






singlyLL.c
~/DSC_Lab/LabCycle1

Open Save

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 struct node
5 {
6     int data;
7     struct node *next;
8 };
9 typedef struct node Node;
10
11 Node* newNode(int data)
12 {
13     Node *n=(Node*)malloc(sizeof(Node));
14     n->data=data;
15     n->next=NULL;
16     return n;
17 }
18
19 struct list
20 {
21     Node *head;
22     Node *tail;
23     int min;
24     int max;
25 };
26 typedef struct list List;
27
28 List* newList()
29 {
30     List *l=(List*)malloc(sizeof(List));
31     l->head=NULL;
32     l->tail=NULL;
33     return l;
34 }
35
36 void insertTail(List *l,int data)
37 {
38     Node *new_node=newNode(data);
39     if(l->head==NULL)
40         l->head=l->tail=new_node;
41     else
42     {
43         l->tail->next=new_node;
44         l->tail=new_node;
45     }
46 }
47
48 void insertAtPos(List *l,int data,int pos)
```

```
singlyLL.c
~/DSC_Lab/LabCycle1

47
48 void insertAtPos(List *l,int data,int pos)
49 {
50     Node *n=l->head,*prev=NULL;
51     int i=0;
52     if(pos<=0)
53     {
54         printf("Position does not exist\n");
55         return;
56     }
57     if(pos==1)
58     {
59         Node *new_node=newNode(data);//put this here,coz if pos doesnot exist then this node
will not be created
60         new_node->next=l->head;
61         l->head=new_node;
62         if(l->tail==NULL)
63             l->tail=l->head;
64         return;
65     }
66     while(n!=NULL && i<pos-1)
67     {
68         prev=n;
69         n=n->next;
70         i++;
71     }
72     if(n==NULL)
73     {
74         printf("Position does not exist\n");
75         return;
76     }
77     Node *new_node=newNode(data);
78     prev->next=new_node;
79     new_node->next=n;//tail will remain as it is, don't need to alter
80 }
81
82 void DeleteHead(List *l)
83 {
84     Node *temp=l->head;
85     if(l->head==NULL)
86         return;
87     else if(l->head==l->tail)
88         l->head=l->tail=NULL;
89     else
90         l->head=l->head->next;
91     free(temp);
92 }
93
```

Open ▾  singlyLL.c ~/DSC_Lab/LabCycle1 Save    

```
93
94 void DeleteValue(List *l,int data)
95 {
96     Node *n=l->head,*prev=NULL;
97     if(l->head->data==data)
98     {
99         DeleteHead(l);
100         return;
101     }
102     while(n!=NULL && n->data!=data)
103     {
104         prev=n;
105         n=n->next;
106     }
107     if(n==NULL)
108     {
109         printf("Element with value %d does not exist in the list\n",data);
110         return;
111     }
112     Node *temp=n;
113     prev->next=n->next;
114     if(l->tail==temp)
115         l->tail=prev;
116     free(temp);
117 }
118
119 void findMin(List *l)
120 {
121     Node *n=l->head;
122     if(l->head==NULL)
123     {
124         printf("There are no elements in the list\n");
125         return;
126     }
127     int min=n->data;
128     n=n->next;
129     while(n!=NULL)
130     {
131         if(n->data<min)
132             min=n->data;
133         n=n->next;
134     }
135     l->min=min;
136 }
137
138 void findMax(List *l)
139 {
140     Node *n=l->head;
```






Open ▾ +

singlyLL.c
~/DSC_Lab/LabCycle1

Save ≡ - □ ×






```
137
138 void findMax(List *l)
139 {
140     Node *n=l->head;
141     if(l->head==NULL)
142     {
143         printf("There are no elements in the list\n");
144         return;
145     }
146     int max=n->data;
147     n=n->next;
148     while(n!=NULL)
149     {
150         if(n->data>max)
151             max=n->data;
152         n=n->next;
153     }
154     l->max=max;
155 }
156
157 void PrintList(List *l)
158 {
159     printf("The List is : ");
160     Node *n=l->head;
161     if(n==NULL)
162         printf("The list is empty");
163     while(n!=NULL)
164     {
165         printf("%d ",n->data);
166         n=n->next;
167     }
168     printf("\n");
169 }
170
171 void freeList(List *l)
172 {
173     Node *n=l->head,*temp;
174     while(n!=NULL)
175     {
176         temp=n;
177         n=n->next;
178         free(temp);
179     }
180 }
181
182 int main()
183 {
184     List *l=newList();
```

C ▾ Tab Width: 4 ▾ Ln 184, Col 1 ▾ INS

Open ▾  singlyLL.c ~/DSC_Lab/LabCycle1 Save    

