

**INFORME ORGANIZACIÓN DEL COMPUTADOR 2**

**Profesores:**

**Carlos Jimenez**

**Marcos D. Aranda**

**alumnos**:

Federico Pardo

DNI: 41.672.835

Lucas Nicolás Alfredo Aguilar

DNI:43.858.348

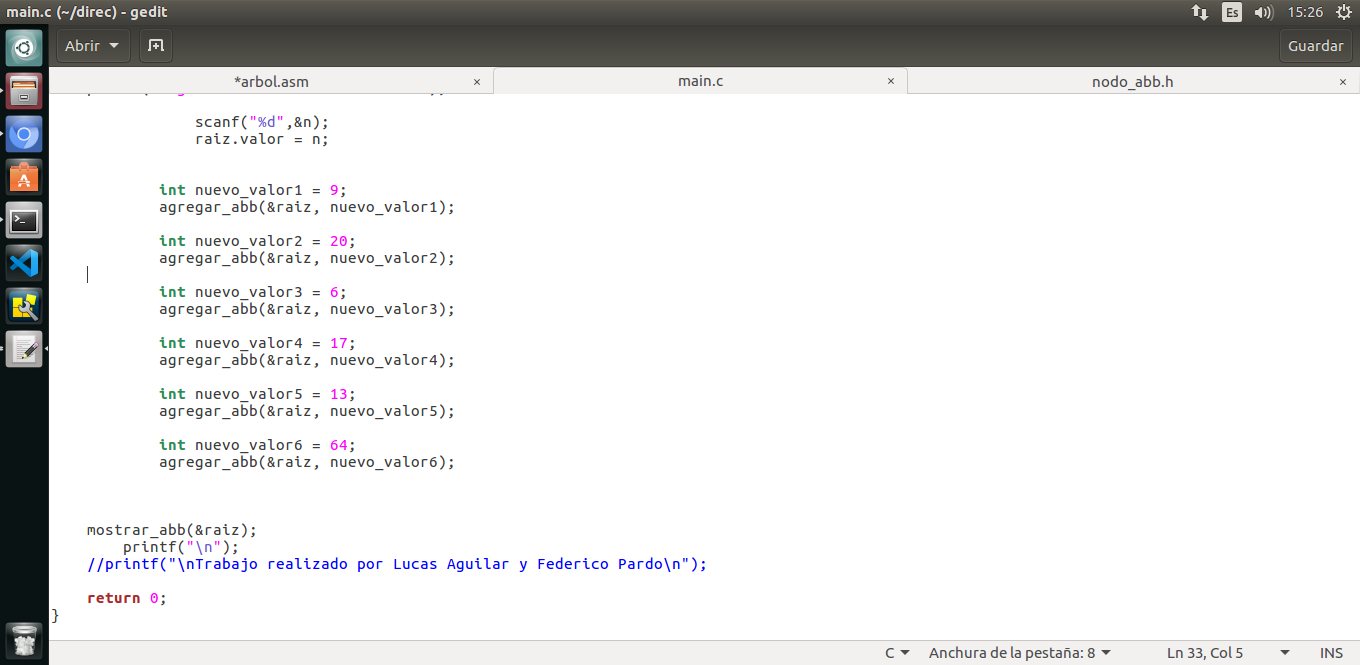
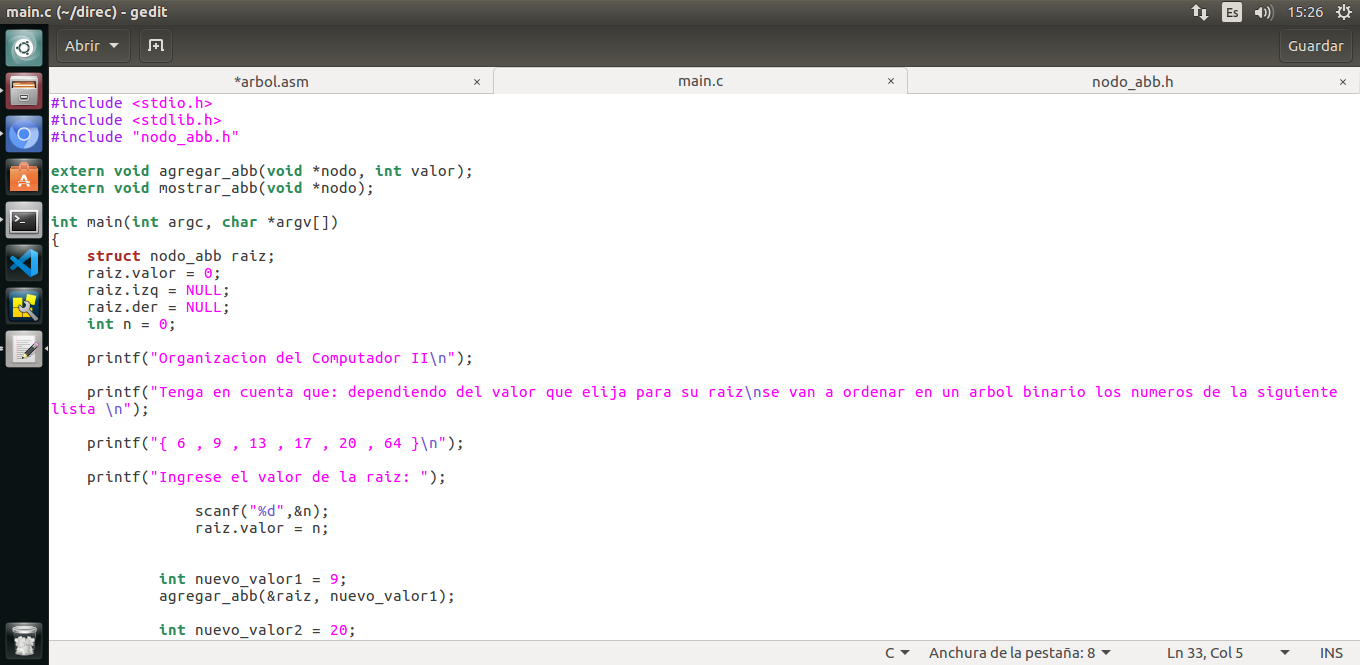
El desarrollo del trabajo práctico involucra la programación de un árbol binario en NASM x86 y su integración con el lenguaje de programación C. El objetivo de este trabajo es agregar en orden los nodos del arbol binario.

