.")
Leer O
Fin Mientras
Fin CrearFinalLista()

```

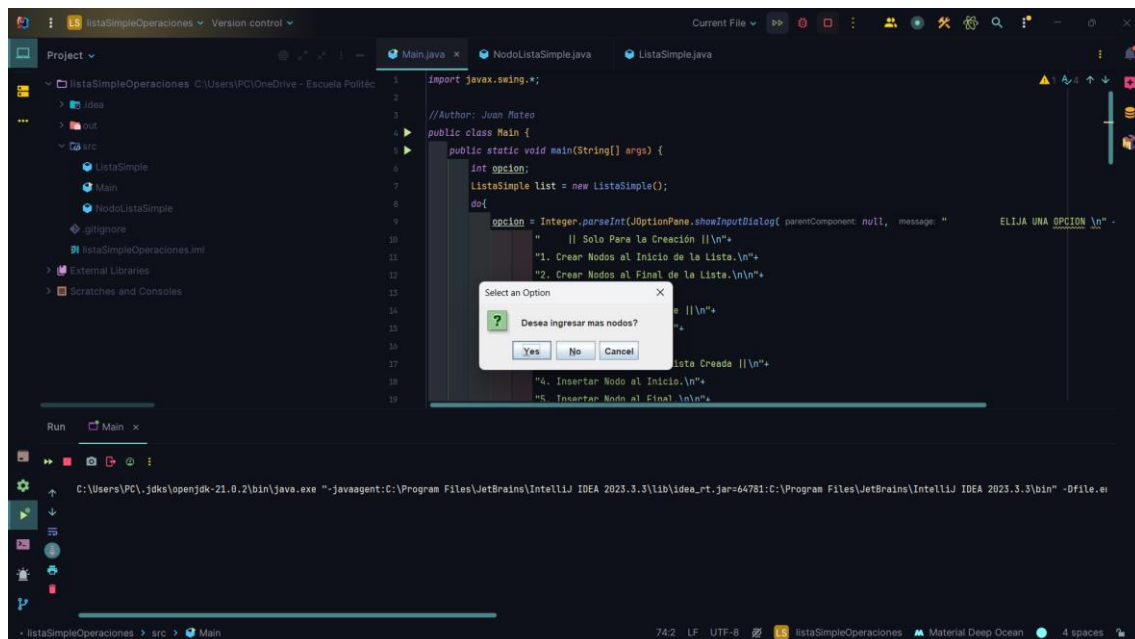
b. Resultados ejecución



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS



AL insertar un Nodo por el Final, se pide primero ingresar la información del mismo.



3. Problema 3. RECORRIDO LISTA

a. Algoritmo Recorrido de nodos en la Lista Simple

Recorrido (P)
P,Q <- Apuntadores
I,L <- camposNodo
Hacer Q <- P
Mientras (Q!= null)
 Escribir Q'.I



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS

Hacer Q <- Q'.L
Fin Mientras
Fin Recorrido

b. Resultados ejecución

```
switch(option){
    case 1 :
        list.crearNodoInicioLista();
        JOptionPane.showMessageDialog( parentComponent: null, message: "LISTA CREADA NODOS POR INICIO\n"+list);
        break;
    case 2 :
        list.crearNodoFinalLista();
        JOptionPane.showMessageDialog( parentComponent: null, message: "LISTA CREADA NODOS POR FINAL\n"+list);
        break;
    case 3 :
        list.recorrer2();
        break;
    case 4 :
        list.insertarNodoInicio();
        break;
    case 5 :
        list.insertarNodoFinal();
        break;
    case 6 :
        int opcion2 = JOptionPane.showConfirmDialog( parentComponent: null, message: "Desea eliminar el primer nodo");
        if (opcion2 == JOptionPane.YES_OPTION){
            list.eliminarNodoInicio();
            JOptionPane.showMessageDialog( parentComponent: null, message: " * Elemento Eliminado con Éxito * \n");
        }
        break;
    case 7 :
        int opcion3 = JOptionPane.showConfirmDialog( parentComponent: null, message: "Está seguro de eliminar el último nodo");
        if (opcion3 == JOptionPane.YES_OPTION){
            list.eliminarNodoFinal();
            JOptionPane.showMessageDialog( parentComponent: null, message: " * Elemento Eliminado con Éxito * \n");
        }
        break;
}
```

Una vez que se han agregado datos en la lista y se accede a la opción 3 que recorre la lista se va a presentar un mensaje con la lista recorrida y otro mensaje de haberlo hecho con éxito, este método es usado también para completar el método de búsqueda.



