Name :M Ijaz

Submitted to

Sir Rasikh

Roll no . 073

Subject . Lab Data structure

Lab .

Answer the following questions

Question no 1.

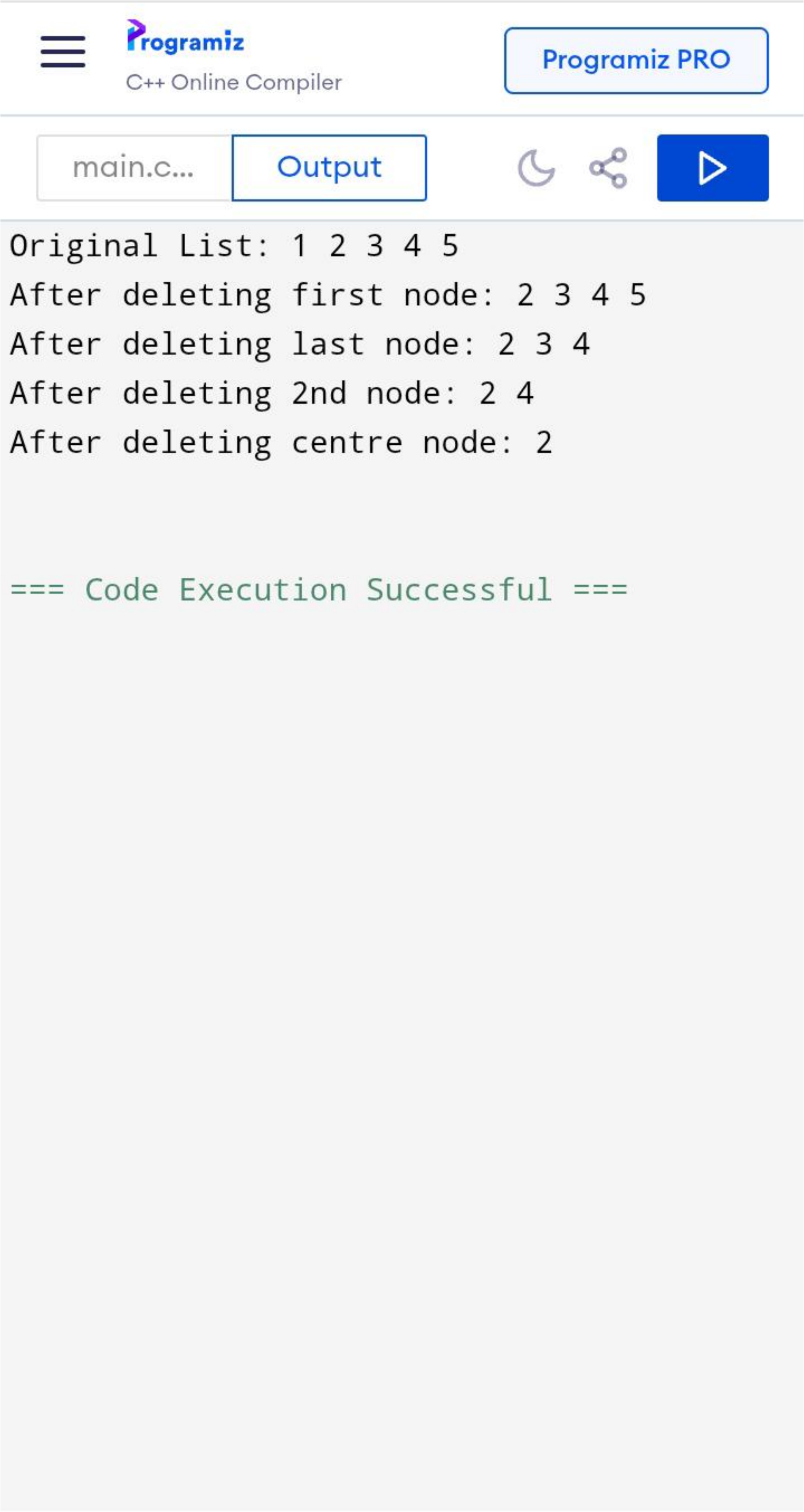
Implement functions to delete the first node, last node, Nth node, and centre node of a singly linked list.

Answer .

```
#include <iostream>
using namespace std;

struct Node {
    int data;
    Node* next;
    Node(int val) : data(val), next(nullptr) {}
};

class LinkedList {
public:
    Node* head;
    LinkedList() : head(nullptr) {}
```

```
void append(int val) {
    Node* newNode = new Node(val);
    if (!head) {
        head = newNode;
        return;
    }
    Node* temp = head;
    while (temp->next) {
        temp = temp->next;
    }
    temp->next = newNode;
}

void deleteFirst() {
    if (!head) return;
    Node* temp = head;
    head = head->next;
    delete temp;
}

void deleteLast() {
    if (!head) return;
    if (!head->next) {
        delete head;
        head = nullptr;
        return;
    }
    Node* temp = head;
    while (temp->next && temp->next->next) {
        temp = temp->next;
    }
    delete temp->next;
    temp->next = nullptr;
}

void deleteNth(int n) {
    if (!head) return;
    if (n == 1) {
        deleteFirst();
        return;
    }
    Node* temp = head;
    for (int i = 1; temp != nullptr && i < n - 1; ++i) {
        temp = temp->next;
    }
    if (temp == nullptr || temp->next == nullptr) return;
    Node* nodeToDelete = temp->next;
    temp->next = temp->next->next;
    delete nodeToDelete;
}

void deleteCentre() {
    if (!head || !head->next) return;
    Node *slow = head, *fast = head, *prev = nullptr;
```

```
        while (fast && fast->next) {
            fast = fast->next->next;
            prev = slow;
            slow = slow->next;
        }
        prev->next = slow->next;
        delete slow;
    }

    void display() {
        Node* temp = head;
        while (temp) {
            cout << temp->data << " ";
            temp = temp->next;
        }
        cout << endl;
    }
};

int main() {
    LinkedList list;
    list.append(1);
    list.append(2);
    list.append(3);
    list.append(4);
    list.append(5);
    cout << "Original List: ";
    list.display();

    list.deleteFirst();
    cout << "After deleting first node: ";
    list.display();

    list.deleteLast();
    cout << "After deleting last node: ";
    list.display();

    list.deleteNth(2);
    cout << "After deleting 2nd node: ";
    list.display();

    list.deleteCentre();
    cout << "After deleting centre node: ";
    list.display();

    return 0;
}
```



```
Original List: 1 2 3 4 5
After deleting first node: 2 3 4 5
After deleting last node: 2 3 4
After deleting 2nd node: 2 4
After deleting centre node: 2


=== Code Execution Successful ===
```