|  |
| --- |
| 0hnQGAuNPo4wojnr77c2x8cWiaT70FGHBZ2h-3__FYQ.png |
| Queue |
| Data Structures Made Easy |
|  |

Dublin city university

# 1. *Bounded Queue*

class bounded\_Queue<T>{

private T[] sequence = (T[])(new Object[10000]);

private int size = 0;

private int head = 0;

private int tail = 0;

public boolean isEmpty(){

return size == 0;

}

public boolean enq(T t){

if(size >= sequence.length){

T[] sequence\_2 = (T[])(new Object[sequence.length \* 2]);

System.arraycopy(sequence, 0, sequence\_2, 0, sequence.length);

sequence = sequence\_2;

}

sequence[tail] = t;

tail = (tail + 1) % sequence.length;

size++;

return true;

}

public T deq(){

if(isEmpty())

return null;

else{

T temp = sequence[head];

head = (head+1)%sequence.length;

size--;

return temp;

}

}

public static void main(String [] args){

bounded\_Queue<Integer> queue = new bounded\_Queue<Integer>();

System.out.println('\n' + "ADDED TO QUEUE");

System.out.println("===============");

for(int index = 10; index <= 100; index += 10){

System.out.print(index + " ");

queue.enq(index);

}

System.out.println();

System.out.println('\n' + "REMOVED FROM QUEUE");

System.out.println("=================");

while(!queue.isEmpty()){

int element = queue.deq();

System.out.print(element + " ");

}

}

}

# 2. *Unbounded Queue*

class unbounded\_Queue<T>{

private static class Node<T>{

private T item;

private Node<T> next = null;

Node(T item0, Node<T> next0){

item = item0;

next = next0;

}

}

private Node<T> head = null;

private Node<T> tail = null;

public boolean isEmpty(){

return head == null;

}

public boolean enq(T t){

Node<T> tNode = new Node<T>(t, null);

if(tail != null)

tail.next = tNode;

else

head = tNode;

tail = tNode;

return true;

}

public T deq(){

if(isEmpty())

tail = null;

T temp = head.item;

head = head.next;

return temp;

}

public static void main(String [] args){

unbounded\_Queue<Integer> queue = new unbounded\_Queue<Integer>();

System.out.println('\n' + "ADDED TO QUEUE");

System.out.println("===============");

for(int index = 10; index <= 100; index += 10){

System.out.print(index + " ");

queue.enq(index);

}

System.out.println();

System.out.println('\n' + "REMOVED FROM QUEUE");

System.out.println("=================");

while(!queue.isEmpty()){

int element = queue.deq();

System.out.print(element + " ");

}

}

}