Activity 11-3: TDD Work Plan

Project: _____

Feature: _____

Feature Owner: _____

Date: _____

Initial Planning

1. **With your teammates**, discuss the initial design of your system with respect to the features that you will be working on for this (and maybe the next) iteration.

   a. What information will the system need to be able to implement the feature(s)?

   b. Can any of this information be represented by a single primitive datatype (i.e., a single integer or string)?

   c. Is any of this information *complex*, i.e., more than can be represented by a single primitive data type? If so, explain what pieces of information are closely related and need to be grouped together.

d. For each complex piece of information identified above, what would be a good name for a class to represent that data?

e. For each piece of information needed for a given feature, where is that information coming from? Is it received from another system component (e.g., a database)? Is it information the user enters? Does the feature produce the information?

f. For information that comes from or is being sent to another system component, how can you mock the other component so that your tests and implementation can focus on the current feature(s) for this iteration?

g. What additional classes do you think will be needed to support the implementation of the features for this iteration?

h. Will any of the identified classes be used in the implementation of more than one feature? If so, decide who will implement them. These classes will need to be implmenented before any others.

2. Write down the name of each class identified so far that you will need to implement for the feature you are in charge of.

3. **With your coding partner**, decide on the first 1-3 tests that you will need to write to drive the implementation of the class. Write down names for these tests.

   When thinking of what tests to write, remember the strategies discussed so far:

   - fake it 'til you make it
   - stairstep tests
   - assert first
   - triangulation

4. Prioritize the tests that you have listed so far. Starting with the first test you plan to write, list each test along with the class it is testing on the TODO sheet for this feature.

    - **As you and your partner are working, you must be dilligent about maintaining your TODO List.** After you finish the "Red-Green-Refactor" cycle for a test, check off the test and related tasks you have completed on the TOOO list. Every time you think of a new test, write it down on the TOOO list (even if it is the very next test you plan to write).

5. Get a "Red-Green-Refactor" pyramid and sit down at a lab computer with your partner. Make sure you have these planning notes, the TOOO list for your feature, something to write with, and some scrap paper. **Whoever is in charge of the feature should login.**

6. Open a web browser and go to: https://github.com/sdp-resources

7. Start intelliJ, then *File -> New -> Project from Version Control -> Git*.

   TODO: Need a few more instructions here about getting the project setup.

Test-Driven Development with Paired Programming

TODO: Additional information here, including committing changes to the repository.