Watering Flowers

Tim Huisman

Problem

You are put in charge of plucking all flowers in an enormously long garden. These flowers are exotic and need special care, so obviously you cannot just pluck them all at once immediately. You have received specific instructions on how to handle them.

In the garden, there is a row of n flowers arranged next to each other. These flowers each have a number assigned to them based on their position, with 1 on the far left and n on the far right.
Given is a sequence a, which is a permutation from 1 to n. That is, all integers from 1 up to and including n appear in a exactly once, in any order. On day i, you do the following things in order:

- You pluck the flower with number a_i.
- You give all the remaining flowers *to the left* of the plucked flower one cup of water.

This continues up until day n, after which all flowers have been plucked.

After n days, how many cups of water will you have given in total?

Note: Because this number can be quite large, you should give the answer modulo $10^9 + 7$.

Input

First line: An integer n ($1 \leq n \leq 10^6$) — The amount of flowers in the garden.
Next line: n space-separated integers $a_1, a_2, ..., a_n$ — The order in which to pluck the flowers.

Output

One integer — The amount of cups of water used after day n, modulo $10^9 + 7$.

Limitations

Time limit: 3 seconds
Memory limit: 2 GB

Examples

Example 1

```
Input:
4
4 2 3 1

Output:
5
```

The first day, you pluck flower **4**, and give a cup of water to flowers **1**, **2** and **3**.
The next day, you pluck flower **2**, and only give a cup of water to flower **1** (not flower 3, as it is not on the left side of flower 2).
The next day, you pluck flower **3**, and only give a cup of water to flower **1** (not flower 2, as that one has already been plucked the day before).
The last day, you pluck flower **1**, and don't give a cup of water to any flower.

$3 + 1 + 1 + 0 \mod (10^9 + 7) = 5$ cups of water have been given.

Example 2

```
Input:
6
1 2 3 4 5 6

Output:
0
```

Example 3

```
Input:
6
6 5 4 3 2 1

Output:
15
```

Example 4

```
Input:
10
4 5 8 7 2 10 3 9 1 6

Output:
23
```

Network Reach

Eames Trinh

In the real world, it is often important to be able to find the "centers" of things so that it can be made sure that at least all of them are within reasonable range of some item(s). There are many ways to do this. Often most situations care about maximizing overall efficiency (i.e. placing hospitals in population clusters so that it can serve the most people the fastest and while not caring about being close to rural residents).
However, making sure that *all* nodes are within a certain range is also important. For example, consider a set of IoT devices that wirelessly communicates to a single router. For this to be possible, all devices need to be in range of this router.
It is important to minimize the amount of energy used by constantly projecting a very powerful signal. One way to do this is to minimize the strength of the signal so that no more energy is used than necessary. We assume two things: the signal propagates outwards from the router in a spherical wave, and we do not care about the signal strength at each node if they can communicate with the router.
Write an algorithm that takes a set of 2D coordinates representing the locations of the IoT devices and returns the coordinates of the router so that the maximum signal distance is minimized. The algorithm should be efficient and able to handle large numbers of devices (e.g., thousands or more).

The input is given as a list of (x, y) coordinates, preceded by the number of points.

Example input:

```
5
3 2
6 8
2 4
7 1
5 -2
```

For the output, return the coordinates as two values, *separated by a space* and *rounded to one decimal place*.

Example output:

```
5.5 3.0
```

Hint 1: Consider writing out some calculations before jumping into the code.
Hint 2: Consider using a recursive formulation. Can you think of base cases?