B. Little Elephant and Sorting

time limit per test: 0.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The Little Elephant loves sortings.

He has an array a consisting of n integers. Let's number the array elements from 1 to n, then the i-th element will be denoted as a_i. The Little Elephant can make one move to choose an arbitrary pair of integers l and r $(1 \le l \le r \le n)$ and increase a_i by 1 for all i such that $l \le i \le r$.

Help the Little Elephant find the minimum number of moves he needs to convert array a to an arbitrary array sorted in the non-decreasing order. Array a, consisting of n elements, is sorted in the non-decreasing order if for any i $(1 \le i < n)$ $a_i \le a_{i+1}$ holds.

Input

The first line contains a single integer n $(1 \le n \le 10^5)$ — the size of array a. The next line contains n integers, separated by single spaces — array a $(1 \le a_i \le 10^9)$. The array elements are listed in the line in the order of their index's increasing.

Output

In a single line print a single integer — the answer to the problem.

Please, do not use the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

Examples

input
3 1 2 3

output
0

input
3 3 2 1

output
2

input
4 7 4 1 47

output
6

Note

In the first sample the array is already sorted in the non-decreasing order, so the answer is 0.

In the second sample you need to perform two operations: first increase numbers from second to third (after that the array will be: `[3, 3, 2]`), and second increase only the last element (the array will be: `[3, 3, 3]`).

In the third sample you should make at least 6 steps. The possible sequence of the operations is: `(2; 3), (2; 3), (2; 3), (3; 3), (3; 3), (3; 3)`. After that the array converts to `[7, 7, 7, 47]`.