En primer lugar ,se explicara el codigo en C:

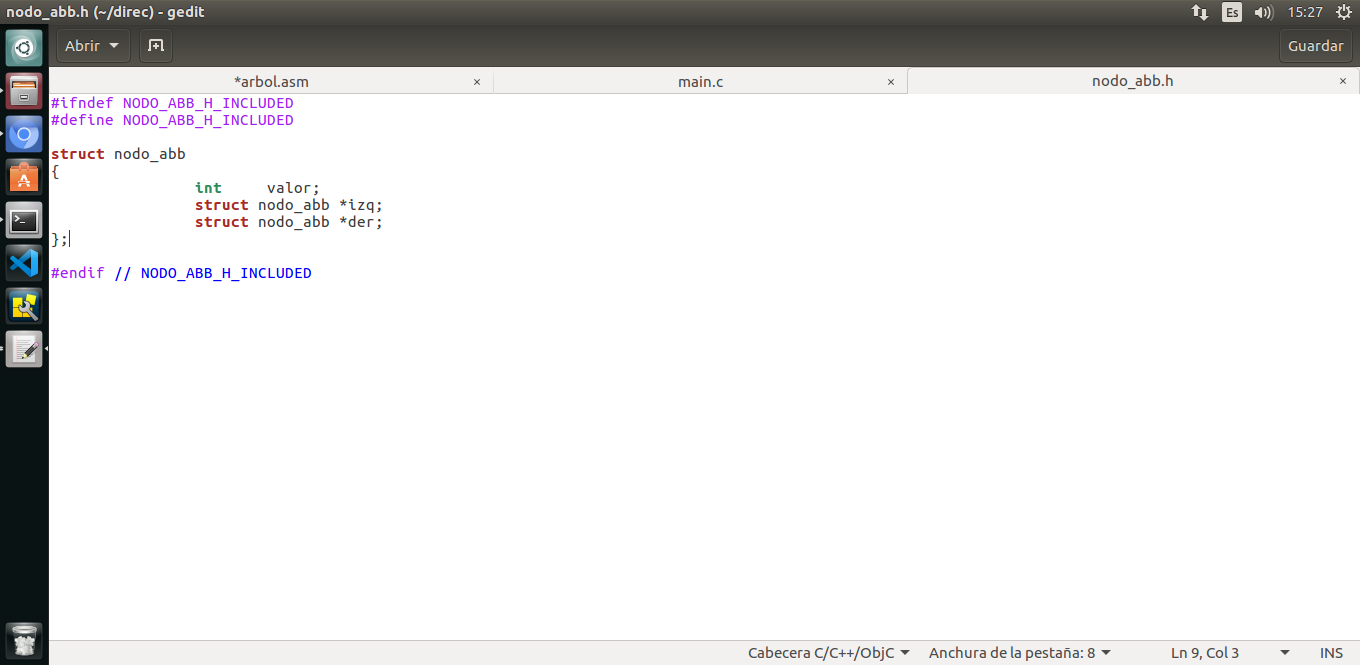
El código C interactua con las funciones implementadas en el código ensamblador para crear un Árbol Binario ,agregar nodos y mostrar el valor de los nodos del arbol.

Declara las funciones externas agregar\_abb y mostrar\_abb, icializa la raíz en cero y los nodos izquierdo y derecho como NULL, y solicita al usuario ingresar el valor para la raíz del arbol,entonces, Llama repetidamente a agregar\_abb para insertar nuevos nodos en el ABB, pasando como parámetros la dirección de la raíz y los valores a agregar.

Finalmente, llama a mostrar\_abb para mostrar el contenido del ABB, pasando como parámetro la dirección de la raíz.; los nodos se ordenan dependiendo del valor que ingrese el usuario.



En segundo lugar, El archivo nodo\_abb.h, define la estructura nodo\_abb utilizada del programa principal y en las funciones del lenguaje ensamblador.