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS

En caso de que la lista esté vacía también se va a presentar una ventana emergente que informa del problema para recorrer una lista vacía.

The screenshot shows an IDE with a project named 'listaSimpleOperaciones'. The code in 'Main.java' is a switch statement handling different operations on a linked list. A message dialog box is displayed over the code, showing an information icon and the text 'La lista está vacía.' with an 'OK' button.

```
switch(option){
    case 1 :
        list.crearNodoInicioLista();
        JOptionPane.showMessageDialog( parentComponent: null, message: "LISTA CREADA NODOS POR INICIO\n"+list);
        break;
    case 2 :
        list.crearNodoFinalLista();
        JOptionPane.showMessageDialog( parentComponent: null, message: "LISTA CREADA NODOS POR FINAL\n"+list);
        break;
    case 3 :
        list.recorrer2();
        break;
    case 4 :
        list.insertarNodoInicio();
        JOptionPane.showMessageDialog( parentComponent: null, message: "INSERTA NODO INICIO DE LISTA CREADA\n"+list);
        break;
    case 5 :
        list.insertarNodoFinal();
        JOptionPane.showMessageDialog( parentComponent: null, message: "INSERTA NODO AL FINAL DE LISTA CREADA\n"+list);
        break;
    case 6 :
        int opcion2 = JOptionPane.showConfirmDialog( parentComponent: null, message: "Desea eliminar el primer nodo");
        if (opcion2 == JOptionPane.YES_OPTION){
            list.eliminarNodoInicio();
            JOptionPane.showMessageDialog( parentComponent: null, message: " * Elemento Eliminado con Éxito * \n");
        }
        break;
    case 7 :
        int opcion3 = JOptionPane.showConfirmDialog( parentComponent: null, message: "Está seguro de eliminar el último nodo");
        if (opcion3 == JOptionPane.YES_OPTION){
            list.eliminarNodoFinal();
            JOptionPane.showMessageDialog( parentComponent: null, message: " * Elemento Eliminado con Éxito * \n");
        }
        break;
}
```

4. Problema 4. INSERTAR NODO EN LISTA CREADA AL INICIO

