Write a program to implement doubly linked list with primitive operations: a) Create a doubly linked list b) Insert nodes at both ends c) Delete nodes at both ends d) Insert a new node to the left of the specified node e) Insert a new node to the right of the specified node f) Delete all key elements g) Display the contents of the list

```
#include<stdio.h>
#include<process.h>
struct node
{
int info;
struct node *llink;
struct node *rlink;
};
typedef struct node *NODE;
NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL)
{
printf("Memory is 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;
 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 is empty.\n");
 return head;
 }
 cur=head->rlink;
 next=cur->rlink;
 head->rlink=next;
 next->llink=head;
 printf("Node deleted is %d",cur->info);
 freenode(cur);
 return head;
}
NODE ddelete_rear(NODE head)
{
 NODE cur,prev;
 if(head->rlink==head)
 {
```

```
 printf("List is empty.\n");
 return head;
 }
 cur=head->llink;
 prev=cur->llink;
 head->llink=prev;
 prev->rlink=head;
 printf("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 is 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)
{
 NODE temp,cur,prev;
 if(head->rlink==head)
 {
 printf("List is empty.\n");
 return head;
```

```
 }
 cur=head->llink;
 while(cur!=head)
 {
 if(item==cur->info)
 break;
 cur=cur->llink;
 }
 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->rlink=prev;
 cur->rlink=temp;
 temp->llink=cur;
 return head;
}
NODE delete_all_key(int item,NODE head)
```

```
{
 NODE prev,cur,next;
 int count;
 if(head->rlink==head)
 {
 printf("List is empty.");
 return head;
 }
 count=0;
 cur=head->rlink;
 while(cur!=head)
 {
 if(item!=cur->info)
 cur=cur->rlink;
 else
 {
 count++;
 prev=cur->llink;
 next=cur->rlink;
 prev->rlink=next;
 next->llink=prev;
 freenode(cur);
 cur=next;
 }
```

```
 }
 if(count==0)
 printf("Key not found.");
 else
 printf("Key found at %d positions and are deleted.\n", count);
 return head;
}
void display(NODE head)
{
 NODE temp;
 if(head->rlink==head)
 {
 printf("List is empty.\n");
 return;
 }
 printf("Contents of the list : \n");
 temp=head->rlink;
 while(temp!=head)
 {
 printf("%d ",temp->info);
 temp=temp->rlink;
 }
 printf("\n");
}
```

```
void main()
{
 NODE head,last;
 int item, choice;
 head=getnode();
 head->rlink=head;
 head->llink=head;
 for(;;)
 {
 printf("\n\n1:Insert front\n2:Insert rear\n3:Delete front\n4:Delete rear\n5:Insert left position\n6:Insert right position\n7:Delete all key elements\n8:Display\n9:Exit\n");
 printf("Enter the choice : ");
 scanf("%d",&choice);
 switch(choice)
 {
 case 1: printf("Enter the item to be inserted at front end : ");
 scanf("%d",&item);
 last=dinsert_front(item,head);
 break;
 case 2: printf("Enter the item to be inserted at rear end : ");
 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 : ");
scanf("%d",&item);
head=insert_leftpos(item,head);
break;
case 6: printf("Enter the key item : ");
scanf("%d",&item);
head=insert_rightpos(item,head);
break;
case 7: printf("Enter the key item : ");
scanf("%d",&item);
head=delete_all_key(item,head);
break;
case 8: display(head);
break;
default:exit(0);
}
}
}
```

```
1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 1
Enter the item to be inserted at front end : 1


1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 2
Enter the item to be inserted at rear end : 13


1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 1
Enter the item to be inserted at front end : 12


1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 8
Contents of the list :
12 1 13


1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 3
Node deleted is 12

1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
```

```
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 4
Node deleted is 13

1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 1
Enter the item to be inserted at front end : 13


1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 1
Enter the item to be inserted at front end : 14


1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 1
Enter the item to be inserted at front end : 15


1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 2
Enter the item to be inserted at rear end : 12


1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 8
Contents of the list :
15 14 13 1 12


1:Insert front
```

```
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 6
Enter the key item : 12
Enter towards right of 12 = 11


1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 5
Enter the key item : 15
Enter towards left of 15 = 16


1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 7
Enter the key item : 1
Key found at 1 positions and are deleted.


1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 8
Contents of the list :
16 15 14 13 12 11


1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 5
Enter the key item : 5
Key not found.


1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
```

```
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice :
6
Enter the key item : 6
Key not found.


1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 7
Enter the key item : 7
Key not found.

1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Insert left position
6:Insert right position
7:Delete all key elements
8:Display
9:Exit
Enter the choice : 9

--------------------------------
Process exited after 98.81 seconds with return value 0
Press any key to continue . . .
```