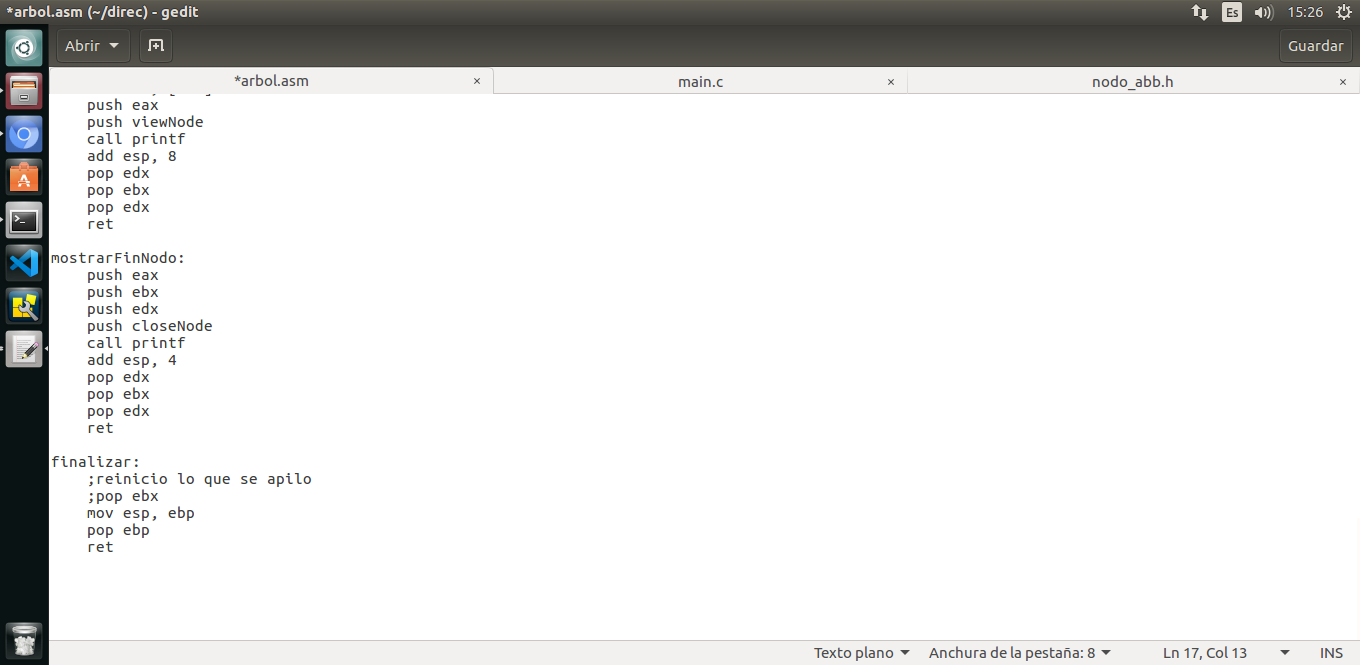
