Experiment-2.2

A menu driven Program for the operations on Circular QUEUE of Characters (Array Implementation of Queue with maximum size MAX)

Student Name: Neha Sharma
UID: 20BCS4576
Branch: 20IOT1
Section/Group: A
Semester: 3rd
Date of Performance: 16/10/21
Subject Name: DATA STRUCTURES LAB
Subject Code: 21O-20CSP-236_20BIT-1

1. Aim/Overview of the practical:

A menu driven Program for operations on Circular Queue.

2. Task to be done:

We have to do different operation on Circular Queue.

3. Algorithm/Flowchart:

Step 1: Start.
Step 2: Initialize queue size to MAX.
Step 3: Insert the elements into circular queue. If queue is full give a message as ‘queue is overflow”
Step 4: Delete an element from the circular queue. If queue is empty give a message as ‘queue is underflow’.
Step 5: Display the contents of the queue.
Step 6: Press 4 to exit from code.
Step 7: Stop.

4. Code for experiment/practical:

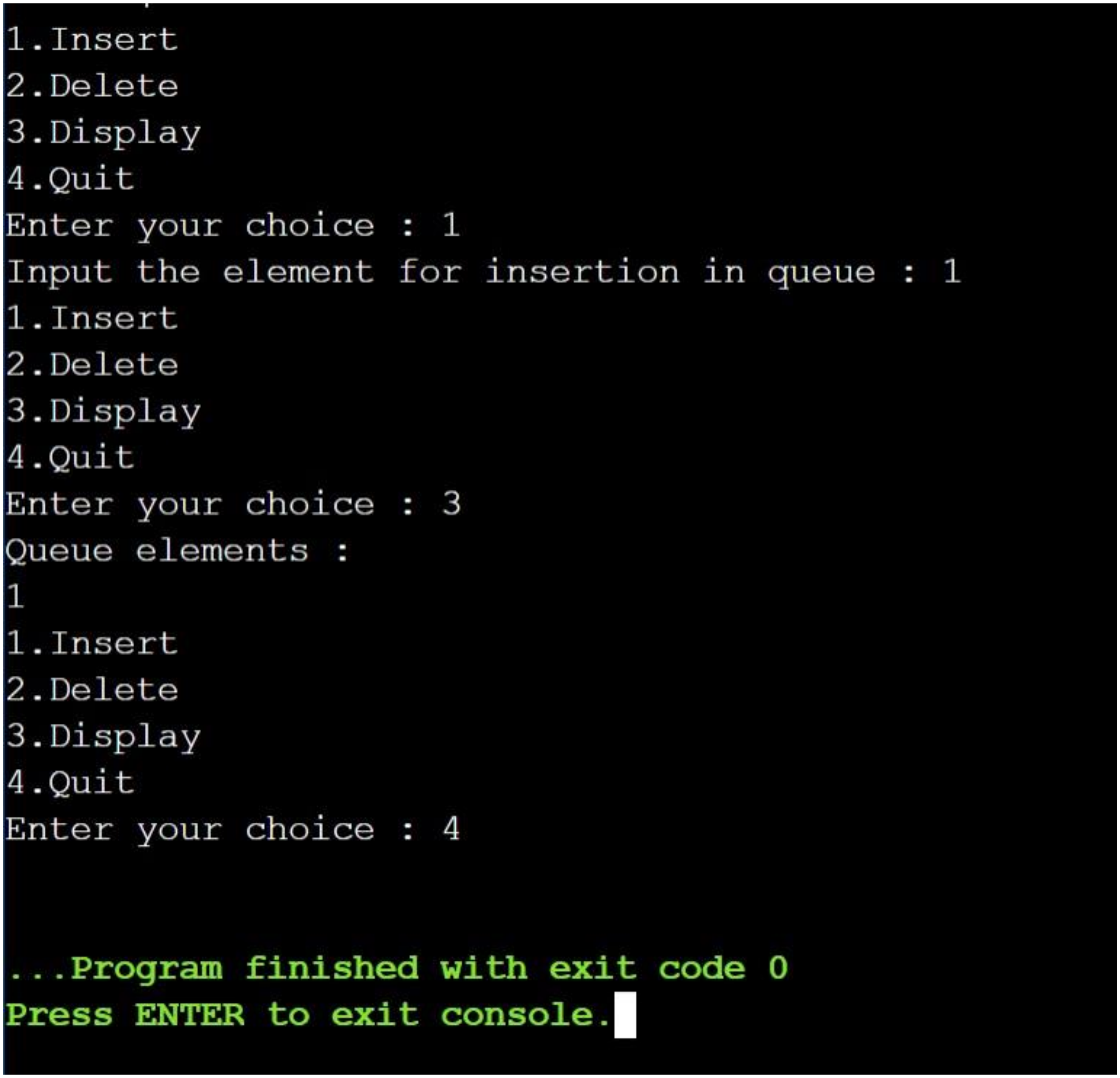
```
#include<stdlib.h>
#include<conio.h>
# include<stdio.h>
# define MAX 5
int cqueue_arr[MAX]; int
front = -1;
```

```
int rear = -1;
void insert(int item)
{
if((front == 0 && rear == MAX-1) || (front == rear+1))
{
printf("Queue Overflow \n");
return;
}
if (front == -1)
{
front = 0;
rear = 0;
}
else
{
if(rear == MAX-1) /*rear is at last position of queue */ rear = 0;
else
rear = rear+1;
}
cqueue_arr[rear] = item ;
}
void del()
{
if (front == -1)
{
printf("Queue Underflow\n"); return ;
}
printf("Element deleted from queue is : %d\
n",cqueue_arr[front]);
if(front == rear) /* queue has only one element */
{
front = -1;
```

```
rear=-1;
}
else
{
if(front == MAX-1) front =
0;
else
front = front+1;
}
}
void display()
{
int front_pos = front,rear_pos = rear; if(front == -1)
{
printf("Queue is empty\n"); return;
}
printf("Queue elements :\n"); if( front_pos
<= rear_pos ) while(front_pos <= rear_pos)
{
printf("%d ",cqueue_arr[front_pos]); front_pos++;
}
else
{
while(front_pos <= MAX-1)
{
printf("%d ",cqueue_arr[front_pos]); front_pos++;
}
front_pos = 0; while(front_pos <=
rear_pos)
{
printf("%d ",cqueue_arr[front_pos]); front_pos++;
}
```

```
}
printf("\n");
}
int main()
{
int choice,item; do
{
printf("1.Insert\n"); printf("2.Delete\n");
printf("3.Display\n"); printf("4.Quit\n");
printf("Enter your choice : ");
scanf("%d",&choice); switch(choice)
{
case 1 :
printf("Input the element for insertion in queue : "); scanf("%d", &item);
insert(item); break;
case 2 :
del(); break;
case 3:
display();
break; case 4:
break;
default:
printf("Wrong choice\n");
}
}while(choice!=4); return
0;
}
```

5. Output: Image of sample output to be attached here



```
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 1
Input the element for insertion in queue : 1
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 3
Queue elements :
1
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 4


...Program finished with exit code 0
Press ENTER to exit console.
```

Learning outcomes (What I have learnt):

- *Add item to the different position of the Linked list.*
- *Remove an item from the different position of the Linked list.*
- *Evaluation of expressions.*
- *Backtracking.*
- *Runtime memory management.*

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			