**Queue**

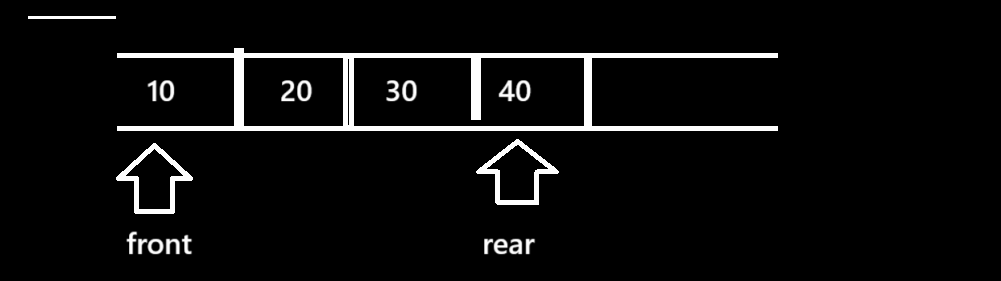
**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Queue is a data structure which is used for first in first out purpose

Means first inserted element removes firstly remove from queue

Means we insert data using one side and remove data using some another side

If data insertion side known as rear and data deletion side known as front means we can say Queue has two pointers.

****

**Q. Can we give real time example where we can use Queue?**

Scheduling algorithm, messaging systems etc

**Types of Queue**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**a) Linear Queue**

**b) Circular Queue**

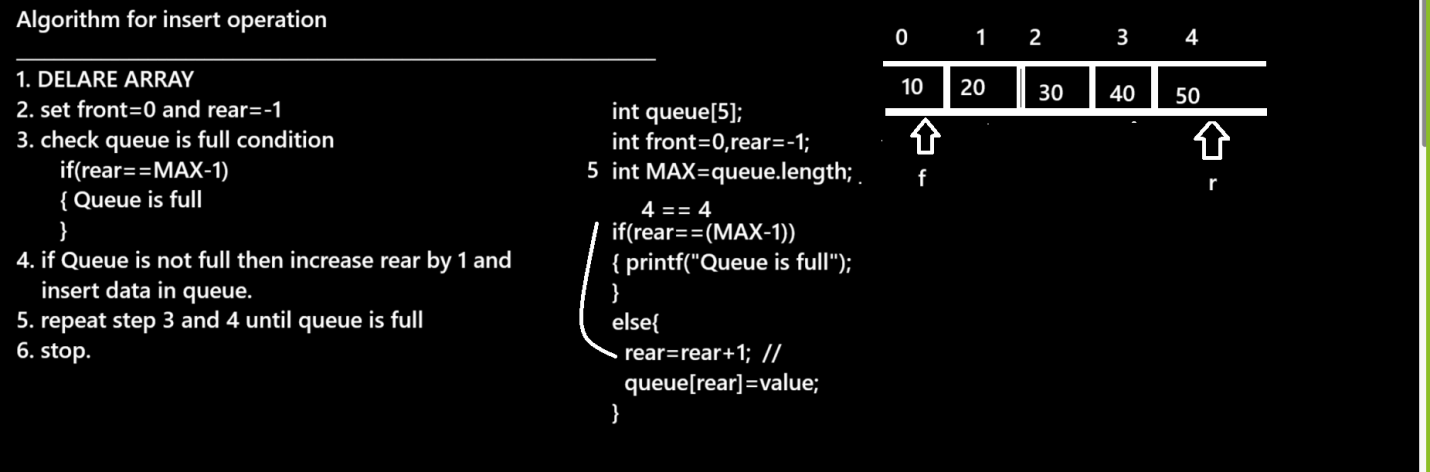
**c) Priority Queue**

**If we want to work with Queue we have following operations.**

**Insert:** insert operation means we insert data in queue when we insert data in queue then

Your rear pointer gets increase by 1 and initially rear=-1 and if we think about insert operation then there is possibility of queue is full

When we try to insert data in queue but there is no space for insert data in queue called as Queue is full.

