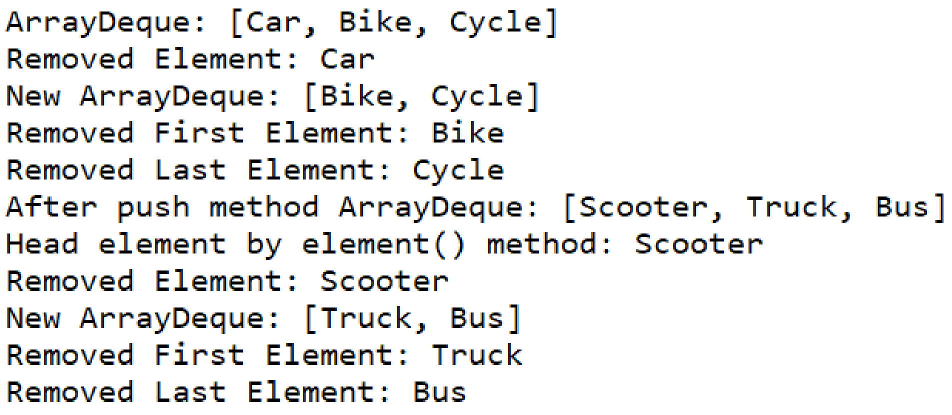
**1. Implement an ArrayDequeue and all of its methods such as add(), addFirst(), addLast(), element(), poll(), push(), remove.**

```
import java.util.ArrayDeque;

public class Sba3_01 {
        public static void main(String[] args) {
                ArrayDeque<String> vehicle = new ArrayDeque<>();
                // Using add()
                vehicle.add("Bike");
                // Using addFirst()
                vehicle.addFirst("Car");
                // Using addLast()
                vehicle.addLast("Cycle");
                System.out.println("ArrayDeque: " + vehicle);
                // Using poll()
                String element = vehicle.poll();
                System.out.println("Removed Element: " + element);
                System.out.println("New ArrayDeque: " + vehicle);
                // Using pollFirst()
                String firstElement = vehicle.pollFirst();
                System.out.println("Removed First Element: " + firstElement);
                // Using pollLast()
                String lastElement = vehicle.pollLast();
                System.out.println("Removed Last Element: " + lastElement);
                // using push()
                vehicle.push("Bus");
                vehicle.push("Truck");
                vehicle.push("Scooter");
                System.out.println("After push method ArrayDeque: " + vehicle);
                // using element()--returns element present in the head
                System.out.println("Head element by element() method: " +
                                vehicle.element());
                // Using remove()
                String element1 = vehicle.remove();
                System.out.println("Removed Element: " + element1);
                System.out.println("New ArrayDeque: " + vehicle);
                // Using removeFirst()
                String firstElement1 = vehicle.removeFirst();
                System.out.println("Removed First Element: " + firstElement1);
                // Using removeLast()
                String lastElement1 = vehicle.removeLast();
                System.out.println("Removed Last Element: " + lastElement1);
                }
}
```

**Output :**

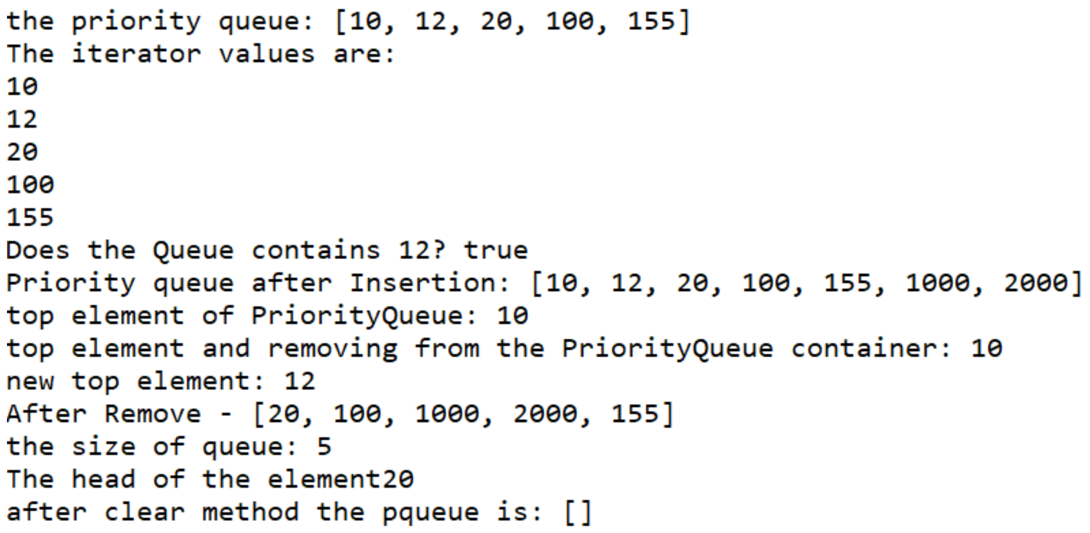


```
ArrayDeque: [Car, Bike, Cycle]
Removed Element: Car
New ArrayDeque: [Bike, Cycle]
Removed First Element: Bike
Removed Last Element: Cycle
After push method ArrayDeque: [Scooter, Truck, Bus]
Head element by element() method: Scooter
Removed Element: Scooter
New ArrayDeque: [Truck, Bus]
Removed First Element: Truck
Removed Last Element: Bus
```

