Problem 2901: Longest Unequal Adjacent Groups Subsequence II

Problem Information

Difficulty: Medium
Acceptance Rate: 51.44%
Paid Only: No
Tags: Array, String, Dynamic Programming

Problem Description

You are given a string array `words`, and an array `groups`, both arrays having length `n`.

The **hamming distance** between two strings of equal length is the number of positions at which the corresponding characters are **different**.

You need to select the **longest** subsequence from an array of indices `[0, 1, ..., n - 1]`, such that for the subsequence denoted as `[i0, i1, ..., ik-1]` having length `k`, the following holds:

* For **adjacent** indices in the subsequence, their corresponding groups are **unequal** , i.e., `groups[ij] != groups[ij+1]`, for each `j` where `0 < j + 1 < k`. * `words[ij]` and `words[ij+1]` are **equal** in length, and the **hamming distance** between them is `1`, where `0 < j + 1 < k`, for all indices in the subsequence.

Return _a string array containing the words corresponding to the indices**(in order)** in the selected subsequence_. If there are multiple answers, return _any of them_.

Note: strings in `words` may be **unequal** in length.

Example 1:

Input: words = ["bab","dab","cab"], groups = [1,2,2]

Output:["bab","cab"]

Explanation: A subsequence that can be selected is `[0,2]`.

* `groups[0] != groups[2]` * `words[0].length == words[2].length`, and the hamming distance between them is 1.

So, a valid answer is `[words[0],words[2]] = ["bab","cab"]`.

Another subsequence that can be selected is `[0,1]`.

* `groups[0] != groups[1]` * `words[0].length == words[1].length`, and the hamming distance between them is `1`.

So, another valid answer is `[words[0],words[1]] = ["bab","dab"]`.

It can be shown that the length of the longest subsequence of indices that satisfies the conditions is `2`.

Example 2:

Input: words = ["a","b","c","d"], groups = [1,2,3,4]

Output:["a","b","c","d"]

Explanation: We can select the subsequence `[0,1,2,3]`.

It satisfies both conditions.

Hence, the answer is `[words[0],words[1],words[2],words[3]] = ["a","b","c","d"]`.

It has the longest length among all subsequences of indices that satisfy the conditions.

Hence, it is the only answer.

Constraints:

* `1 <= n == words.length == groups.length <= 1000` * `1 <= words[i].length <= 10` * `1 <= groups[i] <= n` * `words` consists of **distinct** strings. * `words[i]` consists of lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
vector<string> getWordsInLongestSubsequence(vector<string>& words,
vector<int>& groups) {

}
};
```

Java:

```
class Solution {
public List<String> getWordsInLongestSubsequence(String[] words, int[]
groups) {

}
}
```

Python3:

```
class Solution:
def getWordsInLongestSubsequence(self, words: List[str], groups: List[int])
-> List[str]:
```