a. Algoritmo Inserta nodo Inicio de la Lista Simple

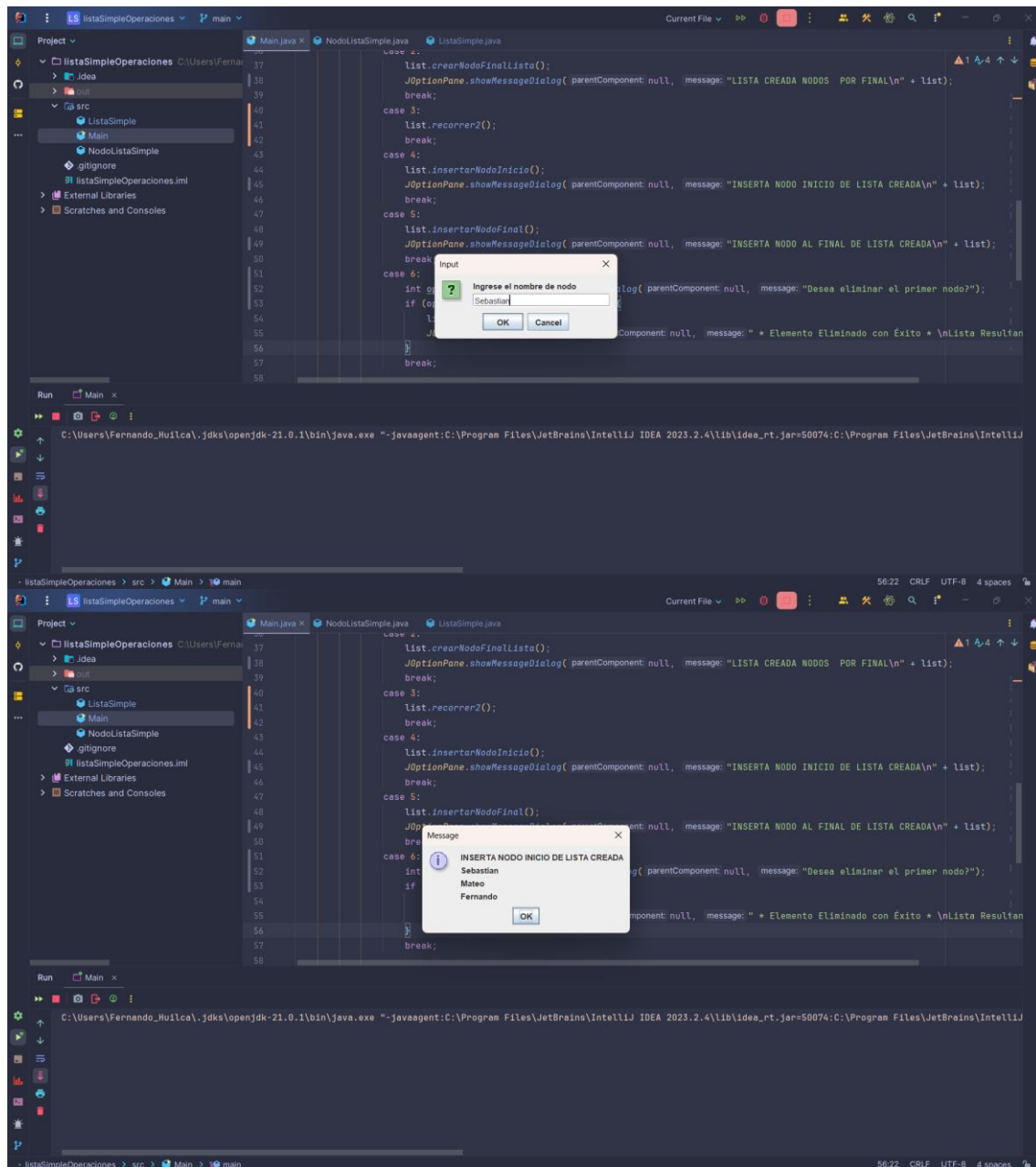
```
InsertarInicio(P,D)
P,Q <- apuntdores
I,L <- camposNodo
Crear Q
Escribir Q'.I <- D
Hacer Q'.L <- P
Hacer P <- Q
Fin InsertarInicio()
```

b. Resultados ejecución

Al principio se va a mostrar la ventana emergente que te va a pedir la información para el nodo y luego se muestra la lista:



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS



5. Problema 5. ELIMINAR NODO EN INICIO LISTA

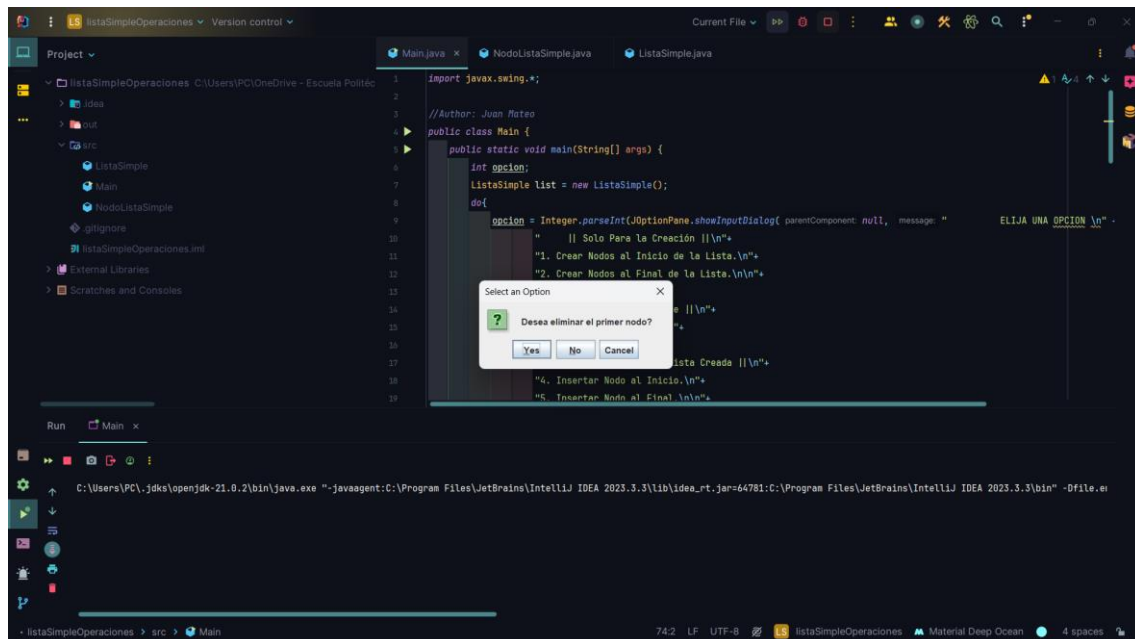
a. Algoritmo Eliminar nodo en inicio Lista Simple

```
EliminarInicio(P)  
P,Q <- apunadores  
I,L <- camposNodo  
Hacer Q <- P  
Hacer P <- Q'.L  
Eliminar (Q)  
Fin EliminarInicio()
```

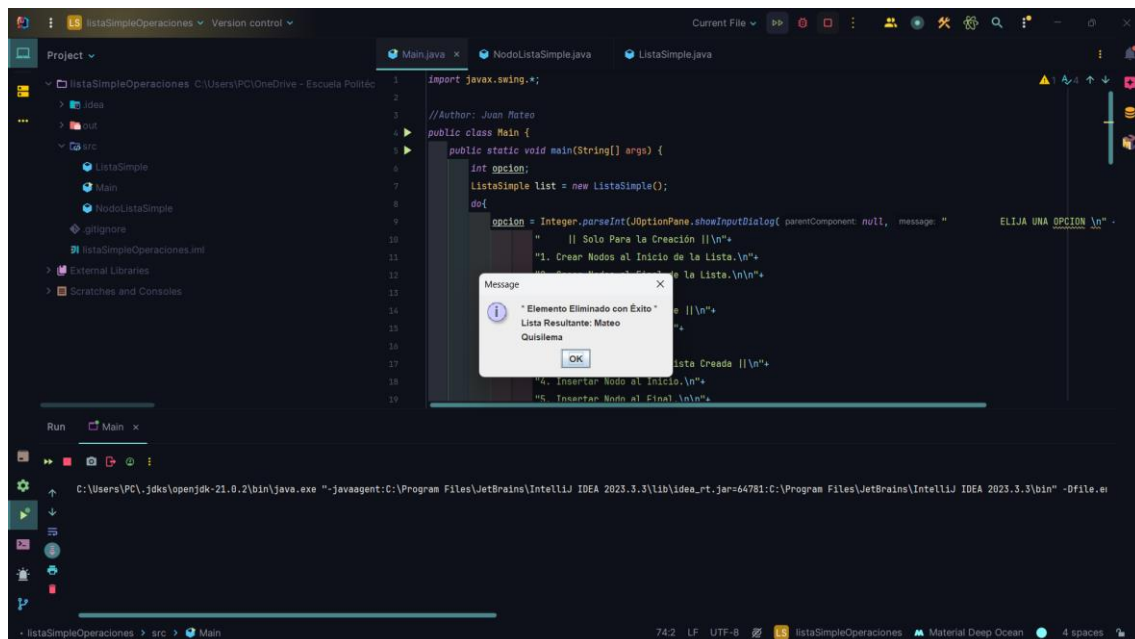
b. Resultados ejecución



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS



AL momento de eliminar un nodo por el inicio, se pide una confirmación de la acción.



6. Problema 6. INSERTAR NODO EN LISTA CREADA POR EL FINAL

a. Algoritmo Inserta nodo final de la Lista Simple.