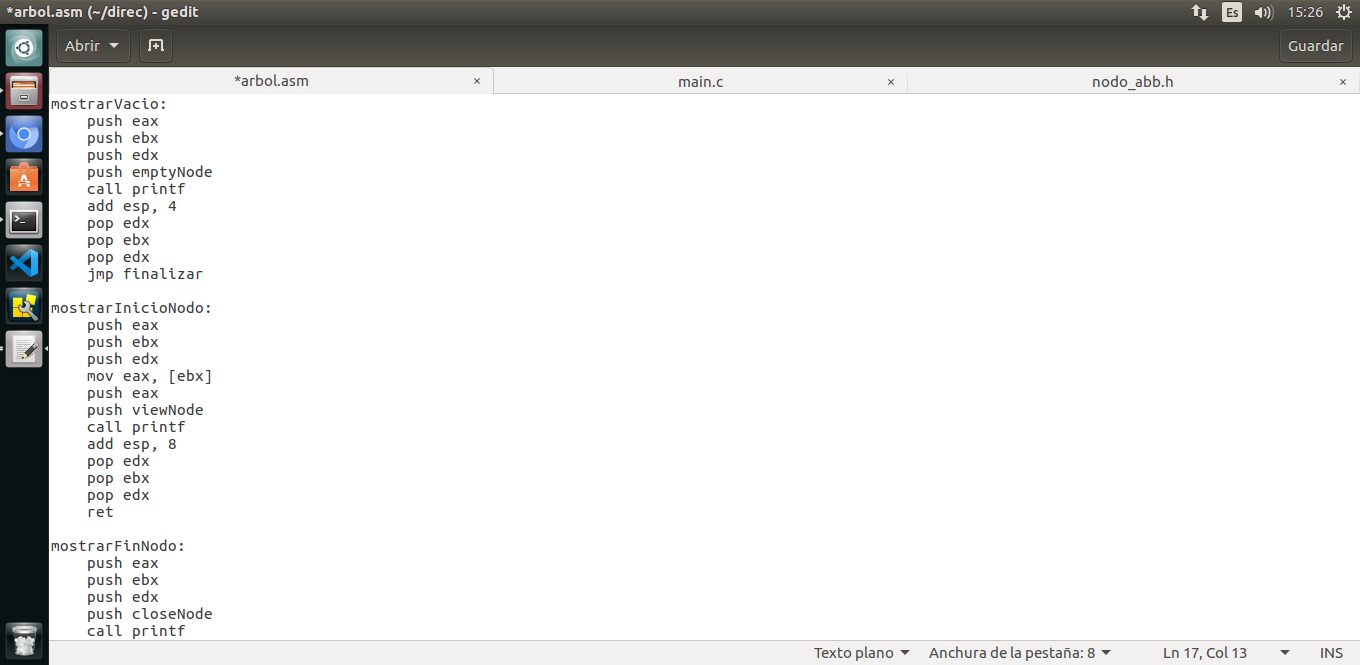
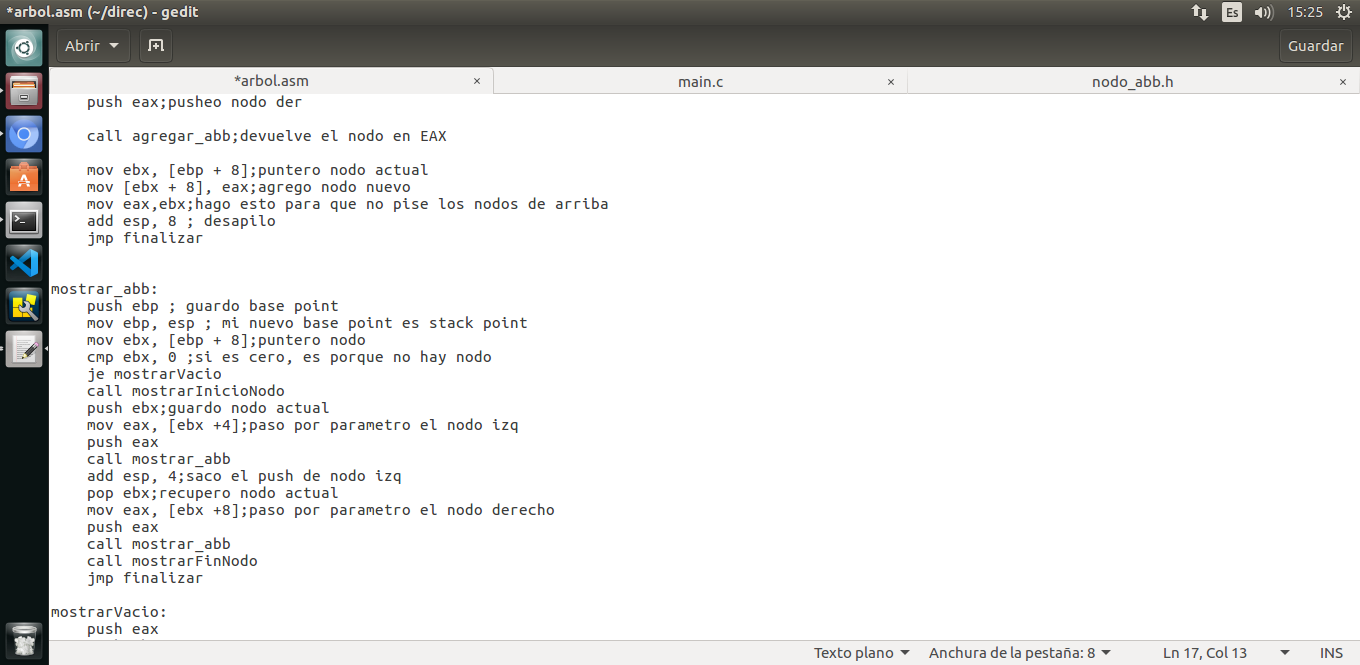
