303. Range Sum Query - Immutable

Easy 3K 1.8K

Companies

Given an integer array `nums`, handle multiple queries of the following type:

1. Calculate the **sum** of the elements of `nums` between indices `left` and `right` **inclusive** where `left <= right`.

Implement the `NumArray` class:

- `NumArray(int[] nums)` Initializes the object with the integer array `nums`.
- `int sumRange(int left, int right)` Returns the **sum** of the elements of `nums` between indices `left` and `right` **inclusive** (i.e. `nums[left] + nums[left + 1] + ... + nums[right]`).

Example 1:

```
Input
["NumArray", "sumRange", "sumRange", "sumRange"]
[[[-2, 0, 3, -5, 2, -1]], [0, 2], [2, 5], [0, 5]]
Output
[null, 1, -1, -3]

Explanation
NumArray numArray = new NumArray([-2, 0, 3, -5, 2, -1]);
numArray.sumRange(0, 2); // return (-2) + 0 + 3 = 1
numArray.sumRange(2, 5); // return 3 + (-5) + 2 + (-1) = -1
numArray.sumRange(0, 5); // return (-2) + 0 + 3 + (-5) + 2 + (-1) = -3
```

Constraints:

- `1 <= nums.length <= 10^4`
- `-10^5 <= nums[i] <= 10^5`
- `0 <= left <= right < nums.length`
- At most `10^4` calls will be made to `sumRange`.

Accepted 464.3K Submissions 761.3K Acceptance Rate 61.0%

Discussion (9)

Similar Questions

Related Topics

Copyright © 2023 LeetCode All rights reserved

Parenthesis CSG - 1140

Bobo has a balanced parenthesis sequence P=p1 p2...pn of length n and q questions.

The i-th question is whether P remains balanced after p_{ai} and p_{bi} swapped. Note that questions are individual so that they have no affect on others.

Parenthesis sequence S is balanced if and only if:

1. S is empty;
2. or there exists balanced parenthesis sequence A,B such that S=AB;
3. or there exists balanced parenthesis sequence S' such that S=(S').

Input

The input contains at most 30 sets. For each set:

The first line contains two integers n,q ($2 \le n \le 10^5, 1 \le q \le 10^5$).

The second line contains n characters p1 p2...pn.

The i-th of the last q lines contains 2 integers ai,bi ($1 \le ai, bi \le n, ai \ne bi$).

Output

For each question, output "Yes" if P remains balanced, or "No" otherwise.

Sample

Input	copy	Output	copy
4 2 (()) 1 3 2 3 2 1 () 1 2		No Yes No	

Sticks Problem POJ - 2452

Xuanxuan has n sticks of different length. One day, she puts all her sticks in a line, represented by S1, S2, S3, ...Sn. After measuring the length of each stick Sk (1 <= k <= n), she finds that for some sticks Si and Sj (1<= i < j <= n), each stick placed between Si and Sj is longer than Si but shorter than Sj.

Now given the length of S1, S2, S3, ...Sn, you are required to find the maximum value j - i.

Input

The input contains multiple test cases. Each case contains two lines.
Line 1: a single integer n (n <= 50000), indicating the number of sticks.
Line 2: n different positive integers (not larger than 100000), indicating the length of each stick in order.

Output

Output the maximum value j - i in a single line. If there is no such i and j, just output -1.

Sample

Input	copy	Output	copy
4 5 4 3 6 4 6 5 4 3		1 -1	

Worst Weather Ever

Kattis - worstweather

“Man, this year has the worst weather ever!”, David said as he sat crouched in the small cave where we had sought shelter from yet another sudden rainstorm.

“Nuh-uh!”, Diana immediately replied in her traditional know-it-all manner.

“Is too!”, David countered cunningly.

Terrific. Not only were we stuck in this cave, now we would have to listen to those two nagging for at least an hour. It was time to cut this discussion short.

“Big nuh-uh. In fact, 93 years ago it had already rained five times as much by this time of year.”

“Duh”, David capitulated, “so it’s the worst weather in 93 years then.”

“Nuh-uh, this is actually the worst weather in 23 years.”, Diana again broke in.

“Yeah, well, whatever”, David sighed, “Who cares anyway?”.

Well, dear contestants, you care, don’t you?

The Problem

Your task is to, given information about the amount of rain during different years in the history of the universe, and a series of statements in the form “Year X had the most rain since year Y”, determine whether these are true, might be true, or are false. We say that such a statement is true if:

- The amount of rain during these two years and all years between them is known.
- It rained at most as much during year X as it did during year Y.
- For every year Z satisfying $Y < Z < X$, the amount of rain during year Z was less than the amount of rain during year X.

We say that such a statement might be true if there is an assignment of amounts of rain

to years for which there is no information, such that the statement becomes true. We say that the statement is false otherwise.

Input

The input will consist of several test cases, each consisting of two parts.

The first part begins with an integer $1 \leq n \leq 50\,000$, indicating the number of different years for which there is information. Next follow n lines. The ith of these contains two integers $-10^9 \leq y_i \leq 10^9$ and $1 \leq r_i \leq 10^9$ indicating that there was r_i millilitres of rain during year y_i (note that the amount of rain during a year can be any nonnegative integer, the limitation on r_i is just a limitation on the input). You may assume that $y_i < y_{i+1}$ for $1 \leq i < n$.

