main.c

```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("mem full\n");
       exit(0);
    }
return x;
}
void freenode(NODE x)
{
   free(x);
}
NODE dinsert_front(int item,NODE head)
{
    NODE temp,cur;
    temp=getnode();
    temp->info=item;
    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;
```

main.c

```
NODE dinsert_rear(int item,NODE head)
{
    NODE temp,cur;
    temp=getnode();
    temp->info=item;
    cur=head->llink;
    head->llink=temp;
    temp->rlink=head;
    temp->llink=cur;
    cur->rlink=temp;
    return head;
}
NODE ddelete_front(NODE head)
{
    NODE cur,next;
    if(head->rlink==head)
    {
       printf("list empty\n");
       return head;
    }
    cur=head->rlink;
    next=cur->rlink;
    head->rlink=next;
    next->llink=head;
    printf("the node deleted is %d",cur->info);
    freenode(cur);
    return head;
}
NODE ddelete_rear(NODE head)
{
    NODE cur,prev;
    if(head->rlink==head)
    {
       printf("list empty\n");
       return head;
    }
    cur=head->llink;
    prev=cur->llink;
    head->llink=prev;
```

main.c

```
    cur=head->llink;
    prev=cur->llink;
    head->llink=prev;
    prev->rlink=head;
    printf("the node deleted is %d",cur->info);
    freenode(cur);
    return head;
}
NODE insert_leftpos(int item,NODE head)
{
    NODE temp,cur,prev;
    if(head->rlink==head)
    {
        printf("list empty\n");
        return head;
    }
    cur=head->rlink;
    while(cur!=head)
    {
        if(item==cur->info)break;
        cur=cur->rlink;
    }
    if(cur==head)
    {
        printf("key not found\n");
        return head;
    }
    prev=cur->llink;
    printf("enter towards left of %d=",item);
    temp=getnode();
    scanf("%d",&temp->info);
    prev->rlink=temp;
    temp->llink=prev;
    cur->llink=temp;
    temp->rlink=cur;
    return head;
}
NODE insert_rightpos(int item,NODE head)
{
```

main.c

```
}
NODE insert_rightpos(int item,NODE head)
{
    NODE temp,cur,prev;
    if(head->rlink==head)
    {
        printf("list empty\n");
        return head;
    }
    cur=head->rlink;
    while(cur!=head)
    {
        if(item==cur->info)break;
        cur=cur->rlink;
    }
    if(cur==head)
    {
        printf("key not found\n");
        return head;
    }
    prev=cur->rlink;
    printf("enter towards right of %d=",item);
    temp=getnode();
    scanf("%d",&temp->info);
    prev->llink=temp;
    temp->llink=cur;
    cur->rlink=temp;
    temp->rlink=prev;
    return head;
}
NODE delete_all_key(int item,NODE head)
{
    NODE prev,cur,next;
    int count;
    if(head->rlink==head)
    {
        printf("List Empty");
        return head;
```

main.c

```
145         printf("List Empty");
146         return head;
147     }
148     count=0;
149     cur=head->rlink;
150     while(cur!=head)
151     {
152         if(item!=cur->info)
153         cur=cur->rlink;
154         else
155         {
156             count++;
157             if(count==1)
158             {
159                 cur=cur->rlink;
160                 continue;
161             }
162             prev=cur->llink;
163             next=cur->rlink;
164             prev->rlink=next;
165             next->llink=prev;
166             freenode(cur);
167             cur=next;
168         }
169     }
170     if(count==0)
171     printf("key not found");
172     else
173     printf("key found at %d positions\n", count);
174     return head;
175 }
176
177 void search_info(int item,NODE head)
178 {
179     NODE cur;
180     if(head->rlink==head)
181     {
182         printf("list empty\n");
```

main.c

```
    if(head->rlink==head)
    {
        printf("list empty\n");
    }
    cur=head->rlink;
    while(cur!=head)
    {
        if(item==cur->info)
        {
            printf("Search Successfull\n");
            break;
        }
        cur=cur->rlink;
    }
    if(cur==head)
    {
        printf("Element not found\n");
    }
}
void display(NODE head)
{
    NODE temp;
    if(head->rlink==head)
    {
        printf("list empty\n");
        return;
    }
    for(temp=head->rlink;temp!=head;temp=temp->rlink)
    printf("%d\n",temp->info);
}
void main()
{
    int item,choice,key;
    NODE head,last;
    head=getnode();
    head->rlink=head;
    head->llink=head;
    for(;;)
    {
```

