**Array Queue Implementation**

public class QueueArray {

String[] items;

int front, rear;

int size =0;

QueueArray() {

items = new String[5];

front = -1;

rear = -1;

}

// insert elements to the queue

void enQueue(String element) {

// if queue is full

if (size == 5) {

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

}

else {

if (front == -1) {

// mark front denote first element of queue

front = 0;

}

rear=(rear+1)%5;

// insert element at the rear

items[rear] = element;

size++;

System.out.println("Insert " + element);

}

}

// delete element from the queue

String deQueue() {

String element;

// if queue is empty

if (size ==0) {

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

return (null);

}

else {

// remove element from the front of queue

element = items[front];

items[front] =null;

front = (front+1)%5;

size--;

}

System.out.println( element + " Deleted");

return (element);

}

}

**Linked List Queue Implementation**

public class QueueLinkedList {

Node<String> front=null;

Node<String> rear=null;

int size =0;

public void enqueue(String item)

{

Node<String> n1 = new Node<String>(item,null);

if(size==0)

{

front = n1;

rear = n1;

size++;

}

else

{

rear.next=n1;

rear = n1;

size++;

}

}

public String dequeue()

{

String item =null;

if(size>0)

{

item = front.data;

front = front.next;

size--;

}

return item;

}

}