Linear Queue

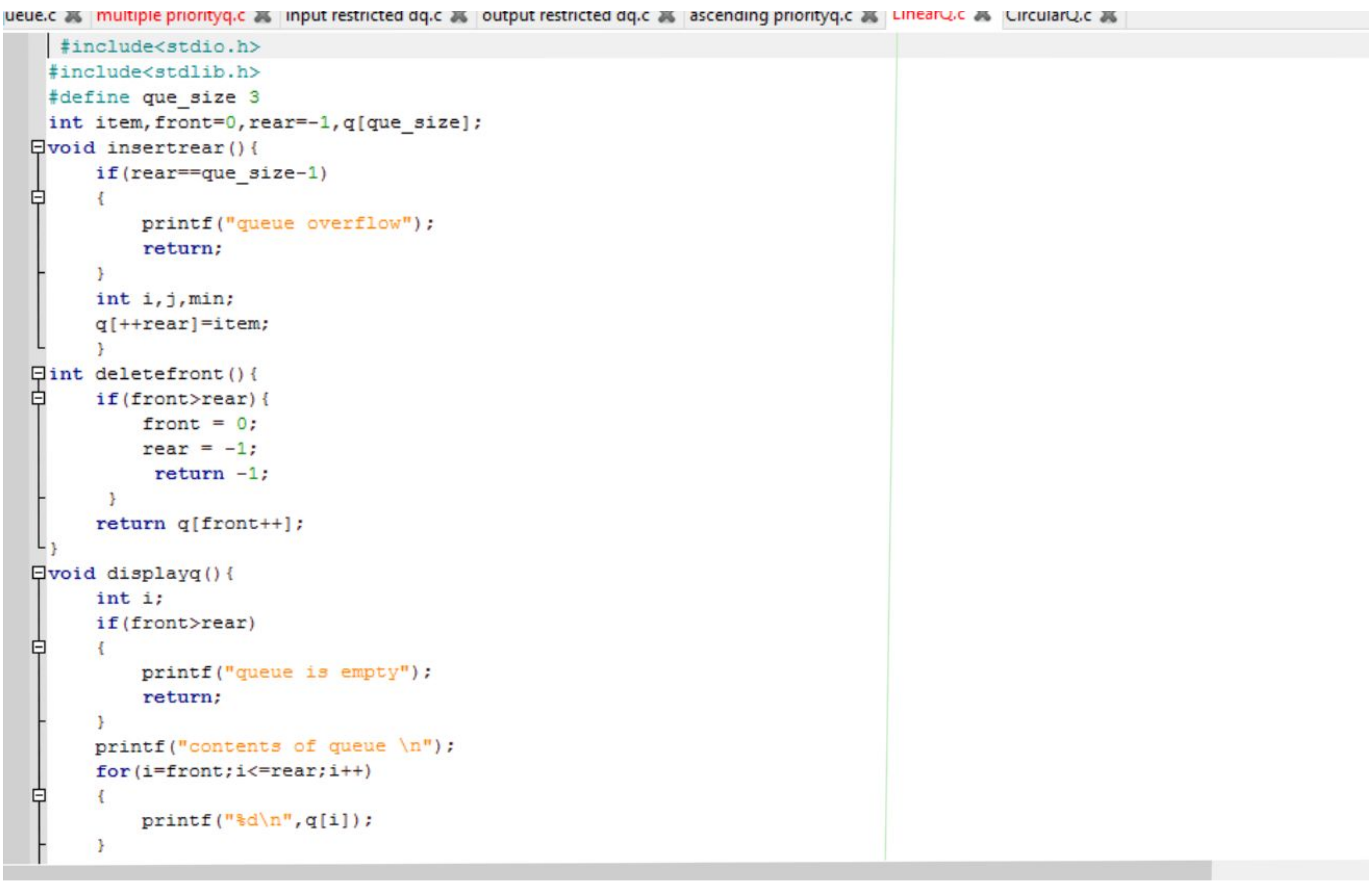
```
# include<stdio.h>
# include<stdlib.h>
# define que-size 3
int item, front=0, rear=-1, q[que-size];
void insertrear () {
 if (rear == que-size-1) {
  printf ("Queue Overflow \n");
   return; }
   rear+=1;
  q[rear] = item;
}
int deletefront () {
  if (front > rear) {
   front =0;
   rear = -1;
   return -1;
 }
  returnq[front++];
}
void displayQ() {
  int i;
  if (front > rear)
 {   printf ("Queue is empty \n");
             return;
  }
printf("Contents of queue \n");
  for (i = front; i <= rear; i++) {
   printf ("%d\n", q[i]);
   }
  }
```

```
int main () {
 int choice; item_deleted ;
 for (; ;) {
 printf ("\n 1. insert rear \n 2. delete front \n 3. display \n 4. exit\n");
 printf(" Enter the choice \n");
 scanf ("%d" &choice);
  switch (choice) {
  case 1:  printf (" Enter the item to be inserted. \n");
           scanf ("%d", & item);
           insert rear() ;
           break;
  case 2:  item deleted = deletefront () ;
             if (item deleted == -1)
               printf ("Queue Underflow \n");
             else.
