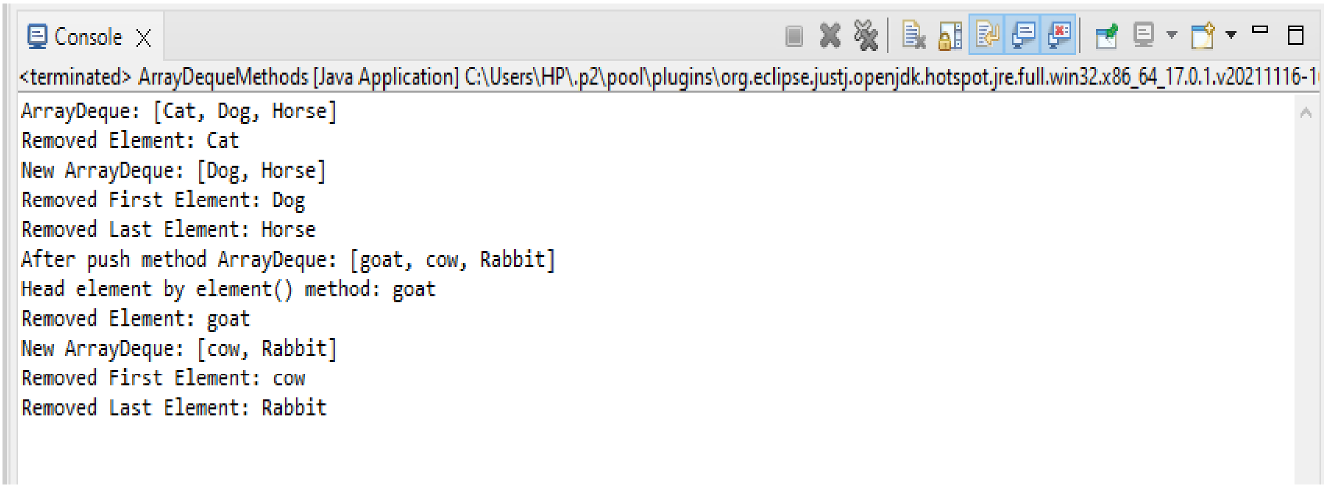
QUESTION 1

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

```
package AssignmentSBA3;
import java.util.ArrayDeque;

public class ArrayDequeMethods {
	public static void main(String[] args) {
		ArrayDeque<String> animals = new ArrayDeque<>();
		// Using add()
		animals.add("Dog");
		// Using addFirst()
		animals.addFirst("Cat");
		// Using addLast()
		animals.addLast("Horse");
		System.out.println("ArrayDeque: " + animals);
		// Using poll()
		String element = animals.poll();
		System.out.println("Removed Element: " + element);
		System.out.println("New ArrayDeque: " + animals);
		// Using pollFirst()
		String firstElement = animals.pollFirst();
		System.out.println("Removed First Element: " + firstElement);
		// Using pollLast()
		String lastElement = animals.pollLast();
		System.out.println("Removed Last Element: " + lastElement);
		// using push()
		animals.push("Rabbit");
		animals.push("cow");
		animals.push("goat");
		System.out.println("After push method ArrayDeque: " + animals);
		// using element()--returns element present in the head
		System.out.println("Head element by element() method: " + 
animals.element());
		// Using remove()
		String element1 = animals.remove();
		System.out.println("Removed Element: " + element1);
		System.out.println("New ArrayDeque: " + animals);
		// Using removeFirst()
		String firstElement1 = animals.removeFirst();
		System.out.println("Removed First Element: " + firstElement1);
		// Using removeLast()
		String lastElement1 = animals.removeLast();
		System.out.println("Removed Last Element: " + lastElement1);
	}
}
```

OUTPUT



```
<terminated> ArrayDequeMethods [Java Application] C:\Users\HP\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1
ArrayDeque: [Cat, Dog, Horse]
Removed Element: Cat
New ArrayDeque: [Dog, Horse]
Removed First Element: Dog
Removed Last Element: Horse
After push method ArrayDeque: [goat, cow, Rabbit]
Head element by element() method: goat
Removed Element: goat
New ArrayDeque: [cow, Rabbit]
Removed First Element: cow
Removed Last Element: Rabbit
```

