Problem 1048: Longest String Chain

Problem Information

Difficulty: Medium
Acceptance Rate: 62.58%
Paid Only: No
Tags: Array, Hash Table, Two Pointers, String, Dynamic Programming, Sorting

Problem Description

You are given an array of `words` where each word consists of lowercase English letters.

`wordA` is a **predecessor** of `wordB` if and only if we can insert **exactly one** letter anywhere in `wordA` **without changing the order of the other characters** to make it equal to `wordB`.

* For example, `"abc"` is a **predecessor** of `"ab _a_ c"`, while `"cba"` is not a **predecessor** of `"bcad"`.

A **word chain** __ is a sequence of words `[word1, word2, ..., wordk]` with `k >= 1`, where `word1` is a **predecessor** of `word2`, `word2` is a **predecessor** of `word3`, and so on. A single word is trivially a **word chain** with `k == 1`.

Return _the**length** of the **longest possible word chain** with words chosen from the given list of _`words`.

Example 1:

Input: words = ["a","b","ba","bca","bda","bdca"] **Output:** 4 **Explanation** : One of the longest word chains is ["a","_b_ a","b _d_ a","bd _c_ a"].

Example 2:

Input: words = ["xbc","pcxbcf","xb","cxbc","pcxbc"] **Output:** 5 **Explanation:** All the words can be put in a word chain ["xb", "xb _c_ ", "_c_ xbc", "_p_ cxbc", "pcxbc _f_ "].

Example 3:

Input: words = ["abcd","dbqca"] **Output:** 1 **Explanation:** The trivial word chain ["abcd"] is one of the longest word chains. ["abcd","dbqca"] is not a valid word chain because the ordering of the letters is changed.

Constraints:

* `1 <= words.length <= 1000` * `1 <= words[i].length <= 16` * `words[i]` only consists of lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
int longestStrChain(vector<string>& words) {

}
};
```

Java:

```
class Solution {
public int longestStrChain(String[] words) {

}
}
```

Python3:

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