Problem 2131: Longest Palindrome by Concatenating Two Letter Words

Problem Information

Difficulty: Medium
Acceptance Rate: 53.61%
Paid Only: No
Tags: Array, Hash Table, String, Greedy, Counting

Problem Description

You are given an array of strings `words`. Each element of `words` consists of **two** lowercase English letters.

Create the **longest possible palindrome** by selecting some elements from `words` and concatenating them in **any order**. Each element can be selected **at most once**.

Return _the**length** of the longest palindrome that you can create_. If it is impossible to create any palindrome, return `0`.

A **palindrome** is a string that reads the same forward and backward.

Example 1:

Input: words = ["lc","cl","gg"] **Output:** 6 **Explanation:** One longest palindrome is "lc" + "gg" + "cl" = "lcggcl", of length 6. Note that "clgglc" is another longest palindrome that can be created.

Example 2:

Input: words = ["ab","ty","yt","lc","cl","ab"] **Output:** 8 **Explanation:** One longest palindrome is "ty" + "lc" + "cl" + "yt" = "tylcclyt", of length 8. Note that "lcyttycl" is another longest palindrome that can be created.

Example 3:

Input: words = ["cc","ll","xx"] **Output:** 2 **Explanation:** One longest palindrome is "cc", of length 2. Note that "ll" is another longest palindrome that can be created, and so is "xx".

Constraints:

* `1 <= words.length <= 105` * `words[i].length == 2` * `words[i]` consists of lowercase English letters.

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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