Applause: Coding Assignment - Tester Matching

Background

A major feature of the Applause platform is our tester-matching algorithm. We are able to drill-down from a community of over 300k *Testers* each with multiple *Devices*, to a sub-set of *Testers* that best meet a *Customer's* needs. As you can imagine, this is a complicated process that takes multiple dimensions into account.

Your goal is to write a simpler matching system, this can even be a `class`, a `module`, a `service`, etc., whatever you think is needed for this exercise, that takes two matching **Criteria** (*Country* and *Device*) and presents a sorted list of results (more on this below).

Data[-sets] Provided

- **bugs.csv**: CSV of all the *Bugs* filed by a *Tester*. Each row corresponds to a single *Bug* filed by a *Tester* and contains the *Tester* and the *Device* the *Bug* was reported on.
- **devices.csv**: CSV of all available *Devices*. Each row corresponds to a single *Device* - This is all the possible *Device* types a *Tester* can have.
- **tester_device.csv**: CSV mapping *Testers* to *Devices*. Each row maps a *Tester* to a *Device*.
- **testers.csv**: CSV of of all *Testers*. Each row corresponds to a *Tester*.

Assignment

Write an application that will match *Testers* based on a *User's* search **Criteria**. The search results should be ranked in order of *Experience*. *Experience* is measured by the number of *Bug(s)* a *Tester* filed for the given *Device(s)*.

You can use any technology as well as any third-party libraries, but be prepared to discuss your rationale behind each.

Search Criteria

- *Country*: Values should be collected from **tester.csv** and should also have an option for `"ALL"`. A *User* of your system should be able to specify one or more *Countries*. Multiple selections are treated as `OR`.
- *Device*: Values should be collected from **devices.csv** and should also have an option for `"ALL"`. A *User* of your system should be able to specify one or more *Devices*. Multiple selections are treated as `OR`.

The walk-through examples below are not based on the data[-set] provided. The output is to help explain how we derived the result(s).

Walk Through Example 1:

- **Criteria:** `Country="ALL"` and `Device="iPhone 4"`
- **Data:** 2 *Testers* (User1 and User2).
  - User1 filed 4 *Bugs* for iPhone 4.
    - 4 *Bugs* filed for *Devices* matching criteria
  - User2 filed 10 *Bugs* for iPhone 4.
    - 10 *Bugs* filed for *Devices* matching criteria
- **Result:** User2 => 10, User1 => 4

Walk Through Example 2:

- **Criteria:** `Country="ALL"` and `Device="iPhone 4"` or `Device="iPhone 5"`
- **Data:** 2 *Testers* (User1 and User2).
  - User1 filed 4 *Bugs* for iPhone 4 and 20 *Bugs* for iPhone 5.
    - 24 *Bugs* filed for *Devices* matching criteria
  - User2 filed 10 *Bugs* for iPhone 4.
    - 10 *Bugs* filed for *Devices* matching criteria
- **Result:** User1 => 24, User2 => 10

Walk Through Example 3:

- **Criteria:** `Country="US"` and `Device="ALL"`
- **Data:** 2 *Testers* (User1 and User2).
  - User1 filed 4 *Bugs* for iPhone 6
    - 4 *Bugs* filed for *Devices* matching criteria
  - User2 filed 0 *Bugs*
    - 0 *Bugs* filed for *Devices* matching criteria
- **Result:** User1 => 4, User2 => 0

Output

How you output the results is up to you, there are no specific UI/UX requirements. However, you should include the *User's* name and *Experience* in your output.

Submission

Compress entire source-code tree, or better yet, commit it to a remote repository (on GitHub, BitBucket, etc.) and share.

Please include details on how to fetch, configure and run your application (include examples!).

Questions

If you have questions or problems, feel free to e-mail us to ask. We're happy to provide input on whatever you like.

Please work with your recuiting team or for general questions reach out to:
recruiting@applause.com