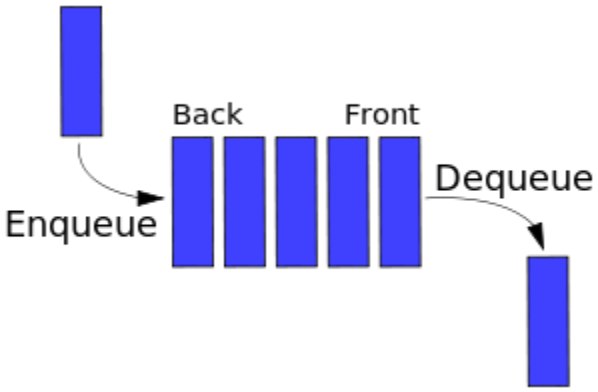
Fifo Queue

In computer science, a queue is a particular kind of abstract data type or collection in which the entities in the collection are kept in order and the principal (or only) operations on the collection are the addition of entities to the rear terminal position, known as *enqueue,* and removal of entities from the front terminal position, known as *dequeue*. This makes the queue a First-In-First-Out (FIFO) data structure. In a FIFO data structure, the first element added to the queue will be the first one to be removed. This is equivalent to the requirement that once a new element is added, all elements that were added before have to be removed before the new element can be removed. Often a *peek* or *front* operation is also implemented, returning the value of the front element without dequeuing it. A queue is an example of a linear data structure, or more abstractly a sequential collection.



CODES (C)

```
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
typedef struct queue
{
        int *a  ;
        int ms,r,f;
}Q;
void init(Q *q,int x)
{
        q->ms=x;
        q->a=(int *)malloc(q->ms*sizeof(int));
        q->f=0;
        q->r=-1;
}
int isfull(Q *q)
{
        if(q->r==q->ms-1&&q->f==0)
                return 1;
        else
                return 0;
}
int isempty(Q *q)
{
        if(q->r==-1&&q->f==0)
                return 1;
        else
                return 0;
}
void insert(Q *q,int z)
{
        if(isfull(q))
                printf("queue full ");
        else
                q->a[++q->r]=z;
}
int del(Q *q)
{
        int j=0;
        if(isempty(q)==1)
        {
                return 1 ;
        }
        else
        {
                j=q->a[q->f++];
                return j;
        }
}
```

CODES (JAVA)

```
import java.io.*;
class queue
{
	int q[];
	int front;
	int rear;
	int temp;
	int max;
	queue(int n)
	{
		max=n;
		q=new int[max];
		front=0;
		rear=-1;
	}
	void push(int a)
	{
		int b;
		b=a;
		if(rear>=max-1)
		{
			System.out.println("\t\t*****queue  is overflow!!!!!******\n");
		}
		else
		{
			rear=rear+1;
			q[rear]=b;

		}
	}
```

```
	void pop()
	{
		if(rear<front)
		{
			System.out.println("\t\tstack is underflow!!!!!\n");
		}
		else
		{
			System.out.println(""+q[front]+"is deleted\n");
			front=front+1;

		}
	}
	void display()
	{
		System.out.println("\n\t\tElements are:");
		for(int i=front;i<=rear;i++)
		{
			System.out.println(""+"\t\t"+q[i]);

		}System.out.println("\n");
	}
}
```