1152. Analyze User Website Visit Pattern

Medium 249 2234 Add to List Share

You are given two string arrays `username` and `website` and an integer array `timestamp`. All the given arrays are of the same length and the tuple `[username[i], website[i], timestamp[i]]` indicates that the user `username[i]` visited the website `website[i]` at time `timestamp[i]`.

A **pattern** is a list of three websites (not necessarily distinct).

- For example, `["home", "away", "love"]`, `["leetcode", "love", "leetcode"]`, and `["luffy", "luffy", "luffy"]` are all patterns.

The **score** of a **pattern** is the number of users that visited all the websites in the pattern in the same order they appeared in the pattern.

- For example, if the pattern is `["home", "away", "love"]`, the score is the number of users `x` such that `x` visited `"home"` then visited `"away"` and visited `"love"` after that.
- Similarly, if the pattern is `["leetcode", "love", "leetcode"]`, the score is the number of users `x` such that `x` visited `"leetcode"` then visited `"love"` and visited `"leetcode"` **one more time** after that.
- Also, if the pattern is `["luffy", "luffy", "luffy"]`, the score is the number of users `x` such that `x` visited `"luffy"` three different times at different timestamps.

Return *the **pattern** with the largest **score***. If there is more than one pattern with the same largest score, return the lexicographically smallest such pattern.

Example 1:

```
Input: username = ["joe","joe","joe","james","james","james","james","mary","mary","mary"],
timestamp = [1,2,3,4,5,6,7,8,9,10], website =
["home","about","career","home","cart","maps","home","home","about","career"]
Output: ["home","about","career"]
Explanation: The tuples in this example are:
["joe","home",1],["joe","about",2],["joe","career",3],["james","home",4],["james","cart",5],
["james","maps",6],["james","home",7],["mary","home",8],["mary","about",9], and
["mary","career",10].
The pattern ("home", "about", "career") has score 2 (joe and mary).
The pattern ("home", "cart", "maps") has score 1 (james).
The pattern ("home", "cart", "home") has score 1 (james).
The pattern ("home", "maps", "home") has score 1 (james).
The pattern ("cart", "maps", "home") has score 1 (james).
The pattern ("home", "home", "home") has score 0 (no user visited home 3 times).
```

Example 2:

```
Input: username = ["ua","ua","ua","ub","ub","ub"], timestamp = [1,2,3,4,5,6], website =
["a","b","a","a","b","c"]
Output: ["a","b","a"]
```

Constraints:

- `3 <= username.length <= 50`
- `1 <= username[i].length <= 10`
- `timestamp.length == username.length`
- `1 <= timestamp[i] <= 10^9`
- `website.length == username.length`
- `1 <= website[i].length <= 10`
- `username[i]` and `website[i]` consist of lowercase English letters.
- It is guaranteed that there is at least one user who visited at least three websites.
- All the tuples `[username[i], timestamp[i], website[i]]` are **unique**.

Accepted 48,116 | Submissions 111,218

Seen this question in a real interview before? Yes No

Companies

0 ~ 6 months 6 months ~ 1 year 1 year ~ 2 years

Amazon | 33

Related Topics

Array Hash Table Sorting

Hide Hint 1

Let's find for every user separately the websites he visited.

Hide Hint 2

Consider all possible 3-sequences, find the number of distinct users who visited each of them.

Hide Hint 3

How to check if some user visited some 3-sequence ?

Hide Hint 4

Store for every user all the 3-sequence he visited.

Java Autocomplete

```java
class Solution {
    public List<String> mostVisitedPattern(String[] username, int[] timestamp, String[] website) {

    }
}
```