题目描述

某学校举行运动会，学生们按编号(1、2、3…n)进行标识，现需要按照身高由低到高排列，对身高相同的人，按体重由轻到重排列；对于身高体重都相同的人，维持原有的编号顺序关系。请输出排列后的学生编号。

输入描述

两个序列，每个序列由n个正整数组成（0 < n <= 100），第一个序列中的数值代表身高，第二个序列中的数值代表体重。

输出描述

排列结果，每个数值都是原始序列中的学生编号，编号从1开始

用例

输入	4 100 100 120 130 40 30 60 50
输出	2 1 3 4
说明	输出的第一个数字2表示此人原始编号为2，即身高为100，体重为30的这个人。 由于他和编号为1的人身高一样，但体重更轻，因此要排在1前面。

输入	3 90 110 90 45 60 45
输出	1 3 2
说明	1和3的身高体重都相同，需要按照原有位置关系让1排在3前面，而不是3 1 2。

题目解析

考察多条件排序。

Java算法源码

```java
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();

    int[] heights = new int[n];
    for (int i = 0; i < n; i++) heights[i] = sc.nextInt();

    int[] weights = new int[n];
    for (int i = 0; i < n; i++) weights[i] = sc.nextInt();

    System.out.println(getResult(n, heights, weights));
  }

  public static String getResult(int n, int[] heights, int[] weights) {
    int[][] students = new int[n][3];

    for (int i = 0; i < n; i++) {
      students[i] = new int[] {heights[i], weights[i], i + 1};
    }

    Arrays.sort(
        students, (a, b) -> a[0] != b[0] ? a[0] - b[0] : a[1] != b[1] ? a[1] - b[1] : a[2] - b[2]);

    StringJoiner sj = new StringJoiner(" ");
    for (int[] student : students) {
      sj.add(student[2] + "");
    }
    return sj.toString();
  }
}
```

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

  if (lines.length === 3) {
    let n = parseInt(lines[0]);
    let heights = lines[1].split(" ").map(Number);
    let weights = lines[2].split(" ").map(Number);

    console.log(getResult(n, heights, weights));

    lines.length = 0;
  }
});

function getResult(n, heights, weights) {
  let persons = [];
  for (let i = 0; i < n; i++) {
    let p = {};
    p.index = i + 1;
    p.height = heights[i];
    p.weight = weights[i];
    persons.push(p);
  }

  return persons
    .sort((a, b) =>
      a.height != b.height
        ? a.height - b.height
        : a.weight != b.weight
        ? a.weight - b.weight
        : a.index - b.index
    )
    .map((p) => p.index)
    .join(" ");
}
```

Python算法源码

```python
# 输入获取
n = int(input())
heights = list(map(int, input().split()))
weights = list(map(int, input().split()))


# 算法入口
def getResult():
    students = []

    for i in range(n):
        students.append([heights[i], weights[i], i + 1])

    students.sort(key=lambda x: (x[0], x[1], x[2]))

    return " ".join(map(lambda x: str(x[2]), students))


# 算法调用
print(getResult())
```