```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_rear(int item, NODE head){
    NODE temp,cur;
    temp=getnode();
    temp->info=item;
    cur=head->llink;
    temp->llink=cur;
    cur->rlink=temp;
    head->llink=temp;
    temp->rlink=head;
    return head;
}
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 ddelete_front(NODE head)
```

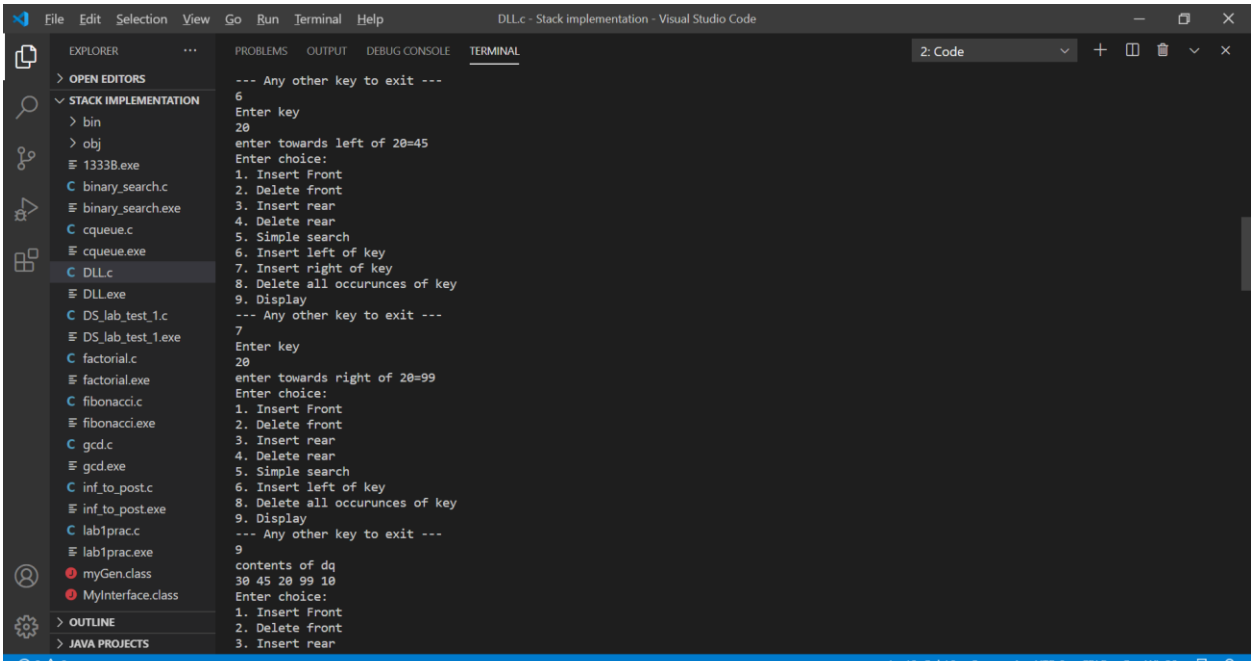
```
{
    NODE cur,next;
    if(head->rlink==head)
    {
        printf("dq empty\n");
        return head;
    }
    cur=head->rlink;
    next=cur->rlink;
    head->rlink=next;
    next->llink=head;
    printf("the item deleted is %d\n",cur->info);
    free(cur);
    return head;
}
NODE ddelete_rear(NODE head)
    {
    NODE cur,prev;
    if(head->rlink==head)
    {
        printf("dq empty\n");
        return head;
    }
    cur=head->llink;
    prev=cur->llink;
    head->llink=prev;
    prev->rlink=head;
    printf("the item deleted is %d\n",cur->info);
    free(cur);
    return head;
}
NODE lsearch(NODE head, int key, int z){
    NODE cur,prev,temp;
    int f=0,c=1;
    if(head->rlink==head)
    {
        printf("list empty\n");
        return head;
    }
    cur=head->rlink;
    while(cur!=head)
    {
        if(cur->info==key){
            f=1;
            break;
```

