【华为OD机考 统一考试机试C卷】英文输入法（C++ Java JavaScript Python C语言）

华为OD机考:统一考试 C卷 + D卷 + B卷 +A卷

目前在考C卷，经过两个月的收集整理，C卷真题已基本整理完毕
抽到原题的概率为2/3到3/3，也就是最少抽到两道原题。请注意：大家刷完C卷真题，最好要把B卷的真题刷一下，因为C卷的部分真题来自B卷。
另外订阅专栏还可以联系笔者开通在线 OJ 进行刷题，提高刷题效率。
真题目录：华为OD机考机试 真题目录（C卷 + D卷 + B卷 + A卷） + 考点说明
专栏：2023华为OD机试(B卷+C卷+D卷)（C++JavaJSPy）
华为OD面试真题精选：华为OD面试真题精选
在线OJ：点击立即刷题，模拟真实机考环境

题目描述

主管期望你来实现英文输入法单词联想功能。
需求如下：

- 依据用户输入的单词前缀，从已输入的英文语句中联想出用户想输入的单词，按字典序输出联想到的单词序列，
- 如果联想不到，请输出用户输入的单词前缀。

注意：

1. 英文单词联想时，区分大小写
2. 缩略形式如”don’t”，判定为两个单词，”don”和”t”
3. 输出的单词序列，不能有重复单词，且只能是英文单词，不能有标点符号

输入描述

输入为两行。

首行输入一段由英文单词word和标点符号组成的语句str；

接下来一行为一个英文单词前缀pre。

- 0 < word.length() <= 20
- 0 < str.length <= 10000
- 0 < pre <= 20

输出描述

输出符合要求的单词序列或单词前缀，存在多个时，单词之间以单个空格分割

用例

输入	**I love you** **He**
输出	He
说明	从用户已输入英文语句”I love you”中提炼出“I”、“love”、“you”三个单词，接下来用户输入“He”，从已输入信息中无法联想到任何符合要求的单词，因此输出用户输入的单词前缀。

输入	**The furthest distance in the world, Is not between life and death, But when I stand in front of you, Yet you don’t know that I love you.** **f**
输出	front furthest
说明	从用户已输入英文语句”The furthestdistance in the world, Is not between life and death, But when I stand in frontof you, Yet you dont know that I love you.”中提炼出的单词，符合“f”作为前缀的，有“furthest”和“front”，按字典序排序并在单词间添加空格后输出，结果为“front furthest”。

C++

```
#include <iostream>
#include <algorithm>
#include <string>
#include <sstream>
#include <set>

```

```cpp
using namespace std;

int main() {
    string sentence, prefix;
    getline(cin, sentence); // 输入一段由英文单词word和标点符号组成的语句
    getline(cin, prefix); // 输入一个英文单词前缀
    replace_if(sentence.begin(), sentence.end(), [](char c){return !isalpha(c);}, ' '); // 将标点符号替换为空格
    stringstream ss(sentence);
    set<string> word_set; // 存储单词的集合，自动去重且按照字典序排序
    string word;
    while (ss >> word) {
        word_set.insert(word);
    }
    string ans;
    for (auto s : word_set) { // 遍历单词集合
        if (s.substr(0, prefix.length()) == prefix) { // 如果单词以前缀开头
            ans += s + " "; // 将单词加入答案字符串
        }
    }
    if (ans.length() > 0) { // 如果答案字符串不为空
        cout << ans << endl; // 输出单词序列
    } else {
        cout << prefix << endl; // 否则输出前缀
    }
    return 0;
}
```

JavaScript

```javascript
const readline = require('readline');

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout
});

rl.on('line', (sentence) => {
  rl.on('line', (prefix) => {
  sentence = sentence.replace(/[^\w\s]/g, ' '); // 将标点符号替换为空格
    const wordSet = new Set(sentence.split(' ')); // 存储单词的集合，自动去重且按照字典序排序
```

```
    let ans = '';
    for (const word of Array.from(wordSet).sort()) { // 遍历单词集合
      if (word.startsWith(prefix)) { // 如果单词以前缀开头
        ans += word + ' '; // 将单词加入答案字符串
      }
    }
    if (ans) { // 如果答案字符串不为空
      console.log(ans); // 输出单词序列
    } else {
      console.log(prefix); // 否则输出前缀
    }
    rl.close();
  });
});
```

