Name: Vangipuram Srinivasa Sarath Chandra
Regd No: 21BCE9853
Title: Digital Assignment Java
Date: 11/June/2022

3. Develp a java program to implement queue interface by using ArrayDeque class and LinkedList class and priority queue class.[5M]

List of methods used in ArrayDeque, LinkedList & PriorityQueue :

add	addAll	addFirst	addLast
removeFirst	removeLast	getFirst	getLast
iterator	clear	peek	poll

Array Deque

```java
import java.util.ArrayDeque;
import java.util.Iterator;

public class ArrayDequeAssign {
    static ArrayDeque<String> createArrayDeque(String... elements) {
        ArrayDeque<String> newArrayDeque = new ArrayDeque<String>();
        for (String el : elements) {
            newArrayDeque.add(el);
        }
        return newArrayDeque;
    }

    public static void main(String[] args) {
        ArrayDeque<String> arrayDeque = createArrayDeque("A", "B", "C", "D", "E", "F", "G");
        System.out.println("arrayDeque: " + arrayDeque);

        // add
        arrayDeque.add("H");
        System.out.println("add H -> arrayDeque: " + arrayDeque);

        // add all
        ArrayDeque<String> addAll = new ArrayDeque<String>(arrayDeque);
        addAll.addAll(createArrayDeque("F", "G", "H", "I", "J", "K", "L"));
        System.out.println("added from F-L" + addAll);

        // add first
        arrayDeque.addFirst("M");
        System.out.println("add M -> arrayDeque: " + arrayDeque);

        // add last
        arrayDeque.addLast("N");
        System.out.println("add N -> arrayDeque: " + arrayDeque);

        // remove first
        arrayDeque.removeFirst();
        System.out.println("remove first -> arrayDeque: " + arrayDeque);

        // remove last
        arrayDeque.removeLast();
        System.out.println("remove last -> arrayDeque: " + arrayDeque);

        // get first
        System.out.println("get first -> arrayDeque: " + arrayDeque.getFirst());

        // get last
```

```
        System.out.println("get last -> arrayDeque: " + arrayDeque.getLast());

        // peek
        System.out.println("peek -> arrayDeque: " + arrayDeque.peek());

        // poll
        System.out.println("poll -> arrayDeque: " + arrayDeque.poll());

        // iterate through iterator
        System.out.print("iterate: ");
        Iterator<String> iterator = arrayDeque.iterator();
        while (iterator.hasNext()) {
            System.out.print(iterator.next() + " ");
        }

        // clear
        arrayDeque.clear();
        System.out.println("\nclear arrayDeque: " + arrayDeque);

    }
}
```

Output:

```
arrayDeque: [A, B, C, D, E, F, G]
add H -> arrayDeque: [A, B, C, D, E, F, G, H]
added from F-L[A, B, C, D, E, F, G, H, F, G, H, I, J, K, L]
add M -> arrayDeque: [M, A, B, C, D, E, F, G, H]
add N -> arrayDeque: [M, A, B, C, D, E, F, G, H, N]
remove first -> arrayDeque: [A, B, C, D, E, F, G, H, N]
remove last -> arrayDeque: [A, B, C, D, E, F, G, H]
get first -> arrayDeque: A
get last -> arrayDeque: H
iterate: A B C D E F G H
clear arrayDeque: []
```

Linked List

```
import java.util.Iterator;
import java.util.LinkedList;

public class LinkedListAsQueueAssign {
        // helper method to create a linked list
    static LinkedList<String> createLinkedList(String... elements) {
        LinkedList<String> newLinkedList = new LinkedList<String>();
        for (String el : elements) {
            newLinkedList.add(el);
        }
        return newLinkedList;
    }
    // main method
    public static void main(String[] args) {
        LinkedList<String> linkedList = createLinkedList("A", "B", "C", "D", "E", "F", "G");
        System.out.println("Linked List: " + linkedList);

        // add
        linkedList.add("H");
        System.out.println("add H -> Linked List: " + linkedList);

        // add all
        LinkedList<String> addAll = new LinkedList<String>(linkedList);
        addAll.addAll(createLinkedList("F", "G", "H", "I", "J", "K", "L"));
        System.out.println("added from F-L" + addAll);

        // add first
        linkedList.addFirst("M");
```

```
        System.out.println("add M -> Linked List: " + linkedList);

        // add last
        linkedList.addLast("N");
        System.out.println("add N -> Linked List: " + linkedList);

        // remove first
        linkedList.removeFirst();
        System.out.println("remove first -> Linked List: " + linkedList);

        // remove last
        linkedList.removeLast();
        System.out.println("remove last -> Linked List: " + linkedList);

        // get first
        System.out.println("get first -> Linked List: " + linkedList.getFirst());

        // get last
        System.out.println("get last -> Linked List: " + linkedList.getLast());

        // peek
        System.out.println("peek -> Linked List: " + linkedList.peek());

        // poll
        System.out.println("poll -> Linked List: " + linkedList.poll());

        // iterate through iterator
        System.out.print("iterate: ");
        Iterator<String> iterator = linkedList.iterator();
        while (iterator.hasNext()) {
            System.out.print(iterator.next() + " ");
        }

        // clear
        linkedList.clear();
        System.out.println("\nclear LinkedList: " + linkedList);

    }
}
```

