```c
#include <stdio.h>
#include <stdlib.h>
struct node
{
int info;
struct node *rlink;
struct node *llink;
};
typedef struct node *NODE;
NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if (x==NULL)
{
printf("Memory full\n");
exit(0);
}
return x;
}
NODE dinsert_front(int item,NODE head)
{
NODE temp,cur;
temp=getnode();
temp->info=item;
temp->llink=NULL;
temp->rlink=NULL;
cur=head->rlink;
head->rlink=temp;
temp->llink=head;
temp->rlink=cur;
cur->llink=temp;
return head;
}
NODE dinsert_rear(int item,NODE head)
{
NODE temp,cur;
temp=getnode();
temp->info=item;
temp->llink=NULL;
temp->rlink=NULL;
```

```
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 ddelete_front(NODE head)
50  {
51  NODE cur,next;
52  if (head->rlink==head)
53  {
54  printf("List is empty\n");
55  return head;
56  }
57  cur=head->rlink;
58  next=cur->rlink;
59  head->rlink=next;
60  next->llink=head;
61  printf("Item deleted at the front end is:%d\n",cur->info);
62  free(cur);
63  return head;
64  }
65  NODE ddelete_rear(NODE head)
66  {
67  NODE cur,prev;
68  if (head->rlink==head)
69  {
70  printf("List is empty\n");
71  return head;
72  }
73  cur=head->llink;
74  prev=cur->llink;
75  prev->rlink=head;
76  head->llink=prev;
77  printf("Item deleted at the rear end is:%d\n",cur->info);
78  free(cur);
```

```
74      prev=cur->llink;
75      prev->rlink=head;
76      head->llink=prev;
77      printf("Item deleted at the rear end is:%d\n",cur->info);
78      free(cur);
79      return head;
80      }
81      void ddisplay(NODE head)
82      {
83      NODE temp;
84      if (head->rlink==head)
85      {
86      printf("List is empty\n");
87      }
88      printf("The contents of the list are:\n");
89      temp=head->rlink;
90      while (temp!=head)
91      {
92      printf("%d\n",temp->info);
93      temp=temp->rlink;
94      }
95      }
96      void dsearch(int key,NODE head)
97      {
98      NODE cur;
99      int count;
100     if (head->rlink==head)
101     {
102     printf("List is empty\n");
103     }
104     cur=head->rlink;
105     count=1;
106     while (cur!=head && cur->info!=key)
107     {
108     cur=cur->rlink;
109     count++;
110     }
111     if (cur==head)
112     {
113     printf("Search unsuccessfull\n");
114     }
```

```
112  {
113  printf("Search unsuccessfull\n");
114  }
115  else
116  {
117  printf("Key element found at the position %d\n",count);
118  }
119  }
120  NODE dinsert_leftpos(int item,NODE head)
121  {
122  NODE cur,prev,temp;
123  if (head->rlink==head)
124  {
125  printf("List is empty\n");
126  return head;
127  }
128  cur=head->rlink;
129  while (cur!=head)
130  {
131  if (cur->info==item)
132  {
133  break;
134  }
135  cur=cur->rlink;
136  }
137  if (cur==head)
138  {
139  printf("No such item found in the list\n");
140  return head;
141  }
142  prev=cur->llink;
143  temp=getnode();
144  temp->llink=NULL;
145  temp->rlink=NULL;
146  printf("Enter the item to be inserted at the left of the given item:\n");
147  scanf("%d",&temp->info);
148  prev->rlink=temp;
149  temp->llink=prev;
150  temp->rlink=cur;
151  cur->llink=temp;
152  return head;
```

```
148     prev->rlink=temp;
149     temp->llink=prev;
150     temp->rlink=cur;
151     cur->llink=temp;
152     return head;
153     }
154     NODE dinsert_rightpos(int item,NODE head)
155     {
156     NODE temp,cur,next;
157     if (head->rlink==head)
158     {
159     printf("List is empty\n");
160     return head;
161     }
162     cur=head->rlink;
163     while (cur!=head)
164     {
165     if (cur->info==item)
166     {
167     break;
168     }
169     cur=cur->rlink;
170     }
171     if (cur==head)
172     {
173     printf("No such item found in the list\n");
174     return head;
175     }
176     next=cur->rlink;
177     temp=getnode();
178     temp->llink=NULL;
179     temp->rlink=NULL;
180     printf("Enter the item to be inserted at the right of the given item:\n");
181     scanf("%d",&temp->info);
182     cur->rlink=temp;
183     temp->llink=cur;
184     next->llink=temp;
185     temp->rlink=next;
186     return head;
187     }
188     NODE ddelete_duplicates(int item,NODE head)
```

```
187 }
188 NODE ddelete_duplicates(int item,NODE head)
189 {
190 NODE prev,cur,next;
191 int count=0;
192 if (head->rlink==head)
193 {
194 printf("List is empty\n");
195 return head;
196 }
197 cur=head->rlink;
198 while (cur!=head)
199 {
200 if (cur->info!=item)
201 {
202 cur=cur->rlink;
203 }
204 else
205 {
206 count++;
207 if (count==1)
208 {
209 cur=cur->rlink;
210 continue;
211 }
212 else
213 {
214 prev=cur->llink;
215 next=cur->rlink;
216 prev->rlink=next;
217 next->llink=prev;
218 free(cur);
219 cur=next;
220 }
221 }
222 }
223 if (count==0)
224 {
225 printf("No such item found in the list\n");
226 }
227 else
```

```
226  }
227  else
228  {
229  printf("Removed all the duplicate elements of the given item successfully\n");
230  }
231  return head;
232  }
233  int main()
234  {
235  NODE head;
236  int item, choice,key;
237  head=getnode();
238  head->llink=head;
239  head->rlink=head;
240  for(;;)
241  {
242  printf("\n1:dinsert front\n2:dinsert rear\n3:ddelete front\n4:ddeleterear\n5:ddisplay\n6:dsearch\n7:dinsert lestpos\n8:dinsert rightpos\n9:ddeleteduplicates\n10:exit\n");
243  printf("enter the choice\n");
244  scanf("%d",&choice);
245  switch(choice)
246  {
247  case 1: printf("Enter the item at front end:\n");
248  scanf("%d",&item);
249  head=dinsert_front(item,head);
250  break;
251  case 2: printf("Enter the item at rear end:\n");
252  scanf("%d",&item);
253  head=dinsert_rear(item,head);
254  break;
255  case 3:head=ddelete_front(head);
256   break;
257  case 4:head=ddelete_rear(head);
258   break;
259  case 5:ddisplay(head);
