**Data Structures and Algorithms**

Logo, company name

Description automatically generated

**Lab report: 4**

|  |  |
| --- | --- |
| **Name:** | **Ali Salman** |
| **Reg no:** | **FA22-BCE-005** |
| **Class:** | **BCE-3A** |
| **Lab Instructor:** | **Dr. Ali Mustafa** |

**Lab 04**

**Implementation of Circular Doubly Linked Lists in C language**

**Task01: Debugging code for errors.**

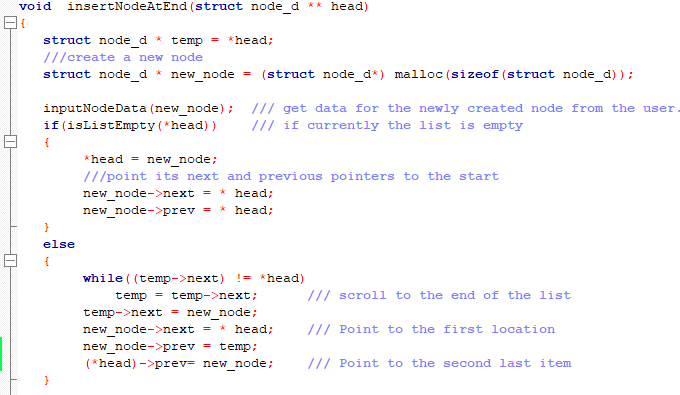
**Syntax errors:**

There are 2 syntax error which are detected by running the program.

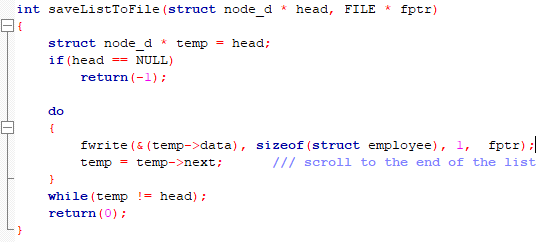
**Logical errors:**

The printf statements print only name statements instead of age and salary in search function.

In below function , the address of new node is not assigned to previous of head pointer , due to which when we use the delete from end function , then two nodes are deleted.

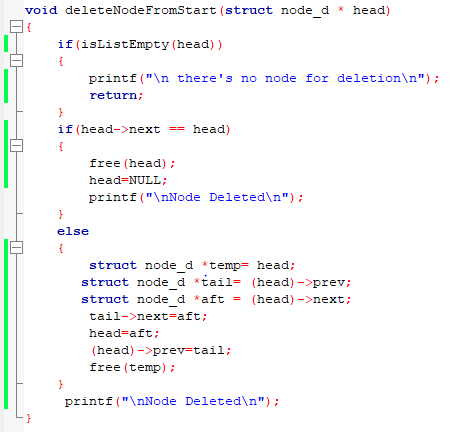


To save a list to file, we must check whether the list is empty. If not, then write the entered data in the file “employee.h” by using fwrite function. We use do while loop instead of while because we want to write the data of last node which cannot be written using while loop because the condition is checked before implementation

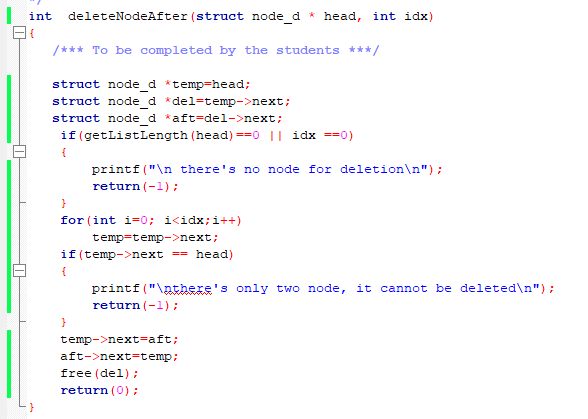


**Task 02: Implementing Node Removal and Node Insertion Tasks**

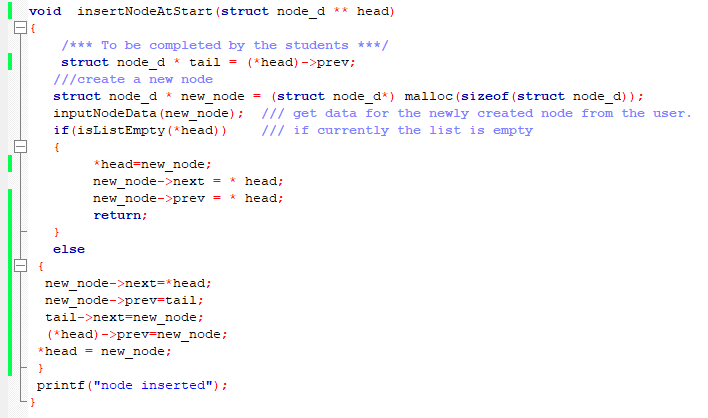
1. **deleteNodeFromStart(struct node\_d \*\* head):**



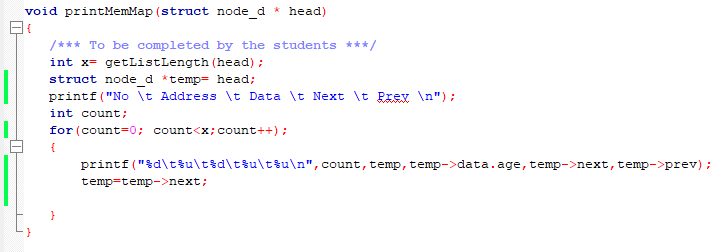
1. **deleteNodeAfter(struct node\_d \* head, int idx):**



1. **void insertNodeAtStart(struct node\_d \*\* head):**



**Void printMemMap(struct node\_d \* head):**



**Critical Analysis:**

In this lab, we understand the implementation of double circularly linked list. It’s confusing to maintain previous and next node of doubly linked list and head and tail pointer of circularly linked list at same time

To delete node from start in double circularly linked list, we have to maintain 3 pointers. 1st is indicating the head node; 2nd is indicating the next of head and 3rd is indicating the previous of head. The previous of head should be given the address of aft node and deallocate the memory of head node. The head pointer should be updated to next node.

In this “deleteNodeAfter(struct node\_d \* head, int idx” function, one node indicates the index specified node and one contain the data of next node. To delete a node after a specified index, we have to maintain 2 pointers. 1st indicates the specified node and 2nd indicates the node to be deleted. We assign the next of to be index the next of index specified node and deallocate the memory for desired node. To insert a node at beginning, a new node is allocated and also assigned the address of head pointer. We must update the head pointer in order not to lose the start of linked list.We must update the address assigned to tail pointer which is of head node.

In “void printMemMap(struct node\_d \* head)” function, we use %u to print the address of every node. It is simply a function to print how the data is linked in linked list by using a for loop.