OkFriend Matching Challenge

OkFriend is an online dating site has an interesting matching algorithm. You're going to implement it. See the enclosed PDF for OkFriend's description of their algorithm.

Write a program in the language of your choice that reads a set of user profiles from stdin (represented in JSON; see included example input file) and writes the top 10 matches for each user profile to stdout (also in JSON, see below), sorted in rank order.

You have 120 minutes to provide your solution (from the time we email you). Email me your program and your output from the provided input.

Important

- The guts of the algorithm is more important to me than the JSON processing (it's not a JSON processing quiz). Consider that when prioritizing if you're short on time or unfamiliar with JSON processing libraries in your language of choice.

- Your primary goal is to produce a quality implementation of the algorithm. Code should be self documenting and easy to read.

- Being time and space efficient is not as important as clear, understandable code here. But think about how space/time inefficient your simple solution is. Think about the data structures and operations you're using.

- There are several optimization opportunities in the implementation. Implement the optimizations if you have time. If you don't have time, feel free to follow up with an optimized version or some notes on how you might make the implementation more time or space efficient.

Input Format

- The 'importance' field is in the range [0,4] and is the index into an array that defines the weights as described in the OkFriend doc, e.g.:

  private static final int[] IMPORTANCE_POINTS = new int[]{0, 1, 10, 50, 250};

- Answers are always in the range [0,3]

- The size of the acceptable answer set is between 1 and 3. 0 and 4 are nonsensical.

Output Format


```
{"results": [
        {"profileId": 0, "matches": [
                {"profileId": 2, "score": 0.87},
                {"profileId": 1, "score": 0.65 }
        ]},
        {"profileId": 1, "matches": [
                {"profileId": 0, "score": 0.65},
                {"profileId": 2,"score": 0.5 }
        ]}
]}
```

HELP TOPICS

Calculating Match Percentages

What exactly those numbers mean.

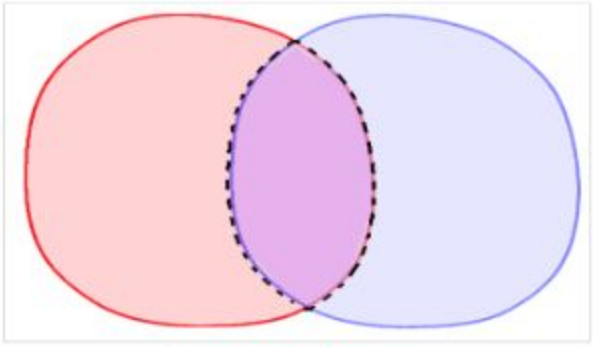
This is a brief, but technical, explanation of how your match percentages are calculated. It's a little complicated, but our method is quite interesting—even unique. Also, there's a patent pending, so no funny business.

Let's get started

We start wanting to calculate a match percentage for you and someone else. And we want to avoid mistakes at all costs! We collect three values for all users. When you answer a question on our Improve Matches page, we learn:

1. your answer,
2. how you'd like someone else to answer, and 3. how important the question is to you.

Your match percentage with a given person on OkFriend, let's call him B, is based on the values of (1), (2), and (3) for questions you've both answered. We'll call that set S later in this explanation:



Now let's look at two example questions and see how we use all this information to make a match.

Example Questions

How messy are you?

1. very messy
2. average
3. very organized

Your answer	3
How you want someone else to answer	2 or 3
The question's importance to you	Very Important
B's answer	2
How B wants someone else to answer	2
The question's importance to B	A Little Important

Have you ever cheated in a relationship?

1. yes 2. no

Your answer	2
How you want someone else to answer	2
the question's importance to you	A Little Important
B's answer	1
How B wants someone else to answer	2
The question's importance to B	Somewhat Important

Calculating The Match

First of all, since we use computers to do this, we need to assign numerical values to ideas such as “somewhat important” and “very important.” We chose the following scale:

Level of Importance	Point Value
Irrelevant	0
A little important	1
Somewhat important	10
Very important	50
Mandatory	250

When we look at how each of your answers satisfied the other’s preferences, we’ll use these values to give our calculations the correct weight. Your match percentage with B is figured by answering the following two questions:

How much did B’s answer make you happy?

