Name:

Wajiha Zahid

Roll No:

S24-040

Subject:

“DSA LAB ”

Section

BSSE-3A

Resource Person:

Sir Rasikh Ali

(Lab Task 5)

Q1: Singly Linked List (Display Nodes)

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

Answer:

```
#include <iostream>

using namespace std;
struct Node {
    int data;
    Node* next;
};
Node* createNode(int data) {
    Node* newNode = new Node();
    newNode->data = data;
    newNode->next = NULL;
    return newNode;
}
void insertAtBeginning(Node** head, int data) {
    Node* newNode = createNode(data);
    newNode->next = *head;
```

```
    *head = newNode;
}
void insertAtEnd(Node** head, int data) {
    Node* newNode = createNode(data);

    if (*head == NULL) {
        *head = newNode;
        return;
    }

    Node* temp = *head;

    while (temp->next != NULL) {
        temp = temp->next;
    }

    temp->next = newNode;
}
void displayFirstNode(Node* head) {
    if (head == NULL) {
        cout << "List is empty." << endl;
        return;
    }

    cout << "First node: " << head->data << endl;
```

```
}
void displayLastNode(Node* head) {
    if (head == NULL) {
        cout << "List is empty." << endl;
        return;
    }

    Node* temp = head;

    while (temp->next != NULL) {
        temp = temp->next;
    }

    cout << "Last node: " << temp->data << endl;
}
void displayNthNode(Node* head, int position) {
    if (head == NULL || position <= 0) {
        cout << "Invalid position." << endl;
        return;
    }

    Node* temp = head;
    int count = 1;

    while (temp != NULL && count < position) {
```

```
        temp = temp->next;
        count++;
    }

    if (temp == NULL) {
        cout << "Position out of range." << endl;
    } else {
        cout << "Node at position " << position << ": " << temp->data << endl;
    }
}
void displayCentreNode(Node* head) {
    if (head == NULL) {
        cout << "List is empty." << endl;
        return;
    }

    Node* slow = head;
    Node* fast = head;

    while (fast != NULL && fast->next != NULL) {
        slow = slow->next;
        fast = fast->next->next;
    }

    cout << "Centre node: " << slow->data << endl;
```

```
}
void deleteNode(Node** head, int position) {

    if (*head == NULL) {
        cout << "List is empty." << endl;
        return;
    }

    if (position == 1) {
        Node* temp = *head;
        *head = (*head)->next;
        delete temp;
        return;
    }

    Node* temp = *head;
    Node* prev = NULL;
    int count = 1;

    while (temp != NULL && count < position) {
        prev = temp;
        temp = temp->next;
        count++;
    }
```

```
    if (temp == NULL) {
        cout << "Position out of range." << endl;
        return;
    }

    prev->next = temp->next;
    delete temp;
    cout << "Node at position " << position << " deleted successfully." << endl;
}
void reverseList(Node** head) {
    Node* prev = NULL;
    Node* current = *head;
    Node* nextNode = NULL;

    while (current != NULL) {
        nextNode = current->next;
        current->next = prev;
        prev = current;
        current = nextNode;
    }

    *head = prev;
}
void displayList(Node* head) {
    Node* temp = head;
```

```
    while (temp != NULL) {
        cout << temp->data << " -> ";
        temp = temp->next;
    }
    cout << "NULL" << endl;
}

int main() {

    Node* head = NULL;
    insertAtBeginning(&head, 5);
    insertAtBeginning(&head, 3);
    insertAtEnd(&head, 8);
    insertAtEnd(&head, 10);
    insertAtEnd(&head, 15);

    cout << "Original list: ";
    displayList(head);

    displayFirstNode(head);
    displayLastNode(head);
    displayCentreNode(head);
    deleteNode(&head, 3);

    cout << "List after deleting node at position 3: ";
```

```
    displayList(head);

    // Reverse the list
    reverseList(&head);

    cout << "Reversed list: ";
    displayList(head);

    return 0;
}
```

Output:

```
Original list: 3 -> 5 -> 8 -> 10 -> 15 -> NULL
First node: 3
Last node: 15
Centre node: 8
Node at position 3 deleted successfully.
List after deleting node at position 3: 3 -> 5 -> 10 -> 15 -> NULL
Reversed list: 15 -> 10 -> 5 -> 3 -> NULL

--------------------------------
Process exited after 0.3791 seconds with return value 0
Press any key to continue . . .
```