             printf ("deleted item is %d", item _deleted");
               break;

   case 3:  displayQ ();
              break;

   case 4:   exit (0); }

   }
}
```

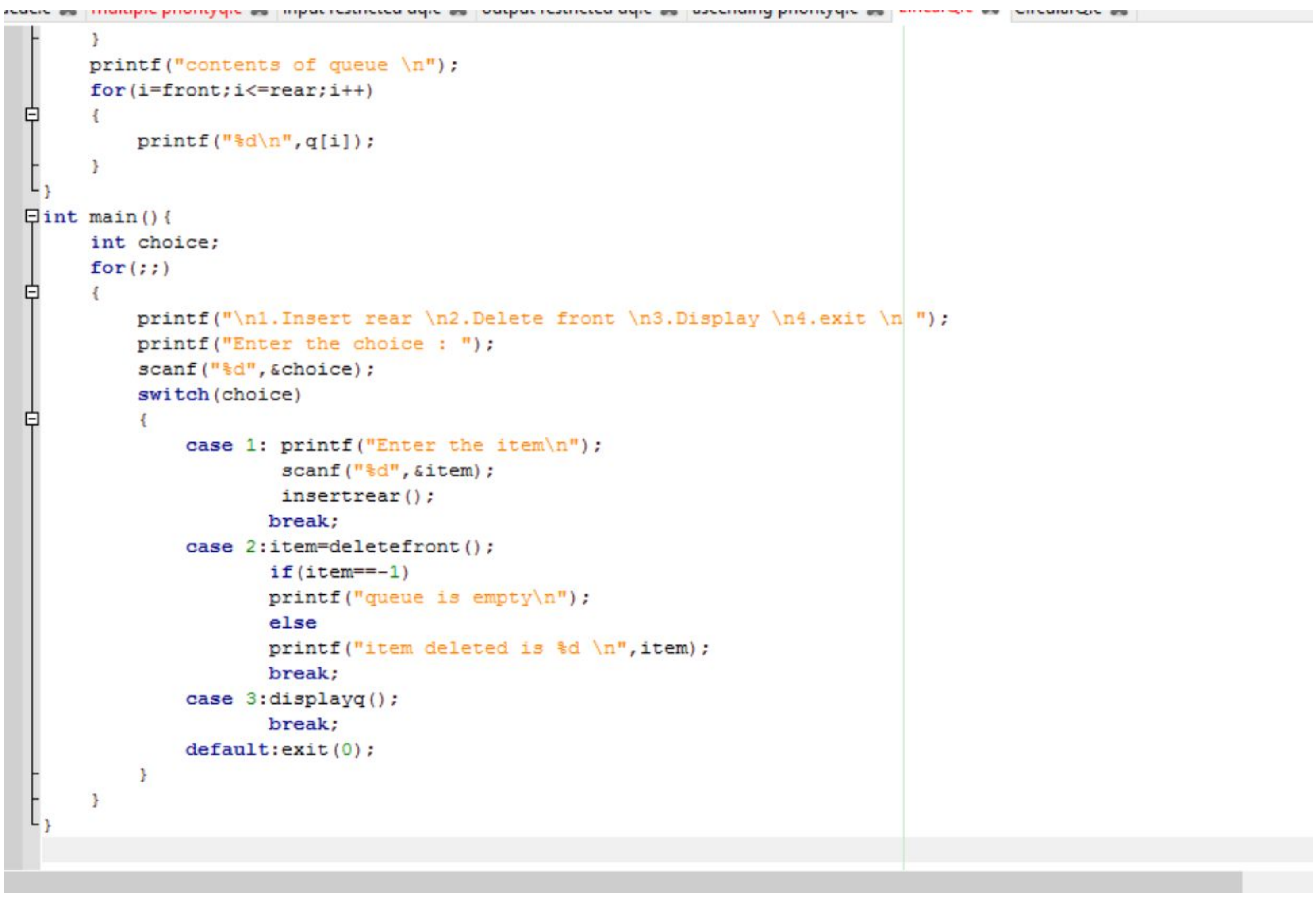
Circular Queue

```
# include <stdio.h>
# include <std lib.h>
# define que-size 3
int item, front=0, rear=-1, q[que-size], count=0;

void insertrear () {
    if (count = = que-size) {
    printf ("Queue Overflow\n");
            return; }
    rear = (rear+1) % que_size;
    q[rear] = item;
    count++;
}
int   deletefront () {
   if (count = =0)
        return -1;
   front = (front +1) % que-size;
     count --;
     return item;
}
void display () {
  int i,f;
  if (count==0) {
     printf ("Queue is empty\n");
       return;
  }
  f= front;
  printf (" Contents of Queue are \n");
  for (i=1; i<= count; i++) {