Output:

```
Linked List: [A, B, C, D, E, F, G]
add H -> Linked List: [A, B, C, D, E, F, G, H]
added from F-L[A, B, C, D, E, F, G, H, F, G, H, I, J, K, L]
add M -> Linked List: [M, A, B, C, D, E, F, G, H]
add N -> Linked List: [M, A, B, C, D, E, F, G, H, N]
remove first -> Linked List: [A, B, C, D, E, F, G, H, N]
remove last -> Linked List: [A, B, C, D, E, F, G, H]
get first -> Linked List: A
get last -> Linked List: H
peek -> Linked List: A
poll -> Linked List: A
iterate: B C D E F G H
clear LinkedList: []
```

Priority Queue

As the implementation of priority queue in java isn't done using a linear data structure we don't have the following methods:

	addFirst	addLast	removeFirst	removeLast	getFirst	getLast

+ Used **Streams**

```java
import java.util.*;
import java.util.stream.Collectors;
import java.util.stream.IntStream;

public class PriorityQueueAssign {
        // helper method to create a priority queue
    static PriorityQueue<Integer> createPriorityQueue(Integer... elements) {
        PriorityQueue<Integer> newPriorityQueue = new PriorityQueue<Integer>();
        for (Integer el : elements) {
            newPriorityQueue.add(el);
        }
        return newPriorityQueue;
    }

            // main method
    public static void main(String[] args) {
        PriorityQueue<Integer> priorityQueue = createPriorityQueue(2, 5, 1, 8, 3, 7, 6, 4);
        System.out.println("priorityQueue: " + priorityQueue);

        // add
        priorityQueue.add(11);
        priorityQueue.add(5);
        System.out.println("add 11 -> priorityQueue: " + priorityQueue);

        // add all
        PriorityQueue<Integer> addAll = new PriorityQueue<Integer>(priorityQueue);
        List<Integer> intList = IntStream.rangeClosed(1, 20)
                .boxed().collect(Collectors.toList());
        addAll.addAll(intList);

        System.out.println("added from 1-20 +List: " + addAll);

        // remove
        priorityQueue.remove(5);
        System.out.println("remove 5 -> priorityQueue: " + priorityQueue);

        // remove first
        Integer ele = priorityQueue.remove();
        System.out.println("remove first -> priorityQueue: " + priorityQueue + " Removed
Element: " + ele);

        // poll
        System.out.println("poll -> priorityQueue: " + priorityQueue.poll());

        // peek
        System.out.println("peek -> priorityQueue: " + priorityQueue.peek());

        // iterate through iterator
        System.out.print("iterate: ");
        Iterator<Integer> iterator = priorityQueue.iterator();
        while (iterator.hasNext()) {
            System.out.print(iterator.next() + " ");
        }

        // iterate through stream
        System.out.print("\niterate through stream: ");
        priorityQueue.stream().forEach(System.out::print);
        System.out.println();

        // clear
        priorityQueue.clear();
        System.out.println("clear -> priorityQueue: " + priorityQueue);
    }
```

```
    }
```

Output:

```
priorityQueue: [1, 3, 2, 4, 5, 7, 6, 8]
add 11 -> priorityQueue: [1, 3, 2, 4, 5, 7, 6, 8, 11, 5]
added from 1-20 +List: [1, 1, 2, 4, 3, 2, 4, 6, 8, 5, 5, 7, 3, 6, 5, 8, 7, 11, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]
remove 5 -> priorityQueue: [1, 3, 2, 4, 5, 7, 6, 8, 11]
remove first -> priorityQueue: [2, 3, 6, 4, 5, 7, 11, 8] Removed Element: 1
poll -> priorityQueue: 2
peek -> priorityQueue: 3
iterate: 3 4 6 8 5 7 11
iterate through stream: 34685711
clear -> priorityQueue: []
```