

Question1.cpp

```
1 // <----Lab 03 - Singly Linked List---->
2
3 // Q1. Implement a singly linked list class with the following functions:
4 // a) Insert a node at head
5 // b) Insert a node at tail/end/back
6 // c) Insert a node at any position
7 // d) Delete a node by value
8 // e) Delete head
9 // f) Delete tail
10 // g) Delete a node at any position.
11
12 #include<iostream>
13 using namespace std;
14
15 class node {
16     public:
17         int data;
18         node* next;
19         node(int value) {
20             data=value;
21             next=NULL;
22         }
23         node(int value,node* nxt) {
24             data=value;
25             next=nxt;
26         }
27 };
28
29
30 class SLL { //SLL = Singly Linked List
31     node* head=NULL;
32     public:
33         void insertAttail(int value) {
34             node* n=new node(value);
35             if(head==NULL) { //Check if Linked List Empty.
36                 head=n;
37                 return;
38             }
39             node* temp=head;
40             while(temp->next!=NULL) {
41                 temp=temp->next;
42             }
43             temp->next=n;
44         }
45         void insertAtPos(int posvalue,int value) {
46             int count=0;
47             node* temp=head;
48             while(temp->next!=NULL&&count<posvalue-1) {
49                 temp=temp->next;
50                 count++;
51             }
52             node* n=new node(value,temp->next);
53             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);

```

```
165         }
166
167     }
168     else if(input==0.5){
169         break;
170     }
171     else{
172         cout<<"Where to Delete from Linked List?\n";
173         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";
174         cin>>input;
175         if(input == 2){
176             list.deleteAttail();
177         }
178         else if(input == 3){
179             int pos;
180             cout<<"Enter the Position to Delete from: ";
181             cin>>pos;
182             list.deleteAtPos(pos);
183         }
184         else if(input == 4){
185             int pos;
186             cout<<"Enter the Value to Delete: ";
187             cin>>value;
188             list.deletion(value);
189         }
190         else{
191             list.deleteAthead();
192         }
193     }
194 }
195 }
196 }
```