Tejas on the Rooftop

time limit per test case: 1 second
memory limit per test case: 1 gigabyte

IIIT's Research Street has N buildings with distinct heights denoted by the array $H = [h_1, h_2, \ldots, h_N]$ of length N ($i \neq j \Rightarrow h_i \neq h_j$). Say Tejas is on the roof of building i, then he can go to roof of building j ($j > i$) with a ladder of length $d(i, j) = j - i + 1$ only if $h_{i+1}, \ldots, h_{j-1} < \min(h_i, h_j)$, i.e. if all the buildings between i and j have height less than both of these buildings.

Tejas is busy playing Elden Ring. So he asks you to compute the **sum of distances** $d(i, j)$ between all pairs (i, j) ($i < j$) of locations such that it is possible to go from the roof of building i to the roof of building j.

Constraints

Subtask 1 (30 marks)

$1 \leq N \leq 10^3$
$1 \leq h_i \leq 10^9$
$i \neq j \Rightarrow h_i \neq h_j$, *OR*, the elements of the array are distinct.

Subtask 2 (70 marks)

$1 \leq N \leq 10^5$
$1 \leq h_i \leq 10^9$
$i \neq j \Rightarrow h_i \neq h_j$, *OR*, the elements of the array are distinct.

Input

The first line contains the integer N, which denotes the size of input array.
The next line contains N integers denoting the heights of the buildings.

Output

Output contains a single integer denoting the sum of distances of all pairs of locations of buildings such that Tejas can go from the roof of the first building to the other.
To avoid overflow, store the output integer in a *long long* variable.

Sample test cases

Test case 1

Input

7
4 3 1 2 5 6 7

Output

24

Explanation

The pairs of required locations in this example are:
$(1, 2), (1, 5), (2, 3), (2, 4), (2, 5), (3, 4), (4, 5), (5, 6), (6, 7)$
And their respective distances are: $2, 5, 2, 3, 4, 2, 2, 2, 2$