     printf ("%d \n", q[i]);
         f= (f+1) % que-size;
    }
}
```

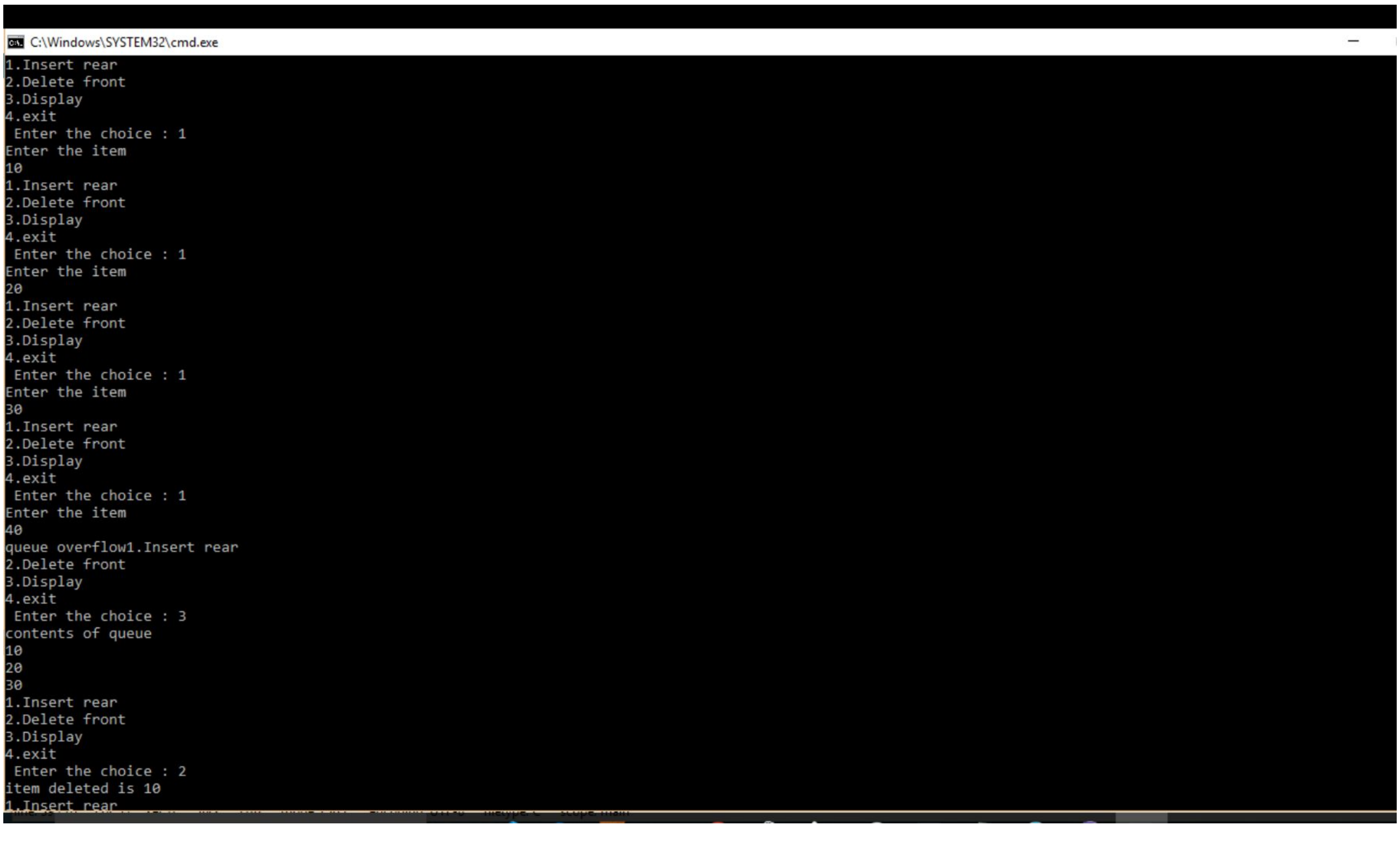
```
int  main () {
int  choice ;
for (; ; )
{
  printf ("\n1. insertrear \n 2. Deletefront \n 3. Display \n 4. exit \n");
  printf (" Enter the choice \n");
  scanf (" %d ", & choice);
  switch (choice) {
  case 1:  printf (" Enter the item to be inserted \n");
           scanf (" %d ", & ~~cho~~ item);
           insertrear ();
           break ;
  case 2:  item = deletefront () ;
           if (item == -1)
           printf (" Queue is empty \n");
           else
           printf (" item deleted is %d \n", item);
           break;

 case 3 :  display () ;
           break ;

 default : exit (0) ;

 }
}
}
```



```c
#include<stdio.h>
#include<stdlib.h>
#define que_size 3
int item,front=0,rear=-1,q[que_size];
