题目描述

给定一个可存储若干单词的字典，找出指定单词的所有相似单词，并且按照单词名称从小到大排序输出。

单词仅包括字母，但可能大小写并存（大写不一定只出现在首字母）。

相似单词说明：给定一个单词X，如果通过任意交换单词中字母的位置得到不同的单词Y，那么定义Y是X的相似单词，如abc、bca即为相似单词（大小写是不同的字母，如a和A算两个不同字母）。

字典序排序：大写字母<小写字母。同样大小写的字母，遵循26字母顺序大小关系。

即*A<B<C<...<X<Y<Z<a<b<c<...<x<y<z*. 如*Bac<aBc<acB<cBa*.

输入描述

第一行为给定的单词个数N（N为非负整数）

从第二行到地N+1行是具体的单词（每行一个单词）

最后一行是指定的待检测单词（用于检测上面给定的单词中哪些是与该指定单词是相似单词，该单词可以不是上面给定的单词）

输出描述

从给定的单词组中，找出指定单词的相似单词，并且按照从小到大字典序排列输出，中间以空格隔开

如果不存在，则输出null（字符串null）

用例

输入	4 abc dasd tad bca abc
输出	abc bca
说明	在给定的输入种，与abc是兄弟单词的是abc bca，且输出按照字典序大小排序，输出的所有单词以空格隔开

输入	4 abc dasd tad bca abd
输出	null
说明	给定的单词组中，没有与给定单词abd是兄弟单词，输出为null（字符串null）

题目解析

简单的排序题，逻辑请看代码。

JavaScript算法源码

```
/* JavaScript Node ACM模式 控制台输入获取 */
const readline = require("readline");

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

const lines = [];
let n;
rl.on("line", (line) => {
  lines.push(line);
```

伏城之外 已关注 0 3 0 专栏目录 已订阅

题目解析

简单的排序题，逻辑请看代码。

JavaScript算法源码

```javascript
/* JavaScript Node ACM模式 控制台输入获取 */
const readline = require("readline");

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

const lines = [];
let n;
rl.on("line", (line) => {
  lines.push(line);

  if (lines.length === 1) {
    n = lines[0] - 0;
  }

  if (n && lines.length === n + 2) {
    lines.shift();
    const target = lines.pop();
    console.log(getResult(lines, target));
    lines.length = 0;
  }
});

function getResult(words, target) {
  const sorted_target = [...target].sort().join("");

  const ans = [];
  for (let word of words) {
    const sorted_word = [...word].sort().join("");

    if (sorted_target === sorted_word) {
      ans.push(word);
    }
  }

  if (ans.length) return ans.sort().join(" ");
  else return "null";
}
```

Java算法源码

```java
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Scanner;
import java.util.StringJoiner;

public class Main {
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);

    int n = sc.nextInt();

    String[] words = new String[n];
    for (int i = 0; i < n; i++) {
      words[i] = sc.next();
    }

    String target = sc.next();

    System.out.println(getResult(words, target));
  }

  public static String getResult(String[] words, String target) {
    String sorted_target = sortStr(target);

    ArrayList<String> ans = new ArrayList<>();
    for (String word : words) {
      String sorted_word = sortStr(word);

      if (sorted_target.equals(sorted_word)) {
        ans.add(word);
      }
    }

    if (ans.size() > 0) {
      ans.sort(String::compareTo);

      StringJoiner sj = new StringJoiner(" ");
      for (String an : ans) sj.add(an);
      return sj.toString();
    } else {
      return "null";
    }
```

伏城之外 已关注 0 3 0 专栏目录 已订阅

Java算法源码

```java
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Scanner;
import java.util.StringJoiner;

public class Main {
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);

    int n = sc.nextInt();

    String[] words = new String[n];
    for (int i = 0; i < n; i++) {
      words[i] = sc.next();
    }

    String target = sc.next();

    System.out.println(getResult(words, target));
  }

  public static String getResult(String[] words, String target) {
    String sorted_target = sortStr(target);

    ArrayList<String> ans = new ArrayList<>();
    for (String word : words) {
      String sorted_word = sortStr(word);

      if (sorted_target.equals(sorted_word)) {
        ans.add(word);
      }
    }

    if (ans.size() > 0) {
      ans.sort(String::compareTo);

      StringJoiner sj = new StringJoiner(" ");
      for (String an : ans) sj.add(an);
      return sj.toString();
    } else {
      return "null";
    }
  }

  public static String sortStr(String word) {
    char[] chars = word.toCharArray();
    Arrays.sort(chars);
    return new String(chars);
  }
}
```

Python算法源码

```python
n = int(input())

words = []
for i in range(n):
    words.append(input())

target = input()

sorted_target = sorted(target)

ans = []

for word in words:
    sorted_word = sorted(word)

    if sorted_target == sorted_word:
        ans.append(word)

if len(ans) > 0:
    ans.sort()
    print(" ".join(ans))
else:
    print("null")
```