

华为OD机试 - 选修课 (Java & JS & Python)

原创

伏城之外

已于 2023-06-02 13:31:40 修改

2118

收藏 9

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

题目描述

现有两门选修课，每门选修课都有一部分学生选修，每个学生都有选修课的成绩，需要你找出同时选修了两门选修课的学生，先按照班级进行划分，班级编号小的先输出，每个班级按照两门选修课成绩和的降序排序，成绩相同时按照学生的学号升序排序。

输入描述

- 第一行为第一门选修课学生的成绩，
- 第二行为第二门选修课学生的成绩，
- 每行数据中学生之间以英文分号分隔，每个学生的学号和成绩以英文逗号分隔，
- 学生学号的格式为8位数字

2位院系编号+入学年份后2位+院系内部1位专业编号+所在班级3位学号

- 学生成绩的取值范围为[0,100]之间的整数，
- 两门选修课选修学生数的取值范围为[1-2000]之间的整数。

输出描述

- 同时选修了两门选修课的学生的学号，如果没有同时选修两门选修课的学生输出NULL，
- 否则，先按照班级划分，班级编号小的先输出，每个班级先输出班级编号(学号前五位)，
- 然后另起一行输出这个班级同时选修两门选修课的学生学号，学号按照要求排序(按照两门选修课成绩和的降序，成绩和相同时按照学号升序学生之间以英文分号分隔。

用例

输入	01202021,75;01201033,95;01202008,80;01203006,90;01203088,100 01202008,70;01203088,85;01202111,80;01202021,75;01201100,88
输出	01202 01202008;01202021 01203 01203088
说明	同时选修了两选修课的学生01202021、01202008、01203088，这三个学生两门选修课的成绩和分别为150、150、185，01202021、01202008属于01202班的学生，按照成绩和降序，成绩相同时按学号升序输出的结果为01202008;01202021，01203088属于01203班的学生，按照成绩和降序，成绩相同时按学号升序输出的结果为01203088，01202的班级编号小于01203的班级编号，需要先输出。
输入	01201022,75;01202033,95;01202018,80;01203006,90;01202066,100 01202008,70;01203102,85;01202111,80;01201021,75;01201100,88
输出	NULL
说明	没有同时选修了两门选修课的学生，输出NULL。

题目解析

本题主要考察：字符串^Q，数组，集合的操作，考察多条件排序。

本题的逻辑如下：

1. 从输入的两行中，解析出学生信息（学号，班号，课程一得分，课程二得分）
2. 过滤出两门课程都有得分的学生
3. 将学生按照班号进行分别统计
4. 按照班号进行升序，然后打印班号，再将该班号下的学生按照：总分进行降序，总分相同，则继续按照学号升序，然后以“;”拼接同班学生的学号，进行打印

难点主要在于第一步，如何从两行输入解析中得到一个完整学生的信息，这里我选用了字典结构，key为学生学号，value为学生信息，解析第二行输入时，可以检查字典是否已存在对应学号的key，若存在则再对应value下补充课程二得分。

```
1  import java.util.ArrayList;
2  import java.util.HashMap;
3  import java.util.Scanner;
4  import java.util.StringJoiner;
5
6  public class Main {
7      // 学生类
8      static class Student {
9          String sid; // 学号
10         String cid; // 班号
11         int score1 = -1; // 课程一 得分
12         int score2 = -1; // 课程二 得分
13
14         // 总分
15         public int getSumScore() {
16             return this.score1 + this.score2;
17         }
18     }
19
20     public static void main(String[] args) {
21         Scanner sc = new Scanner(System.in);
22
23         String s1 = sc.nextLine();
24         String s2 = sc.nextLine();
25
26         getResult(s1, s2);
27     }
28 }
```

```

29 public static void getResult(String s1, String s2) {
30     // key是学号, value是学生对象
31     HashMap<String, Student> students = new HashMap<>();
32
33     // 解析学生信息
34     divide(s1, 1, students);
35     divide(s2, 2, students);
36
37     // 过滤出有两个课程得分的学生
38     Student[] suits =
39         students.values().stream()
40             .filter(stu -> stu.score1 != -1 && stu.score2 != -1)
41             .toArray(Student[]::new);
42
43     // 如果不存在这样的学生, 则返回NULL
44     if (suits.length == 0) {
45         System.out.println("NULL");
46         return;
47     }
48
49     // key是班号, value是该班级的学生集合
50     HashMap<String, ArrayList<Student>> ans = new HashMap<>();
51     for (Student stu : suits) {
52         ans.putIfAbsent(stu.cid, new ArrayList<>());
53         ans.get(stu.cid).add(stu);
54     }
55
56     ans.keySet().stream()
57         .sorted(String::compareTo) // 按照班号升序
58         .forEach(
59             cid -> {
60                 System.out.println(cid); // 打印班号
61
62                 ArrayList<Student> studs = ans.get(cid);
63                 studs.sort(
64                     (a, b) ->
65                         a.getSumScore() != b.getSumScore()
66                             ? b.getSumScore() - a.getSumScore()
67                             : a.sid.compareTo(b.sid)); // 同一班的学生按照总分降序, 总分相
同, 则按照学号升序

```

```
68
69         // 打印同一班的学号，以分号分隔
70         StringJoiner sj = new StringJoiner(";");
71         for (Student stud : studs) sj.add(stud.sid);
72         System.out.println(sj);
73     });
74 }
75
76 public static void divide(String str, int courseId, HashMap<String, Student>
students) {
77     for (String sub : str.split(";")) {
78         String[] tmp = sub.split(",");
79
80         String sid = tmp[0]; // 学号
81         String cid = sid.substring(0, 5); // 班号
82         int score = Integer.parseInt(tmp[1]); // 课程得分
83
84         students.putIfAbsent(sid, new Student()); // 是否已记录过该学生，若没有则生成对应学生
对象
85
86         // 初始化学生对象
87         Student stu = students.get(sid);
88
89         stu.sid = sid;
90         stu.cid = cid;
91
92         if (courseId == 1) stu.score1 = score;
93         else stu.score2 = score;
94     }
95 }
96 }
```