Problem D. Prof. Slim

Time limit 1000 ms
Mem limit 262144 kB

One day Prof. Slim decided to leave the kingdom of the GUC to join the kingdom of the GIU. He was given an easy online assessment to solve before joining the GIU. Citizens of the GUC were ~~happy~~ sad to see the prof leaving, so they decided to hack into the system and change the online assessment into a harder one so that he stays at the GUC. After a long argument, they decided to change it into the following problem.

Given an array of n integers $a_1, a_2, \ldots, a_n$, **where** $a_i \neq 0$, check if you can make this array sorted by using the following operation any number of times (possibly zero). An array is sorted if its elements are arranged in a non-decreasing order.

1. select two indices i and j ($1 \leq i, j \leq n$) such that a_i and a_j have **different signs**. In other words, one must be positive and one must be negative.
2. swap the **signs** of a_i and a_j. For example if you select $a_i = 3$ and $a_j = -2$, then they will change to $a_i = -3$ and $a_j = 2$.

Prof. Slim saw that the problem is still too easy and isn't worth his time, so he decided to give it to you to solve.

Input

The first line contains a single integer t ($1 \leq t \leq 10^4$) — the number of test cases. Then t test cases follow.

The first line of each test case contains a single integer n ($1 \leq n \leq 10^5$) — the length of the array a.

The next line contain n integers $a_1, a_2, \ldots, a_n$ ($-10^9 \leq a_i \leq 10^9$, $a_i \neq 0$) separated by spaces describing elements of the array a.

It is guaranteed that the sum of n over all test cases doesn't exceed 10^5.

Output

For each test case, print "`YES`" if the array can be sorted in the non-decreasing order, otherwise print "`NO`". You can print each letter in any case (upper or lower).

Sample 1

Input	Output

4
7
7 3 2 -11 -13 -17 -23
6
4 10 25 47 71 96
6
71 -35 7 -4 -11 -25
6
-45 9 -48 -67 -55 7
	NO
	YES
	YES
	NO

Note

In the first test case, there is no way to make the array sorted using the operation any number of times.

In the second test case, the array is already sorted.

In the third test case, we can swap the sign of the 1-st element with the sign of the 5-th element, and the sign of the 3-rd element with the sign of the 6-th element, this way the array will be sorted.

In the fourth test case, there is no way to make the array sorted using the operation any number of times.