The Maximum Subarray

by sh4d0wkn1ght

Given an array $A = \{a_1, a_2, \ldots, a_N\}$ of N elements, find the maximum possible sum of a

1. Contiguous subarray
2. Non-contiguous (not necessarily contiguous) subarray.

Empty subarrays/subsequences should not be considered.

Input Format

First line of the input has an integer T. T cases follow.
Each test case begins with an integer N. In the next line, N integers follow representing the elements of array A.

Constraints:

- $1 \leq T \leq 10$
- $1 \leq N \leq 10^5$
- $-10^4 \leq a_i \leq 10^4$

The subarray and subsequences you consider should have at least one element.

Output Format

Two, space separated, integers denoting the maximum contiguous and non-contiguous subarray. At least one integer should be selected and put into the subarrays (this may be required in cases where all elements are negative).

Sample Input

```
2
4
1 2 3 4
6
2 -1 2 3 4 -5
```

Sample Output

```
10 10
10 11
```

Explanation

In the first case:
The max sum for both contiguous and non-contiguous elements is the sum of ALL the elements (as they are all positive).

In the second case:
[2 -1 2 3 4] --> This forms the contiguous sub-array with the maximum sum.
For the max sum of a not-necessarily-contiguous group of elements, simply add all the positive elements.

Related Topics

Dynamic Programming Basics

Submissions: 26388
Max Score: 50
Difficulty: Easy

More

```java
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class should be named Solution. */
        Scanner read = new Scanner(System.in);
        int t = read.nextInt();
        for(int i=0;i<t;i++){
            int n = read.nextInt();
            int[] arr = new int[n];
            for(int j=0;j<n;j++){
                arr[j]=read.nextInt();
            }
            int maxSumSoFar=Integer.MIN_VALUE;
            int maxEnding = 0;
            int nonContSum=0;
            for(int x: arr){
                maxEnding = Math.max(x, maxEnding+x);
                maxSumSoFar = Math.max(maxEnding, maxSumSoFar);


            }
                System.out.print(maxSumSoFar);

            //2) For max non-continuous sub array, order doesn't matter
            Arrays.sort(arr);
            int sum = 0;

            //if there is none positive value in entire array
            if(arr[arr.length-1] <= 0){
                sum = arr[arr.length - 1];
                //System.out.println("neg case: "+sum);
            //accumulate all positive values in entire array
            }else{
                for(int x : arr){
                    if(x > 0)
                        sum += x;
                }
            }
            System.out.println(" " + sum);        }
    }
}
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code