Data Structure Design

Haverford CS 106 - Introduction to Data Structures

Lab 1 (two weeks)

1 Designing a Data Structure

1.1 Understanding the Data

When designing any data structure, we first need to understand the details of the data that we will store and what types of queries we may want to make about that data. For this lab, the data we'll be working with comes from a ProPublica story about a risk assessment tool called COMPAS. In order to understand the data, start by reading the article.

1. Read the ProPublica article:
   `https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing`
   and the document describing how they did their analysis:
   `https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm`

2. Look at the (cleaned) data in "compas-scores.csv".

1.2 Determining Names and Adding Fields

1. Create a class that will hold the information from a single row in the CSV. Think carefully about what to name it, based on your understanding of the data, so that it correctly indicates what data is being stored. Using Javadoc style, write a comment to describe the contents of this class.

2. Determine what the constructor for this class should take as input. Add any necessary fields, getters, and setters to the class. Think carefully about what the types of each of these fields should be and remember to include `enum` types as a possibility in your thinking. Be sure to use appropriate coding style and to add Javadoc comments as appropriate.

3. Make a main class with a main method and use it to manually create a row object to test your work. Continue to test your work this way throughout this lab.

1.3 Adding Methods

1. Determine what methods (beyond the constructors, getters, and setters you have already created) should be added to the class holding a row that you made in order to help you replicate ProPublica's analysis shown in the chart, "Prediction Fails Differently for Black Defendants." Add and implement these methods. Since you are only storing one row of the data for this lab, you will not yet be able to replicate this analysis (we will do that in the next lab).

2. Update the comments to the class you created to include descriptive details about the data that will be stored as well as the queries that can be performed on that data.

2 Data Validation

We have not yet imported the data from the csv into our data structure. Before we do (in the next lab), we're gong to put some safeguards in place to make sure that we don't have any data errors when we do the import from the csv.

1. Create a second constructor for the class holding a row of the data that takes only Strings as input — this will be useful when reading in from the file in the next lab.

2. Determine what the valid options are for each field you've added to your row class (you'll likely want to look at "compas-scores.csv" again). Use Javadoc style to document these preconditions.

3. Add methods to ensure that all fields only ever hold valid values. Think carefully about what method(s) this validity should be ensured within. Invalid values should throw an exception.

4. Talk to someone else in the class and find out what preconditions they've decided on for each of the fields. Are theirs the same as yours? Update any preconditions in the comments if you would like to, or document any potential conflicts between the two of you in what you believe the data should look like.

5. Test that this worked by creating both a valid row and an invalid row within your main method.