void insertrear(){
    if(rear==que_size-1)
    {
        printf("queue overflow");
        return;
    }
    int i,j,min;
    q[++rear]=item;
    }
int deletefront(){
    if(front>rear){
        front = 0;
        rear = -1;
         return -1;
     }
    return q[front++];
}
void displayq(){
    int i;
    if(front>rear)
    {
        printf("queue is empty");
        return;
    }
    printf("contents of queue \n");
    for(i=front;i<=rear;i++)
    {
        printf("%d\n",q[i]);
    }
```

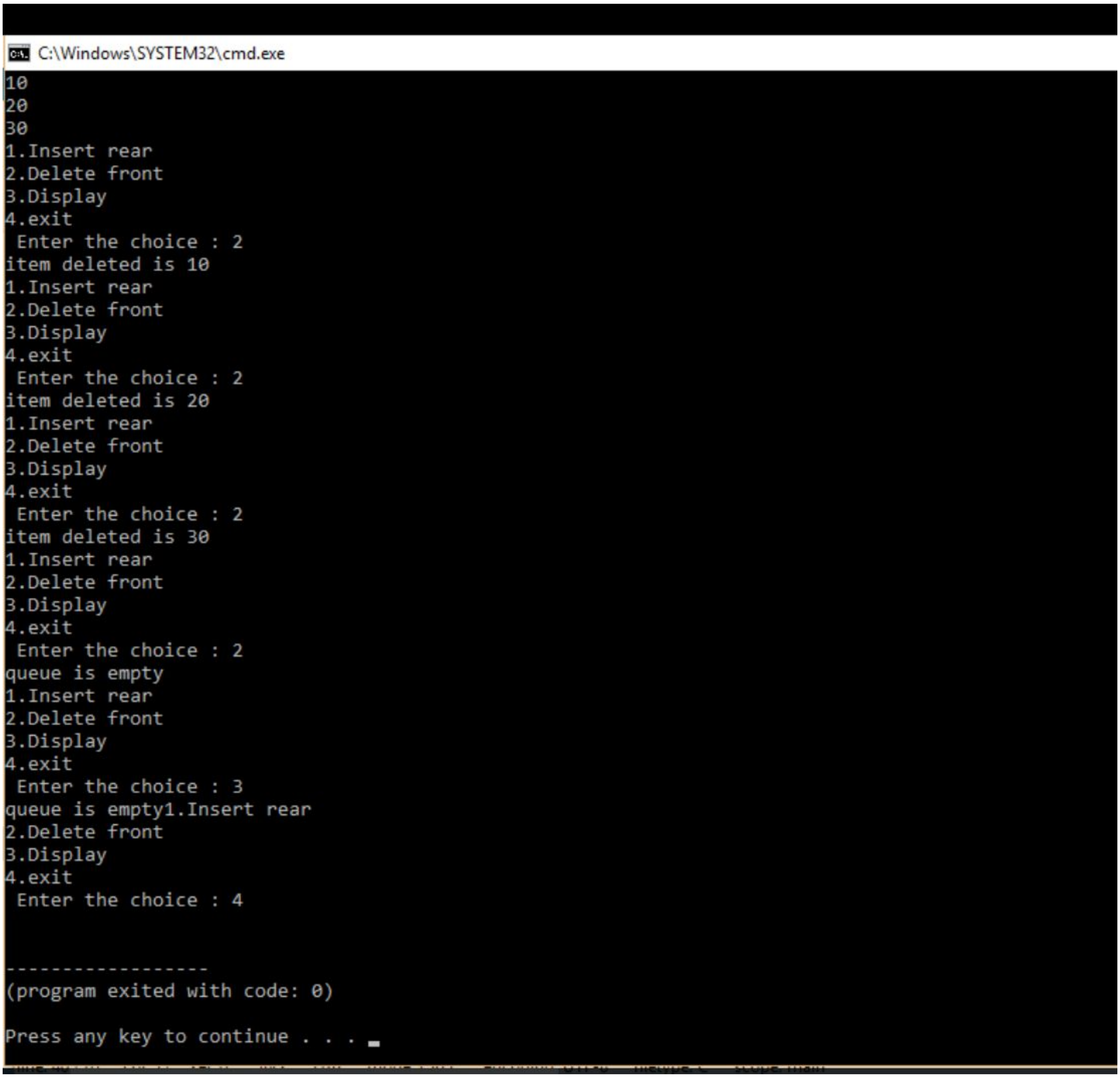


```
    }
    printf("contents of queue \n");
    for(i=front;i<=rear;i++)
    {
        printf("%d\n",q[i]);
    }
}
int main(){
    int choice;
    for(;;)
    {
        printf("\n1.Insert rear \n2.Delete front \n3.Display \n4.exit \n ");
