/*

Implement different operations on Priority Queue .i.e. adding element, removing element, size of priority queue, print the queue and top element of queue.

The first line of input contains an integer denoting the no of test cases. For each test case, the first line of input contains an integer denoting the no of queries followed by space separated queries.

A query can be of the following types:

1 x (Adding x to the priority queue and print the queue)

2 (Removing the element from the top of the priority queue and printing that element)

3 (Get the element at the top of the priority queue)

4 (Get the size of the priority queue)

5 (Print the priority queue)

For each test case, the output is according to the query performed and if the priority queue is empty the output is

1<=T<=100

1<=Q<=100

1

8

1 6

1 1

1 7

4

3

2

5

2

6

6 1

7 6 1

3

7

7

6 1

6

1 6 means adding 6 in the queue and printing that, similarly adding 1 and 7 in the

queue and printing the queue i.e. 7 6 1. By 4 it returns the size of the queue i.e 3.

With 3 as input, it returns the element at the top i.e 7. With 2 it removes the top

element i.e 7 from the queue and prints the element i.e. 7. Having 5 as input, it

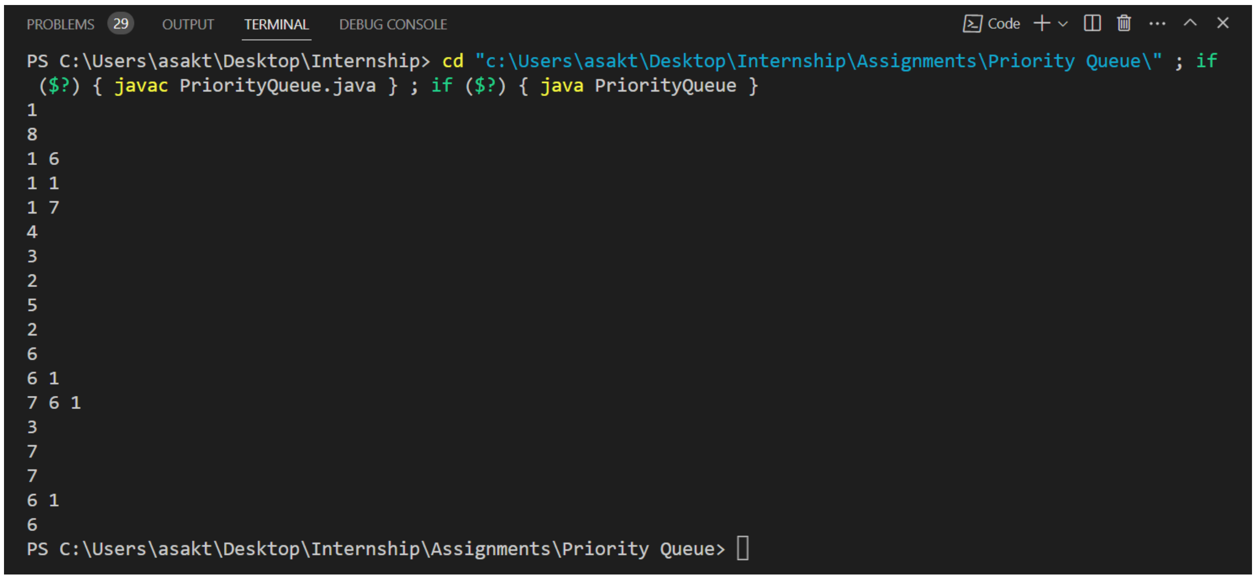
prints the queue i.e. 6 1 and again 2 remove the element and prints that i.e 6.

***/**

```
// Solution
import java.util.Scanner;
class Node{
    int data;
    Node next;
    Node(int data){
        this.data = data;
        this.next = null;
    }
}
public class PriorityQueue {
    static Node head = null;
    static int size = 0;

    public static void insert(int data){
        Node newNode = new Node(data);
        if(head == null){
            head = newNode;
        }
        else{
            Node temp = head;
            while(temp.next != null){
                if(temp.data < newNode.data || temp.next.data <= newNode.data)
                    break;
                temp = temp.next;
            }
            if(temp == head){
                if(temp.data < newNode.data){
                    newNode.next = head;
                    head = newNode;
                }
                else{
                    temp.next = newNode;
                }
            }
            else{
                newNode.next = temp.next;
                temp.next = newNode;
            }
        }
        size++;
    }

    public static void remove(){
        if(head == null){
            System.out.println("Priority Queue is empty");
            return;
        }
        head = head.next;
        size--;
    }

    public static void queueSize(){
        System.out.println(size);
    }

    public static void topElement(){
        if(head == null){
            System.out.println("Priority Queue is empty");
            return;
        }
        System.out.println(head.data);
    }
    public static void display(){
        Node temp = head;
        if(temp == null){
            System.out.println("Priority Queue is empty");
            return;
        }
        while(temp != null){
            System.out.print(temp.data + " ");
            temp = temp.next;
        }
        System.out.println();
    }
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        int testCase = scan.nextInt(); // getting no of test cases

        while(testCase-- > 0){
            int noOfQuery = scan.nextInt(); // getting no of queries
            while(noOfQuery-- > 0){
                int choice = scan.nextInt(); // getting the choice
                switch(choice){
                    case 1:
                        int x = scan.nextInt(); // getting the data to insert and
```

```
                  insert(x);  // displaying the queue elements
                  display();
                  break;
               case 2:
                  topElement(); // printing top element and
                  remove();     // removing the top element
                  break;
               case 3:
                  topElement(); // printing top element
                  break;
               case 4:
                  queueSize(); // printing queue size
                  break;
               case 5:
                  display(); // displaying the queue elements
                  break;
               default:
                  System.out.println("Please enter a valid choice!!!");
                  noOfQuery++;
            }
         }
      }
      scan.close();
   }
}
```

// Output



```
PS C:\Users\asakt\Desktop\Internship> cd "c:\Users\asakt\Desktop\Internship\Assignments\Priority Queue\" ; if
 ($?) { javac PriorityQueue.java } ; if ($?) { java PriorityQueue }
1
8
1 6
1 1
1 7
4
3
2
5
2
6
6 1
7 6 1
3
7
7
6 1
6
PS C:\Users\asakt\Desktop\Internship\Assignments\Priority Queue> 
```