EXPERIMENT NO. 4

ARRAY REPRESENTATION OF QUEUE

Aim: Write A Program To Implement Queue Using Array.

Theory: Queue is a particular kind of ***Abstract Data Type*** or ***Collection*** in which the entities in the collection are kept in order and the principal operations on the collection are the addition of entities to the ***rear*** terminal position, known as ***Enqueues***, and removal of entities from the ***front*** terminal position, known as **Dequeues.** This makes the queue a ***First-In-First-Out (FIFO) Data Structure.*** In a FIFO data structure, ***the first element added to the queue will be the first one to be removed.*** A queue is an example of a ***Linear Data Structure***, or more abstractly a sequential collection.

Queues are common in computer programs, where they are implemented as data structures coupled with access routines, as an ***Abstract Data Structure*** or in object-oriented languages as Classes. Common implementations are ***Circular Buffers*** and ***Linked Lists.***

Some Of The Types Of Queue Are:

- Simple or Linear queue
- Circular Queue
- Priority Queue
- DE queue (Double Ended Queue)

OPERATIONSONQUEUE

Enqueue

This operation is used to add an item to the queue at the rear end .This operation will be performed at the rear end of the queue.

Dequeue

This operation is used to remove an item from the queue at the front end. This operation will be performed at the front end of the queue.

Display

It Is Used To Display The Elements Of A Queue.

Algorithm:

Enqueue Operation:

```
1. If (REAR = = (Size-1)) Then [Check For Overflow]
2. Print: Queue Overflow
3. Else
4. If (FRONT And REAR = = -1) Then [Check if QUEUE is empty]
(a) Set FRONT = 0
(b) Set REAR = 0
5.      Else
6.      Set REAR = REAR + 1 [Increment REAR by 1] [End of Step 4 If]
7.      QUEUE [REAR] = ITEM
8.      Print: ITEM inserted [End of Step 1 If] 9. Exit.
```

Dequeue Operation:

```
1. If (FRONT = = -1) Then [Check for Underflow]
2. Print: Queue Underflow
3. Else
4. ITEM = QUEUE [FRONT]
5. If (FRONT = = REAR) Then [Check if Only One Element is left]
(a) Set FRONT = -1
(b) Set REAR = -1
6. Else
7. Set FRONT = FRONT + 1 [Increment FRONT by 1] [End of Step 5 If]
8. Print: ITEM Deleted [End of Step 1 If]
9. Exit
```

Display Operation:

```
1. If (FRONT = = -1) Then [Check For Queue Empty]
```

```
2. Print: Queue Empty
3. Else
4. Initialize Integer i
5. For (i=Front; i<=Rear; i++) [Repeat Step 6]
6. Print: Element [i].
7. Exit.
```

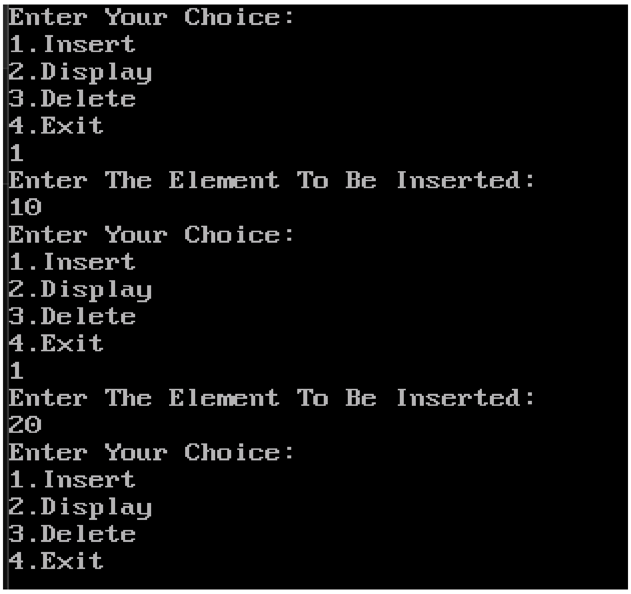
Program:

```
#include<stdio.h>
#define size 5
int empty();
int full();
void insert(int x);
void delet();
void display();
int queue[size];
int f=-1,r=-1;
void main()
{
int x,c;
do
{
printf("Enter Your Choice:\n");
printf("1.Insert\n2.Display\n3.Delete\n4.Exit\n");
scanf("%d",&c);
switch(c)
{
case 1:printf("Enter The Element To Be Inserted:\n");
scanf("%d",&x);
insert(x);
break;
case 2:display();
break;
case 3:
delet();
break;
}
}
while(c!=4);
return 0;
}
int empty()
```