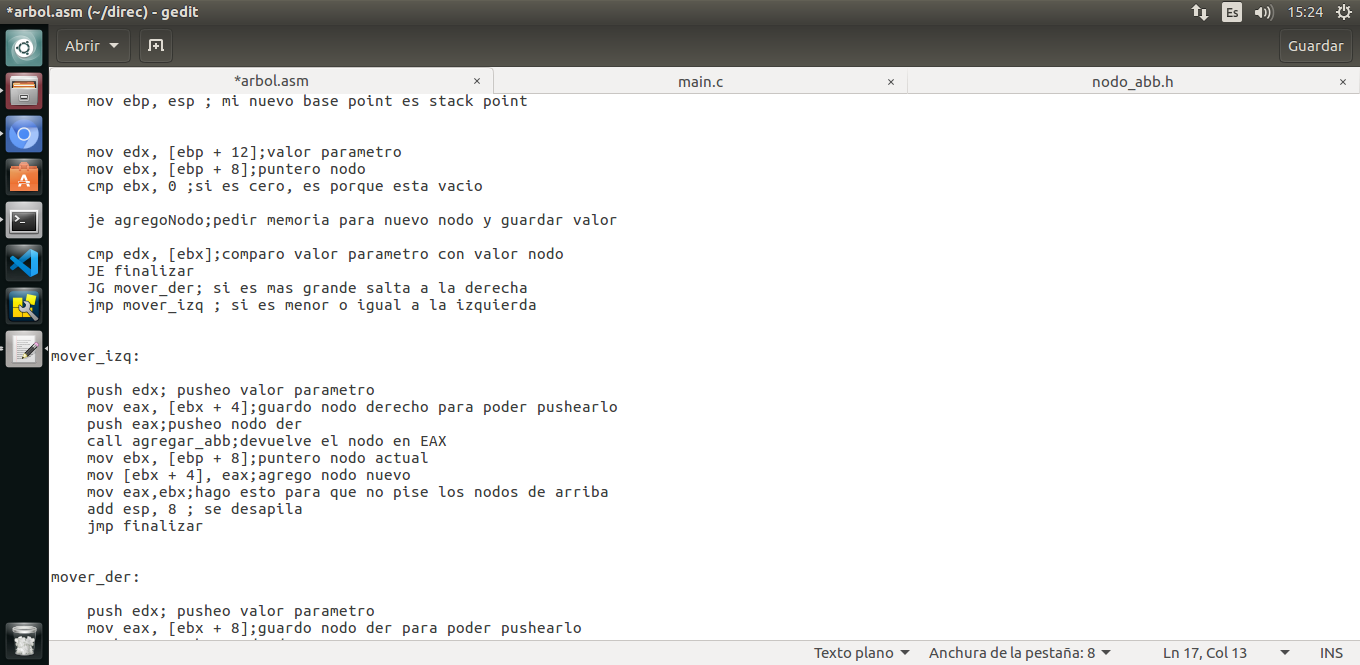
