**Swap 3 HW**

**CSSE375-02&03, Spring 2014**

This is the **final** of **multiple “swap” activities** among small teams, doing coding as a HW activity.

The learning goals are as follows:

1. Considering strategic system goals in deciding what to refactor.
2. Evaluating the “quality” of peers’ code, in these ways.
3. Using their recommendations in adding new features. (And *supplying* recommendations to the team you gave your code to.)
4. Applying more of the specific coding skills we’re learning in 375, about refactoring.

This time, you’ll be adding features and fixing bugs to a system that came from Swap 2.

At the end of **this week**, again, you will pass your code to another team. They will then do the following:

1. Grade your code immediately, based on the rubrics of that particular code swap.
2. Add to and bug fix your code, making it “their code” for the next week. This time, bug fixing doesn’t count as doing the assignment, per se, but let me know if you run into serious bugs that must be fixed. Otherwise, note them in the bug list below the rubric.

* **Repeat Warning:** We’ll do the “exchange of code” during next Tuesday’s class, so it really MUST be done and running by then!

**Here’s what will we do during the third week of this Swap HW assignment:**

I. “Today,” on Monday, Mar 31, do the following in class:

1. **Exchange 2**: See the revised “Teams for everything - Projects and Swaps!” Spreadsheet on Moodle. Notice that you will be emailing a zip file to one team (send it to both people – or all 3), and receiving one from a different team!   
     
   Note: Do not zap every copy of the code you are sending to another team. You may well want to see it, later on! (E.g., if they send you questions, in today’s grading.)
2. Also during class today, use the Swap 2 Code Dropbox to turn in, to me and the TA’s, the same zip file.
3. Revise your code-sharing system so that you are now sharing the new program you received.
4. Bring your laptops: Both partners on a team need to get the program running during class today, and verify what functions you believe it does or doesn’t do. If the team who worked on it changed something, and it just works in a different way, this gives you a chance to clarify that with them.
5. Start grading it, to see how that feels. Use the rubric for Swap 2, the one included in *last* week’s assignment. Attach the bug list below it, if there are bugs. **This is what you will turn in to the Swap 2 dropbox by 11:55 tonight.** Be sure you graded everything worth points, so that it adds up to as many points as the other team should get!
6. Ask questions about the enhancements (“features”) they recommended but didn’t add (see *below*).

II. During the week, do the following things with your partner, to your copy of this new code, verifying after each activity that the system still runs:

1. On every coding change or comment you make, **precede that line or area of code with a comment line** which says: // SWAP 3, TEAM XX where XX is your team number. This way, it’s easy for the next team to find all the changes you made, and use the rubric to grade those.
2. **In Swap 2, the other team was asked to refactor as many of the 10 code smells as possible – up to 7.**   
     
   They were also told in both Swap 1 and Swap 2, “On each SMELL you identify, also **describe the kind of enhancement** that the change would enable. **They were told not to do those changes**, but to comment them in enough detail that someone else could do so.”  
     
   Ok – Now, for Swap 3, your job is to take them up on those recommendations, and make the enhancements. Whatever were the 7 or so enhancements they thought some refactoring would enable, do just the refactoring for **4 of these**. Pick the **best** 4, so you don’t have to add in the comments that one of them was an unworkable feature, and all you can show is a best try at it. What are the best 4? Those that appear to meet some strategic system goals for this system, and what it’s supposed to do. The more important the feature is, to the success of the system, to more reason you should do it. So, you may want to start off listing all 7 smells they found, and the features they thought would be enabled by fixing them, and decide what is important.  
     
   On each, put a comment line like, // ENHANCEMENT FROM REFACTORING. Then, in your comments go on to describe the following:
   1. What *really* their refactoring enabled. If they claimed it would do great and glorious things, but the feature you could add with the refactoring is actually much more modest, say so, and say why. (You actually have both Swap 1 and Swap 2 advice to go on, for many of these possible enhancements.)
   2. In comments, also give your analysis of how successful the refactoring was, toward building this enhancement!
   3. Be sure to describe what strategic goal of the system your enhancement relates to, how it adds value to the system relative to the other things the system is supposed to do.

Note: If they did bug fixes for Swap 2, in lieu of doing all 7 of the refactorings, you may be left with fewer areas to choose from. If this appears to be a problem in the code you got, please let me know ASAP.

Let me know if you have questions!

**The “turnin” for this assignment**

1. As with Swaps 1 and 2, this will be “due” at the start of a class, in this case, on the following Tuesday, Apr 8. That’s a hard deadline, because if you miss it, the team getting it will have nothing to grade!.
2. In class, you will find out the names for the team you will send it to. You can email the zipped files to them directly, at the start of class.
3. During class, you also need to turn in the zipped files to me, in a drop box on Moodle! One copy per team.

**The grading**

Here’s the rubric to use in grading the system you receive. If there’s time, you can start grading during class on Tuesday, Apr 8. You will need to finish grading, and turn-in your rubric form as a .docx file, in the Moodle drop box for this, by 11:55 pm that Tuesday night.

SWAP 3 RUBRIC:

CSSE 375 Section # \_\_\_\_ From Team # \_\_\_\_ , Graded by Team # \_\_\_\_

|  |  |  |  |
| --- | --- | --- | --- |
| **Capability / Feature / Assignment** | **Grade (“works perfectly” or “done perfectly” = full credit in each case)** | **Grader Comments – Explain anything that is missing, doesn’t work, or has issues or concerns.** | **Rubric Comments and Examples** |
| **0. It still runs** – Does all the “Required” and “Additional” features for the original CSSE221 assignment, that it did before the team changed it. | \_\_\_\_ out of -30 |  | This time, not running is a negative score. Take off for things that don’t run. If they don’t have anything to give you next Tuesday, don’t show up in class, or the whole thing blows up when you try to run it, extend this grade to a 0 on the whole assignment. |
| 0’. They made wholesale changes to the system that were not asked for. | \_\_\_\_ out of -100 |  | This is a negative grade if they did it. **Zero** if they avoided the temptation to do this. -100 if they rewrote the whole program without commenting where they made changes. -30 if they changed key data structures, like how the calendar is represented. |
| 0’’. They didn’t follow the documentation guidelines to make it easy for you to grade this. | \_\_\_\_ out of -20 |  | Also a negative grade. Take off some points if they didn’t put the ENHANCEMENT 1… comment lines in. |
| B. They developed a reasonably-sized feature, for each of 4 different places that the code had been refactored in Swap 2. | \_\_\_\_ out of 20 for each of the 4 asked for (80 total) | Please make a list, so they can see clearly what you took off for. | Grade proportionally if you believe they could have tried harder to build on the refactoring, or if the resulting “feature” they created seems trivial or doesn’t run. Note: In this case, duplicate refactoring types are **ok**, but they should not be too easy. |
| C. Feature explanation | \_\_\_\_ out of 5 for each of the 4 asked for (20 total) | Explain why you agree or disagree with their value. | Did they pick features that accomplish **strategic system goals**? Did they explain why the enhancements they chose add value to the system, so that you understand their function relative to other things the system does. |
| Total | \_\_\_\_ out of 100 |  |  |

Below the table, in the .docx that you hand in, note additional bugs found that need to be fixed:

1. Description:
2. Etc.

Please observe that I also plan to grade you on the quality of your grading, based on my (or my TAs’) own review of the same code!

**The last development cycle is this Swap 3!**

Next Tuesday we’ll swap one last time, to do the grading of these.