```
InsertarNodoFinal()  
P,Q,T <- Apuntadores  
I,L <- camposNodo  
Q <- P  
Mientras (Q'.L != null)  
    Q<-Q'.L
```

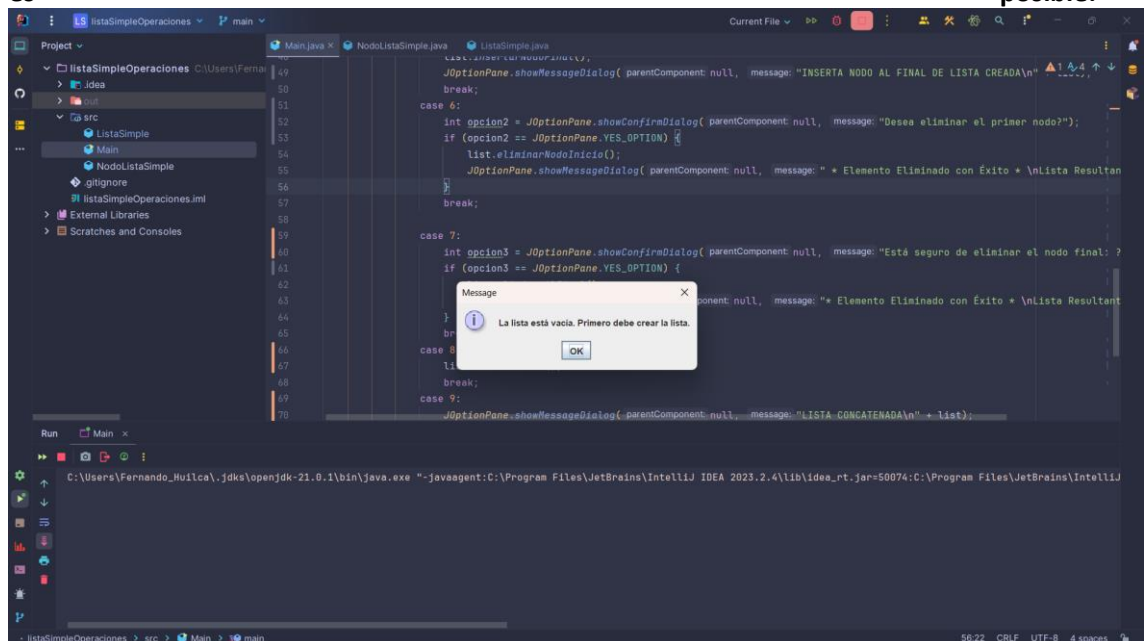



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS

Fin Mientras
Crear(T)
Leer T'.l
Hacer Q'.L <- T
Hacer T'.L <-- null
Fin InsertarNodoFinal()

b. Resultados ejecución.

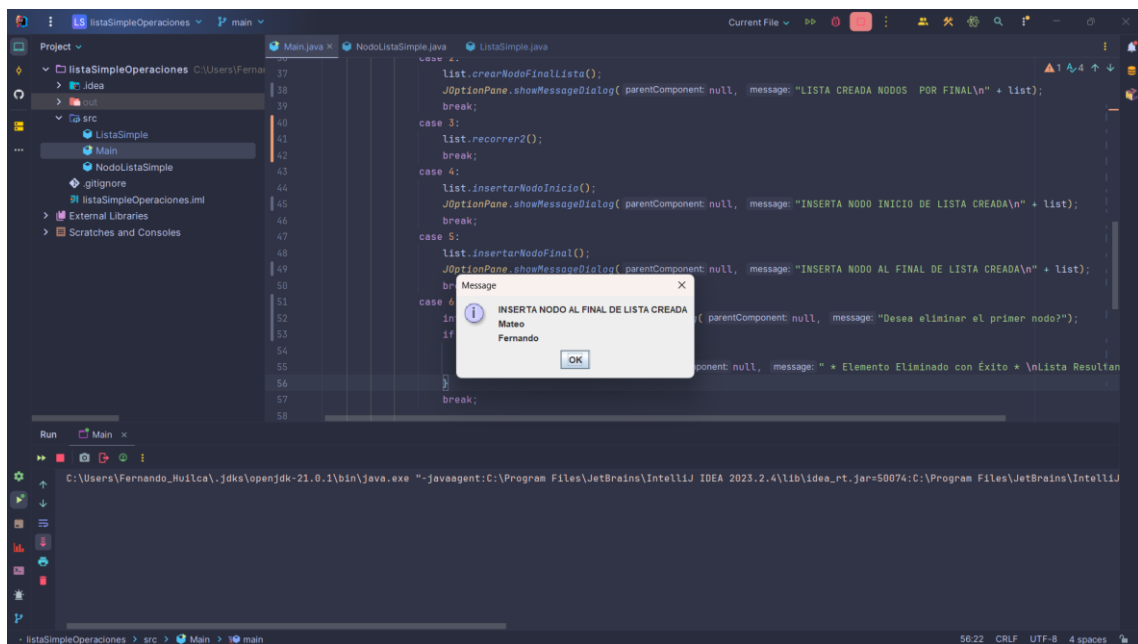
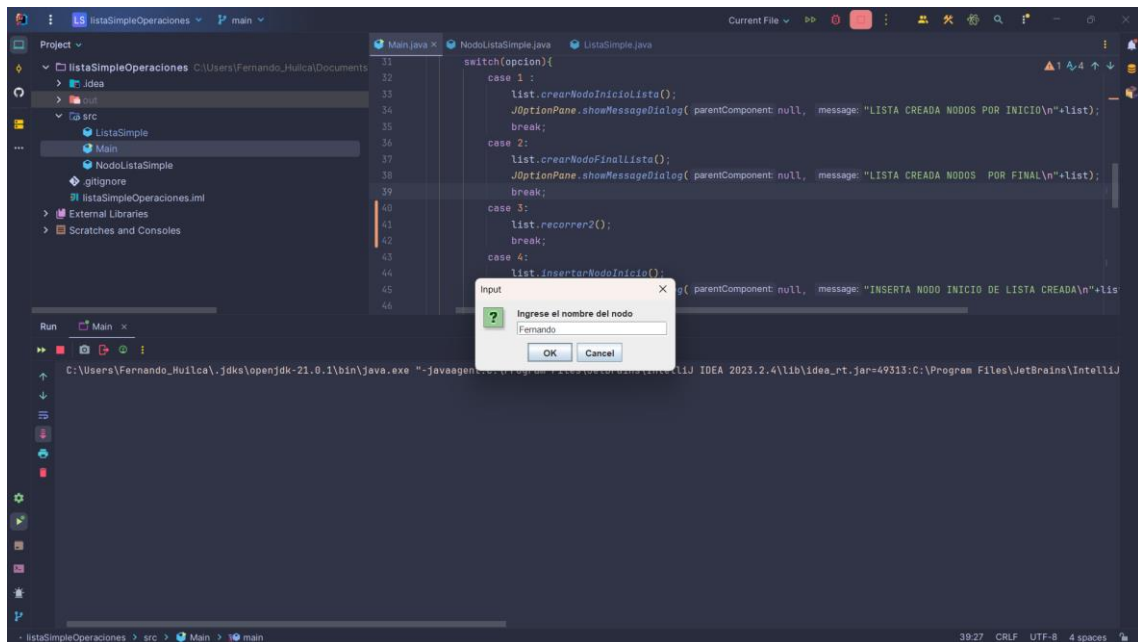
En caso de que la lista esté vacía, va a salir un mensaje de error ya que esta operación no es posible:



Una vez creada la lista y si se quiere insertar un nodo por el final primero va a pedir la información y luego presentar la lista con el nuevo nodo:



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS



7. Problema 7. CREACION DE UN NODO
a. Algoritmo Crea un nodo

```
CrearNodoListaS()  
P <- apuntador  
I,L <- camposNodo  
Crear (P)  
Hacer P'.T <- Dato  
Hacer P'.L <- null  
Fin CrearNodoListaS()
```

b. Resultados ejecución.



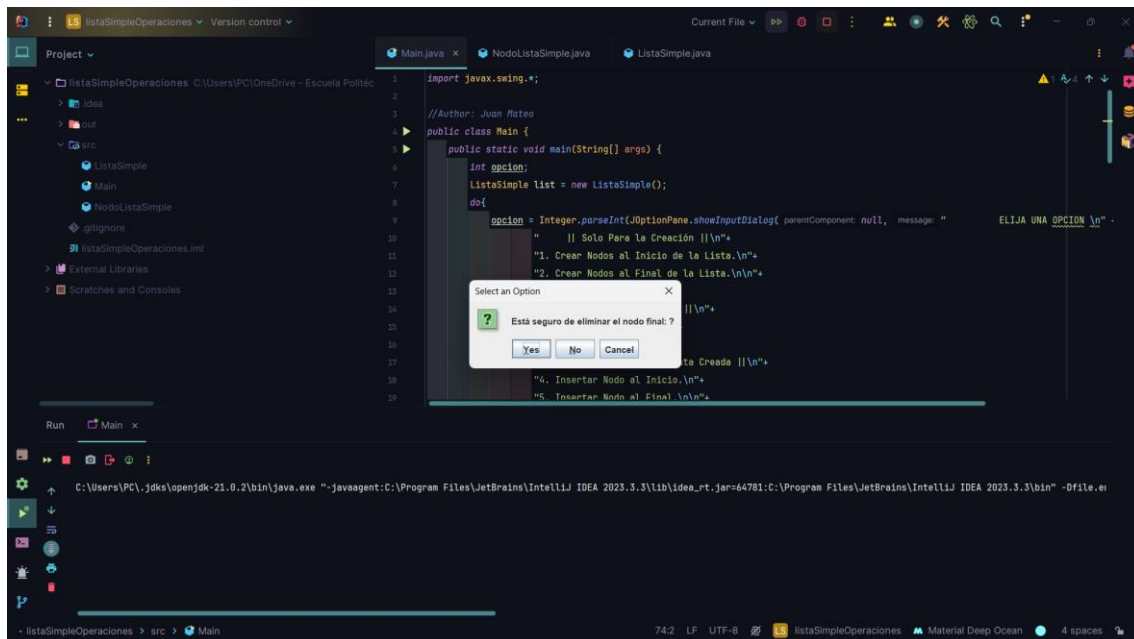
ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS

8. Problema 8. ELIMINAR NODO EN FINAL LISTA

a. Algoritmo Eliminar nodo en final Lista Simple

```
EliminarFinal()  
P,Q,T <- apunadores  
I,L <- campos  
Q <- P  
T <- Q  
Mientras (Q'.L != null)  
T <- Q y Q <- Q'.L  
Fin Mientras  
Eliminar (Q)  
T'.L <- null  
Fin Eliminar
```

c. Resultados ejecución.



