

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  struct node
4  {
5      int info;
6      struct node *rlink;
7      struct node *llink;
8  };
9  typedef struct node *NODE;
10 NODE getnode()
11 {
12     NODE x;
13     x=(NODE)malloc(sizeof(struct node));
14     if (x==NULL)
15     {
16         printf("Memory is full\n");
17         exit(0);
18     }
19     return x;
20 }
21 NODE dinser_front(int item,NODE head)
22 {
23     NODE temp,cur;
24     temp=getnode();
25     temp->info=item;
26     temp->llink=NULL;
27     temp->rlink=NULL;
28     cur=head->rlink;
29     head->rlink=temp;
30     temp->llink=head;
31     temp->rlink=cur;
32     cur->llink=temp;
33     return head;
34 }
35 NODE dinser_rear(int item,NODE head)
36 {
37     NODE temp,cur;
38     temp=getnode();
39     temp->info=item;
40     temp->llink=NULL;
41     temp->rlink=NULL;
42     cur=head->llink;

```

```

43     head->llink=temp;
44     temp->rlink=head;
45     cur->rlink=temp;
46     temp->llink=cur;
47     return head;
48 }
49 NODE dinsert_leftpos(int item,NODE head)
50 {
51     NODE cur,prev,temp;
52     if (head->rlink==head)
53     {
54         printf("LIST IS EMPTY.\n");
55         return head;
56     }
57     cur=head->rlink;
58     while (cur!=head)
59     {
60         if (cur->info==item)
61         {
62             break;
63         }
64         cur=cur->rlink;
65     }
66     if (cur==head)
67     {
68         printf("NO ITEM FOUND IN LIST.\n");
69         return head;
70     }
71     prev=cur->llink;
72     temp=getnode();
73     temp->llink=NULL;
74     temp->rlink=NULL;
75     printf("Enter the item to be inserted at the left of the given item:\n");
76     scanf("%d",&temp->info);
77     prev->rlink=temp;
78     temp->llink=prev;
79     temp->rlink=cur;
80     cur->llink=temp;
81     return head;
82 }
83 NODE ddeletepos(int pos, NODE head)

```

```

85     NODE cur, prev, temp;
86     int count=1, flag=0;
87     if (head->rlink==head)
88     {
89         printf("LIST IS EMPTY.\n");
90         return head;
91     }
92     if(pos==1)
93     {
94         cur=head->rlink;
95         prev=cur->rlink;
96         head->rlink=prev;
97         prev->llink=head;
98         printf("THE NODE DELETED IS %d", cur->info);
99         free(cur);
100        return head;
101    }
102    prev=head;
103    cur=head->rlink;
104    while (cur!=head)
105    {
106        if (count==pos)
107        {
108            flag=1;
109            break;
110        }
111        count++;
112        cur=cur->rlink;
113        prev=cur->llink;
114    }
115    if(flag==0)
116    {
117        printf("Invalid Position.\n");
118        return head;
119    }
120    printf("ITEM DELETED AT POSITION %d is %d\n", pos, cur->info);
121    temp=cur->rlink;
122    prev->rlink=cur->rlink;
123    temp->llink=prev;
124    free(cur);
125    return head;

```

```

127 }
128 void ddisplay(NODE head)
129 {
130     NODE temp;
131     if (head->rlink==head)
132     {
133         printf("LIST IS EMPTY.\n");
134     }
135     printf("The contents of the list are:\n");
136     temp=head->rlink;
137     while (temp!=head)
138     {
139         printf("%d\n",temp->info);
140         temp=temp->rlink;
141     }
142 }
143 int main()
144 {
145     NODE head;
146     int item, choice, key, pos;
147     head=getnode();
148     head->llink=head;
149     head->rlink=head;
150     for(;;)
151     {
152         printf("1.Insert front\n2.Insert rear\n3.Insert at Left Position\n4.Delete at specified Position\n5.Display\n6.exit\n");
153         printf("enter the choice\n");
154         scanf("%d",&choice);
155         switch(choice)
156         {
157             case 1: printf("Enter the item at front end:\n");
158                     scanf("%d",&item);
159                     head=dinsert_front(item,head);
160                     break;
161             case 2: printf("Enter the item at rear end:\n");
162                     scanf("%d",&item);
163                     head=dinsert_rear(item,head);
164                     break;
165             case 3: printf("Enter the key element, the left of which an item is to be inserted:\n");
166                     scanf("%d",&key);
167                     head=dinsert_leftpos(key,head);
168                     break;

```

```
        break;
    case 3: printf("Enter the key element, the left of which an item is to be inserted:\n");
            scanf("%d",&key);
            head=dinsert_leftpos(key,head);
            break;
    case 4: printf("Enter position of node to be deleted:\n");
            scanf("%d",&pos);
            head=ddeletepos(pos,head);
            break;
    case 5: ddisplay(head);
            break;
    default: exit(0);
}
}
return 0;
}
```



```
1.Insert front
2.Insert rear
3.Insert at Left Position
4.Delete at specified Position
5.Display
6.exit
```

enter the choice

1

Enter the item at front end:

23

```
1.Insert front
2.Insert rear
3.Insert at Left Position
4.Delete at specified Position
5.Display
6.exit
```

enter the choice

1

Enter the item at front end:

65

```
1.Insert front
2.Insert rear
3.Insert at Left Position
4.Delete at specified Position
5.Display
6.exit
```

enter the choice

2

Enter the item at rear end:

89

```
1.Insert front
2.Insert rear
3.Insert at Left Position
4.Delete at specified Position
5.Display
6.exit
```

enter the choice

2

Enter the item at rear end:

12

```
1.Insert front
2.Insert rear
3.Insert at Left Position
4.Delete at specified Position
5.Display
6.exit
```

enter the choice

5

The contents of the list are:

65

The contents of the list are:

65

23

89

12

1.Insert front

2.Insert rear

3.Insert at Left Position

4.Delete at specified Position

5.Display

6.exit

enter the choice

3

Enter the key element, the left of which an item is to be inserted:

23

Enter the item to be inserted at the left of the given item:

145

1.Insert front

2.Insert rear

3.Insert at Left Position

4.Delete at specified Position

5.Display

6.exit

enter the choice

5

The contents of the list are:

65

145

23

89

12

1.Insert front

2.Insert rear

3.Insert at Left Position

4.Delete at specified Position

5.Display

6.exit

enter the choice

4

Enter position of node to be deleted:

4

ITEM DELETED AT POSITION 4 is 89

1.Insert front

2.Insert rear

3.Insert at Left Position

4.Delete at specified Position

5.Display

6.exit

enter the choice

5

ITEM DELETED AT POSITION 4 is 89

- 1.Insert front
- 2.Insert rear
- 3.Insert at Left Position
- 4.Delete at specified Position
- 5.Display
- 6.exit

enter the choice

5

The contents of the list are:

65

145

23

12

- 1.Insert front
- 2.Insert rear
- 3.Insert at Left Position
- 4.Delete at specified Position
- 5.Display
- 6.exit

enter the choice

6

Process returned 0 (0x0) execution time : 63.378 s

Press any key to continue.

■