

Compare Version Numbers

Given two version numbers, version1 and version2, compare them.

Version numbers consist of **one or more revisions** joined by a dot '.'. Each revision consists of **digits** and may contain leading **zeros**. Every revision contains **at least one character**. Revisions are **0-indexed from left to right**, with the leftmost revision being revision 0, the next revision being revision 1, and so on. For example 2.5.33 and 0.1 are valid version numbers.

To compare version numbers, compare their revisions in **left-to-right order**. Revisions are compared using their **integer value ignoring any leading zeros**. This means that revisions 1 and 001 are considered **equal**. If a version number does not specify a revision at an index, then **treat the revision as 0**. For example, version 1.0 is less than version 1.1 because their revision 0s are the same, but their revision 1s are 0 and 1 respectively, and $0 < 1$.

Return the following:

- If version1 < version2, return -1.
- If version1 > version2, return 1.
- Otherwise, return 0.

Example 1:

Input: version1 = "1.01", version2 = "1.001"

Output: 0

Explanation: Ignoring leading zeroes, both "01" and "001" represent the same integer "1".

Example 2:

Input: version1 = "1.0", version2 = "1.0.0"

Output: 0

Explanation: version1 does not specify revision 2, which means it is treated as "0".

Example 3:

Input: version1 = "0.1", version2 = "1.1"

Output: -1

Explanation: version1's revision 0 is "0", while version2's revision 0 is "1". $0 < 1$, so version1 < version2.

Constraints:

- $1 \leq \text{version1.length}, \text{version2.length} \leq 500$
- version1 and version2 only contain digits and '.'.
- version1 and version2 **are valid version numbers**.
- All the given revisions in version1 and version2 can be stored in a **32-bit integer**.