260   break;
261   case 6:printf("Enter the key element to be searched:\n");
262   scanf("%d",&key);
263   dsearch(key,head);
264   break;
265   case 7:printf("Enter the key element:\n");
266   scanf("%d",&key);
```

```
printf("\n1:dinsert front\n2:dinsert rear\n3:ddelete front\n4:ddeleterear\n5:ddisplay\n6:dsearch\n7:dinsert lestpos\n8:dinsert rightpos\n9:ddeleteduplicates\n10:exit\n");
printf("enter the choice\n");
scanf("%d",&choice);
switch(choice)
{
case 1: printf("Enter the item at front end:\n");
scanf("%d",&item);
head=dinsert_front(item,head);
break;
case 2: printf("Enter the item at rear end:\n");
scanf("%d",&item);
head=dinsert_rear(item,head);
break;
case 3:head=ddelete_front(head);
 break;
case 4:head=ddelete_rear(head);
 break;
case 5:ddisplay(head);
 break;
 case 6:printf("Enter the key element to be searched:\n");
 scanf("%d",&key);
 dsearch(key,head);
 break;
 case 7:printf("Enter the key element:\n");
 scanf("%d",&key);
 head=dinsert_leftpos(key,head);
 break;
case 8:printf("Enter the key element:\n");
 scanf("%d",&key);
 head=dinsert_rightpos(key,head);
 break;
case 9:printf("Enter the key element whose duplicates should be removed:\n");
 scanf("%d",&key);
 head=ddelete_duplicates(key,head);
 break;
default:exit(0);
}
}
return 0;
}
```

```
1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
1
Enter the item at front end:
11

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
1
Enter the item at front end:
12

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
2
Enter the item at rear end:
13

1:dinsert front
2:dinsert rear
3:ddelete front
```

```
2
Enter the item at rear end:
13

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
2
Enter the item at rear end:
14

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
3
Item deleted at the front end is:12

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
3
Item deleted at the front end is:11

1:dinsert front
2:dinsert rear
3:ddelete front
```

```
Item deleted at the front end is:11

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
3
Item deleted at the front end is:13

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
3
Item deleted at the front end is:14

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
3
List is empty

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
```

```
10:exit
enter the choice
3
List is empty

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
2
Enter the item at rear end:
11

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
2
Enter the item at rear end:
13

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
2
Enter the item at rear end:
14
```

```
enter the choice
2
Enter the item at rear end:
14

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
4
Item deleted at the rear end is:14

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
4
Item deleted at the rear end is:13

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
4
Item deleted at the rear end is:11

1:dinsert front
2:dinsert rear
3:ddelete front
```

```
Item deleted at the rear end is:11

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
4
List is empty

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
5
List is empty
The contents of the list are:

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
1
Enter the item at front end:
11

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
```

```
1
Enter the item at front end:
11

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
1
Enter the item at front end:
12

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
1
Enter the item at front end:
13

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
5
The contents of the list are:
13
12
11
```

```
5
The contents of the list are:
13
12
11

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
6
Enter the key element to be searched:
12
Key element found at the position 2

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
7
Enter the key element:
11
Enter the item to be inserted at the left of the given item:
19

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
```

```
The contents of the list are:
13
12
19
11

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
8
Enter the key element:
55
No such item found in the list

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
8
Enter the key element:
13
Enter the item to be inserted at the right of the given item:
55

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
```

```
10:exit
enter the choice
8
Enter the key element:
13
Enter the item to be inserted at the right of the given item:
55

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
5
The contents of the list are:
13
55
12
19
11

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
1
Enter the item at front end:
11

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
```

```
1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
5
The contents of the list are:
11
13
55
12
19
11

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
9
Enter the key element whose duplicates should be removed:
11
Removed all the duplicate elements of the given item successfully

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
5
The contents of the list are:
11
```

```
enter the choice
9
Enter the key element whose duplicates should be removed:
11
Removed all the duplicate elements of the given item successfully

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
5
The contents of the list are:
11
13
55
12
19

1:dinsert front
2:dinsert rear
3:ddelete front
4:ddeleterear
5:ddisplay
6:dsearch
7:dinsert lestpos
8:dinsert rightpos
9:ddeleteduplicates
10:exit
enter the choice
10

Process returned 0 (0x0)   execution time : 269.365 s
Press any key to continue.
```