        printf("Enter the choice : ");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: printf("Enter the item\n");
                     scanf("%d",&item);
                     insertrear();
                    break;
            case 2:item=deletefront();
                    if(item==-1)
                    printf("queue is empty\n");
                    else
                    printf("item deleted is %d \n",item);
                    break;
            case 3:displayq();
                    break;
            default:exit(0);
        }
    }
}
```

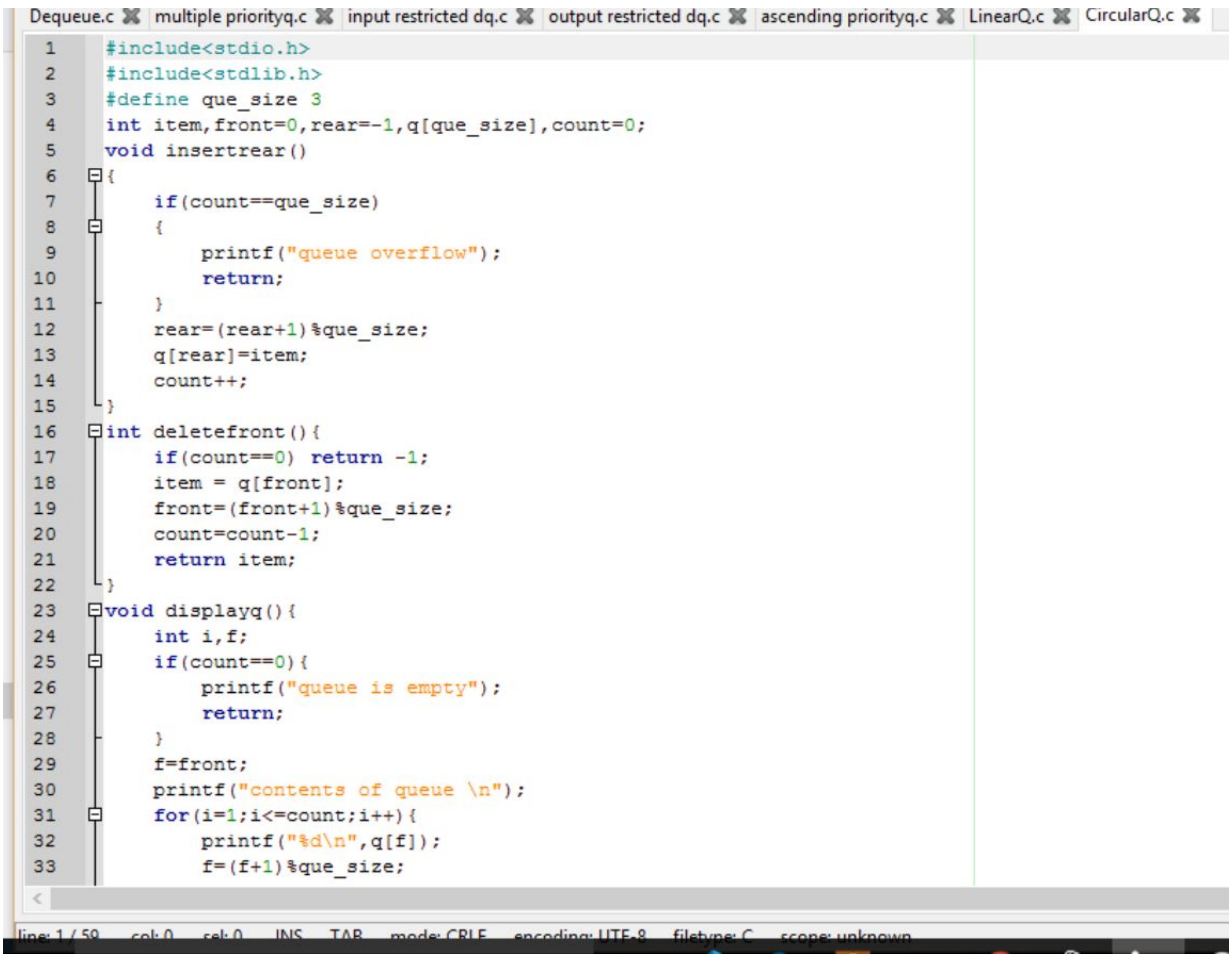


C:\Windows\SYSTEM32\cmd.exe

```
1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 1
Enter the item
10
1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 1
Enter the item
20
1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 1
Enter the item
30
