Question1.cpp

```cpp
// <----Lab 03 - Singly Linked List---->

// Q1. Implement a singly linked list class with the following functions:
// a) Insert a node at head
// b) Insert a node at tail/end/back
// c) Insert a node at any position
// d) Delete a node by value
// e) Delete head
// f) Delete tail
// g) Delete a node at any position.

#include<iostream>
using namespace std;

class node {
    public:
        int data;
        node* next;
        node(int value) {
            data=value;
            next=NULL;
        }
        node(int value,node* nxt) {
            data=value;
            next=nxt;
        }
};


class SLL {  //SLL = Singly Linked List
    node* head=NULL;
    public:
        void insertAttail(int value) {
            node* n=new node(value);
            if(head==NULL) {  //Check if Linked List Empty.
                head=n;
                return;
            }
            node* temp=head;
            while(temp->next!=NULL) {
                temp=temp->next;
            }
            temp->next=n;
        }
        void insertAtPos(int posvalue,int value) {
            int count=0;
            node* temp=head;
            while(temp->next!=NULL&&count<posvalue-1) {
                temp=temp->next;
                count++;
            }
            node* n=new node(value,temp->next);
            temp->next=n;
        }
        void display() {
            node* temp=head;
            cout<<"[HEAD] ";
            while(temp!=NULL) {
                cout<<temp->data<<"| "<<temp->next<<" -> ";
                temp=temp->next;
            }
            cout<<"NULL [TAIL]"<<endl;
        }
        void insertAthead(int value) {
            node* n=new node(value);
            n->next=head;
            head=n;
        }
        void deletion(int value) {
            if(head==NULL) {
                return;
            }
            node* temp=head;
            if(head==NULL){
                cout<<"Empty Linked List, returning\n";
                return;
            }
            if(head->data==value){
                head=head->next;
                return;
            }
            while(temp->next->data!=value ) {
                if(temp->next->next==NULL){
                    cout<<"Value not found... Returning\n";
                    return;
                }
                temp=temp->next;
            }
            node* todelete=temp->next;
            temp->next=temp->next->next;

            delete todelete;
        }
        void deleteAthead() {
            if(head==NULL) {
                return;
            }
            node* todelete=head;
            head=head->next;
            delete todelete;
        }
        void deleteAtPos(int posvalue) {
            if(head==NULL) {
                return;
            }
            int count=0;
            node* temp=head;
            while(temp->next!=NULL && count<posvalue-1) {
                temp=temp->next;
```

```cpp
                count++;
            }
            node* todelete=temp->next;
            temp->next=temp->next->next;
            delete todelete;
        }
        void deleteAttail() {
            if(head==NULL) { //If linked list empty.
                return;
            }
            node* temp=head;
            if(head->next==NULL) { //If linked list has 1 item only.
                head=NULL;
                delete temp;
                return;
            }
            while(temp->next->next!=NULL) {
                temp=temp->next;
            }
            delete temp->next;
            temp->next=NULL;
        }

};


int main() {
    SLL list;
    float input=0;
    int value;
    while(input!=0.5) {
        cout<<"----------------------------------------------------\n";
        cout<<"CURRENT LINKED LIST:\n";
        list.display();
        cout<<"----------------------------------------------------\n";
        cout<<"What would you like to do with the linked list?\n";
        cout<<"1. Insert\t2. Delete\nEnter 0.5 to Exit\n[Anything else will default to Delete]\n";
        cin>>input;
        if(input==1) {
            cout<<"Enter Value to insert: ";
            cin>>value;
            cout<<"Where to Insert in Linked List?\n";
            cout<<"1. At head\t2. At tail\t3. At specified Position\n[Any other value will default to Insertion at Head]\n";
            cin>>input;
            if(input == 2){
                list.insertAttail(value);
            }
            else if(input == 3){
                int pos;
                cout<<"Enter the Position to insert into: ";
                cin>>pos;
                list.insertAtPos(pos,value);
            }
            else{
                list.insertAthead(value);
            }

        }
        else if(input==0.5){
            break;
        }
        else{
            cout<<"Where to Delete from Linked List?\n";
            cout<<"1. At head\t2. At tail\t3. At specified Position\t 4. Delete a specific Value\n[Any other value will default to Deletion from Head]\n";
            cin>>input;
            if(input == 2){
                list.deleteAttail();
            }
            else if(input == 3){
                int pos;
                cout<<"Enter the Position to Delete from: ";
                cin>>pos;
                list.deleteAtPos(pos);
            }
            else if(input == 4){
                int pos;
                cout<<"Enter the Value to Delete: ";
                cin>>value;
                list.deletion(value);
            }
            else{
                list.deleteAthead();
            }
        }

    }
}
```