Question1.cpp

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

// 1. 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;
```

```
54          }
55          void display() {
56              node* temp=head;
57              cout<<"[HEAD] ";
58              while(temp!=NULL) {
59                  cout<<temp->data<<"| "<<temp->next<<" -> ";
60                  temp=temp->next;
61              }
62              cout<<"NULL [TAIL]"<<endl;
63          }
64          void insertAthead(int value) {
65              node* n=new node(value);
66              n->next=head;
67              head=n;
68          }
69          void deletion(int value) {
70              if(head==NULL) {
71                  return;
72              }
73              node* temp=head;
74              if(head==NULL){
75                  cout<<"Empty Linked List, returning\n";
76                  return;
77              }
78              if(head->data==value){
79                  head=head->next;
80                  return;
81              }
82              while(temp->next->data!=value ) {
83                  if(temp->next->next==NULL){
84                      cout<<"Value not found... Returning\n";
85                      return;
86                  }
87                  temp=temp->next;
88              }
89              node* todelete=temp->next;
90              temp->next=temp->next->next;
91
92              delete todelete;
93          }
94          void deleteAthead() {
95              if(head==NULL) {
96                  return;
97              }
98              node* todelete=head;
99              head=head->next;
100             delete todelete;
101         }
102         void deleteAtPos(int posvalue) {
103             if(head==NULL) {
104                 return;
105             }
106             int count=0;
107             node* temp=head;
108             while(temp->next!=NULL && count<posvalue-1) {
109                 temp=temp->next;
```

```
110                 count++;
111             }
112             node* todelete=temp->next;
113             temp->next=temp->next->next;
114             delete todelete;
115         }
116         void deleteAttail() {
117             if(head==NULL) { //If linked list empty.
118                 return;
119             }
120             node* temp=head;
121             if(head->next==NULL) { //If linked list has 1 item only.
122                 head=NULL;
123                 delete temp;
124                 return;
125             }
126             while(temp->next->next!=NULL) {
127                 temp=temp->next;
128             }
129             delete temp->next;
130             temp->next=NULL;
131         }
132 
133 };
134 
135 
136 int main() {
137     SLL list;
138     float input=0;
139     int value;
140     while(input!=0.5) {
141         cout<<"-------------------------------------------------------\n";
142         cout<<"CURRENT LINKED LIST:\n";
143         list.display();
144         cout<<"-------------------------------------------------------\n";
145         cout<<"What would you like to do with the linked list?\n";
146         cout<<"1. Insert\t2. Delete\nEnter 0.5 to Exit\n[Anything else will default to Delete]\n";
147         cin>>input;
148         if(input==1) {
149             cout<<"Enter Value to insert: ";
150             cin>>value;
151             cout<<"Where to Insert in Linked List?\n";
152             cout<<"1. At head\t2. At tail\t3. At specified Position\n[Any other value will default
    to Insertion at Head]\n";
153             cin>>input;
154             if(input == 2){
155                 list.insertAttail(value);
156             }
157             else if(input == 3){
158                 int pos;
159                 cout<<"Enter the Position to insert into: ";
160                 cin>>pos;
161                 list.insertAtPos(pos,value);
162             }
163             else{
164                 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();
            }
        }

    }
}
```