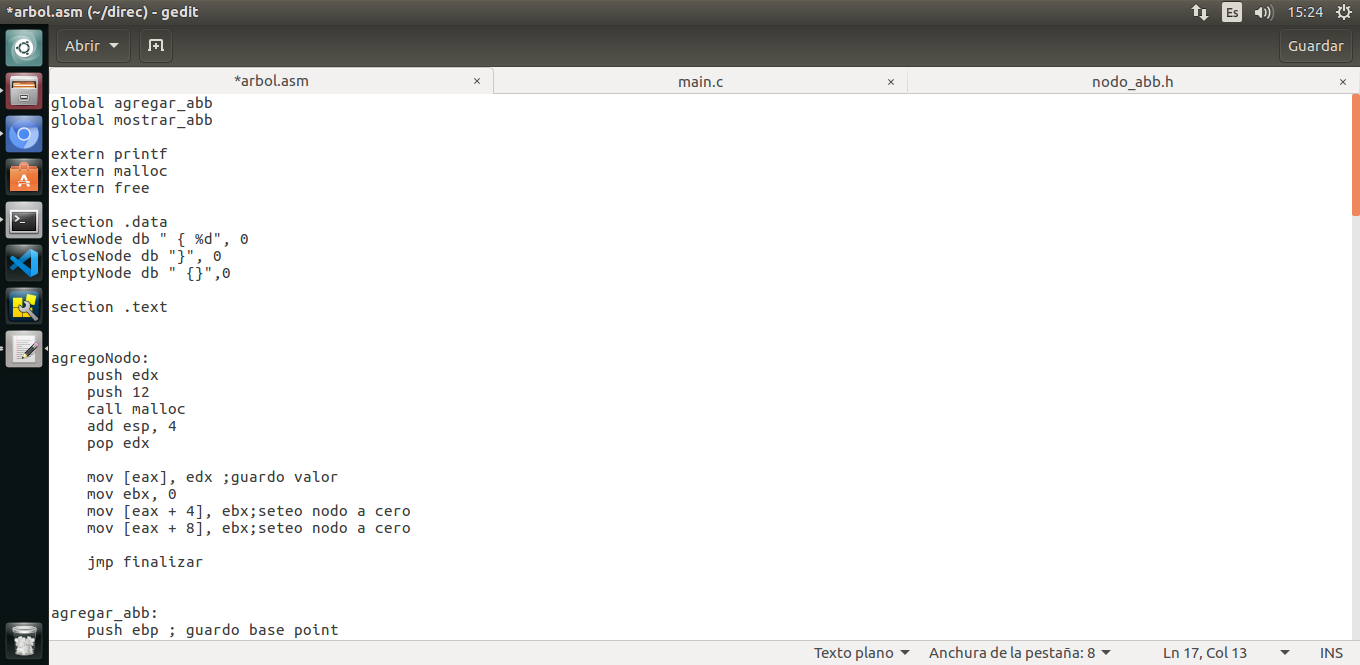


