Parsing Uniprot files—with ArrayLists

In this assignment you will further modify the code you wrote by replacing arrays with Java’s ArrayList data structure. There are some additional modifications to the interface that your classes need to have. As in the previous assignment, the parse method in the UniprotParser class needs to return a ProteinDatabase object. The ProteinDatabase needs to have the following instance variables and methods (note that you can have additional supporting methods, but these are the ones we will check for):

- An ArrayList of Protein objects.
- A constructor that receives no parameters. You will use an add method to add proteins to the database instead.
- An add method that receives a Protein object, and adds it to the database. The your parse method will use this method to add proteins to the database as they are being parsed.
- A find method that receives an AC number and returns the protein that has that AC number, or null if that protein does not exist in the database (i.e. returns a Protein object).
- A findProteinLength method that returns the protein whose length is closest to the given length parameter (what should it do if the database has no proteins in it?) If there are several proteins with that given length, return one of them.
- A length method that returns the number of proteins in the database.
- A findDuplicates that returns an ArrayList with the AC numbers of all proteins that have a duplicate in the database. Two proteins are defined to be duplicates if they have the same sequence (this array shouldn’t have duplicate AC numbers!) In the previous assignment you had to do a nested for loop to find which proteins had a duplicate in the database. In this assignment you need to program this using the contains method of ArrayList. To do this you will need to be clever about how you define the .equals method of the Protein class.
- A toString method that prints out something useful.
Specifications, notes, and hints

Your program needs to meet the following specifications:

• You need to submit three files: `Protein.java`, `UniprotParser.java`, and `ProteinDatabase.java`. We will use our own driver class in order to test your classes. We recommend you write your own driver class in order to test that your program is working correctly. To test the `findDuplicates` we suggest creating a custom Uniprot file that indeed has duplicates. Custom files for testing other functionalities are also advised.

• Except for the requested changes the `UniprotParser` and `Protein` should be structured as in the second assignment.

• Put comments at the top of the file with your name, EID, date and course, and a short (one or two line) description of what the program does. We will be testing the code on the machines in the CS computer lab, so make sure your code runs on those machines.

• All instance variables should be private.

• Submit your source code files via the checkin program by the due date (read the course syllabus for the late policy).