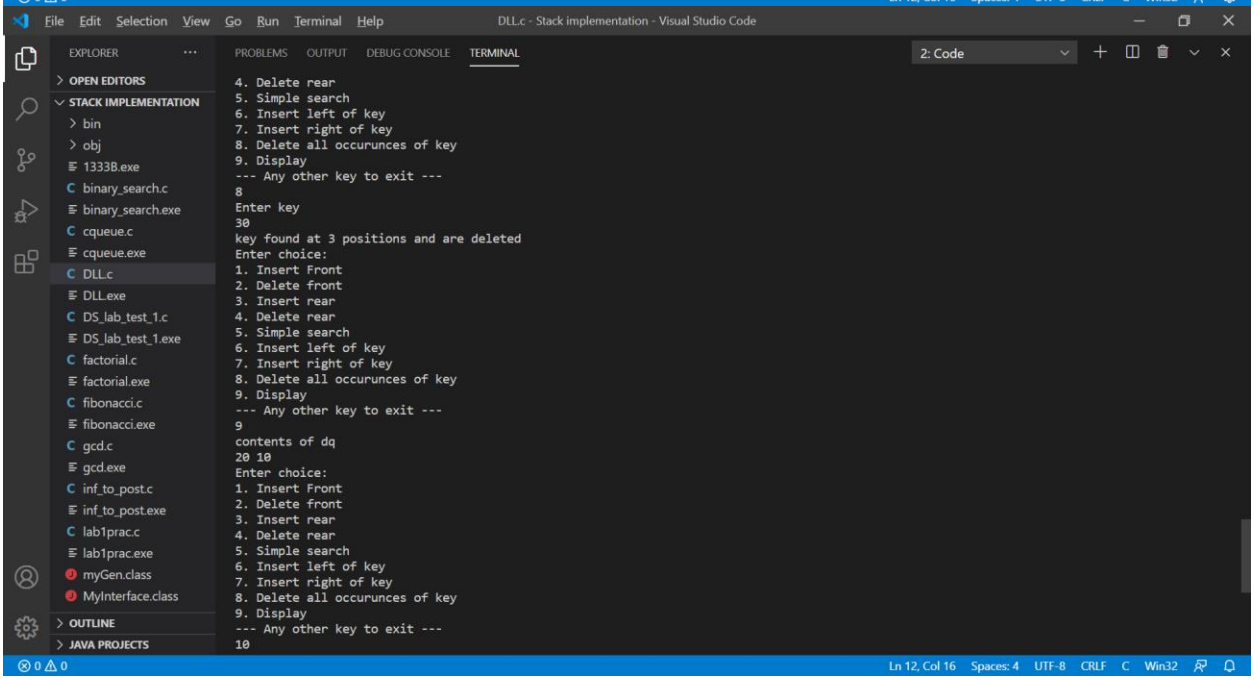
```
        }
        cur=cur->rlink;
        c++;
    }
    if(f==1 && z==0) {
    printf("Search successful, found at index %d\n",c);
    return head;
    }
    if(f==1 && z==1){
    prev=cur->llink;
    printf("enter towards left of %d=",key);
    temp=getnode();
    scanf("%d",&temp->info);
    prev->rlink=temp;
    temp->llink=prev;
    cur->llink=temp;
    temp->rlink=cur;
    return head;
    }
    if(f==1 && z==2){
    prev=cur;
    cur=cur->rlink;
    printf("enter towards right of %d=",key);
    temp=getnode();
    scanf("%d",&temp->info);
    prev->rlink=temp;
    temp->llink=prev;
    cur->llink=temp;
    temp->rlink=cur;
    return head;
    }
    printf("Search unsuccessful\n");
}
NODE delete_all_key(int item,NODE head)
{
NODE prev,cur,next;
int count;
   if(head->rlink==head)
    {
     printf("List Empty\n");
     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;
  free(cur);
  cur=next;
}
}
if(count==0)
  printf("key not found\n");
  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("dq empty\n");
        return;
    }
    printf("contents of dq\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("Enter choice:\n1. Insert Front\n2. Delete front\n3. Insert rear\n
4. Delete rear\n5. Simple search\n6. Insert left of key\n7. Insert right of key\n
8. Delete all occurunces of key\n9. Display\n--- Any other key to exit ---\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 3: printf("enter the item at rear end\n");
            scanf("%d",&item);
            head=dinsert_rear(item,head);
            break;
        case 2:
            head=ddelete_front(head);
            break;
        case 4:
            head=ddelete_rear(head);
            break;
        case 5:printf("Enter key\n");
        scanf("%d",&item);
        head=lsearch(head,item,0);
        break;
        case 6:printf("Enter key\n");
        scanf("%d",&item);
        head=lsearch(head,item,1);
        break;
        case 7:printf("Enter key\n");
        scanf("%d",&item);
        head=lsearch(head,item,2);
        break;
        case 8: printf("Enter key\n");
        scanf("%d",&item);
        head=delete_all_key(item,head);
        break;
        case 9: display(head);
            break;
        default:exit(0);

        }

    }

}
```

