Sorting: Comparator ☆

Your Sorting: Comparator submission got 35.00 points. Share Tweet

Try the next challenge

Problem | Submissions | Leaderboard | Editorial

Check out the resources on the page's right side to learn more about sorting. The video tutorial is by Gayle Laakmann McDowell, author of the best-selling interview book Cracking the Coding Interview.

Comparators are used to compare two objects. In this challenge, you'll create a comparator and use it to sort an array. The Player class is provided in the editor below. It has two fields:

1. $name$: a string.
2. $score$: an integer.

Given an array of n Player objects, write a comparator that sorts them in order of decreasing score. If 2 or more players have the same score, sort those players alphabetically ascending by name. To do this, you must create a Checker class that implements the Comparator interface, then write an int compare(Player a, Player b) method implementing the Comparator.compare(T o1, T o2) method. In short, when sorting in ascending order, a comparator function returns -1 if $a < b$, 0 if $a = b$, and 1 if $a > b$.

For example, given $n = 3$ Player objects with $Player.name, Player.score$ values of $data = [[Smith, 20], [Jones, 15], [Jones, 20]]$, we want to sort the list as $data_{sorted} = [[Jones, 20], [Smith, 20], [Jones, 15]]$.

Function Description

Declare a Checker class that implements the comparator method as described. It should sort first descending by score, then ascending by name. The code stub reads the input, creates a list of Player objects, uses your method to sort the data, and prints it out properly.

Input Format

Locked stub code in the Solution class handles the following input from stdin:

The first line contains an integer, n, the number of players.
Each of the next n lines contains a player's respective $name$ and $score$, a string and an integer.

Constraints

- $0 \leq score \leq 1000$
- Two or more players can have the same name.
- Player names consist of lowercase English alphabetic letters.

Output Format

You are not responsible for printing any output to stdout. Locked stub code in Solution will create a Checker object, use it to sort the Player array, and print each sorted element.

Sample Input

```
5
amy 100
david 100
heraldo 50
aakansha 75
aleksa 150
```

Sample Output

```
aleksa 150
amy 100
david 100
aakansha 75
heraldo 50
```

Explanation

As you can see, the players are first sorted by decreasing score and then sorted alphabetically by name.

Current Buffer (saved locally, editable)

Java 8

```
import ↔;

class Player {
    String name;
    int score;

    Player(String name, int score) {
        this.name = name;
        this.score = score;
    }
}
class Checker implements Comparator<Player> {
    // complete this method
    public int compare(Player a, Player b) {
        if(b.score == a.score) {
            return (a.name.toString().compareTo(b.name.toString()));
        }else{
            return (b.score - a.score);
        }
    }
}



public class Solution {
```

```
24
25      public static void main(String[] args) {
26          Scanner scan = new Scanner(System.in);
27          int n = scan.nextInt();
28
29          Player[] player = new Player[n];
30          Checker checker = new Checker();
31
32          for(int i = 0; i < n; i++){
33              player[i] = new Player(scan.next(), scan.nextInt());
34          }
35          scan.close();
36
37          Arrays.sort(player, checker);
38          for(int i = 0; i < player.length; i++){
39              System.out.printf("%s %s\n", player[i].name, player[i].score);
40          }
41      }
42  }
```

Line: 16 Col: 50

Upload Code as File | Test against custom input | Run Code | Submit Code

Congratulations

You solved this challenge. Would you like to challenge your friends? Next Challenge

Testcase 0 | Testcase 1 | Testcase 2 | Testcase 3 | Testcase 4 | Testcase 5 | Testcase 6

Input (stdin) Download

```
5
amy 100
david 100
```

Expected Output Download

```
aleksa 150
amy 100
david 100
```

Compiler Message

```
Success
```