Por ultimo, el codigo ensamblador nasm:

Reserva memoria para los nodos utilizando la función malloc . Guarda el valor pasado como parámetro en el nodo. Guarda el puntero de la base actual (base point) en ebp y establece un nuevo punto de base en el punto de la pila actual (esp).

Si el nodo no está vacío, compara el valor pasado como parámetro con el valor del nodo actual y decide si se debe mover a la izquierda (Se obtiene el puntero al enlace izquierdo del nodo actual desde [EBX + 4] y se guarda en EAX) o a la derecha (Se obtiene el puntero al enlace derecho del nodo actual desde [EBX + 8] y se guarda en EAX).

mover\_izq y mover\_der realizan llamadas recursivas a agregar\_abb para el enlace izquierdo o derecho del nodo actual, respectivamente. mostrarVacio, mostrarInicioNodo y mostrarFinNodo son funciones auxiliares utilizadas por mostrar\_abb para mostrar mensajes específicos.La función finalizar realiza las operaciones necesarias para terminar las funciones: restablece el punteo de la pila a la base anterior y tambien retorna.



**Complicaciones:**

Primera complicación: Mal manejo de memoria al utilizar la función agregar\_abb de NASM.

Durante la ejecución del programa, se encontró un problema relacionado con el manejo de la memoria al intentar utilizar la función agregar\_abb de NASM. Este problema provocó un comportamiento inesperado y errores en el programa.

Segunda complicación: Error de compilación resuelto con la adición de la opción -no-pie.

Durante el proceso de compilación, se encontró un error que resultaba en una compilación incorrecta del código. Para solucionar este problema, se agregó la opción -no-pie al compilar. Esta opción permitió que el código se compile correctamente y evitó el error que se encontraba previamente.

Tercera complicación: Limitación en la cantidad de valores ingresados durante un ciclo while.

Durante la implementación de un ciclo while para solicitar al usuario que ingrese valores para agregar al árbol, se observó una limitación inesperada. El programa solo permitía realizar dos solicitudes de ingreso de valores antes de finalizar prematuramente. Para resolver este inconveniente, se decidió modificar el enfoque y solicitar manualmente los valores del árbol, únicamente pidiendo al usuario que ingrese la raíz del árbol para permitir una mayor variabilidad en su orden.

**En conclusión:**

Al realizar el trabajo práctico nos hizo comprender el funcionamiento de estas estructuras de datos y las indicaciones de la programación a bajo nivel. Este trabajo práctico proporciona una comprensión de la implementación de un árbol binario en NASM x86, su integración con C y la importancia de garantizar una gestión adecuada de la memoria.