The second part of a test case starts with an integer $1 \leq m \leq 10\,000$, indicating the number of queries to process. The following m lines each contain two integers $-10^9 \leq Y < X \leq 10^9$ indicating two years.

There is a blank line between test cases. The input is terminated by a case where $n = 0$ and $m = 0$. This case should not be processed.

Technical note: Due to the size of the input, the use of cin/cout in C++ might be too slow in this problem. Use scanf/printf instead. In Java, make sure that both input and output is buffered.

Output

There should be m lines of output for each test case, corresponding to the m queries. Queries should be answered with "`true`" if the statement is true, "`maybe`" if the statement might be true, and "`false`" if the statement is false.

Separate the output of two different test cases by a blank line.

Sample 1

Input	copy	Output	copy
4 2002 4920 2003 5901 2004 2832 2005 3890 2 2002 2005 2003 2005 3 1985 5782 1995 3048 2005 4890 2 1985 2005 2005 2015 0 0		false true maybe maybe	

Surveillance

Kattis - surveillance

The International Corporation for Protection and Control (ICPC) develops efficient technology for, well, protection and control. Naturally, they are keen to have their own headquarters protected and controlled. Viewed from above, the headquarters building has the shape of a convex polygon. There are several suitable places around it where cameras can be installed to monitor the building. Each camera covers a certain range of the polygon sides (building walls), depending on its position. ICPC wants to minimize the number of cameras needed to cover the whole building.

Input

The input consists of a single test case. Its first line contains two integers n and k ($3 \leq n \leq 10^6$ and $1 \leq k \leq 10^6$), where n is the number of walls and k is the number of possible places for installing cameras. Each of the remaining k lines contains two integers a_i and b_i ($1 \leq a_i, b_i \leq n$). These integers specify which walls a camera at the i^{th} place would cover. If $a_i \leq b_i$ then the camera covers each wall j such that $a_i \leq j \leq b_i$. If $a_i > b_i$ then the camera covers each wall j such that $a_i \leq j \leq n$ or $1 \leq j \leq b_i$.

Output

Display the minimal number of cameras that suffice to cover each wall of the building. The ranges covered by two cameras may overlap. If the building cannot be covered, display `impossible` instead.

Sample 1

Input	Output
100 7 1 50 50 70 70 90 90 40 20 60 60 80 80 20	3

Sample 2

Input	copy	Output	copy
8 2 8 3 5 7		impossible	

Sample 3

Input	copy	Output	copy
8 2 8 4 5 7		2	

493. Reverse Pairs

Hint

Hard 5.5K 237

Companies

Given an integer array `nums`, return *the number of **reverse pairs** in the array*.

A **reverse pair** is a pair `(i, j)` where:

- `0 <= i < j < nums.length` and
- `nums[i] > 2 * nums[j]`.

Example 1:

```
Input: nums = [1,3,2,3,1]
Output: 2
Explanation: The reverse pairs are:
(1, 4) --> nums[1] = 3, nums[4] = 1, 3 > 2 * 1
(3, 4) --> nums[3] = 3, nums[4] = 1, 3 > 2 * 1
```

Example 2:

```
Input: nums = [2,4,3,5,1]
Output: 3
Explanation: The reverse pairs are:
(1, 4) --> nums[1] = 4, nums[4] = 1, 4 > 2 * 1
(2, 4) --> nums[2] = 3, nums[4] = 1, 3 > 2 * 1
(3, 4) --> nums[3] = 5, nums[4] = 1, 5 > 2 * 1
```

Constraints:

- $1 <= nums.length <= 5 * 10^4$
- $-2^{31} <= nums[i] <= 2^{31} - 1$

Accepted 143.1K Submissions 468.9K Acceptance Rate 30.5%

Discussion (22)

Similar Questions

Related Topics

Copyright © 2023 LeetCode All rights reserved

1395. Count Number of Teams

Hint

Medium 2.6K 183

Companies

There are `n` soldiers standing in a line. Each soldier is assigned a **unique** `rating` value.

You have to form a team of 3 soldiers amongst them under the following rules:

- Choose 3 soldiers with index (`i`, `j`, `k`) with rating (`rating[i]`, `rating[j]`, `rating[k]`).
- A team is valid if: (`rating[i] < rating[j] < rating[k]`) or (`rating[i] > rating[j] > rating[k]`) where (`0 <= i < j < k < n`).

Return the number of teams you can form given the conditions. (soldiers can be part of multiple teams).

Example 1:

```
Input: rating = [2,5,3,4,1]
Output: 3
Explanation: We can form three teams given the conditions. (2,3,4), (5,4,1), (5,3,1).
```

Example 2:

```
Input: rating = [2,1,3]
Output: 0
Explanation: We can't form any team given the conditions.
```

Example 3:

```
Input: rating = [1,2,3,4]
Output: 4
```

Constraints:

- `n == rating.length`
- `3 <= n <= 1000`
- `1 <= rating[i] <= 10^5`
- All the integers in `rating` are **unique**.

Accepted 104.2K Submissions 157K Acceptance Rate 66.4%

Discussion (9)

Related Topics

Copyright © 2023 LeetCode All rights reserved