FHIR Competition

July 2020

Welcome to the competition! This competition starts at 11 am on Thursday 16th July and ends at 11 am on Thursday 23rd July. There's a few administrative points you should read through before attempting the problems, detailed below.

The Data

The dataset along with sample input and output files and the recordings of the workshops are all available here (click).
This data is the only thing you should need to download over the entire competition - problems will require very little external input, excluding this data.

The dataset contains records of organizations, practitioners, patients, encounters and observations. You can find the list of attributes each of these data types have here. Each of the files in the dataset represents the set of all patients under care of a particular practitioner in a particular organization (hospital). You may have the following assumptions regarding the dataset:

- Each patient is under care of exactly one practitioner.
- All of the objects grouped together are related to each other. In other words, a bundle containing a patient and an organization and a few observations means that this patient belongs to this organization's records and these observations have been recorded for this patient.
- Each practitioner may be working in several organizations, but each patient belongs to exactly one organization.

Obviously, you should not make any changes to the dataset since it will make your output files incorrect. In case you accidentally did make changes to the dataset, you should download the dataset again.

Submissions

Submissions for this competition are made to DOMjudge. You should have received an email with a link, as well as a username and password to login. Each problem has 2 submissions - One for answer submissions, and another for code submissions.
The answer submission should be your code's output for the problem (zipped if necessary), and will be checked for validity in order to determine your score. The code submission is there to ensure the answer file was created using your own code - and we will be analysing these codes before determining prize winners. As such it is of utmost importance that you **submit to BOTH to ensure you are applicable to receive prizes**.

The DOMjudge answer submission will hang on pending - do not worry about it, it's not meant to show your submission's score until the competition ends. You can, however, use the provided sample input and output files for each question to make sure your program is working and generating the output file as expected.
At the end of the competition, the actual scores of all best submissions from each competitor will be released, independent of DOMjudge.

If your code is not a simple program written in a widely used programming language (e.g. a single python file), please provide in a ZIP file the instructions of how your code should be run.

Please note that you should **strictly follow the output formats** specified in the problem statements. If a problem states that the output file should be answers to the queries separated by empty lines and query numbers at the beginning, you should generate your output exactly as such or it will not be marked correctly. For your convenience, each problem comes with a sample output file so you can see the format of output your program is expected to generate.

You can login with the username and password sent to you by email here to submit your files.

The competition starts at 11:00 am 16 July 2020 and ends at 11:00 am 23 July 2020 i.e. no more submissions will be accepted after 11 am on 23rd July.

Code Run-time

All problems listed can be solved by a program in less than 10 minutes on a low-end laptop, so don't leave code running overnight trying to beat a specific input, try instead to optimise your code (All problems are solvable with run-time complexity linear with the number of patients).
If there are many high scoring contestants, then prizes will likely be decided based on the most efficient code submissions.

Scoring

Scoring for competition submissions are based on how many inputs your program has generated the correct output for (excluding Q1, which is binary in outcome). Each problem has a maximum score of 100. This isn't the only thing considered when determining prizes, as mentioned above.

For each question (excluding Q1) you will get an input file, containing many queries formatted as explained in the statement of the problem. Your program needs to generate answers to each query and then put all of the answers with the format specified in the problem statement in a file - which you will then submit to the judge. Your score for the problem will be calculated based on the number of queries you answered correctly. Hence, it is better to submit answers to at least a few queries if you can rather than not attempting the problem.

Although your code efficiency in terms of time may be taken into consideration for breaking ties in the scoreboard, your submission time will not be considered as a scoring factor. Two submissions that both generate the same output and are similar in terms of efficiency will always get the same score even if one is submitted in the first day of the competition and the other is submitted in the last day.

Questions During the Contest

When you login to the judge website, you will find a section for clarification requests on the right hand side. You can use it when you have questions to ask. You should click on the ‘request clarification‘ button and write your message. It is very important to have these points in mind before sending a clarification request:

- This section should only be used for clarification requests for the problems not any question you might have with the contest. Questions like "why isn’t my program working?" and other help requests of such will not be answered.
- Questions will be answered typically in a couple of days. Do not expect fast replies here as there are a lot of contestants.
- Some of the questions you ask may be answered as an announcement instead of a direct reply to you. Some of the questions other people ask too might be seen to be frequently asked and therefore answered as an announcement so all contestant can see them. In both of those cases, you should have a look at announcements before submitting a new clarification request as there might be an announcement sent already answering your question.

FAQ
When will the results of the contest be released?
- One week after the competition ends. We will inform you about that by an email.

I have submitted my program and its output but the judge shows ‘pending‘ as my verdict. How long is this going to take?
- You will be provided with the correct output for the sample inputs you will be given and you can check your program’s correctness locally. The judge will not mark your programs until the competition ends.

When does the competition ends? When is the last time we can make submissions.
- Thursday 23rd July, 11 am.

1 Patient Genders

Hospital administration would like some broad metrics about the patients currently admitted to the system. Each patient in the dataset is indicated as active or inactive. This report is going to be only about the currently active patients.

In this part, we are looking only at the information we have on patients' genders and we want to know how many patients in the dataset identify as male, how many identify as female and how many identify as other.

Problem - Gender Classification

Create a program that reads all patients data and reports the number of patients in each of the classes (of those that are active).

Input and Output

This problem does not have any input files.

For the output, your file should show the number of active patients in each category in this format:

```
# <category>:
<answer>
{empty line}
# <category>:
<answer>
{empty line}
# <category>:
<answer>
{empty line}
```

Here category is one of male/female/other.
The order in which you print the categories does not matter. To see an example of how the output file should look like please see the file sample/Q1/output.txt.

2 Hospital Substitution

As a part of protocol for unexpected emergency situations, we need to know the options available for moving a patient from their current hospital (or more generally, their current associated organization) to another. Given the data available, we assume that any two organizations with the same postal code are suitable options for patients to be transferred between them.

Problem - Individual Patient Transfer

Write a program which, when given a patient ID as input, generates the most suitable replacement organization. The best replacement organization is that with the least currently active patients (Then sorted alphabetically by name to break ties). If no such replacement exists, then print 'None'.

It is guaranteed that each patient has exactly one associated organization in their bundle in the dataset.

Input and Output

The input file for this problem contains several lines, each line containing only the ID of the patient to be queried. It will look like this:

```
<patient id for the first query>
<patient id for the second query>
...
```

To see an example of how the input file will look like please see the file sample/Q2/input.txt.
For the output, your file should show the ID of the best replacement organization for each query in this format:

```
Patient <patient id>:
<answer organization id>
{empty line}
Patient <patient id>:
<answer organization id>
{empty line}
...
```

The order in which you print the answers to the queries does not matter. To see an example of how the output file should look like please see the file sample/Q2/output.txt.

3 Healthy Patients

Practitioners have recently seen an influx of new patients under their care, and as such need some new insights into the health and blood quality of these patients, for their own safety as well as to evaluate their eligibility for blood donation.

Problem - Healthy Patients

There are a number of useful entries per patient in the data-set. In order to evaluate the health of patients, the following metric is used:

$$\text{BH} := \frac{(\text{Systolic Blood Pressure} - 2 \times \text{Diastolic Blood Pressure})^2}{\text{Low Density Lipoprotein Cholesterol}}$$

All three of these entries are defined in observations. Patients can have 0, 1, or multiple readings for each of these datapoints - always favour the information in an observation with a later 'effectiveDateTime'.
If a patient has no reading for any of these entries, then they are given a BH of 0.
Your program should take a practitioner ID as input, and output the 3 patients with the highest 'BH' value.

Input and Output
The input file for this problem contains several lines, each line containing only the ID of the practitioner to be queried. It will look like this:

```
<practitioner id for the first query>
<practitioner id for the second query>
...
```

To see an example of how the input file will look like please see the file sample/Q3/input.txt.
For the output, your file should print the IDs of the three best patients, numbered 1 to 3, after each practitioner:

```
Practitioner <practitioner id>:
1: <patient id>
2: <patient id>
3: <patient id>
{empty line}
Practitioner <practitioner id>:
1: <patient id>
2: <patient id>
3: <patient id>
{empty line}
...
```

The order in which you print the answers to the queries does not matter. To see an example of how the output file should look like please see the file sample/Q3/output.txt.

4 Patient Pairing

Even still, it seems that patients in urgent need of blood may not be able to be connected to the optimal donor in time, and so we wish to begin an initiative to pair up like patients with a shared practitioner, so that if one requires blood, the other can always give.

Problem - Patient Partners

As mentioned above, we want to pair patients with the same practitioners with similar blood characteristics. Patients A and B can only be paired if the difference between their 'Leukocytes [#/volume] in Blood by Automated count' is less than 1 Million per millilitre. If a patient has no readings for leukocytes, then they cannot be paired. If a patient has multiple readings for leukocytes, use the one with the latest 'effective-DateTime'.

Subtask 1:
Create a program which takes a practitioner id as input, and then attempts to pair as many patients as possible given the above rules. The program should output all pairs of patient IDs.

Subtask 2:
In order to measure the viability of patient pairings, we measure the likelihood of successful transfer by:

$$V(A, B) := 1 - \frac{(L_A - L_B)^2}{1e^{12}}$$

Where L_A and L_B measure the Leukocytes per millilitre of blood for patients A and B respectively. Complete Subtask 1, now maximising the sum of all viability scores between patient pairs. (If patients A, B and C, D are paired, then the total score is $V(A, B) + V(C, D)$.

Input and Output
The input file for this problem contains several lines, each line containing only the ID of the practitioner to be queried - exactly the same as Q3.
To see an example of how the input file will look like please see the file sample/Q4/input.txt.
For the output, your file should print the pairs of patients:

```
Practitioner <practitioner id>:
<patient id> <patient id>
<patient id> <patient id>
<patient id> <patient id>
{empty line}
...
```

The order in which you print the answers to the queries does not matter, the order of the pairs, or the inner ordering of each pair, does not matter.
To see an example of how the output file should look like please see the file sample/Q4/P1/output.txt or sample/Q4/P2/output.txt.