A purple text on a black background

AI-generated content may be incorrect.

**LAB 5 - SINGLY LINKED LIST (DISPLAY NODES)**

Name: Omar Afef Mohammed

Roll number: SU92-BSSEM-S24-086

Section: 3A

Teacher: Mr. Rasikh Ali

Subject: DSA LAB

Date: February 3, 2025

Task: Implement functions to display the first node, last node, Nth node, and center node of a singly linked list.

**ANSWER**

**CODE**

#include <iostream>

using namespace std;

class Node{

    public:

        int data;

        Node\* next;

        Node(int data){

            this->data=data;

            next=NULL;

        }

};

class LinkedList{

    public:

        Node\* head;

        LinkedList(){

            head=NULL;

        }

        //display

        void display(){

            Node\* tem=head;

            while(tem!=NULL){

                cout<<tem->data<<"->";

                tem=tem->next;

            }

            cout<<"NULL\n";

        }

            //at start

        void insertAtStart(int d){

            Node\* newNode=new Node(d);

            newNode->next=this->head;

            this->head=newNode;

        }

    //display first node

        void DisplayFirst(){

                Node\* tem=head;

                if(tem==NULL){

                    cout<<"NULL\n";

                    return;

                }

                else if(tem->next==NULL){

                    cout<<tem->data<<" | NULL\n";

                    return;

                }

                else{

                    cout<<tem->data<<" | "<<tem->next<<endl;

                    return;

                }

            }

            //display last node

            void DisplayLast(){

                 if (head == NULL) {

                    cout << "NULL\n";

                    return;

                }

                Node\* tem=head;

                while(tem->next!=NULL){

                    tem=tem->next;

                }

                cout<<tem->data<<" | Null\n";

            }

            //at Nth node

            void DisplayNth(int num){

                Node\* tem=head;

                num--;

                while( num>0){

                    if(tem->next==NULL){

                        cout << "Position out of bounds.\n";

                        return;

                    }

                    tem=tem->next;

                    num--;

                }

                if (tem->next == NULL) {

                    cout << tem->data << " | Null\n";

                } else {

                    cout << tem->data << " | " << tem->next << endl;

                }

            }

            //at center node

            void DisplayCenter(){

                Node\* tem=head;

                int num=0;

                while(tem->next!=NULL){

                    num++;

                    tem=tem->next;

                }

                num=num/2;

                tem=head;

                while( num>0){

                    tem=tem->next;

                    num--;

                }

                cout << tem->data << " | " << tem->next << endl;

            }

};

int main(){

    LinkedList link;

    link.insertAtStart(4);

    link.insertAtStart(3);

    link.insertAtStart(2);

    link.insertAtStart(1);

    link.insertAtStart(0);

    cout<<"The first of the link:\n";

    link.DisplayFirst();

    cout<<endl;

    cout<<"The last of the link:\n";

    link.DisplayLast();

    cout<<endl;

    cout<<"The 2th of the link:\n";

    link.DisplayNth(2);

    cout<<endl;

    cout<<"The center of the link:\n";

    link.DisplayCenter();

    cout<<endl;

    cout<<"The whole of the link:\n";

    link.display();

}

**How it works:**

1. **First Node: Print the head node.**
2. **Last Node: Traverse until next == NULL and print the last node.**
3. **Nth Node: Traverse n-1 times, checking if n is out of range.**
4. **Center Node:**
   * **Find the total number of nodes.**
   * **Divide by 2 to get the middle index.**
   * **Traverse again to that position.**

**Why it works:**

* **O(1) for the first node, since we directly access head.**
* **O(n) for the last, Nth, and center nodes, since we traverse the list**

**OUTPUT**

**A computer screen with white text

AI-generated content may be incorrect.**