College of the North Atlantic
CP3566 – Applied Java Programming
Assignment 1

Due Date: Sunday, January 23th

Part 1: Database Setup

It's assumed that your database setup is complete on your development machine. But if not, Install and configure MariaDB by following the installation guide posted on D2L.

Part 2: Book Database

Using HeidiSQL (or your management interface of choice), which was configured in Part 1, set up the **books** database by following these steps:

1. Open HeidiSQK
2. Create a new database called "books"
3. Open a query in the books database and copy in all the SQL statements contained in the "books.sql" file. (Alternatively, use another method of your choice)
4. Execute all the statements and confirm that the data has been correctly inserted into the database. The select statements at the end of the script should echo out all the data that you can see in the script.

   Note: You can remove the "drop" statements at the beginning of the SQL file the first time you run the script. Also, you can remove the select statements at the end as you populate the database. Technically, you can leave both in, but you will see errors related to the drop statements the first time, as there will be nothing to drop.

Part 3: Using the Book Database in Java

1. Create a "Book" class which will represent the books from the "titles" table in the books database. Each book object will have a list of its authors – a private attribute *List<Author> authorList*. (Make sure to populate the authorList attribute when loading the objects from the database.)
2. Create an "Author" class which will represent the authors from the "authors" table in the books database. Each author object will have a list of the books they have written – a private attribute *List<Book> bookList*. (Make sure to populate the bookList attribute when loading the objects from the database.)
3. Create a "BookDatabaseManager" class which will handle all database connections and queries as well as store lists of Books and Authors to be used by the main application. It is recommended that you do the following:
   - Stores the database URL and login as static private members. Note, any time you use a database connection ensure that you close the connection when finished – don't allow connections to persist.
   - Create a private **loadDatabase** method that coordinates the queries for both the books and authors and their relationships. Note: you can use other private methods to simplify the code. (Open and close the database connection within this method.)

- To simplify this method, you may want to create two additional methods (e.g. loadBooks and loadAuthors).
- Create an addNewBook method and addNewAuthor method that inserts a book or author into the database from user input form the terminal.

NOTE: We will not be concerned with any update queries. To simply the solution it is assumed that any entered authors or books are legitimate (e.g. we don't care if two authors have the same name).

Part 4: Create a Simple Application

1. Create a Java class called "BookApplication" with a main method that prompts the user to select any of the following options:
    1. Print all the books from the database (showing the authors)
    2. Print all the authors from the database (showing the books)
    3. Add a book to the database for an existing author
    4. Add a new author
    5. Quit

2. Note that the user should be able to continue making choices until they quit.

Export your project as a zip file and submit to the dropbox.

Below is a sample solution that outlines the necessary methods/classes. Note that you could use additional methods or a different approach to achieve the same solution or to simplify code, feel free to modify as necessary.