牛客网-华为机试练习题 47

题目描述

信号测量的结果包括测量编号和测量值。存在信号测量结果丢弃及测量结果重复的情况。

1.测量编号不连续的情况，认为是测量结果丢弃。对应测量结果丢弃的情况，需要进行插值操作以更准确的评估信号。

采用简化的一阶插值方法,由丢失的测量结果两头的测量值算出两者中间的丢失值。

假设第M个测量结果的测量值为A，第N个测量结果的测量值为B。则需要进行(N-M-1)个测量结果的插值处理。进行一阶线性插值估计的第N+i个测量结果的测量值为A+((B-A)/(N-M))*i （注：N的编号比M大。）

例如：只有测量编号为4的测量结果和测量编号为7的测量结果，测量值分别为4和10

- 则需要补充测量编号为5和6的测量结果。
- 其中测量编号为5的测量值=4 + ((10-4)/(7-4))*1 = 6
- 其中测量编号为6的测量值=4 + ((10-4)/(7-4))*2 = 8

- 2.测量编号相同，则认为测量结果重复，需要对丢弃后来出现的测量结果。

请根据以上规则进行测量结果的整理。

详细描述：

接口说明

原型：

```
intCleanUpMeasureInfo(MEASURE_INFO_STRUCT* pOriMeasureInfo,intnOriMINum,intnMaxMIRst,
MEASURE_INFO_STRUCT* pMeasureInfoRst);
```

输入参数：

- MEASURE_INFO_STRUCT* pOriMeasureInfo: 原始测量结果内容，以结构数组方式存放。测量编号已经按升序排列。MEASURE_INFO_STRUCT定义包含编号和测量值，见OJ.h
- int nOriMINum: 原始测量结果个数。
- int nMaxMIRst: 整理的测量结果最大个数。

输入参数：

- MEASURE_INFO_STRUCT* pMeasureInfoRst: 整理的测量结果

返回值：

- Int
- 整理的测量结果个数

输入描述:

输入说明
1 输入两个整数m， n
2 输入m个数据组

输出描述:

输出整理后的结果

示例1

输入

2 3
4 5
5 7

输出

4 5
5 7

解决代码:

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class Main {
    public static String process(int [][] values){
        int len=values.length;
        List<String> toFill=new ArrayList<String>();
        int curNo=values[0][0], curV=values[0][1];
        for(int i=1;i<=len-1;i++){
            if(values[i][0]-curNo>1){
                cal(curNo,curV,values[i][0],values[i][1],toFill);
                curNo=values[i][0];
                curV=values[i][1];
                continue;
            }else if(values[i][0]-curNo==1 || values[i][0]-curNo<0){
                toFill.add(curNo+" "+curV);
                curNo=values[i][0];
                curV=values[i][1];
                continue;
            }else if(values[i][0]-curNo==0){//编号相等
                continue;
            }
        }
        toFill.add(curNo+" "+curV);
        StringBuilder res=new StringBuilder();
        for(String str : toFill){
            res.append(str+"\n");
        }
        return res.substring(0,res.length()-1);
    }

    public static void cal(int smallNo,int smallV,int bigNo,int bigV,List<String> toFill){
        toFill.add(smallNo+" "+smallV);
        int nums=bigNo-smallNo-1;
        for(int i=1;i<=nums;i++){
            int curNo=smallNo+i;
```

```
            int curV=smallV+((bigV-smallV)/(bigNo-smallNo))*i;
            toFill.add(curNo+" "+curV);
        }
    }
    public static void main(String args[]){
        Scanner in=new Scanner(System.in);
         while (in.hasNext()) {
            int M=in.nextInt();
            int N=in.nextInt();
            int [][] values=new int[M][2];
            for(int i=0;i<M;i++){
                values[i][0]=in.nextInt();
                values[i][1]=in.nextInt();
            }
            System.out.println(process(values));
         }
        in.close();
    }
}
```