Software Engineering Internship
Take Home Project

Overview

Hi - We're excited that you're interested in joining the identifeye HEALTH team as a Software Engineering Intern this summer! As part of our interview process, we'd like you to build a small project that meets the requirements listed below.

This is designed to be a short exercise, so please don't spend too much time on it. At a high-level, we want to see that you've thought through the problem, developed an easy-to-understand solution, and tested your work to ensure accuracy.

The final steps in the interview process will include an introduction to two of our Software Engineering team members, where you will be able to talk through your project and explain the decisions you made.

Project Description

We want you to create a simple service that aggregates patient and exam data. Your project should be able to read in a .txt file that conforms to the specifications below and then outputs a summary as described in the "Output Specification" section of this document.

Technical Specifications

- Please feel free to use whatever programming language you feel most comfortable in
- You can assume that all input files and commands are properly formatted (i.e. you **do not** need to handle any errors or edge cases beyond what is stated in this document)
- Please write some sort of test case(s) for your project that demonstrates it works as you expect
- Please store all data in memory (**do not** use a database, etc)
- If anything is unclear in this document, feel free to make an assumption - please just add a comment in your code to let us know what your assumption was
- How you structure your code is up to you, but please submit it as a link to a public git repo (Github, GitLab, Bitbucket, SourceHut, etc) including a meaningful commit history.
- Keep it simple and straightforward
- Have fun with this!

Input File Specification

The input file will be a .txt file containing a series of instructions that will instruct your program to: add a patient record, add an exam record, delete a patient record, or delete an exam record. There will be one instruction on each line in the file. Each instruction can further be broken down into a series of **space-delimited** segments (detailed below for each type of instruction):

Add a Patient Record

The command to add a new patient record looks like this:

```
ADD PATIENT 123 JOHN DOE
```

The command will begin with the word ADD, followed by the word PATIENT, followed by a patient identifier, and finally any remaining text on this line should be considered the patient's name.

Edge case: If a patient with the given identifier already exists, then ignore this command and continue processing the rest of the file.

Add an Exam Record

The command to add a new exam record looks like this:

```
ADD EXAM 123 456
```

The command will begin with the word ADD, followed by the word EXAM, followed by a patient identifier, and then an exam identifier.

Edge case: If a patient with the given patient identifier does **not** already exist, then ignore this command and continue processing the rest of the file.

Edge case: If an exam with the given exam identifier already exists, then ignore this command and continue processing the rest of the file.

Delete a Patient Record

The command to delete a patient record looks like this:

```
DEL PATIENT 123
```

The command will begin with the word DEL, followed by the word PATIENT, followed by a patient identifier. This command should remove the patient record from your system **along with** any exams that have been created for the patient.

Edge case: If a patient with the given identifier does **not** already exist, then ignore this command and continue processing the rest of the file.

Delete an Exam Record

The command to delete an exam record looks like this:

```
DEL EXAM 456
```

The command will begin with the word DEL, followed by the word EXAM, followed by an exam identifier. This command should remove the exam record from your system.

Edge case: If an exam with the given identifier does **not** already exist, then ignore this command and continue processing the rest of the file.

Output Specification

When you've finished processing the file, your program should print out a summary of the patients in your system. This should include the patient identifier, the patient name, and the number of exams that each patient has in your system. For example, you could output something like this for each patient:

```
Name: JOHN DOE, Id: 123, Exam Count: 1
```

Example

Given the following input:

```
ADD PATIENT 123 JOHN DOE
ADD PATIENT 321 JOE SCMOE
ADD PATIENT 321 JOHN SNOW
ADD PATIENT 789 JANE CROW
ADD EXAM 321 444
ADD EXAM 789 445
ADD EXAM 789 554
DEL PATIENT 321
```

Your program should output the following:

```
Name: JOHN DOE, Id: 123, Exam Count: 0
Name: JANE CROW, Id: 789, Exam Count: 2
```