

```

1  /*
2  TITLE:IMPLEMENTATION OF DOUBLY LINKED LIST
3  NAME:Tauseef Mushtaque Ali Shaikh
4  CLASS: S.Y.[C0]
5  ROLLNO: 18C063
6  SUBJECT: DS
7  DATE: 23/9/19
8  DISCRIPTION: In this Program a doubly linked list is created and different
9  function is performed i.e. insert, remove, display.
10 */
11 #include<stdio.h>
12 #include<stdlib.h>
13
14 struct DLL
15 {
16     int data;
17     struct DLL *next, *prev;
18 };
19
20 struct DLL *insertAtEnd(struct DLL *h,int d)
21 {
22     struct DLL *p,*tmp;
23     p=(struct DLL *)malloc(sizeof(struct DLL));
24     p->data=d;
25     p->next=NULL;
26     p->prev=NULL;
27     if(h==NULL)
28     {
29         h=p;
30     }
31     else
32     {
33         tmp=h;
34         while(tmp->next!=NULL)
35             tmp=tmp->next;
36         tmp->next=p;
37         p->prev=tmp;
38     }
39     return h;
40 }
41
42 struct DLL *insertAtStart(struct DLL *h,int d)
43 {
44     struct DLL *p;
45     p=(struct DLL *)malloc(sizeof(struct DLL));
46     p->data=d;
47     p->next=h;
48     p->prev=NULL;
49     if(h!=NULL)
50         h->prev=p;
51     h=p;
52     return h;
53 }
54
55 struct DLL *insertAfter(struct DLL *h,int key,int d)
56 {
57     struct DLL *p,*tmp;
58     p=(struct DLL *)malloc(sizeof(struct DLL));
59     p->data=d;
60     p->next=NULL;
61     p->prev=NULL;
62     if(h==NULL)
63     {
64         h=p;

```

```

64 }
65 else
66 {
67     tmp=h;
68     while(tmp!=NULL && tmp->data!=key )
69     tmp=tmp->next;
70     if(tmp!=NULL)
71     {
72         p->next=tmp->next;
73         p->prev=tmp;
74         if(tmp->next!=NULL)
75             (tmp->next)->prev=p;
76         tmp->next=p;
77     }
78     else
79         printf("\n\tGiven Node %d does not exist in the Linked List.",key);
80 }
81 return h;
82 }
83
84 struct DLL *removelast(struct DLL *h)
85 {
86     struct DLL *tmp;
87     tmp=h;
88     if(h!=NULL)
89     {
90         if(h->next!=NULL)
91         {
92             while(tmp->next!=NULL)
93             tmp=tmp->next;
94             (tmp->prev)->next=NULL;
95         }
96         else
97             h=NULL;
98         free(tmp);
99     }
100     else
101         printf("\nLL is empty.");
102     return h;
103 }
104
105 struct DLL *removeAfter(struct DLL *h,int key)
106 {
107     struct DLL *tmp,*p;
108     tmp=h;
109     if(h!=NULL)
110     {
111         while(tmp!=NULL && tmp->data!=key)
112             tmp=tmp->next;
113         if(tmp!=NULL)
114         {
115             if(tmp->next!=NULL)
116             {
117                 p=tmp->next;
118                 if(p->next!=NULL)
119                     (p->next)->prev=tmp;
120                 tmp->next=p->next;
121                 p->next=NULL;
122                 p->prev=NULL;
123                 free(p);
124             }
125             else
126                 printf("\nGiven Node is the last Node.");
127         }

```

```

128 else
129 printf("\nGiven key does not exist.");
130 }
131 else
132 printf("\nLL is empty.");
133 return h;
134 }
135
136 void display(struct DLL *h)
137 {
138 struct DLL *tmp;
139 tmp=h;
140 if(h!=NULL)
141 {
142 printf("\n\n\t\tLinked List Contents..\n");
143 while(tmp!=NULL)
144 {
145 printf("\t%d\n",tmp->data);
146 tmp=tmp->next;
147 }
148 }
149 else
150 printf("\nLL is empty.");
151 }
152
153 void displayRev(struct DLL *h)
154 {
155 struct DLL *tmp;
156 tmp=h;
157 if(h!=NULL)
158 {
159 printf("\n\n\t\tLinked List Contents in Reverse Order..\n");
160 while(tmp->next!=NULL)
161 tmp=tmp->next;
162 do
163 {
164 printf("\t%d\n",tmp->data);
165 tmp=tmp->prev;
166 }while(tmp!=NULL);
167 }
168 else
169 printf("\nLL is empty.");
170 }
171
172 int main()
173 {
174 struct DLL *head;
175 int ch,d,k;
176 head=NULL;
177 while(1)
178 {
179
180 printf("\n\t\tMENU");
181 printf("\n\n1. INSERT AT END\n2. INSERT AFTER\n3. INSERT AT START\n4.
REMOVE FROM LAST\n5. REMOVE AFTER\n6. DISPLAY\n7. DISPLAY REVERSE\n0.
EXIT\n");
182 printf("\nENTER THE CHOICE: ");
183 scanf("%d",&ch);
184
185 switch(ch)
186 {
187 case 1:
188 printf("\nEnter the element to be insert: ");
189 scanf("%d",&d);

```

```

190         head=insertAtEnd(head,d);
191         break;
192
193     case 2:
194         printf("Enter the KEY: ");
195         scanf("%d",&k);
196         printf("\nEnter the element to be insert: ");
197         scanf("%d",&d);
198         head=insertAfter(head,k,d);
199         break;
200
201     case 3:
202         printf("\nEnter the element to be insert: ");
203         scanf("%d",&d);
204         head=insertAtStart(head,d);
205         break;
206
207     case 4:
208         head=removeLast(head);
209         break;
210
211     case 5:
212         printf("Enter the KEY: ");
213         scanf("%d",&k);
214         head=removeAfter(head,k);
215         break;
216
217     case 6:
218         display(head);
219         break;
220
221     case 7:
222         displayRev(head);
223
224     case 0:
225         exit(0);
226         break;
227
228     default:
229         printf("INVALID CHOICE!!");
230
231     }
232
233 }
234
235 return 0;
236 }
237

```