

华为OD机试 – 字符串重新排列、字符串重新排序（Java & JS & Python）

原创

伏城之外

已于 2023-04-02 14:33:56 修改

7699

收藏 23

版权

分类专栏：

华为OD机试（Java & JS & Python）

华为OD机试2023A

文章标签：

算法

JavaScript 华为机试 Java Python

OD

华为OD机试2... 同时被 2 个专栏收录

¥49.90
¥99.00

253 订阅 132 篇文章

已订阅

该专栏为热销专栏榜 第46名

题目描述

给定一个字符串s，s包括以空格分隔的若干个单词，请对s进行如下处理后输出：

- 1、单词内部调整：对每个单词字母重新按字典序排序
- 2、单词间顺序调整：

1) 统计每个单词出现的次数，并按次数降序排列

2) 次数相同，按单词长度升序排列

3) 次数和单词长度均相同，按字典升序排列

请输出处理后的字符串，每个单词以一个空格分隔。

输入描述

一行字符串，每个字符取值范围：[a-zA-z0-9]以及空格，字符串长度范围：[1, 1000]

输出描述

输出处理后的字符串，每个单词以一个空格分隔。

用例

输入	This is an apple
输出	an is This aelpp
说明	无

输入	My sister is in the house not in the yard
输出	in in eht eht My is not adry ehosu eirsst
说明	无

题目解析

本题需要注意的是，先进行单词内部调整，然后再进行单词间顺序。

考察的是排序。

Python解法的第16行排序原理，请看：

Python3 - list sort方法实现多条件自定义排序_伏城之外的博客-CSDN博客

JavaScript算法源码

```
1  /* JavaScript Node ACM模式 控制台输入获取 */
2  const readline = require("readline");
3
4  const rl = readline.createInterface({
5    input: process.stdin,
6    output: process.stdout,
7  });
8
9  rl.on("line", (line) => {
10    const arr = line.split(" ");
11    console.log(getResult(arr));
12  });
13
14  function getResult(arr) {
15    arr = arr.map((str) => [...str].sort().join(""));
16
17    const count = arr.reduce((p, c) => {
18      p[c] ? p[c]++ : (p[c] = 1);
19      return p;
20    }, {});
21
22    arr.sort((a, b) =>
23      count[a] !== count[b]
24        ? count[b] - count[a]
25        : a.length !== b.length
26          ? a.length - b.length
27          : a > b
28            ? 1
29            : -1
30    );
31
32    return arr.join(" ");
33  }
```

Java算法源码

```
1  import java.util.Arrays;
2  import java.util.HashMap;
3  import java.util.Scanner;
4  import java.util.StringJoiner;
5
6  public class Main {
7      public static void main(String[] args) {
8          Scanner sc = new Scanner(System.in);
9
10         String[] arr = sc.nextLine().split(" ");
11         System.out.println(getResult(arr));
12     }
13
14     public static String getResult(String[] arr) {
15         arr =
16             Arrays.stream(arr)
17                 .map(
18                     str -> {
19                         char[] cArr = str.toCharArray();
20                         Arrays.sort(cArr);
21                         return new String(cArr);
22                     })
23                 .toArray(String[]::new);
24
25         HashMap<String, Integer> count = new HashMap<>();
26         for (String s : arr) {
27             count.put(s, count.getOrDefault(s, 0) + 1);
28         }
29
30         Arrays.sort(
31             arr,
32             (a, b) ->
33                 !count.get(a).equals(count.get(b))
34                     ? count.get(b) - count.get(a)
35                     : a.length() != b.length() ? a.length() - b.length() : a.compareTo(b)
36         );
37
38         StringJoiner sj = new StringJoiner(" ", "", "");
39         for (String s : arr) {
40             sj.add(s);
41         }
42         return sj.toString();
43     }
```

Python算法源码

```
1  # 输入获取
2  arr = input().split()
3
4
5  # 算法入口
6  def getResult(arr):
7      for i in range(len(arr)):
8          arr[i] = "".join(sorted(arr[i]))
9
10     count = {}
11     for c in arr:
12         if count.get(c) is None:
13             count[c] = 0
14             count[c] += 1
15
16     # arr.sort(key=lambda x: (-count[x], len(x), [ord(char) for char in x]))
17     arr.sort(key=lambda x: (-count[x], len(x), x))
18
19     # return " ".join(map(str, arr))
20     return " ".join(arr)
21
22
23 # 算法调用
24 print(getResult(arr))
```