**Experiment No-11**

C++ program for simulating job queue

# Objectives: -

Understand the function to add a job and delete a job from queue.

.

# Problem Statement: -

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.

# Theory-

**Basic Concept of queue:-**

Definition:- The queue can be formally defined as ordered collection of elements that has two ends named as front and rear. From the front end one can delete the elements and from the rear end one can insert the elements.

Following figure represents the queue of few elements.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 10 | 20 | 30 | 40 | 50 | 60 | 70 |

# front rear

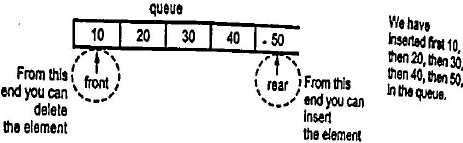
**Figure: - Queue**

# Queue Operations:-

1. **Insertion of element into the queue:-**

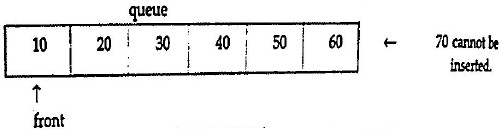
The insertion of any element in the queue will always take place from the rear

end.



# Figure: - Representing the insertion

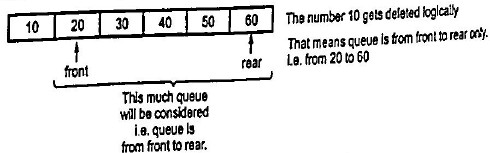
Before performing insertion operation you must check whether the queue is full or not If rear pointer is going beyond the maximum size of the queue the queue overflow occurs.



# Figure: - Representing the queue overflow

1. **Deletion of element from the queue:-**

The deletion of any element in the queue takes placed by the front end always.



Before performing deletion operation you must check whether the queue is empty or not. If the queue is empty, you cannot perform the deletion . The result of illegal attempt to deletion an elements from the empty queue is called as the queue underflow condition.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |

# front rear Figure: - Representing the Queue underflow

**Algorithm:-**

**1.Enqueue function( to add element ‘ x ‘ in the queue)**

i) Increment rear counter by 1

ii)Insert element at rear position as data[rear]=x

**2.Delqueue function( to delete element from the queue)**

i) Increment front counter by 1

ii)Take out front element of queue in x as x=data[front]

**3.Display function( to display all elements from the queue)**

i) Initialize i=front +1

ii)display data[i]

iii)increment i

iv)repeat step i), ii) ,iii) till i<=rear

**4.queuefull function( to check if queue is full)**

i)if rear=MAX-1

return 1to indicate queue is full

ii) if rear!=MAX-1

return 0 to indicate queue is not full

**4.queueempty function( to check if queue is empty)**

i)if front =rear

return 1to indicate queue is empty

ii) if rear!=front

return 0 to indicate queue is not empty

**Conclusion:**

By this way, we can learn how to add a job and delete a job from queue.