# 题目

给你一个字符串sentence作为句子并指定检索词为 searchWord ，其中句子由若干用单个空格分隔的单词组成。请你检查检索词 searchWord 是否为句子 sentence 中任意单词的前缀。

如果 searchWord 是某一个单词的前缀，则返回句子 sentence 中该单词所对应的下标（下标从 1 开始）。如果 searchWord 是多个单词的前缀，则返回匹配的第一个单词的下标（最小下标）。如果 searchWord 不是任何单词的前缀，则返回 -1 。

字符串 s 的 前缀 是 s 的任何前导连续子字符串。

示例 1：

输入：sentence = "i love eating burger", searchWord = "burg"

输出：4

解释："burg" 是 "burger" 的前缀，而 "burger" 是句子中第 4 个单词。

示例 2：

输入：sentence = "this problem is an easy problem", searchWord = "pro"

输出：2

解释："pro" 是 "problem" 的前缀，而 "problem" 是句子中第 2 个也是第 6 个单词，但是应该返回最小下标 2 。

示例 3：

输入：sentence = "i am tired", searchWord = "you"

输出：-1

解释："you" 不是句子中任何单词的前缀。

提示：

1 <= sentence.length <= 100

1 <= searchWord.length <= 10

sentence 由小写英文字母和空格组成。

searchWord 由小写英文字母组成。

# 分析

## 方法一：模拟/遍历

思路：

代码：

class Solution {

public:

int isPrefixOfWord(string sentence, string searchWord) {

int index = 1; // 单词的下标从 1 开始

int wordStart = 0; // 单词的起始位置

// 遍历句子中的每个字符

for (int i = 0; i <= sentence.length(); ++i) {

if (i == sentence.length() || sentence[i] == ' ') { // 如果遇到空格或者遍历完整个句子

// 提取出一个单词

string word = sentence.substr(wordStart, i - wordStart);

// 检查该单词是否以 searchWord 开头

if (word.find(searchWord) == 0) {

return index; // 如果是，则返回该单词的下标

}

index++; // 更新单词的下标

wordStart = i + 1; // 更新下一个单词的起始位置

}

}

return -1; // 如果遍历完所有单词都没有找到符合条件的单词，则返回 -1

}

};