

华为OD机试 - 报文重排序 (Java & JS & Python)

按 F11 即可退出全屏模式

原创 伏城之外 已于 2023-06-04 14:15:22 修改 1745 收藏 3 版权

分类专栏: 华为OD机试AB (Java & JS & Python) 文章标签: 算法 华为机试 Java JavaScript Python

OD 华为OD机试AB (Java & JS & Python) 同时被 2 个专栏收录

该专栏为热销专栏榜 第2名

¥59.90  
¥99.00

3382 订阅 371 篇文章

已订阅

题目描述

对报文进行重传和重排序是常用的可靠性机制，重传缓冲区内有有一定数量的子报文，每个子报文在原始报文中的顺序已知，现在需要恢复出原始报文。

输入描述

输入第一行为N，表示子报文的个数， $0 < N \leq 1000$ 。  
输入第二行为N个子报文，以空格分开，子报文格式为：

字符串报文内容+后缀顺序索引

字符串报文内容由[a-z,A-Z]组成，后缀为整型值，表示顺序。  
顺序值唯一，不重复。

输出描述

输出恢复出的原始报文，按照每个子报文的顺序的升序排序恢复出原始报文，顺序后缀需要从恢复出的报文中删除掉

用例

输入	4 rolling3 stone4 like1 a2
输出	like a rolling stone
说明	4个子报文的内容分别为 "rolling", "stone", "like", "a", 顺序值分别为3, 4, 1, 2, 按照顺序值升序并删除顺序后缀，得到恢复的原始报文: "like a rolling stone"

输入	8 gifts6 and7 Exchanging1 all2 precious5 things8 kinds3 of4
输出	Exchanging all kinds of precious gifts and things
说明	无

题目解析

考察字符串操作。  
这题解析“字符串报文内容”、“后缀顺序索引”时，我使用了正则匹配的捕获组。

2023.06.04

本题需要考虑下后缀顺序索引不连续的情况，即不是严格的0-N-1序号

```
1 import java.util.ArrayList;
```

```
2 import java.util.Scanner;
3 import java.util.StringJoiner;
4 import java.util.regex.Matcher;
5 import java.util.regex.Pattern;
6
7 public class Main {
8     static class Word {
9         int id;
10        String content;
11
12        public Word(int id, String content) {
13            this.id = id;
14            this.content = content;
15        }
16    }
17
18    public static void main(String[] args) {
19        Scanner sc = new Scanner(System.in);
20
21        int n = Integer.parseInt(sc.nextLine());
22        String[] arr = sc.nextLine().split(" ");
23
24        System.out.println(getResult(n, arr));
25    }
26
27    public static String getResult(int n, String[] arr) {
28        ArrayList<Word> ans = new ArrayList<>();
29
30        Pattern pattern = Pattern.compile("([a-zA-Z]+)(\\d+)");
31
32        for (String s : arr) {
33            Matcher matcher = pattern.matcher(s);
34            if (matcher.find()) {
35                String content = matcher.group(1);
36                int i = Integer.parseInt(matcher.group(2)) - 1;
37                ans.add(new Word(i, content));
38            }
39        }
40
41        ans.sort((a, b) -> a.id - b.id);
```

```
42
43     StringJoiner sj = new StringJoiner(" ");
44     for (Word an : ans) {
45         sj.add(an.content);
46     }
47     return sj.toString();
48 }
49 }
```