```
181
182 int main()
183 {
184     List *l=newList();
185     char ch;
186     printf("Menu :\n");
187     printf("Insert At Tail(a)\n");
188     printf("Insert At Position(b)\n");
189     printf("Delete At Head(c)\n");
190     printf("Delete perticular value(d)\n");
191     printf("Find minimum element(e)\n");
192     printf("Find maximum element(f)\n");
193     printf("Exit(*)\n");
194     printf("Enter option : ");
195     scanf("%c",&ch);
196     while(ch!='*')
197     {
198         if(ch=='a')
199         {
200             int data;
201             printf("Enter the data : ");
202             scanf("%d",&data);
203             insertTail(l,data);
204         }
205         else if(ch=='b')
206         {
207             int data,pos;
208             printf("Enter the data and position : ");
209             scanf("%d%d",&data,&pos);
210             insertAtPos(l,data,pos);
211         }
212         else if(ch=='c')
213             DeleteHead(l);
214         else if(ch=='d')
215         {
216             int data;
217             printf("Enter the data to be deleted : ");
218             scanf("%d",&data);
219             DeleteValue(l,data);
220         }
221         else if(ch=='e')
222         {
223             findMin(l);
224             printf("The minimum element is %d\n",l->min);
225         }
226         else if(ch=='f')
227         {
228             findMax(l);
```

C ▾ Tab Width: 4 ▾ Ln 228, Col 1 ▾ INS

Open ▾  singlyLL.c ~/DSC_Lab/LabCycle1 Save    

```
200     l->data,
201     printf("Enter the data : ");
202     scanf("%d",&data);
203     insertTail(l,data);
204 }
205 else if(ch=='b')
206 {
207     int data,pos;
208     printf("Enter the data and position : ");
209     scanf("%d%d",&data,&pos);
210     insertAtPos(l,data,pos);
211 }
212 else if(ch=='c')
213     DeleteHead(l);
214 else if(ch=='d')
215 {
216     int data;
217     printf("Enter the data to be deleted : ");
218     scanf("%d",&data);
219     DeleteValue(l,data);
220 }
221 else if(ch=='e')
222 {
223     findMin(l);
224     printf("The minimum element is %d\n",l->min);
225 }
226 else if(ch=='f')
227 {
228     findMax(l);
229     printf("The maximum element is %d\n",l->max);
230 }
231 PrintList(l);
232 printf("\n");
233 printf("Menu :\n");
234 printf("Insert At Tail(a)\n");
235 printf("Insert At Position(b)\n");
236 printf("Delete At Head(c)\n");
237 printf("Delete perticular value(d)\n");
238 printf("Find minimum element(e)\n");
239 printf("Find maximum element(f)\n");
240 printf("Exit(*)\n");
241 printf("Enter option : ");
242     scanf(" %c",&ch);
243 }
244 freeList(l);
245 free(l);
246 return 0;
247 }
```

Bracket match found on line: 183 C ▾ Tab Width: 4 ▾ Ln 247, Col 2 ▾ INS

```
deven@deven-VirtualBox: ~/DSC_Lab/LabCycle1
deven@deven-VirtualBox: ~/DSC_Lab/LabCycle1$ ./a.out
Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : a
Enter the data : 1
The List is : 1

Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : b
Enter the data and position : 2 1
The List is : 2 1

Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : b
Enter the data and position : 3 5
Position does not exist
The List is : 2 1

Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : c
```

```
deven@deven-VirtualBox: ~/DSC_Lab/LabCycle1
Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : c
The List is : 1

Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : d
Enter the data to be deleted : 1
The List is : The list is empty

Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : a
Enter the data : 4
The List is : 4

Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : a
Enter the data : 5
The List is : 4 5
```



```
deven@deven-VirtualBox: ~/DSC_Lab/LabCycle1
Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : d
Enter the data to be deleted : 7
Element with value 7 does not exist in the list
The List is : 4 5

Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : e
The minimum element is 4
The List is : 4 5

Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : f
The maximum element is 5
The List is : 4 5

Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : *
deven@deven-VirtualBox:~/DSC Lab/LabCycle1$
```