Practice Compete Jobs Rank Leaderboard gadhiya

Dashboard > Algorithms > Dynamic Programming > The Maximum Subarray

Badge Progress ★ ★ ★ ★

Points: 4727.88 Rank: 491

The Maximum Subarray

by sh4d0wkn1ght

Problem Submissions Leaderboard Discussions Editorial Tutorial

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.

Submissions: 49267
Max Score: 50
Difficulty: Medium

Rate This Challenge:
☆☆☆☆☆

Need Help?
Dynamic Programming Basics

More

Current Buffer (saved locally, editable)

Java 8

```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) throws IOException{

        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int tst = Integer.parseInt(br.readLine());

        for(int i = 0 ; i < tst ; i++){

            int N = Integer.parseInt(br.readLine());

            String line = br.readLine();
            String[] numbers = line.split("\\s");

            long[] arr = new long[N];
            long[] out = new long[N];

            long seq = 0;
            long nonSeq = 0;

            for(int j = 0 ; j < N ; j++){
                arr[j] = Long.parseLong(numbers[j]);

               if(arr[j] >= 0){
                   nonSeq += arr[j];
               }

                if(j == 0){
                    out[j] = arr[j];
                    continue;
                }

                if(arr[j] + out[j - 1] < arr[j]){
                    out[j] = arr[j];
                }
                else{
                    out[j] = arr[j] + out[j - 1];
                }
            }

            Arrays.sort(out);

            seq = out[out.length - 1];

            if(seq <= 0){
                nonSeq = seq;
            }

            System.out.println(seq + " " + nonSeq);

        }

    }
}
```

Line: 1 Col: 1

Upload Code as File □ Test against custom input

Run Code Submit Code

Join us on IRC at #hackerrank on freenode for hugs or bugs.

Contest Calendar | Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature