Digital Health Competition

July 2020

Welcome to the competition! This competition starts at Tuesday 8th Dec, 14:30 and ends at Tuesday 15th Dec, 14:30. 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.
The dataset is organised into bundles; One for each datatype.
Each bundle then contains all instances of that particular object, be it organizations, practitioners, patients, etc.
We are particularly interested in a few links between the objects:

- A patient has a single `managingOrganization`, which denotes the ID of the organization the patient is linked to.
- A patient can have multiple `generalPractitioners`. These practitioners will all work for the `managingOrganization` (This 'working' relationship is not explicitly present in the data).
- Multiple encounters occur per patient. Encounters have a list of `participants` (Practitioners) and a single `subject` (Patient). The `participants` of an encounter will always be `generalPractitioners` of the `subject`.
- Multiple observations occur per encounter. Observations are linked to Encounters by `context`, as well as the patient, via `subject`.

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, and must be a `.txt` file. This file will be checked for validity in order to determine your score at the end of the contest. The code submission (which must be a .zip file) is there to ensure the answer file

was created using your own code - and we will be analysing these code submissions before determining prize winners. As such it is of utmost importance that you **submit to BOTH the answer and code 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 scores of each contestants 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 the .zip file 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 test 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.

The competition starts at Tuesday 8th Dec, 14:30 and ends at Tuesday 15th Dec, 14:30 i.e. **no more submissions will be accepted after Tuesday 15th Dec, 14:30**.

Code Run-time

For each problem, our solution program solves the problem in less than 10 minutes on our home computers, 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 $n\sqrt{n}$ or less, where n is the number of objects in the data).

Scoring

Each problem needs to be solved for a number of test cases. As such you will be scored based on how many test cases you can solve for a particular input. **You do not need to submit all test case answers in a submission for it to be scored.** For example, you can just submit answers for test case 1, 3 and 7 and get the score for these 3 results. As such if certain test cases are too large for your solution to handle, feel free to skip them.
You will receive a mark for your submission based on the percentage of test cases you solve. The relationship between test cases solved and marks awarded is not linear however. To encourage having a go, but also to stress the importance of particularly hard test cases, we use this curve to model the relationship between correct answers and marks awarded. So the first 20% of correct test cases are worth 40% of the marks, as are the final 20% of correct test cases.
As an example, correctly guessing 80% of the test case answers will only net you 57.58% of the marks available for that question.

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 're-quest 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, and not any question you might have with the contest. Questions like "why isn't my program working?" and other help requests unrelated to the specific contest problems will not be answered.

- Questions will be answered typically in a couple of days. Do not expect fast replies here as the competition runs over a long timespan.
- 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?
- This will stay as pending. Domjudge is simply there for us to accept submissions, not mark them. You can check the correctness of your program using the sample input and output.

When does the competition end? When is the last time we can make submissions?.
- **Tuesday 15th Dec, 14:30**

Problem 1 - Passport Number

A new study you wish to conduct requires some patients to move to another country, in order to use the latest equipment. To do this, you need to collect the patient's passport number, to get them a visa if needed.

You decide to automate the process to fetch any patient's passport ID.

Problem - Passport Number **(40 marks)**

Write a program which takes as input the ID of a patient and prints their passport number in the output. The passport number is one of the items in the "identifier" field, with the type code of "PPN". If the passport number for the given patient is recorded, print it. Otherwise, print "Not Recorded". (Output IS case sensitive)

Input and Output

The first line of the input will be a single integer N denoting the number of queries. Each of the next N lines will contain a single string, the patient ID for that query.

Print N lines in the output, each line showing either the passport number of the patient or, if no passport number is recorded for the patient, "Not Recorded".

Example problem

For the following input:

```
2
00765d42-bf48-45cc-8c80-d659da47142e
01299d69-c6fb-4b5a-8f9d-574d4b61b119
```

The correct output is:

```
Test 1: X74182220X
Test 2: Not Recorded
```

Problem 2 - Inconsistent Observations

In some recent studies conducted by your organization, you've found some inconsistencies in the smoking status reported for subjects. Looking into the issue, you find that some of the observations record the patient as never having smoked, despite previous observations contradicting this.

In particular, the problem is with the Tobacco smoking status observations (loinc code 72166-2). Apparently, some patients show "Never smoker" as their status in an observation while they have had an older observation showing status as a current or former smoker. We are interested in knowing which patients have these inconsistencies in their observations so we can exclude them from smoking-related studies.

The list of all possible values and codes for smoking status observations can be found here. The observation codes of interest are "Never smoker" (code 266919005) and "Unknown if ever smoked" (code 266927001). A particular patient has inconsistent records if they have at least 2 observations, where one observation has entry "Never smoker" and an earlier observation has entry which is neither "Never smoker" nor "Unknown if ever smoked".

For example a patient with observation "Never smoker" 2 days ago, and an observation "Former smoker" from last week would be considered an inconsistency.
A newer observation is one with a more recent "effectiveDateTime". It has nothing to do with the related Encounter object.

Problem - Organization-level inconsistency **(60 marks)**

Organisations want to know how badly this inconsistency has affected previous studies. You are expected to write a program where an organization ID is given to your program as input and, the program writes to output the number of patients being managed by that organization with inconsistent records.
(A patient is managed by an organization if the "managingOrganization" field of the patient is the organization's ID).

Input and Output
The first line of the input will be an integer N denoting the number of queries. Each of the next N lines will contain a single string, the organization ID for that query.

Print N lines in the output, each line containing a single integer showing the number of patients managed by that organization with inconsistent records.

Example problem
Suppose we have 2 organizations with IDs H123 and G123. H123 has 3 patients, Alice, Bob and Carly, and G123 has 2 patients, David and Emily.
The following observations are then recorded, in the following order:

- Bob is recorded as "Never Smoker"
- David is recorded as "Former Smoker"
- Bob is recorded as "Former Smoker"
- Emily is recorded as "Never Smoker"
- Emily is recorded as "Current Light tobacco smoker"

- Bob is recorded as "Never Smoker"

For the following input:

```
2
H123
G123
```

The correct output is:

```
Test 1: 1
Test 2: 0
```

As the only inconsistency stems from Bob.

Problem 3 - Limited Serum

Bad news! You've just discovered that everyone will be turned to stone in 1 day!
Good news! You've discovered a serum which would make you immune!
Bad news! You've only made a limited amount, and it takes too long to create more!

You find that the amount of serum needed to make someone immune correlates with their Platelet mean volume reading (loinc code 32623-1). In particular, if someone has a Platelet mean volume reading of x fL (fL = $10^9/\text{mm}^3$), then

$$20x + 10$$

millilitres of serum are required.

Problem - Maximum Saved **(40 marks)**

You decide it is best to administer this serum to your patients (with their consent) since you already know this quantity for many of them.
You can administer serum to a patient if:

1. The patient is under your care. (you are in their list of 'generalPractioner's)
2. The patient has an 'Observation' record listing their Platelet mean volume.
3. You have enough serum to make the patient immune.

If the patient has multiple Platelet mean volume observations, then use the most recent value in your calculations (The Observation with the most recent 'effectiveDateTime').
How many patients can you save?

Complexity note: Each patient can have at most 5 'generalPractitioner' entries.

Input and Output
The first line of input is the number of tests.
Every test is a single line containing the ID of a single practitioner and an integer $0 \leq S \leq 10^9$, the amount of serum you have, measured in millilitres, separated by a space.

For each test case your output should be one line; the maximum number of patients that this practitioner can save, given that they start with S millilitres of serum. (Prepended by "Test x: " where x is the test number, starting at 1.)

Example problem
Suppose we have a practitioner Senku (with ID 1sh1g4m1), and 4 patients, Beatrice, Carly, David and Eustace, with the following characteristics:

- Beatrice is under the care of Senku, and her latest measurement of Platelet mean volume was 4.2.
- Carly is under the care of Senku, but she **never** has had her Platelet mean volume measured.
- David is **not** under the care of Senku, and his latest measurement of Platelet mean volume was 0.2.
- Eustace is under the care of Senku, and his latest measurement of Platelet mean volume was 4.0.

Then for the following input:

```
2
1sh1g4m1 120
1sh1g4m1 200
```

The correct output is:

```
Test 1: 1
Test 2: 2
```

This is because Senku cannot apply the serum to Carly and David, but can apply it to Beatrice or Eustace. Applying the serum to both of them would require 184 millilitres of serum, so in the first test case you have enough to apply to Beatrice (94) or Eustace (90) by themselves.

Problem - Maximum Chance of Civilization **(60 marks)**

Thinking a bit further ahead, you don't just want to maximise the number of people saved, you want to maximise the chance that the people saved are going to be able to restart civilization.

For any set of people you save, you denote the 'Civilization score' to be the minimum of the number of men and the number of women (So a set with 3 men and 5 women has Civilization score 3).

Following the same rules as before, your aim is to maximise this Civilization score, and **then** maximise the number of people saved (So you prefer saving 2 women and 2 men over 1 woman and 10000 men).

Input and Output

As with before, you receive the ID of the practitioner and an integer $0 \leq S \leq 10^9$, the amount of serum in millilitres on a single line per test case.

Your output per test case should be one line; The maximum Civilization score they can achieve, and the maximum number of patients they can save with this civilization score, separated by a space.

Example Problem

Assume we have the same patients and practitioners as before (plus a two extra patients, Frank and George), but instead:

- All patients are under the care of Senku.
- Beatrice's last measurement of Platelet mean volume was 17.9.
- Carly's last measurement of Platelet mean volume was 15.6.
- David's last measurement of Platelet mean volume was 5.3.
- Eustace's last measurement of Platelet mean volume was 4.6.
- Frank's last measurement of Platelet mean volume was 4.8.
- George's last measurement of Platelet mean volume was 4.2.

Then for the following input:

```
1
1sh1g4m1 550
```

The correct output is:

```
Test 1: 1 3
```

Because while Senku can save all 4 men, he maximises the Civilization score by saving Carly, and two of the men instead.

Problem 4 - Patient Discounts

Practitioners are trialling a new scheme where repeat patients receive discounts, in the form of single use coupons. However, practitioners don't want to give two coupons to the same patient. As such the practitioners need to devise a scheme whereby all practitioners are able to give away their coupons, without any patient receiving multiple.
Furthermore, a practitioner can only give a coupon to a patient if that patient is under their care (They are in the patients list of ‘generalPractitioners‘).

Problem 1 - Best patients **(40 marks)**

Each practitioner receives 3 coupons. It is then agreed that most reasonable scheme for allocating coupons is for each practitioner to give a coupon to their top 3 patients.

For each practitioner, patients are ranked by the number of encounters where the ‘subject‘ is the patient, and the ‘participant‘ list includes the practitioner. Ties in the number of encounters are then broken by preferring whichever patient has an encounter with the practitioner with an earlier ‘period/start‘ time (So long term patients are rewarded).

The practitioner then gives a coupon each to their top 3 ranked patients.

Your task is to calculate whether this scheme allows for every practitioner to get rid of their 3 coupons, without any patient receiving 2 or more.

Complexity note: Each patient can have at most 5 ‘generalPractitioner‘ entries, and so each patient has at most 5 practitioners that show up in encounters.

Input and Output
Your input to this problem will be a positive integer N, the number of tests, followed by N lines.
Each of the next N lines will contain the ID of an organization.

For test number x (1 to N), your output should be one line:

- ‘Test x: POSSIBLE‘ if all practitioners working for this organization can get rid of their coupons (Assuming they follow the strategy above, and no other practitioners receive coupons).
- ‘Test x: IMPOSSIBLE‘ if following the above strategy a practitioner working for this organization still has a coupon, or some patient receives 2 or more coupons from practitioners in this organization.

Example problem

Suppose we have 4 patients, Alice, Bob, Candice, Darren, and 2 practitioners, Edward and Fiona. Edward and Fiona both work for organization Greater Health (ID G123), while Edward also works for Hospitals r us (ID H123).
Then the following encounters occur, in the following order:

- Alice sees Edward for a blood test,
- Bob sees Fiona for a routine checkup,
- Bob sees Edward for a blood test,
- Candice sees Fiona for a prescription,

- Candice sees Edward for a routine checkup,
- Darren sees Edward for a routine checkup,
- Darren sees Edward for a blood test.

Then for the following input:

```
2
G123
H123
```

The correct output is:

```
Test 1: IMPOSSIBLE
Test 2: POSSIBLE
```

This is because Edward's top 3 patients are Darren, Alice, and Bob, and Fiona's top 3 patients are Bob and Candice. For organization G123, this is problematic for two reasons:

- Bob receives 2 coupons.
- Fiona still has a coupon remaining.

But for H123, Edward can give his 3 coupons to Darren, Alice and Bob.

Problem 2 - Any scheme **(60 marks)**

Because our previous scheme had some failures, the practitioners would like to know if there is some other scheme where they get rid of all their coupons.
Rather than each practitioner giving a coupon to the top 3 patients, each practitioner can now give their coupons to whomever they want, as long as an encounter exists with ‘subject‘ the patient and ‘participant‘ the practitioner. The question still remains, can everyone get rid of their coupons?

Because this set of rules is relaxed, each practitioner needs to give away 10 coupons, rather than 3.

Input and Output
The input, and expected output, follow the same format as before.