Milestone 2

due at midnight on Mon Feb 15 (125 points)

For this milestone, you can continue working in your sparkdemo project, but I will ask you to create new classes and programs within that project. Please pay close attention to class and method names — following my instructions accurately will make it more convenient and less error-prone for me to evaluate your code.

Commit and push to the Git server as often as you like — it's a good way to keep backups of your work. When you have a commit candidate that you think is your final submission, **please include #milestone2 in the first line of the commit message** — I will search for that when figuring out what to grade.

1. Create a class called DatabaseConfig. The entire purpose of this class is to hold a static constant for the JDBC **connection string** that other classes should use. Something like this:

```
public class DatabaseConfig {
    public static final String JDBC_TEST_URL = "jdbc:h2:~/tmp/cs164db";
}
```

The actual filename specified after h2: can be whatever you like, but the name of the constant should be exactly JDBC_TEST_URL. This makes it easier for me to plug in a DB name that is suitable for my system, and to distinguish different students' databases from one another.

2. Create a class called SampleProducts, and a static main method within it. When run, this class should connect to the database (using the JDBC_TEST_URL) and do a few things:

   a. If a table called product exists already, just remove all its records and delete it. This can be done with the single SQL statement:

   ```
   DROP TABLE IF EXISTS product;
   ```

   b. Now create a table called product. The table should contain an integer id, a varchar name, a decimal price, and a text description. It's up to you to determine the precise SQL syntax, but see the comment at the top of my ProductDemo class for an example.

   c. Insert 5 or more sample products into the product table. The names can be whatever you want. At least two of them should have non-null, non-blank description fields.

The effect of doing steps a, b, c (in that order) is that, whenever this program is run, the sample product table will be recreated from scratch. That's dangerous when using on a live database, but very helpful to have for testing.

3. Create a class called `ProductBrowser` and a static `main` method within it. When run, this should start a Spark server. It will work very similarly to the ProductDemo I did in class on 8 February. The home page should list all the products in the database, in alphabetical order by name. They should be hyperlinks to a product detail page, which displays the name, price, and description.

   The `ProductBrowser` class should also make use of `DatabaseConfig.JDBC_TEST_URL` to connect to the database.