Output:


File Edit Selection View Go Run Terminal Help
DLL.c - Stack implementation - Visual Studio Code
EXPLORER
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
2: Code
OPEN EDITORS
STACK IMPLEMENTATION
bin
obj
1333B.exe
binary_search.c
binary_search.exe
cqueue.c
cqueue.exe
DLL.c
DLL.exe
DS_lab_test_1.c
DS_lab_test_1.exe
factorial.c
factorial.exe
fibonacci.c
fibonacci.exe
gcd.c
gcd.exe
inf_to_post.c
inf_to_post.exe
lab1prac.c
lab1prac.exe
myGen.class
MyInterface.class
OUTLINE
JAVA PROJECTS
--- Any other key to exit ---
6
Enter key
20
enter towards left of 20=45
Enter choice:
1. Insert Front
2. Delete front
3. Insert rear
4. Delete rear
5. Simple search
6. Insert left of key
7. Insert right of key
8. Delete all occurunces of key
9. Display
--- Any other key to exit ---
7
Enter key
20
enter towards right of 20=99
Enter choice:
1. Insert Front
2. Delete front
3. Insert rear
4. Delete rear
5. Simple search
6. Insert left of key
8. Delete all occurunces of key
9. Display
--- Any other key to exit ---
9
contents of dq
30 45 20 99 10
Enter choice:
1. Insert Front
2. Delete front
3. Insert rear
Ln 12, Col 16 Spaces: 4 UTF-8 CRLF C Win32


File Edit Selection View Go Run Terminal Help
DLL.c - Stack implementation - Visual Studio Code
EXPLORER
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
2: Code
OPEN EDITORS
STACK IMPLEMENTATION
bin
obj
1333B.exe
binary_search.c
binary_search.exe
cqueue.c
cqueue.exe
DLL.c
DLL.exe
DS_lab_test_1.c
DS_lab_test_1.exe
factorial.c
factorial.exe
fibonacci.c
fibonacci.exe
gcd.c
gcd.exe
inf_to_post.c
inf_to_post.exe
lab1prac.c
lab1prac.exe
myGen.class
MyInterface.class
OUTLINE
JAVA PROJECTS
4. Delete rear
5. Simple search
6. Insert left of key
7. Insert right of key
8. Delete all occurunces of key
9. Display
--- Any other key to exit ---
8
Enter key
30
key found at 3 positions and are deleted
Enter choice:
1. Insert Front
2. Delete front
3. Insert rear
4. Delete rear
5. Simple search
6. Insert left of key
7. Insert right of key
8. Delete all occurunces of key
9. Display
--- Any other key to exit ---
9
contents of dq
20 10
Enter choice:
1. Insert Front
2. Delete front
3. Insert rear
4. Delete rear
5. Simple search
6. Insert left of key
7. Insert right of key
8. Delete all occurunces of key
9. Display
--- Any other key to exit ---
10
Ln 12, Col 16 Spaces: 4 UTF-8 CRLF C Win32