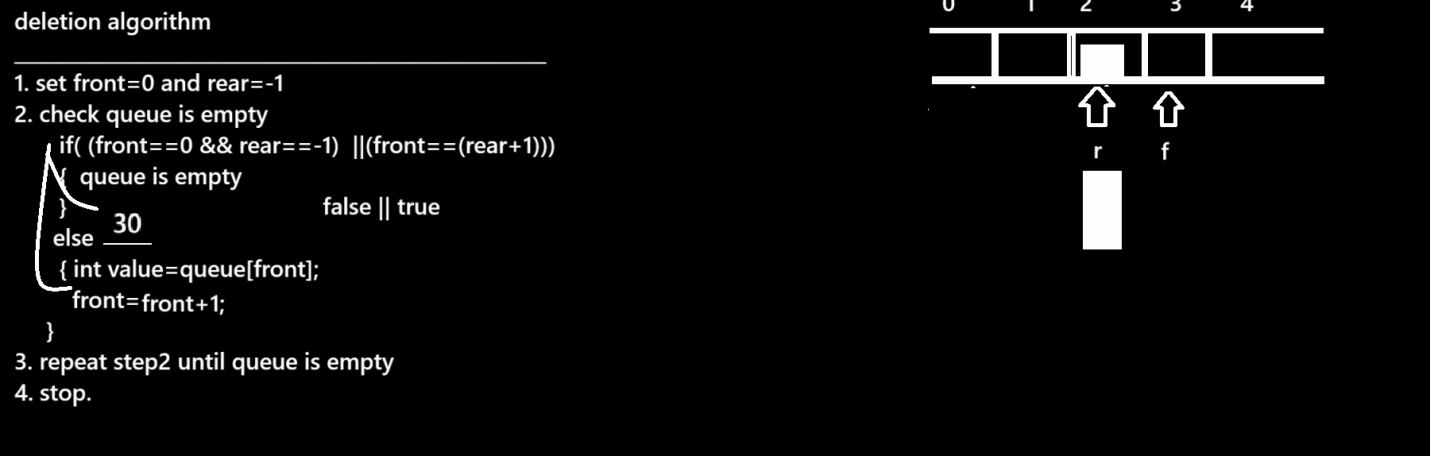


**Delete:** delete means we try to remove data from queue when we remove data from queue then front increase by 1 and if we think about delete operation then there is possibility queue is empty.

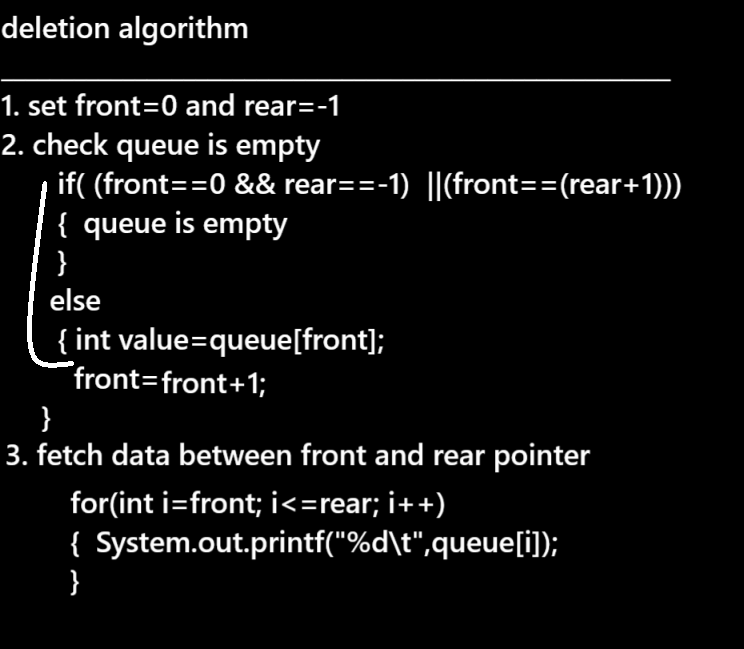
If we think about queue empty possibility occurs two times.

a) When front =0 and rear=-1

b) When front==rear+1



**Display:** if we want to fetch data from queue then can fetch between front and rear



Example with source code

import java.util.\*;

public class QueueAlgo

{ static int queue[]=new int[5];

static int front=0,rear=-1;

static int choice;

static int index=-1;

public static void main(String x[])

{ Scanner xyz = new Scanner(System.in);

do{

System.out.println("\n1:INSERT");

System.out.println("2:DELETE");

System.out.println("3:DISPLAY");

System.out.println("4:EXIT");

System.out.println("Enter your choice");

choice=xyz.nextInt();

switch(choice)

{

case 1:

System.out.println("Enter data in queue");

int data=xyz.nextInt();

insert(data);

break;

case 2:

int index=pop();

if(index==-1)

{ System.out.println("Queue is empty");

}

else{

System.out.println("Deleted value is "+queue[index]);

}

break;

case 3:

display();

break;

case 4:

System.exit(0);

break;

default:

System.out.println("Wrong choice");

}

}while(true);

}

public static void insert(int ele)

{

if(rear==(queue.length-1)){

System.out.println("Queue is full");

}

else

{

rear=rear+1;

queue[rear]=ele;

}

}

public static int pop()

{

if((front==0 && rear==-1) ||(front==rear+1))

{ return -1;

}

else

{

int value=queue[front];

front=front+1;

return (front-1);

}

}

public static void display()

{

if((front==0 && rear==-1) ||(front==rear+1))

{ System.out.println("Queue is empty");

}

else{

for(int i=front;i<=rear;i++)

{ System.out.printf("%d\t",queue[i]);

}

}

}

}