Para la eliminación de un nodo por el final, de igual forma se pide primero una confirmación de la acción.



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS

The screenshot shows an IDE with a project named 'listaSimpleOperaciones'. The code in 'Main.java' includes imports for 'javax.swing.*' and 'JOptionPane', a comment '//Author: Juan Mateo', and a 'public class Main' with a 'main' method. The 'main' method creates a 'ListaSimple' object and enters a loop where it prompts the user to choose an option (1 to 5). A message dialog box is displayed over the code, showing the message 'Elemento Eliminado con Éxito' (Element removed successfully) and 'Lista Resultante: Mateo Quisilema' (Resulting List: Mateo Quisilema). The dialog box has an 'OK' button. The IDE's status bar at the bottom indicates '74:2 LF UTF-8' and 'listaSimpleOperaciones Material Deep Ocean 4 spaces'.

9. Problema 9. BUSQUEDA EN LA LISTA

a. Algoritmo Búsqueda en la lista.

```
BuscarListaS()  
P,Q <- apunadores  
O <- variable a buscar  
I,L <- camposNodo  
Hacer Q<-P  
Escribir ("Ingrese el carácter a buscar")  
Leer O  
Mientras (Q'.I != O)  
Q <- Q'.L  
Fin Mientras  
Escribir ("Valor encontrado!" + O)  
Fin BuscarListaS()
```

b. Resultados ejecución.

Cuando se quiere buscar un nodo con una información que no se encuentra almacenada en la lista se va a presentar un mensaje de error:



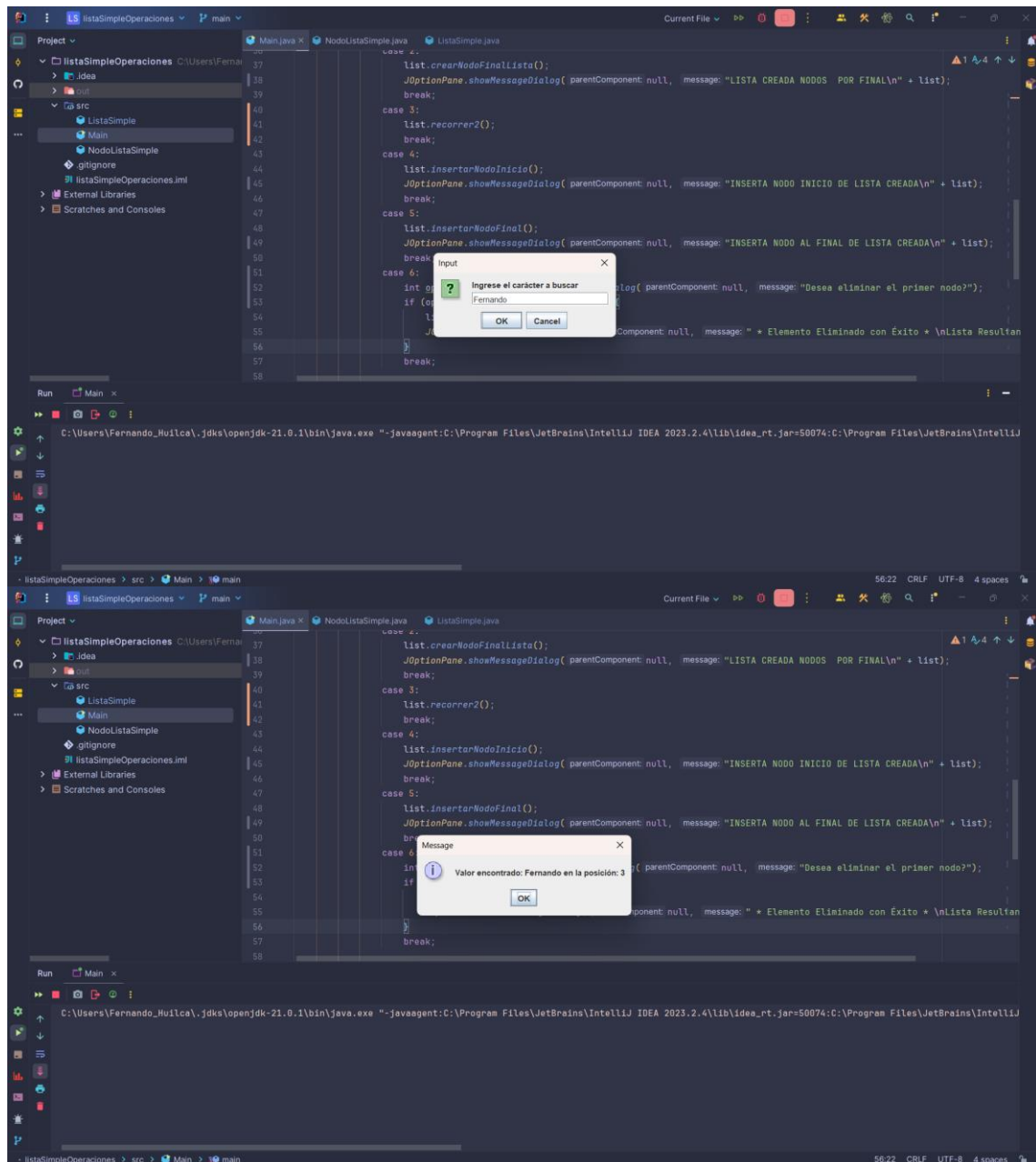
ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS

The screenshot displays the IntelliJ IDEA IDE with a project named 'listaSimpleOperaciones'. The code is written in Java and implements a linked list with operations like creating, inserting, and deleting nodes. A dialog box titled 'Input' is shown, asking the user to 'Ingresar el carácter a buscar' (Enter the character to search for). The user has entered 'Mayra'. Below the code, the 'Run' console shows the execution path. Another dialog box titled 'Message' is shown, displaying the error message 'Valor no encontrado: Mayra' (Value not found: Mayra).