1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 1
Enter the item
40
queue overflow1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 3
contents of queue
10
20
30
1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 2
item deleted is 10
1.Insert rear
```

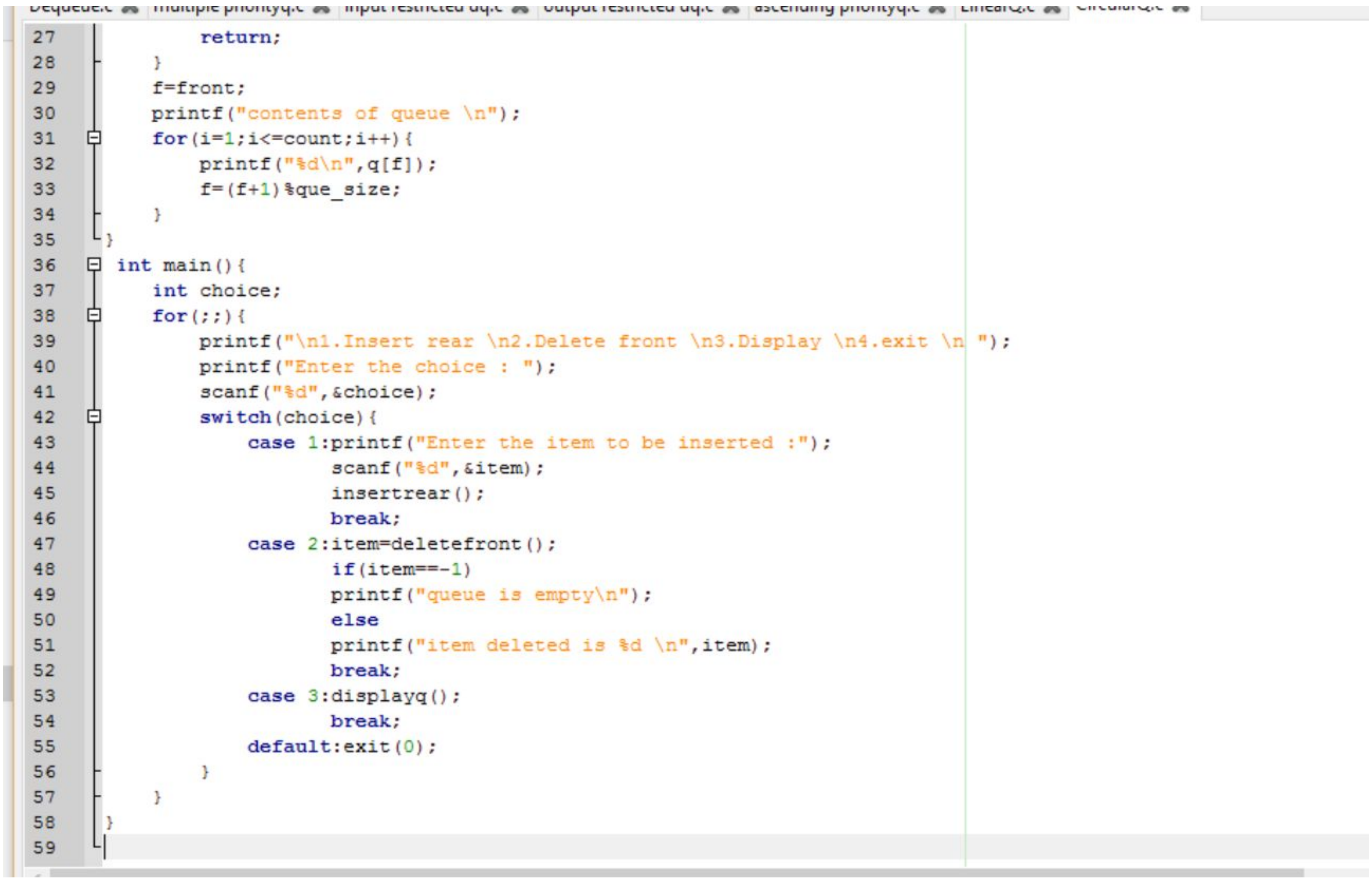


```
10
20
30
1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 2
item deleted is 10
1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 2
item deleted is 20
1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 2
item deleted is 30
1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 2
queue is empty
1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 3
queue is empty1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 4


------------------
(program exited with code: 0)

Press any key to continue . . .
