# 题目

给你一个字符串数组 words 和一个字符 separator ，请你按 separator 拆分 words 中的每个字符串。

返回一个由拆分后的新字符串组成的字符串数组，不包括空字符串 。

注意

separator 用于决定拆分发生的位置，但它不包含在结果字符串中。

拆分可能形成两个以上的字符串。

结果字符串必须保持初始相同的先后顺序。

示例 1：

输入：words = ["one.two.three","four.five","six"], separator = "."

输出：["one","two","three","four","five","six"]

解释：在本示例中，我们进行下述拆分：

"one.two.three" 拆分为 "one", "two", "three"

"four.five" 拆分为 "four", "five"

"six" 拆分为 "six"

因此，结果数组为 ["one","two","three","four","five","six"] 。

示例 2：

输入：words = ["$easy$","$problem$"], separator = "$"

输出：["easy","problem"]

解释：在本示例中，我们进行下述拆分：

"$easy$" 拆分为 "easy"（不包括空字符串）

"$problem$" 拆分为 "problem"（不包括空字符串）

因此，结果数组为 ["easy","problem"] 。

示例 3：

输入：words = ["|||"], separator = "|"

输出：[]

解释：在本示例中，"|||" 的拆分结果将只包含一些空字符串，所以我们返回一个空数组 [] 。

提示：

1 <= words.length <= 100

1 <= words[i].length <= 20

words[i] 中的字符要么是小写英文字母，要么就是字符串 ".,|$#@" 中的字符（不包括引号）

separator 是字符串 ".,|$#@" 中的某个字符（不包括引号）

# 分析

## 方法一：模拟

思路：

我们首先创建结果数组res，然后我们将字符串数组words中每一个字符串按照separator 进行拆分，并将每一个拆分出来的非空字符串加入答案数组res即可。

代码：

class Solution {

public:

vector<string> splitWordsBySeparator(vector<string>& words, char separator) {

vector<string> res;

for (string &word : words) {

stringstream ss(word);

string sub;

while (getline(ss, sub, separator)) {

if (!sub.empty()) {

res.push\_back(sub);

}

}

}

return res;

}

};

或：

class Solution {

public:

std::vector<std::string> splitWordsBySeparator(std::vector<std::string>& words, char separator) {

std::vector<std::string> result;

for (const std::string& word : words) {

size\_t start = 0;

size\_t found;

while ((found = word.find(separator, start)) != std::string::npos) {

// 如果不是空字符串，则添加到结果中

if (found > start) {

result.push\_back(word.substr(start, found - start));

}

start = found + 1; // 跳过分隔符

}

// 处理最后一个子串（如果存在）

if (start < word.length()) {

result.push\_back(word.substr(start));

}

}

return result;

}

};

复杂度分析：

时间复杂度：O(n×m)，其中n为字符串数组words的长度，m为字符串数组 words中最长字符串的长度。

空间复杂度：O(m)，其中m为字符串数组words中最长字符串的长度。主要为存储每一个字符串拆分结果的空间开销。注意返回的答案数组的不计入空间复杂度。