main.c

```
    head->llink=head;
    for(;;)
    {
        printf("1:Insert Front\n2:Insert Rear\n3:Delete Front\n4:Delete Rear\n5.Insert_left of node");
        printf("\n6.Insert_right of node\n7.Delete Duplicates\n8.Searh Item\n9.Display\n10.exit\n");
        printf("enter the choice:\n");
        scanf("%d",&choice);
        printf("----\n");
        switch(choice)
        {
            case 1: printf("enter the item at front end\n");
                    scanf("%d",&item);
                    last=dinsert_front(item,head);
                    break;
            case 2: printf("enter the item at rear end\n");
                    scanf("%d",&item);
                    last=dinsert_rear(item,head);
                    break;
            case 3:last=ddelete_front(head);
                   break;
            case 4:last=ddelete_rear(head);
                   break;
            case 5:printf("enter the key item\n");
                   scanf("%d",&item);
                   head=insert_leftpos(item,head);
                   break;
            case 6:printf("enter the key item\n");
                   scanf("%d",&item);
                   head=insert_rightpos(item,head);
                   break;
            case 7:printf("enter the key item\n");
                   scanf("%d",&item);
                   head=delete_all_key(item,head);
                   break;
            case 8:printf("enter the key item\n");
                   scanf("%d",&item);
                   search_info(item,head);
                   break;
            case 9:display(head);
```

main.c

```
221         printf("enter the choice:\n");
222         scanf("%d",&choice);
223         printf("----\n");
224         switch(choice)
225         {
226             case 1: printf("enter the item at front end\n");
227                     scanf("%d",&item);
228                     last=dinsert_front(item,head);
229                     break;
230             case 2: printf("enter the item at rear end\n");
231                     scanf("%d",&item);
232                     last=dinsert_rear(item,head);
233                     break;
234             case 3:last=ddelete_front(head);
235                    break;
236             case 4:last=ddelete_rear(head);
237                    break;
238             case 5:printf("enter the key item\n");
239                    scanf("%d",&item);
240                    head=insert_leftpos(item,head);
241                    break;
242             case 6:printf("enter the key item\n");
243                    scanf("%d",&item);
244                    head=insert_rightpos(item,head);
245                    break;
246             case 7:printf("enter the key item\n");
247                    scanf("%d",&item);
248                    head=delete_all_key(item,head);
249                    break;
250             case 8:printf("enter the key item\n");
251                    scanf("%d",&item);
252                    search_info(item,head);
253                    break;
254             case 9:display(head);
255                    break;
256             default:exit(0);
257         }
258     }
259 }
```

input

```
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5.Insert_left of node
6.Insert_right of node
7.Delete Duplicates
8.Searh Item
9.Display
10.exit
enter the choice:
1
----
enter the item at front end
45
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5.Insert_left of node
6.Insert_right of node
7.Delete Duplicates
8.Searh Item
9.Display
10.exit
enter the choice:
2
----
enter the item at rear end
67
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5.Insert_left of node
6.Insert_right of node
7.Delete Duplicates
```

```
enter the choice:
9
----
45
67
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5.Insert_left of node
6.Insert_right of node
7.Delete Duplicates
8.Searh Item
9.Display
10.exit
enter the choice:
5
----
enter the key item
45
enter towards left of 45=16
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5.Insert_left of node
6.Insert_right of node
7.Delete Duplicates
8.Searh Item
9.Display
10.exit
enter the choice:
6
----
enter the key item
67
enter towards right of 67=89
```

input

```
9.Display
10.exit
enter the choice:
9
----
16
45
67
89
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5.Insert_left of node
6.Insert_right of node
7.Delete Duplicates
8.Searh Item
9.Display
10.exit
enter the choice:
2
----
enter the item at rear end
45
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5.Insert_left of node
6.Insert_right of node
7.Delete Duplicates
8.Searh Item
9.Display
10.exit
enter the choice:
1
----
```

input

```
enter the choice:
1
----
enter the item at front end
45
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5.Insert_left of node
6.Insert_right of node
7.Delete Duplicates
8.Searh Item
9.Display
10.exit
enter the choice:
9
----
45
16
45
67
89
45
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5.Insert_left of node
6.Insert_right of node
7.Delete Duplicates
8.Searh Item
9.Display
10.exit
enter the choice:
7
----
```

```
10.exit
enter the choice:
7
----
enter the key item
45
key found at 3 positions
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5.Insert_left of node
6.Insert_right of node
7.Delete Duplicates
8.Searh Item
9.Display
10.exit
enter the choice:
9
----
45
16
67
89
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5.Insert_left of node
6.Insert_right of node
7.Delete Duplicates
8.Searh Item
9.Display
10.exit
enter the choice:
8
----
```

```
enter the choice:
8
----
enter the key item
67
Search Successfull
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5.Insert_left of node
6.Insert_right of node
7.Delete Duplicates
8.Searh Item
9.Display
10.exit
enter the choice:
8
----
enter the key item
99
Element not found
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5.Insert_left of node
6.Insert_right of node
7.Delete Duplicates
8.Searh Item
9.Display
10.exit
enter the choice:
10
----
```