

Linear Queue

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

void enqueue() {
    if (rear == que-size-1) {
        printf("Queue Overflow\n");
        return;
    }
    rear++;
    q[rear] = item;
}

int deletefront() {
    if (front > rear) {
        front = 0;
        rear = -1;
        return -1;
    }
    return q[front++];
}

void display() {
    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 ("In 1. insert rear In 2. delete front In 3. display In 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: Display();
                    break;
            case 4: exit (0);
        }
    }
}

```


Circular Queue

```
#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\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[f]);
        f = (f + 1) % que-size;
    }
}
```



```

int main() {
    int choice;
    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", &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);
        }
    }
}

```

queue.c multiple priorityq.c input restricted dq.c output restricted dq.c ascending priorityq.c LinearQ.c CircularQ.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
```

C:\Windows\SYSTEM32\cmd.exe

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 . . .

Deque.c X multiple priorityq.c X input restricted dq.c X output restricted dq.c X ascending priorityq.c X LinearQ.c X CircularQ.c X

```
1  #include<stdio.h>
2  #include<stdlib.h>
3  #define que_size 3
4  int item,front=0,rear=-1,q[que_size],count=0;
5  void insertrear()
6  {
7      if(count==que_size)
8      {
9          printf("queue overflow");
10         return;
11     }
12     rear=(rear+1)%que_size;
13     q[rear]=item;
14     count++;
15 }
16 int deletefront(){
17     if(count==0) return -1;
18     item = q[front];
19     front=(front+1)%que_size;
20     count=count-1;
21     return item;
22 }
23 void displayq(){
24     int i,f;
25     if(count==0){
26         printf("queue is empty");
27         return;
28     }
29     f=front;
30     printf("contents of queue \n");
31     for(i=1;i<=count;i++){
32         printf("%d\n",q[f]);
33         f=(f+1)%que_size;
```

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

```

27         return;
28     }
29     f=front;
30     printf("contents of queue \n");
31     for(i=1;i<=count;i++){
32         printf("%d\n",q[f]);
33         f=(f+1)%que_size;
34     }
35 }
36 int main(){
37     int choice;
38     for(;;){
39         printf("\n1.Insert rear \n2.Delete front \n3.Display \n4.exit \n ");
40         printf("Enter the choice : ");
41         scanf("%d",&choice);
42         switch(choice){
43             case 1:printf("Enter the item to be inserted :");
44                     scanf("%d",&item);
45                     insertrear();
46                     break;
47             case 2:item=deletefront();
48                     if(item==-1)
49                         printf("queue is empty\n");
50                     else
51                         printf("item deleted is %d \n",item);
52                     break;
53             case 3:displayq();
54                     break;
55             default:exit(0);
56         }
57     }
58 }
59

```

```
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
```


C:\Windows\SYSTEM32\cmd.exe

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