You indicated that B’s answer to the first question was very important to you. And that his answer to the second question was not. So we placed 50 importance points on the first question and 1 point on the second question. Of those 51 possible points, B earned 50 by answering the first question how you wanted. So B’s answers were 50/51 = 98% satisfactory.

How much did your answers make B happy?

Well, B placed 1 importance point on your answer to the first question and 10 on your answer to the second. Of those 11, you earned 10 points. So your answers were 10/11 = 91% satisfactory.

To get a match percentage for you and B, we just multiply your satisfactions, and then take the square root: **sqrt(91% * 98%) = 94%.**

This is a mathematical expression of how happy you’d be with each other... if these two questions were the only things that mattered in a relationship!

Any questions?

Why do you multiply (as opposed to say, average) the two match scores together, to get a final score?
Because we like to think of each match percentage as the probability you'd get along. That's the product of them, assuming they're independent. Intuitively, this makes more sense anyway; two people matching each other 95% are a better match than two others who match 90% and 100%.

What if a user and I have only answered one question in common, and we happen to satisfy each other's requirements? Does that mean we're suddenly a 100% match?
Even though two users have satisfied each other on a few common questions, they may not actually be a good match. That is, while the set of questions you've both answered, S, is small, we can't have much confidence in the match percentage yielded by the above calculations. With any poll, there's a margin of error that needs accounting for, and here's how we do it:
True Match = Calculated Match +/ Reasonable Margin of Error

We've toyed with multiple formulas for confidence, as there are subtle forces at play. For example, if we're too aggressive, people with few questions answered will never show up in match results. If we're too lenient, you might see too many matches who just got lucky on a few questions. Currently, we're defining the reasonable margin of error as 1/(size of S).

In OkFriend, when the size of S = 50, meaning you and someone else have answered 50 of the same questions, and we've calculated your match to be, say, 84% based on your answers, that means your "True Match" is between 82% and 86%.

To give you the most confidence in the match process, we always publish the lowest possible percentage your match can be. In this example, that would be 82%.

So when we were comparing you and B above, your calculated match was 94.5%, but you'd only answered 2 questions. The margin of error for a S of that size is 50%! So the published match percentage of you and B would only be 44.5%, which is 94.5% – 50%, as per our "True Match" formula.

Examine the following:

Size of S	Margin of Error	Highest Possible Match
1	1.00	0%
2	0.50	50%
3	0.33	67%
4	0.25	75%
5	0.20	80%
10	0.10	90%
20	0.05	95%
50	0.02	98%
100	0.01	99%
500	0.002	99.8%
1000	0.001	99.9%

You have to answer 100 questions for a 99% match to be possible. A consequence of this is that we're highly confident in our published match scores... we've chosen the lowest statistically valid value. Our users have to tell us a lot about themselves before we can pretend like we know them.

How are questions chosen?

We have a system for sorting questions by how well they divide the population. Users are exposed typically to the best questions they haven't answered yet.

What if I check all the "acceptable answers" boxes, or none of them?

We record your answer, but the question's importance is cast as "Irrelevant" when matching people to you. Your answer may obviously still affect the match, of course.

Shouldn't "Mandatory" be some kind of filter?

Using the "Mandatory" and "Very important" votes selectively will heavily focus matches on users who meet your most important criteria. However, purely filtering matches by the "Mandatory" vote would upset many users who use the term more liberally. As a rule of thumb, save that vote for the case where you couldn't POSSIBLY IMAGINE dating someone who answered incorrectly. Still, keep an open mind.

Why can't I place different importance values on each acceptable answer?

It’s likely you would get confused and screw up your matches.

The importance values you mentioned above (0, 1, 10, 50, 250) seem wrong. I know what is important to me and I want to assign my own values. Ok?

The best way to think about those numbers is to see what they imply about the relative values of questions. For example, 10 “a little important” questions are equal to 1 “somewhat important” question. And 5 “very important” questions are worth 1 “mandatory“ question. If we let you edit them, you might put in something ridiculous like (0, 1, 2, 3, 4) and that would be bad for your matching. Nonetheless, we might add this as a feature soon... It would be very easy to program, it’s just a question of whether or not we trust you.

What besides user questions affects my match percentages?

That’s it!