Assignment No. 13

Title: Write a C++ program to implement queue.

Objectives:-

- To add elements from rear end of Queue.
- To delete elements from front end of Queue.

Problem Statement:-

Pizza parlor accepting maximum M orders. Orders are served in first come first served basis. Order once placed cannot be cancelled. Write C++ program to simulate the system using circular queue using array..

Software Requirements:-

Fedora with Linux 3.11.10, Eclipse (Editor).

Hardware Requirements :-

Any Processor above Pentium 4.

Theory and Concepts:-

Double Ended Queue:-The word dequeue is a short form of double ended queue.

It is representation of both stack and queue and can be used as stack and queue. In a dequeue, insertion as well as deletion can be carried out either at the rear end or the front end. In practice, it becomes necessary to fix type of operation to be performed on front and rear end. Dequeue can be classified into two types;

A) Input Restricted Dequeue

The following operation are possible in an input restricted dequeue;

1) Insertion of an element at the rear end
2) Deletion of an element from front end
3) Deletion of an element from rear end

B) Output Restricted Dequeue

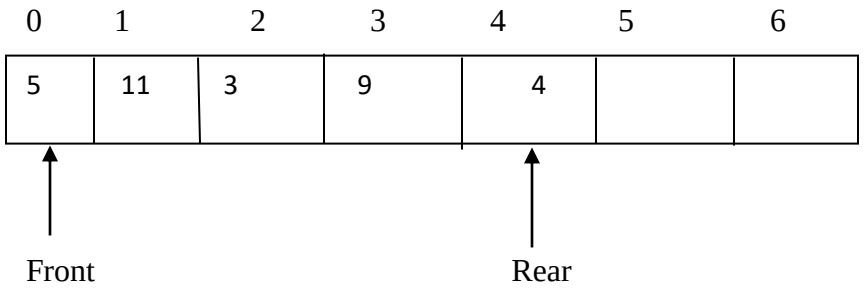
The following operation are possible in an output restricted dequeue

1) Deletion of an element from front end
2) Insertion of an element at rear end

3) Insertion of an element at the front end

There are various methods to implement a dequeue

a) Using a circular array
b) Using a singly linked list
c) Using a singly circular linked list
d) Using a doubly linked list
e) Using a doubly circular linked list



a) Initialization

```
Void init(structque *q)
        {
                q->front=-1;
                q->rear=-1;
        }
```

b) Print:-

```
void print(structque q)
        {
                int i;
                i=q.front;
                while(i!=q.rear)
                {
                        cout<<"\t"<<q.arr[i];
                          i=(i+1)%MAX;
                }
                cout<<"\t"<<q.arr[q.rear];
        }
```

c) Add Front:-

```
void addf(structque *q,int data)
        {
```

```
            if(isempty(*q))
            {
               q->front=q->rear=0;
                q->arr[q->front]=data;
            }
        else
            {
                    q->front=(q->front-1+MAX)%MAX;
                    q->arr[q->front]=data;
            }
    }
```

D) Add Rear:-

```
Void addr(structque *q, int data)
    {
    if(isempty(*q))
            {
                    q->front=q->rear=0;
                    q->arr[q->rear]=data;
            }
    else
            {
                     q->rear=(q->rear+1)%MAX;
                    q->arr[q->rear]=data;
            }
    }
```

E) Delete Front:-

```
Int delf(structque *q)
  {
    int data1;
    data1=q->arr[q->front];
    if(q->front==q->rear)
            init(q);
    else
              q->front=(q->front+1)%MAX;
    return data1;
   }
```

F) Delete Rear:-

```
Int delr(structque *q)
   {
    int data1;
```

```
        data1=q->arr[q->rear];
        if(q->front==q->rear)
                init(q);
        else
                  q->rear=(q->rear-1+MAX)%MAX;
                return data1;
    }
```

Algorithm:-

Step1:- START

Step 2:- Declare integer array, integer front, rear.

Step 3:- Declare integer data, ch.

Step 4:- Accept integer array from user.

