LAB MANAUAL 4

Linked List

AIM: To write a program that implements the basic operations of the linked list in C++

Linked List Overview

List is a collection of components, called nodes. Every node (except the last
node) contains the address of the next node. Every node in a linked list has two components:

1. one to store the relevant information (that is, data)
2. one to store the address, called the link or next , of the next node in the list.

- ✓ The address of the first node in the list is stored in a separate location, called

the head or first

- ✓ The address of the last node in the list is stored in a separate location, called

the tail or last

Linked list

SINGLY LINKED LIST

AIM:

To perform all the singly linked list operations

Write a C++ program to perform the following operations :

a) Create a SLL of integers.
b) Delete an integer from SLL,
c) insert an integer at middle of the list,
d) Show the elements in the list
e) Check if list is empty
f) Count the total element in the list

ALGORITHM STEPS

Step 1: Declare the functions to create, display, delete, count and check empty list. (You may use class)
Step 2: Declare the variables in the main function.
Step 3: In a switch case get each functions number.
Step 4: To append the list created in the memory
Step 5: Assign a variable temp using pointers.
Step 6: To delete a node create a dummy variable.
Step 7: Check if the list is empty otherwise display the list using for statement. Step 8: To insert a node in as first element

INPUT/OUTPUT

Singly Linked List Menu:
1. Create or Append List
2. Insert in Beginning
4. Remove from the List
5. Count element in the list
6. Display
7. Exit

Lab Exercise 1.
Use the code given in the Lab section to do create the following function:

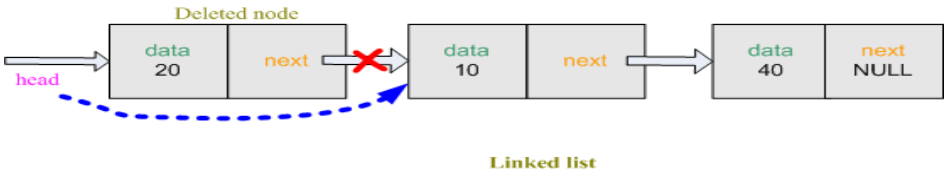
1. Add a function that Insert in Middle of Linkedlist
2. Modify Add a function that will enable you to Enter more items in insert function using (Y/N): E.g

   Enter your Choice: 1
   Enter number to add to list: 12
   Enter more(y/n): y
   Enter number to add to list: 13
   Enter more(y/n): n

3. Add a function (Count()) that count total element added in Linkedlist
4. Sort the nodes in the linked list

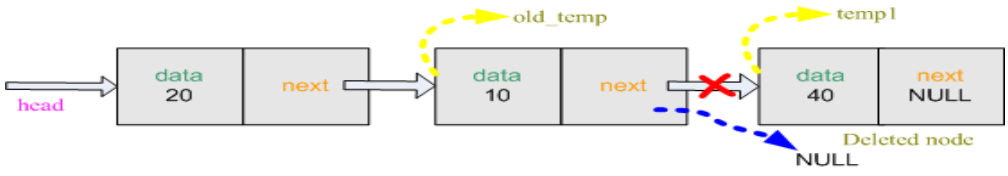
   Hint on sorting : Linked list sorting is very simple. It is just like ordinary array sorting. First we create two temporary node node *temp1, *temp2 and allocate space for it. Transfer the address of first node to temp1 and address of second node to temp2 . Now check if temp1->data is greater than temp2->data . If yes then exchange the data. Similarly, we perform this checking for all the nodes.

5. Delete from front/head of list



Linked list

6. Delete from last/tail of list



Linked list

Your final code should have the following in the menu

Singly Linked List Menu:
1. Create or Append List
2. Insert in Beginning
3. Insert in Middle

4. Remove from the List (Head/Tail) H/T or you can write a separate function for it
5. Count Element in List
6. Show List of Elements
7. Exit

All the Singly Linked List operations are performed.

Group Assignment 2.

Use the doubly linked list code given in the lab, modify it and complete the following functions;

1. Add to list using cin>>
2. Remove from item the list
3. Count Element in List
4. Show List of Elements forward and reverse
5. Exit

Submission on Monday 19th October, 2015 before or by 5pm