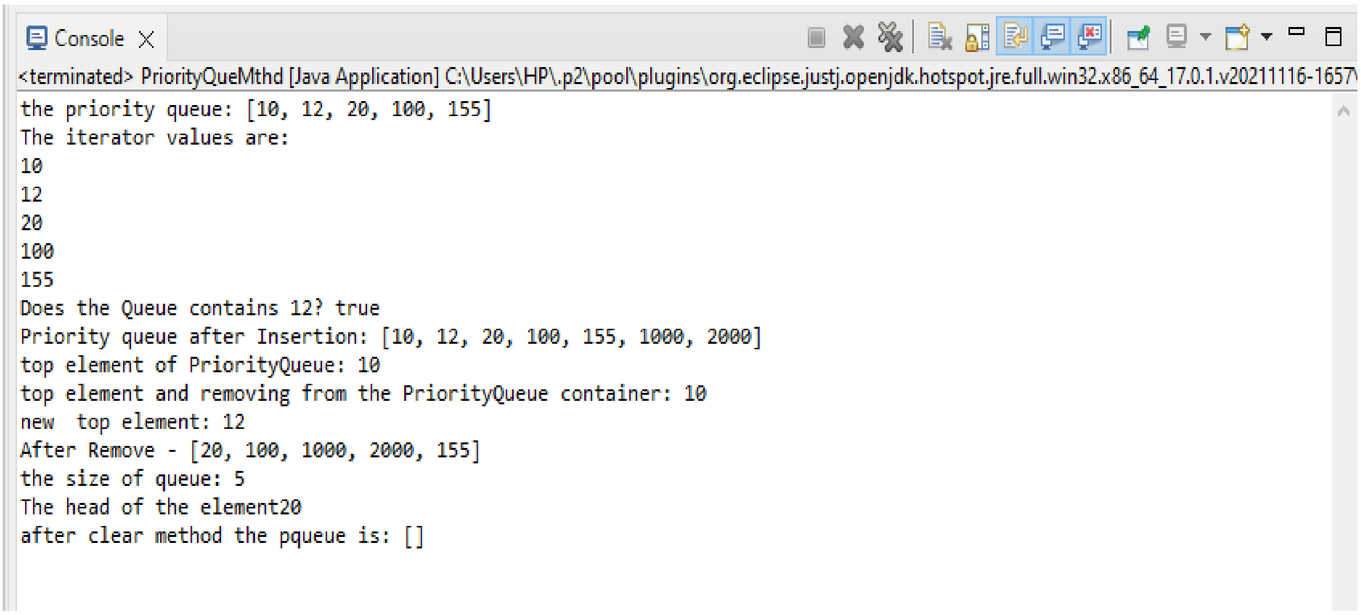
QUESTION 2

Implement a PriorityQueue and use all the methods.

```java
package AssignmentSBA3;
import java.util.Iterator;
import java.util.PriorityQueue;
public class PriorityQueMthd {

        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



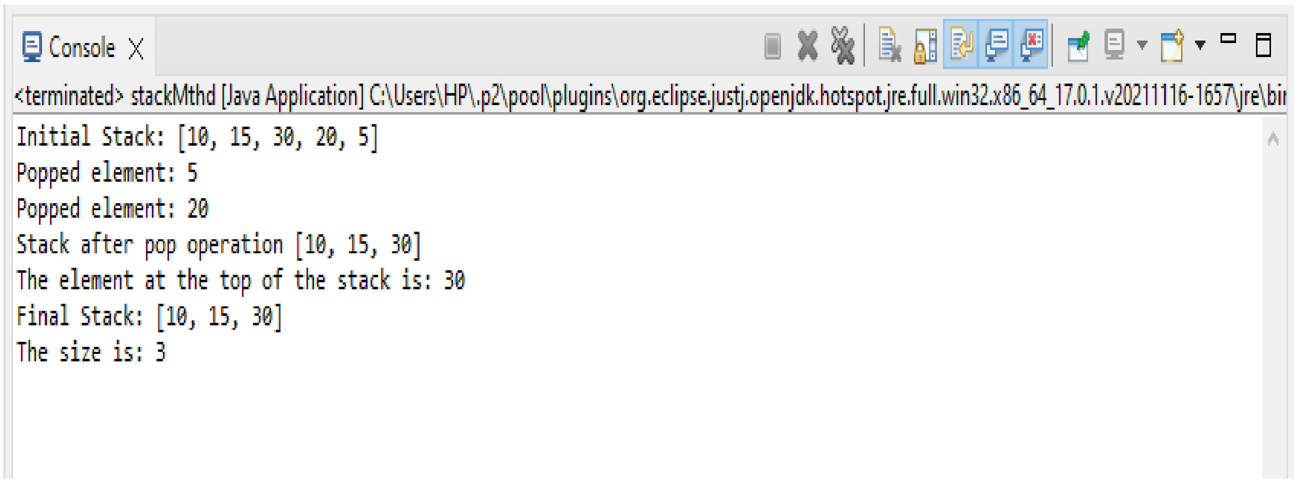
```
<terminated> PriorityQueMthd [Java Application] C:\Users\HP\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657
the priority queue: [10, 12, 20, 100, 155]
The iterator values are:
10
12
20
100
155
Does the Queue contains 12? true
Priority queue after Insertion: [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: []
```

QUESTION 3

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

