Full Name:	Xing Li
Email:	li.xing1@husky.neu.edu
Test Name:	**INFO 6205 Spring2019 Section_05 Mid-term**
Taken On:	26 Feb 2019 16:21:36 EST
Time Taken:	37 min 30 sec/ 50 min
Work Experience:	< 1 years
Invited by:	Robin
Invited on:	26 Feb 2019 15:53:20 EST
Tags Score:	

100%

60/60

scored in **INFO 6205 Spring2019 Section_05 Mid-term** in 37 min 30 sec on 26 Feb 2019 16:21:36 EST

Recruiter/Team Comments:

No Comments.

Plagiarism flagged

We have marked questions with suspected plagiarism below. Please review.

	Question Description	Time Taken	Score	Status
Q1	Rank List > **Coding**	37 min 11 sec	60/ 60	!

QUESTION 1

Needs Review

Score 60

Rank List > Coding

QUESTION DESCRIPTION

After the mid-term grades of a particular subject are announced, you are curious to know your current position (rank) in the class. Therefore, you decide to make a rank list . You are given the name , scholar number and marks of every student in your class. You must create an accurate rank list, i.e. the student having the maximum marks is at the top and, if two students have the same marks, the student with the (lexicographically) smaller name comes first. If both the names and marks of two students are the same, then the student having the smaller scholar number comes first.

Input:

First line of input contains N - Total number of students in class

Next N line contains name of student , scholar number and marks scored in exam .

Output:

Print the ranklist of students as explained above.

Constraints

1 <= N <= 1000

1 <= length of name <= 10

1 <= scholar number <= 1000

0 <= marks <= 30

SAMPLE INPUT

```
5
arun 8 28
harshit 10 30
surya 7 26
satyam 27 6
arun 1 28
```

SAMPLE OUTPUT

```
harshit 10 30
arun 1 28
arun 8 28
surya 7 26
satyam 27 6
```

CANDIDATE ANSWER

Language used: **Java 8**

```java
import java.io.*;
import java.math.*;
import java.security.*;
import java.text.*;
import java.util.*;
import java.util.concurrent.*;
import java.util.regex.*;
import java.io.BufferedReader;
import java.io.InputStreamReader;

class Student {

    String name;
    int scholar;
    int marks;

    public Student(String name, int scholar, int marks) {
        this.name = name;
        this.scholar = scholar;
        this.marks = marks;
    }

    public static boolean compare(Student s1, Student s2) {

        // TODO- Implement the compare method to sompare two Student Objects
        if (s1.marks > s2.marks) return true;
        else if (s1.marks < s2.marks) return false;
        char[] cs1 = s1.name.toCharArray();
        char[] cs2 = s2.name.toCharArray();
        for (int i = 0; i < cs1.length && i < cs2.length; i++) {
            if (cs1[i]-cs2[i] > 0) return false;
            else if (cs1[i]-cs2[i] < 0) return true;
        }
        if (s1.scholar > s2.scholar) return false;
        else return true;

```

```
    }

    public String toString() {
        return this.name + " " + this.scholar + " " + this.marks;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) {
            return true;
        }

        if (o instanceof Student) {
            Student tmp = (Student) o;
            if (name.equals(tmp.name) && scholar == tmp.scholar) {
                return true;
            }
        }

        return false;
    }

    @Override
    public int hashCode() {
        final int prime = 31;
        int result = 1;
        result = prime * scholar;
        result = prime * (name.hashCode()) + result;
        return result;
    }
}

class Solution {

    private static void merge(Student[] a, Student[] aux, int lo, int mid,
int hi) {

        // TODO- Implement the   Merge function.
        for (int k = lo; k <= hi; k++) {
            aux[k] = a[k];
        }
        int i = lo;
        int j = mid + 1;
        for (int k = lo; k <= hi; k++) {
            if (i > mid) a[k] = aux[j++];
            else if (j > hi) a[k] = aux[i++];
            else if (Student.compare(aux[j], aux[i])) a[k] = aux[j++];
            else a[k] = aux[i++];
        }


    }

    // mergesort a[lo..hi] using auxiliary array aux[lo..hi]
    private static void sort(Student[] a, Student[] aux, int lo, int hi) {
        if (hi <= lo) return;
        int mid = lo + (hi - lo) / 2;
        sort(a, aux, lo, mid);
        sort(a, aux, mid + 1, hi);
        merge(a, aux, lo, mid, hi);
    }

    public static void main(String args[]) throws Exception {
        BufferedReader br = new BufferedReader(new
```

```
InputStreamReader(System.in));
        String line = br.readLine();
        int N = Integer.parseInt(line);

        Student[] list = new Student[N];

        for (int i = 0; i < N; i++) {
            String[] split = br.readLine().split(" ");
            list[i] = new Student(split[0], Integer.parseInt(split[1]),
Integer.parseInt(split[2]));
        }

        Student aux[] = new Student[N];

        sort(list,aux,0,N-1);

        for (Student s : list) {
            System.out.println(s);
        }
    }
}
```

TESTCASE	DIFFICULTY	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Success	10	0.23 sec	26.5 MB
Testcase 1	Easy	Success	10	0.11 sec	24.4 MB
Testcase 2	Easy	Success	10	0.23 sec	26.5 MB
Testcase 3	Easy	Success	10	0.22 sec	26.6 MB
Testcase 4	Easy	Success	10	0.22 sec	26.6 MB
Testcase 5	Medium	Success	10	0.11 sec	24.4 MB

No Comments