Queues are frequently used in computer programming, and a typical example is the creation of a job queue by an operating system. If the operating system does not use priorities, then the jobs are processed in the order they enter the system. Write C++ program for simulating job queue. Write functions to add job and delete job from queue

```
#include <iostream>
#define MAX 10
using namespace std;



struct queue
{

 int data[MAX];
        int front,rear;
};



class Queue
{

struct queue q;
  public:
    Queue()

{
        q.front=q.rear=-1;
        }
            int isempty();
            int isfull();
            void enqueue(int);
            int delqueue();
            void display();
        };


        int Queue::isempty()
        {
                if (q.front==q.rear)
                 return 1;
                 else
                 return 0;

}


int Queue::isfull()
```

```
{

if(q.rear==MAX-1)

        return 1;

else

        return 0;

}


void Queue::enqueue(int x)
{

q.data[++q.rear]=x;

}


int Queue::delqueue()
{

return q.data[++q.front];

}


void Queue::display()
{

 int i;
   cout<<"\n";
   for(i=q.front+1;i<=q.rear;i++)
            cout<<q.data[i]<<" ";
}


int main()
{

Queue obj;
        int ch,x;
        do{
         cout<<"\n 1. insert job\n 2.delete job\n 3.display\n 4.Exit\n Enter your choice:";
                  cin>>ch;
                 switch(ch)
                 {
         case 1: if (!obj.isfull())
                            {
```

```
                 cout<<"\n Enter data:";
                        cin>>x;
                        obj.enqueue(x);
                   }
                 else
                         cout<< "Queue is overflow";
                         break;

 case 2: if(!obj.isempty())
                         cout<<"\n Deleted Element="<<obj.delqueue();
                  else
                       {
                         cout<<"\n Queue is underflow";
                         }
                         cout<<"\nremaining jobs :";
                         obj.display();
                         break;

  case 3: if (!obj.isempty())
               {
                 cout<<"\n Queue contains:";
                                    obj.display();
                 }
                 else
                         cout<<"\n Queue is empty";
                         break;
case 4: cout<<"\n Exit";
        }
    }while(ch!=4);
return 0;
}
```

```
/**************************************OUTPUT****************************
 1. insert job
 2.delete job
 3.display
 4.Exit
 Enter your choice:1
 Enter data:34

 1. insert job
 2.delete job
 3.display
 4.Exit
 Enter your choice:1
 Enter data:64
 1. insert job
 2.delete job
 3.display
 4.Exit
 Enter your choice:1
 Enter data:84

 1. insert job
 2.delete job
 3.display
 4.Exit
 Enter your choice:1
 Enter data:93

 1. insert job
 2.delete job
 3.display
 4.Exit
 Enter your choice:3
 Queue contains:
34 64 84 93

 1. insert job
 2.delete job
 3.display
 4.Exit
 Enter your choice:2
 Deleted Element=34
remaining jobs :
64 84 93

 1. insert job
 2.delete job
 3.display
 4.Exit
 Enter your choice:3
 Queue contains:
64 84 93

 1. insert job
 2.delete job
 3.display
 4.Exit
 Enter your choice:4
 Exit */
```