```
package AssignmentSBA3;
import java.util.Stack;

public class stackMthd {
        public static void main(String[] args) {
                // Creating an empty Stack
          Stack<Integer> stk = new Stack<Integer>();

          // Use add() method to add elements
          stk.push(10);
          stk.push(15);
          stk.push(30);
          stk.push(20);
          stk.push(5);

          // Displaying the Stack
          System.out.println("Initial Stack: " + stk);

          // Removing elements using pop() method
          System.out.println("Popped element: "
                        + stk.pop());
          System.out.println("Popped element: "
                        + stk.pop());

          // Displaying the Stack after pop operation
          System.out.println("Stack after pop operation "
                        + stk);
        // Fetching the element at the head of the Stack
          System.out.println("The element at the top of the"
                        + " stack is: " + stk.peek());

          // Displaying the Stack after the Operation
          System.out.println("Final Stack: " + stk);
        // Displaying the size of stack
          System.out.println("The size is: " + stk.size());

        }

}
```

OUTPUT



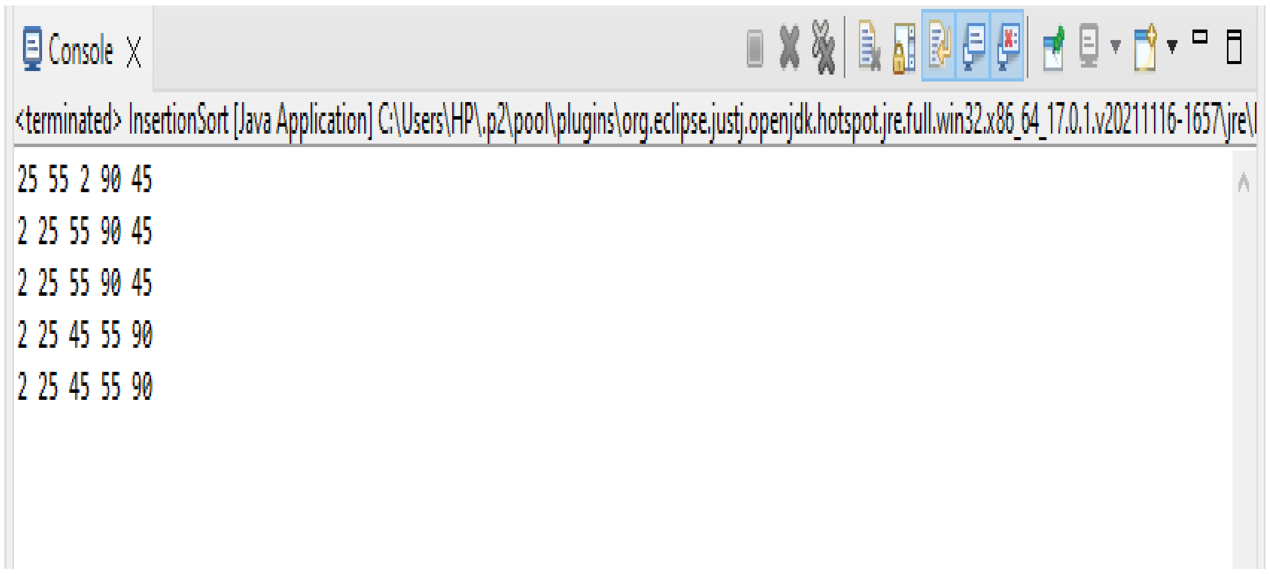
```
<terminated> stackMthd [Java Application] C:\Users\HP\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bi
Initial Stack: [10, 15, 30, 20, 5]
Popped element: 5
Popped element: 20
Stack after pop operation [10, 15, 30]
The element at the top of the stack is: 30
Final Stack: [10, 15, 30]
The size is: 3
```

QUESTION 4

Write a program to implement insertion sort.

```
package Assignment;
public class InsertionSort {

	public static void main(String[] args) {
		int a[]= {25,55,2,90,45};
		int temp,j;
		for(int i=1;i<a.length;i++)
		{
			temp=a[i];
			j=i;
			while(j>0 && a[j-1]>temp)
			{
				a[j]=a[j-1];
				j=j-1;
			}
			a[j]=temp;
			for (int k=0; k<a.length; ++k)
			{
				System.out.print(a[k]+" ");
			}
			System.out.println();
		}
		for(int i=0;i<a.length;i++)
		{
			System.out.print(a[i]+ " ");
		}
	}

}
```

OUTPUT



```
<terminated> InsertionSort [Java Application] C:\Users\HP\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\
25 55 2 90 45
2 25 55 90 45
2 25 55 90 45
2 25 45 55 90
2 25 45 55 90
```