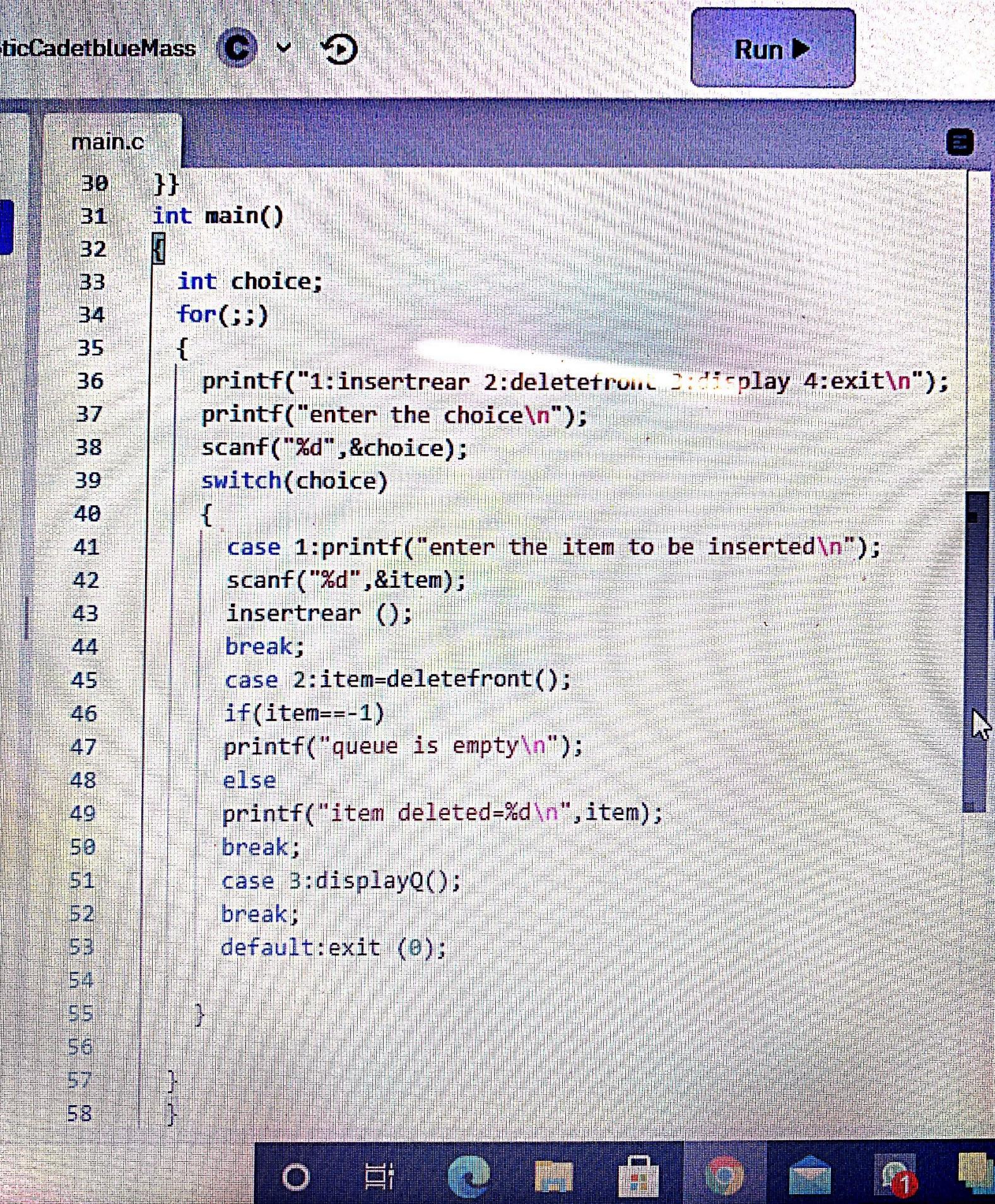


Ordinary Queue

```
#include <stdio.h>
#include <stdlib.h>
#define QUE_SIZE 3
int item, front = 0, rear = -1, q[10];
void insertrear()
{ if (rear == QUE_SIZE - 1)
{
    printf("queue overflow \n");
    return;
}
rear = rear + 1
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 ;
for(;;)
{
    printf("1: insertrear 2: deletefront 3. display 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);
        insertrear();
        break;
        case 2: item = deletefront();
        if(item == -1)
            printf("Queue is empty \n");
        else
            printf("Item deleted = %d \n", item);
        break;
        case 3: display();
        break;
        default: exit(0);
    }
}
```

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



```
• clang-7 -pthread -lm -o main main.c
• ./main
1:insertrear 2:deletefront 3:display 4:exit
enter the choice
1
enter the item to be inserted
32
1:insertrear 2:deletefront 3:display 4:exit
enter the choice
1
enter the item to be inserted
34
1:insertrear 2:deletefront 3:display 4:exit
enter the choice
3
contents of queue
32
34
1:insertrear 2:deletefront 3:display 4:exit
enter the choice
2
item deleted=32
1:insertrear 2:deletefront 3:display 4:exit
enter the choice
2
item deleted=34
1:insertrear 2:deletefront 3:display 4:exit
enter the choice
2
queue is empty
1:insertrear 2:deletefront 3:display 4:exit
enter the choice
• []
```

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");
    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=0; i<=count; i++)
{
    printf("%d\n", q[i]);
    f=f+1;
}
qne_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);
        }
    }
}

```

main.c

```
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 {
18     if(count==0) return -1;
19     item = q[front];
20     front=(front+1)%que_size;
21     count=count-1;
22     return item;
23 }
24 void displayq()
25 {
26     int i,f;
27     if(count==0)
28     {
29         printf("queue is empty");
30     }
31 }
```

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

```
c
|   |   |   |   |   printf("queue is empty\n");
|   |   |   |   | else
|   |   |   |   |   printf("item deleted is %d \n",item);
|   |   |   |   |   break;
|   |   |   | case 3:displayq();
|   |   |   |   | break;
|   |   | default:exit(0);
|   |
|   | }
| }
| }
```

```
$ clang-7 -pthread -lm -o main main.c  
$ ./main
```

```
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 : 1
```

```
Enter the item to be inserted :21
```

```
1.Insert rear  
2.Delete front  
3.Display  
4.exit
```

```
Enter the choice : 3
```

```
contents of queue
```

```
21
```

```
0
```

```
1.Insert rear  
2.Delete front  
3.Display  
4.exit
```

```
Enter the choice :
```