```

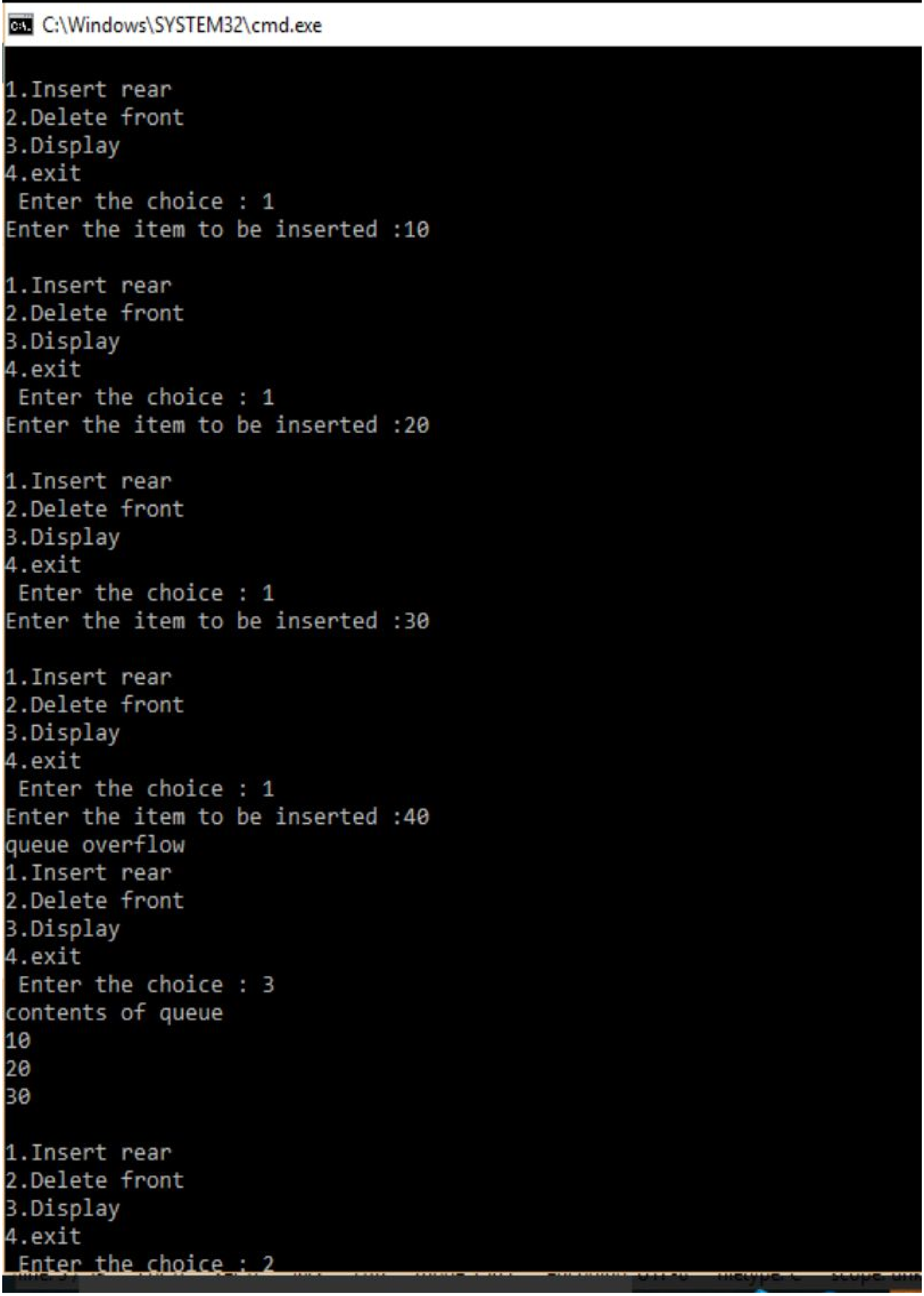


```c
#include<stdio.h>
#include<stdlib.h>
#define que_size 3
int item,front=0,rear=-1,q[que_size],count=0;
void insertrear()
{
    if(count==que_size)
    {
        printf("queue overflow");
        return;
    }
    rear=(rear+1)%que_size;
    q[rear]=item;
    count++;
}
int deletefront(){
    if(count==0)  return -1;
    item = q[front];
    front=(front+1)%que_size;
    count=count-1;
    return item;
}
void displayq(){
    int i,f;
    if(count==0){
        printf("queue is empty");
        return;
    }
    f=front;
    printf("contents of queue \n");
    for(i=1;i<=count;i++){
        printf("%d\n",q[f]);
        f=(f+1)%que_size;
```

line: 1 / 59   col: 0   sel: 0   INS   TAB   mode: CRLF   encoding: UTF-8   filetype: C   scope: unknown

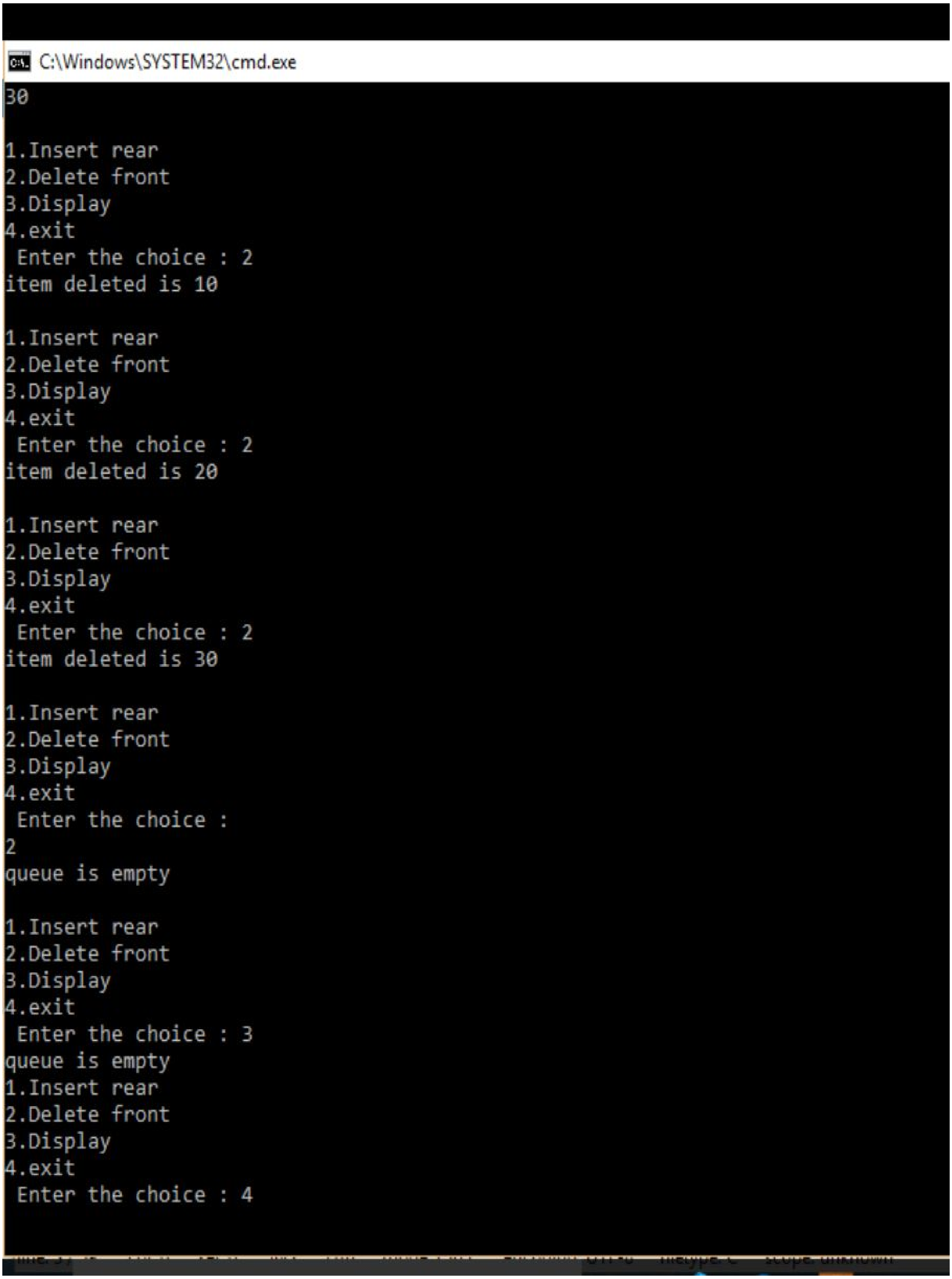


```c
            return;
        }
        f=front;
        printf("contents of queue \n");
        for(i=1;i<=count;i++){
            printf("%d\n",q[f]);
            f=(f+1)%que_size;
        }
}
int main(){
    int choice;
    for(;;){
        printf("\n1.Insert rear \n2.Delete front \n3.Display \n4.exit \n ");
        printf("Enter the choice : ");
        scanf("%d",&choice);
        switch(choice){
            case 1:printf("Enter the item to be inserted :");
                    scanf("%d",&item);
                    insertrear();
                    break;
            case 2:item=deletefront();
                    if(item==-1)
                    printf("queue is empty\n");
                    else
                    printf("item deleted is %d \n",item);
                    break;
            case 3:displayq();
                    break;
            default:exit(0);
        }
    }
}
```



C:\Windows\SYSTEM32\cmd.exe

```

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 1
Enter the item to be inserted :10

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 1
Enter the item to be inserted :20

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 1
Enter the item to be inserted :30

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 1
Enter the item to be inserted :40
queue overflow
1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 3
contents of queue
10
20
30

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 2
```



```
30

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 2
item deleted is 10

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 2
item deleted is 20

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 2
item deleted is 30

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice :
2
queue is empty

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 3
queue is empty
1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 4
```