Step 5:- Call functionaddf

Step 6:- Call function addr

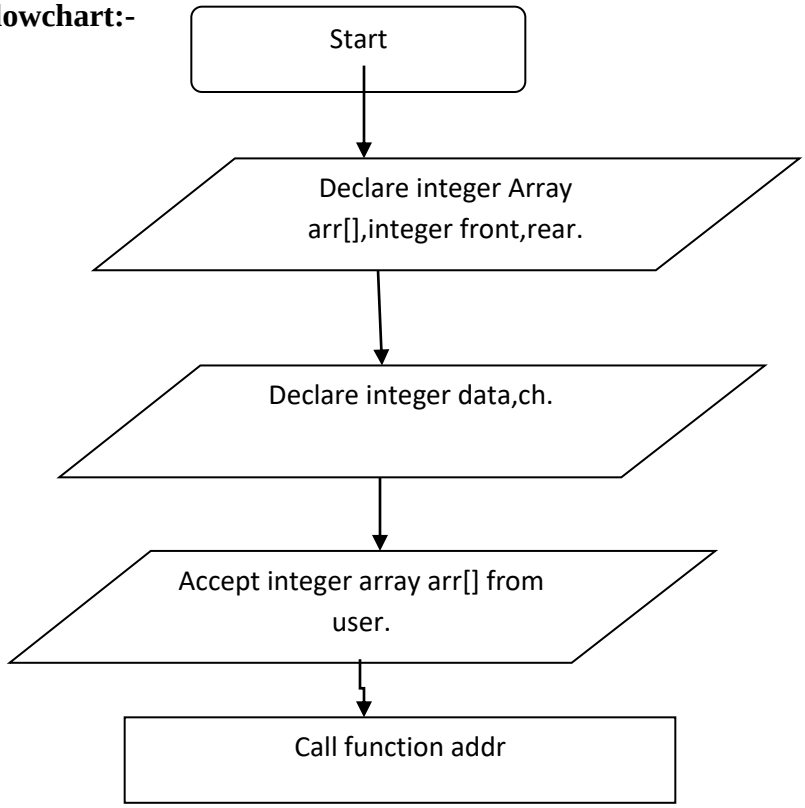
Step 7:- Call function delf

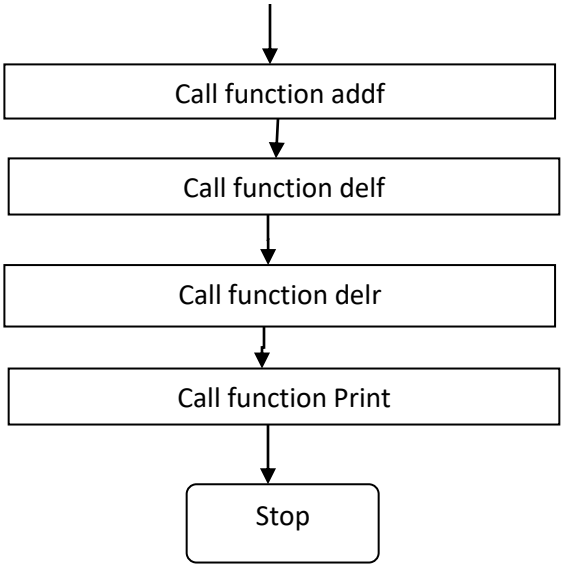
Step 8:- Call function delr

Step 9:-Call function Print.

Stop 10:- Stop

Flowchart:-





Conclusion: By using above code we successfully implemented Queue .

/*pizza parler acceting maximum 'm' order orders are stored in first come first serve basis otrder once placed cannot be cancelled. write c++ program to simulate system using circular queue using array. */

```
#include<iostream>

#define MAX 3

using namespace std;

class pizza

{

public:

        int front,rear,p_order[MAX];

        pizza()

  {

        front=-1;

        rear=-1;

  }

        void isfull();

        void isempty();

        void insert();

        void del();

        void display();

};

void pizza::isfull()

{

        if(front==(rear+1)%MAX)

        {

                cout<<"all orders full";

        }
```

```
}
void pizza::isempty()
{
	if(front ==-1 && rear==-1)
	{
		cout<<"no orders ";
	}
}
void pizza::insert()
{
	int ch;
	isfull();
	cout<<"\n 1.onion pizza.";
	cout<<"\n 2.chees pizza.";
	cout<<"\n 3.ytfyr";
	cout<<"\n 4.ygdshgf";
	cout<<"\n enter a pizza choice";
	cin>>ch;

	if(front==-1)
	{
		front=rear=0;
		p_order[rear]=ch;
	}
	else
```

```
        {
                rear=(rear+1)%MAX;
                p_order[rear]=ch;
        }
}

void pizza::del()
{
        isempty();
        if(front == rear)
        {
                cout<<"make payment Of"<<p_order[front];
                front=rear=-1;
        }
        else
        {
                cout<<"make payment Of"<<p_order[front];
                 front=(front+1)%MAX;
        }

}

void pizza::display()
{
        int i;
```

```
        i=front;
        while(i!=rear)
        {
                cout<<p_order[i];
                i=(i+1)%MAX;
        }
}
int main()
{
        int ch;
        char ans;
pizza p;
do
{
cout<<"\n 1.pizza order.";
cout<<"\n 2.make payment";
cout<<"\n 3.display pending order.";
cout<<"\n enter a choice";
cin>>ch;
switch(ch)
{
case 1:
        p.insert();
break;
case 2:
```

```
            p.del();
            break;
case 3:
            p.display();
            break;
default:
            cout<<"wrong choice";
            break;
}
cout<<"do you want to contineu if yes enter y";
cin>>ans;
}while(ans=='y');
return 0;
}
```