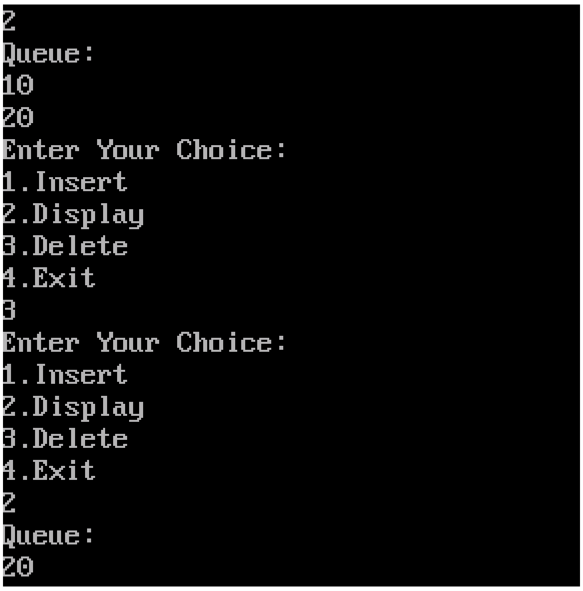
```
{
if(f==-1)
return 1;
else
return 0;
}
int full()
{
if(r==(size-1))
return 1;
else
return 0;
}
void display()
{
int i;
if(empty()==1)
printf("Queue Empty\n");
else
{
printf("Queue:\n");
for(i=f;i<=r;i++)
{
printf("%d\n",queue[i]);
}
}
}
void delet()
{
if(empty()==1)
printf("Queue Empty\n");
else if(f==r)
{
f=-1;
r=-1;
}
else
{
f=f+1;
}
}
void insert(int x)
{
if(full()==1)
printf("Queue Overflow\n");
else if(f==-1)
{
```

```
f=f+1;
r=r+1;
queue[r]=x;
}
else
{
r=r+1;
queue[r]=x;
}
}
```

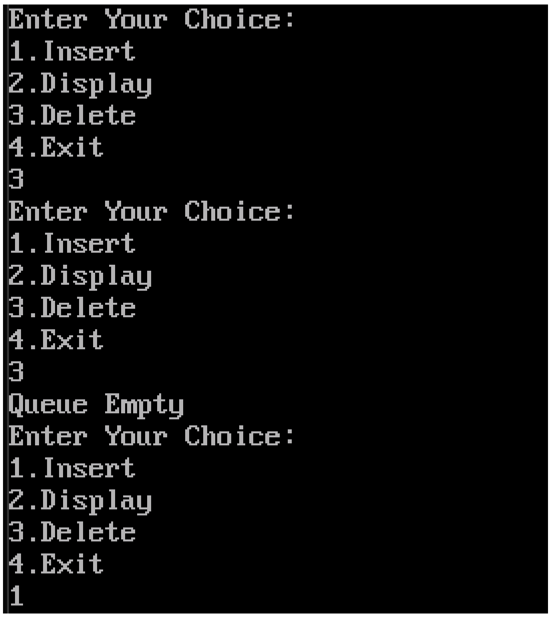
Output:



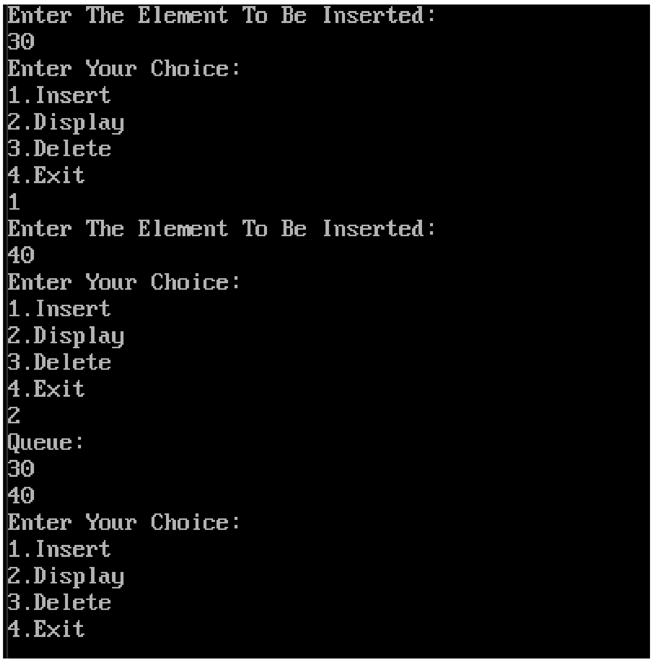
```
Enter Your Choice:
1.Insert
2.Display
3.Delete
4.Exit
1
Enter The Element To Be Inserted:
10
Enter Your Choice:
1.Insert
2.Display
3.Delete
4.Exit
1
Enter The Element To Be Inserted:
20
Enter Your Choice:
1.Insert
2.Display
3.Delete
4.Exit
```



```
2
Queue:
10
20
Enter Your Choice:
1.Insert
2.Display
3.Delete
4.Exit
3
Enter Your Choice:
1.Insert
2.Display
3.Delete
4.Exit
2
Queue:
20
```



```
Enter Your Choice:
1.Insert
2.Display
3.Delete
4.Exit
3
Enter Your Choice:
1.Insert
2.Display
3.Delete
4.Exit
3
Queue Empty
Enter Your Choice:
1.Insert
2.Display
3.Delete
4.Exit
1
```



```
Enter The Element To Be Inserted:
30
Enter Your Choice:
1.Insert
2.Display
3.Delete
4.Exit
1
Enter The Element To Be Inserted:
40
Enter Your Choice:
1.Insert
2.Display
3.Delete
4.Exit
2
Queue:
30
40
Enter Your Choice:
1.Insert
2.Display
3.Delete
4.Exit
```