Java

```
import java.util.*;
import java.io.*;

public class Main {
    public static void main(String[] args) throws IOException {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        String sentence = br.readLine(); // 输入一段由英文单词word和标点符号组成的语句
        String prefix = br.readLine(); // 输入一个英文单词前缀
        sentence = sentence.replaceAll("[^a-zA-Z]", " "); // 将标点符号替换为空格
        Set<String> wordSet = new TreeSet<>(); // 存储单词的集合，自动去重且按照字典序排序
        String[] words = sentence.split("\\s+");
        for (String word : words) {
            wordSet.add(word);
        }
        StringBuilder ans = new StringBuilder();
        for (String s : wordSet) { // 遍历单词集合
            if (s.startsWith(prefix)) { // 如果单词以前缀开头
                ans.append(s).append(" "); // 将单词加入答案字符串
            }
        }
        if (ans.length() > 0) { // 如果答案字符串不为空
            System.out.println(ans.toString().trim()); // 输出单词序列
        } else {
```

```
            System.out.println(prefix); // 否则输出前缀
        }
    }
}
```

Python

```python
import string

sentence = input() # 输入一段由英文单词word和标点符号组成的语句
prefix = input() # 输入一个英文单词前缀
sentence = sentence.translate(str.maketrans(string.punctuation, ' ' * len(string.punctuation))) # 将标点符号替换为空格
word_set = set(sentence.split()) # 存储单词的集合，自动去重且按照字典序排序
ans = ''
for s in sorted(word_set): # 遍历单词集合
    if s.startswith(prefix): # 如果单词以前缀开头
        ans += s + ' ' # 将单词加入答案字符串
if ans: # 如果答案字符串不为空
    print(ans) # 输出单词序列
else:
    print(prefix) # 否则输出前缀
```

文章目录

用例3
用例4
用例5
用例6
用例7
用例8
用例9
用例10

C语言

```c
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include <stdlib.h>

#define MAX_WORDS 1000
#define MAX_WORD_LENGTH 21
#define MAX_SENTENCE_LENGTH 10001

```

```c

int compare(const void *a, const void *b) {
    return strcmp(*(const char **)a, *(const char **)b);
}

int main() {
    char sentence[MAX_SENTENCE_LENGTH], prefix[MAX_WORD_LENGTH];
    fgets(sentence, MAX_SENTENCE_LENGTH, stdin); // 输入一段由英文单词和标点符号组成的语句
    fgets(prefix, MAX_WORD_LENGTH, stdin);       // 输入一个英文单词前缀

    // 去除前缀字符串末尾的换行符
    size_t prefix_len = strlen(prefix);
    if (prefix[prefix_len - 1] == '\n') {
        prefix[prefix_len - 1] = '\0';
        prefix_len--;
    }

    // 将标点符号替换为空格
    for (int i = 0; sentence[i] != '\0'; i++) {
        if (!isalpha(sentence[i])) {
            sentence[i] = ' ';
        }
    }

    // 存储单词的数组
    char *words[MAX_WORDS];
    int word_count = 0;
    char *word = strtok(sentence, " ");

    // 分割单词并存储
    while (word != NULL) {
        words[word_count] = (char *)malloc(strlen(word) + 1);
        strcpy(words[word_count], word);
        word_count++;
        word = strtok(NULL, " ");
    }

    // 对单词数组进行排序
    qsort(words, word_count, sizeof(char *), compare);

```

```
    // 输出结果
    int found = 0;
    for (int i = 0; i < word_count; i++) {
        if (strncmp(words[i], prefix, prefix_len) == 0) {
            printf("%s ", words[i]);
            found = 1;
        }
        free(words[i]); // 释放分配的内存
    }

    // 如果没有找到任何匹配的单词，输出前缀
    if (!found) {
        printf("%s", prefix);
    }

    return 0;
}
```

完整用例

用例1

```
I love you
He
```

用例2

```
The furthest distance in the world, Is not between life and death, But when I stand in front of you, Yet you don't know that I love you.
f
```

用例3

```
Hello world
W
```

用例4

```
I am a student
s
```

用例5

```
This is a test
T
```

用例6

```
I love you
L
```

用例7

```
The furthest distance in the world, Is not between life and death, But when I stand in front of you, Yet you don't know that I love you.
d
```

用例8

```
This is a test. This is only a test.
o
```

用例9

```
Hello world, how are you? I'm fine, thank you.
h
```

用例10

```
I am a student. I study in a university.
u
```