Experiment No. 4: Impalement of Linear Queue ADT using Array

Aim: To implement a Queue using arrays.

Objective:

1 Understand the Queue data structure and its basis operations.

2. Understand the method of defining Queue ADT and its basic operations.

3. Learn how to create objects from an ADT and member function are invoked.

Theory:

A Queue is an ordered collection of items from which items may be deleted at one end (called the front of the queue) and into which items may be inserted at the other end (the rear of the queue). Queues remember things in first-in-first-out (FIFO) order. The basic operations in a queue are: Enqueue - Adds an item to the end of queue. Dequeue - Removes an item from the front

A queue is implemented using a one dimensional array. FRONT is an integer value, which contains the array index of the front element of the array. REAR is an integer value, which contains the array index of the rear element of the array. When an element is deleted from the

queue, the value of front is increased by one. When an element is inserted into the queue, the value of rear is increased by one.

Algorithm:

ENQUEUE(item)

1. If (queue is full)

Print “overflow”

2. if (First node insertion)

Front++

3. rear++

Queue[rear]=value

DEQUEUE()

1. If (queue is empty)

Print “underflow”

2. if(front=rear)

Front=-1 and rear=-1

3. t = queue[front]

4. front++

5. Return t

ISEMPTY()

1. If(front = -1)then

return 1

2. return 0

ISFULL()

1. If(rear = max)then

return 1

2. return 0

Code :

```
#include <stdio.h>
#include<conio.h>
#include <stdlib.h>
#define max 10
```

```
int queue[max], front= -1, rear= -1;
void insert();
void delete();
void peek();
void display();

int main(){
   int option;
   clrscr();
   do {
   printf("\n\n*****QUEUES*****\n");
   printf("\n[1] Insert an element");
   printf("\n[2] Delete an element");
   printf("\n[3] Peek");
   printf("\n[4] Display");
   printf("\n[5] Exit");
   printf("\n\nEnter your choice: ");
   scanf("%d", &option);

   switch (option) {
      case 1:
         insert();
         break;
      case 2:
         delete();
         break;
      case 3:
         peek();
         break;
```

```
        case 4:
            display();
            break;
        case 5:
            break;
        default:
            printf("Invalid Choice");
            break;
    }
    }while (option!=5);
     getch();
     return 0;
}

void insert()
{
    int num;
    if(front == 0 && rear == max-1) {
        printf("\nQueue is full");
    }
    else
    {
        printf("\nEnter the data to be inserted : "); scanf("%d",&num);
        if(front==-1 && rear==-1)
        {
            front = rear = 0;
            queue[rear] = num;
        }
        else if(front!=0 && rear == max-1)
```

```
    { rear = 0;
      queue[rear] = num; }
    else
    {
      rear++;
      queue[rear] = num;
    }
  }}

void delete()
{
  if(front==-1 && rear==-1)
  {
    printf("Queue is Empty");
  }
  else
  {
    printf("\nThe deleted element is %d", queue[front]); if(front == rear)
    {
      front = rear = -1;
    }
    else if(front == max-1)
    {
      front = 0;
    }
    else
    {
      front++;
    }
```

```
    }
}

void peek()
{
    if(front== -1 && rear== -1)
    {
        printf("Queue is Empty");
    }
    else
    {
        printf(" \nThe first element in queue is %d",queue[front]);
    }
}

void display()
{ int i;
    if(front== -1 && rear== -1)
    {
        printf(" \nQueue is Empty"); }

    else
    {
        printf(" \nThe elements in the queue are : ");
    }
    if(front<rear)
    {
        for(i=front;i<rear;i++)
        {
```

```
            printf("%d\t",queue[i]);
        }
    }

    else if (front>rear) {
        for (i=front; i<max; i++) {
            printf("%d\t", queue[i]);
        }
        for (i=0; i<=rear; i++) {
            printf("%d\t", queue[i]);
        }
    }
    else {
        printf("%d\t", queue[front]);
    }
}
```

Output:

*****QUEUES*****

[1] Insert an element

[2] Delete an element

[3] Peek

[4] Display

[5] Exit

Enter your choice: 1

Enter the data to be inserted : 5

*****QUEUES*****

[1] Insert an element

[2] Delete an element

[3] Peek

[4] Display

[5] Exit

Enter your choice:

Conclusion: Code Works!