Sin embargo, cuando se quiere buscar en la lista un nodo con la información que sí está dentro de la lista se va a presentar un mensaje de búsqueda exitosa además se va a mencionar en qué posición se encuentra dicho nodo:



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS



B. Conclusiones y recomendaciones:

Permite evidenciar la capacidad del estudiante para analizar y concluir en base a lo que se llevó a cabo en el laboratorio. Las conclusiones son objetivas y deben indicar lo siguiente:

- Se logró comprender de manera exitosa el funcionamiento y concepto de las listas simples, por medio del análisis y creación de algoritmos para realizar diferentes operaciones y la implementación de los mismos.
- Se utilizó un nuevo recurso conocido como listas el cual permite guardar una gran cantidad de datos de modo dinámico.
- Esta estructura de datos estudiada (la lista) nos ha abierto la mente a una nueva posibilidad de almacenamiento de datos más flexible y dinámicos en comparación con



ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS

los arreglos que son más estáticos y con algunas restricciones como puede ser el tamaño definido previamente.

- Hemos adquirido cierta consciencia al momento de aplicar varias de las estructuras de datos estudiadas segun el problema al que nos enfrentamos al momento de resolverlo con código.

C. Bibliografía:

La bibliografía será descrita en formato IEEE.

[1]. Estructura de Datos, GARCIA Ivan, GARCIA Magariño, ISBN: 8445419358 ISBN-13, Edición 2011

[2].

D. Anexos

Captura evidencia trabajo en equipo

