Sort the Summary

Given an array of integers, create a 2-dimensional array where the first element is a distinct *value* from the array and the second element is that value's *frequency* within the array.  Sort the resulting array descending by frequency.  If multiple values have the same frequency, they should be sorted ascending.

****Example****

*arr = [3, 3, 1, 2, 1]*

* There are two values, 3 and 1, each with a frequencyof 2, and one value 2 with a frequency of 1: [[3, 2], [1, 2], [2, 1]]
* Sort the 2-dimensional array descending by frequency: [[3,2], [1, 2], [2, 1]]
* Sort the 2-dimensional array ascending by value for values with matching frequencies: [[1,2], [3, 2], [2, 1]]

****Function Description****

Complete the function *groupSort* in the editor below.

groupSort has the following parameter(s):

*int* *arr[n]*: an array of integers

Returns:

*int[n][2]:* a 2-d array of integers sorted as described

**Constraints**

* 1 ≤ *n* ≤ 105
* 1 ≤ *arr[i]* ≤ 105

**Input Format Format for Custom Testing**

Input from stdin will be processed as follows and passed to the function*.*

 The first line contains a single integer, *n*, the size of *arr*.

Each of the next *n* lines contains an integer, the item id for *arr[i]*.

**Sample Case 0**

**Sample Input**

STDIN Function

----- --------

4 → arr[] size n = 4

2      → arr = [2, 1, 2, 2]

1

2

2

**Sample Output**

2 3

1 1

**Explanation**

* The value 2 occurs 3 times and 1 occurs 1 time: [[2, 3], [1, 1]]
* Sort the 2-dimensional array descending by frequency: [[2, 3], [1, 1]]
* Sort the 2-dimensional array ascending by value for values with matching frequencies: [[2, 3], [1, 1]]

Sample Case 1

**Sample Case 1**

**Sample Input**

STDIN Function

----- --------

3 → arr size n = 3

7      → arr = [7, 12, 3]

12

3

**Sample Output**

3 1

7 1

12 1

**Explanation**

* There are three values, 7, 12 and 3, with 1 occurrence each: [[7, 1], [12, 1], [3, 1]]
* Sort the 2-dimensional array descending by number of occurrences: [[7, 1], [12, 1], [3, 1]]
* Sort the 2-dimensional array ascending by value for values with matching frequencies: [[3, 1], [7, 1], [12, 1]]

import java.io.\*;

import java.math.\*;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.function.\*;

import java.util.regex.\*;

import java.util.stream.\*;

import static java.util.stream.Collectors.joining;

import static java.util.stream.Collectors.toList;

class Result {

/\*

\* Complete the 'groupSort' function below.

\*

\* The function is expected to return a 2D\_INTEGER\_ARRAY.

\* The function accepts INTEGER\_ARRAY arr as parameter.

\*/

public static List<List<Integer>> groupSort(List<Integer> arr) {

// Write your code here

}

}

public class Solution {

public static void main(String[] args) throws IOException {

BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(System.in));

BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

int arrCount = Integer.parseInt(bufferedReader.readLine().trim());

List<Integer> arr = IntStream.range(0, arrCount).mapToObj(i -> {

try {

return bufferedReader.readLine().replaceAll("\\s+$", "");

} catch (IOException ex) {

throw new RuntimeException(ex);

}

})

.map(String::trim)

.map(Integer::parseInt)

.collect(toList());

List<List<Integer>> result = Result.groupSort(arr);

result.stream()

.map(

r -> r.stream()

.map(Object::toString)

.collect(joining(" "))

)

.map(r -> r + "\n")

.collect(toList())

.forEach(e -> {

try {

bufferedWriter.write(e);

} catch (IOException ex) {

throw new RuntimeException(ex);

}

});

bufferedReader.close();

bufferedWriter.close();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*