**2. Implement a PriorityQueue and use all the methods.**

```java
import java.util.Iterator;
import java.util.PriorityQueue;

public class Sba3_02 {
        public static void main(String[] args) {
                // Creating empty priority queue
                PriorityQueue<Integer> pQueue = new PriorityQueue<Integer>();
                // Adding items to the pQueue using add()
                pQueue.add(10);
                pQueue.add(12);
                pQueue.add(20);
                pQueue.add(100);
                pQueue.add(155);
                System.out.println("the priority queue: " + pQueue);
                // Creating an iterator
                Iterator <Integer>value =pQueue.iterator();
                // Displaying the values after iterating through the queue
                System.out.println("The iterator values are: ");
                while (value.hasNext()) {
                System.out.println(value.next());
                }
                // Check for "4" in the queue
                System.out.println("Does the Queue contains 12? "+pQueue.contains(12));
                // Inserting using offer()
                pQueue.offer(1000);
                pQueue.offer(2000);
                // Displaying th final Queue
                System.out.println("Priority queue after Insertion: " +pQueue );
                // Printing the top element of PriorityQueue
                System.out.println("top element of PriorityQueue: " + pQueue.peek());
                // Printing the top element and removing it
                // from the PriorityQueue container
                System.out.println("top element and removing from the PriorityQueue container: " + pQueue.poll());
                // Printing the top element again
                System.out.println("new top element: " + pQueue.peek());
                // using the method
                pQueue.remove(12);
                System.out.println("After Remove - " + pQueue);
                //to find size
                System.out.println("the size of queue: "+pQueue.size());
                //element()
                System.out.println("The head of the element"+pQueue.element());
                // Creating an iterator
                //clear()
                pQueue.clear();
                System.out.println("after clear method the pqueue is: "+pQueue);
                }
}
```

**Output :**

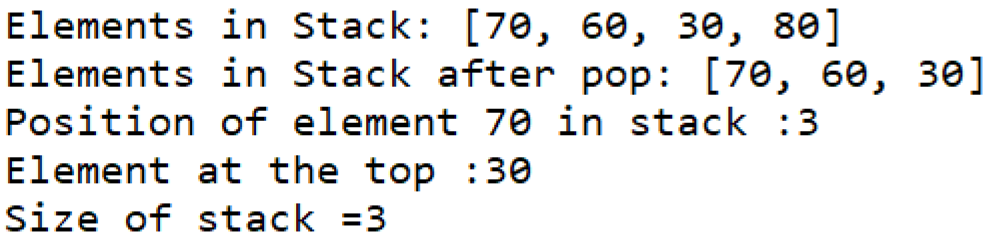


```
the priority queue: [10, 12, 20, 100, 155]
The iterator values are:
10
12
20
100
155
Does the Queue contains 12? true
Priority queue after Insertion: [10, 12, 20, 100, 155, 1000, 2000]
top element of PriorityQueue: 10
top element and removing from the PriorityQueue container: 10
new top element: 12
After Remove - [20, 100, 1000, 2000, 155]
the size of queue: 5
The head of the element20
after clear method the pqueue is: []
```

**3. Implement a Stack and all of its methods peek(), push(), pop(), and to determine the size of the stack.**

```
import java.util.Stack;

public class Sba3_03 {
        public static void main(String[] args) {
                Stack<Integer> stk = new Stack<>();
                // boolean result = stk.empty();
                // System.out.println("Is the stack empty? " + result);
                stk.push(70);
                stk.push(60);
                stk.push(30);
                stk.push(80);
                System.out.println("Elements in Stack: " + stk);
                // result = stk.empty();
                // System.out.println("Is the stack empty? " + result);

                stk.pop();
                System.out.println("Elements in Stack after pop: " + stk);

                System.out.println("Position of element 70 in stack :" + stk.search(70));

                System.out.println("Element at the top :" + stk.peek());

                System.out.println("Size of stack =" + stk.size());
        }
}
```

**Output :**

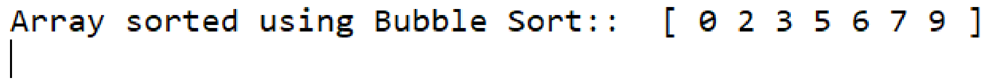


```
Elements in Stack: [70, 60, 30, 80]
Elements in Stack after pop: [70, 60, 30]
Position of element 70 in stack :3
Element at the top :30
Size of stack =3
```

**4. Write a program to implement insertion sort.**

```
public class Sba3_04 {

    static int[] insertionsort(int[] arr) {
        int n = arr.length;
        for (int j = 1; j < n; j++) {
            int key = arr[j];
            int i = j - 1;
            while ((i > -1) && (arr[i] > key)) {
                arr[i + 1] = arr[i];
                i--;
            }
            arr[i + 1] = key;
        }
        return arr;
    }

    // static method to print array
    static void printarr(String s, int[] arr) {
        System.out.print(s + " [ ");
        for (int i : arr) {
            System.out.print(i + " ");
        }
        System.out.println("]");
    }

    public static void main(String[] args) {
        int[] numArr = { 5, 9, 7, 3, 6, 0, 2 };
        printarr("Array sorted using Bubble Sort:: ", insertionsort(numArr));
    }

}
```

**Output :**



```
Array sorted using Bubble Sort::  [ 0 2 3 5 6 7 9 ]
```