Homework 4

Code Reviews

In case you want to do HW4, please signup as a pair by **11:59 pm CDT on Friday, April 14.**

HW4 is due by **11:59 pm CDT on Thursday, April 27.** (note that HW4 is optional).

Why Do Code Reviews?

The purpose of code reviews is to have a fresh pair of eyes look at your code. Having someone new look at your code usually identifies a whole bunch of problems that you might have otherwise missed. Some motivations for doing code reviews are: finding faults, suggesting code improvements, transferring knowledge and suggesting alternate solutions.

Running a successful Code Review

- Read Chapter 21.3 of Code Complete 2 for the guidelines on how to run a successful code review.
- Revisit the lecture portions (Lecture 15) that discuss how to go about doing a code review. The instructions below summarize the lecture; if you find something unclear between the lecture and the instructions, please post on piazza.

Signing up and Organization

As many students did not want this homework on peer code reviews, this homework is optional. For this homework, you need to first pair up with a member from your project team. In case you wish to do the homework, please sign up on Homework 4 Signup (Requires Login), by mentioning the NetIDs of the pair separated by a comma, by the sign up deadline specified above. If your team has an odd number of members and all of them want to do the homework, then one of the "pairs" may include three members. Once the pairs have signed up, we will team up pairs from different project groups to conduct an inter-team review. The teams for the homework will be updated on the Homework 4 Signup page (Requires Login) once everyone has signed up.

Getting Started

Once the teams have been updated by us on the Homework 4 Signup page (Requires Login) after the sign up phase, contact the pair that has been matched to your pair. This homework will involve two exercises-- (1) your code will be reviewed by the pair matched to yours and (2) their code will be reviewed by you. Specifically,

1. Each pair will play the role of a moderator/author/scribe in one review. We shall refer to this pair as MAS in the following paragraphs.
2. Each pair will also play the role of the reviewers in another review. We shall refer to this pair as RR in the following paragraphs.

Before the Review

Things to do for the MAS pair

- You should decide (together with RR) on a time and location for the code review meeting. The duration of the meeting should be **1 hour.** You are required to e-mail this location/time to your team's supervising TA as well in case the TA is able to stop by (but you don't have to include the TA's schedule when deciding a time).
- You should select a component to be reviewed by RR. A component consists of several classes and test cases that address some user story. The component should neither be too simple nor too complicated. A good rule of thumb is to have about 6 classes (and their corresponding test cases). Ideally, the MAS pair should have written the code under review, and the RR pair should not have.
- You should provide the selected component to RR at least 48 hours before the review. Make sure there is a canonical, stable version of the component for review so that everyone is on the same page. The best way to do this is to use the tags feature in version control. So you should tag the code under review for (or send a zip copy of that code to) RR. This allows development to continue in parallel with the review.
- Finally, you should also provide to RR a short description of the user story that the component is supposed to address.
Things to do for the RR pair

- You should prepare a checklist of things to review. The checklist should not be too long (you should be able to discuss all issues in the checklist during the review session). However, the checklist must be specific enough so that the grader can identify what problems you are trying to detect. Since there are at least 2 students in RR, it's a good idea for each student to come up with a checklist independently and then merge them, to maximize the number of things you check for during the review. The checklist needs to address:
  1. Design: Check if there are code smells both within classes and between classes. The list of design smells should help. Here is another generic example of good coding practices. Feel free to be more specific and augment them depending on your technologies.
  2. Tests: Check if the test code actually tests the system adequately. For instance, the test cases should address the cases when something is working properly (i.e., the ideal case) and when something is not working as expected (for example, when an exception has occurred).
  3. Requirements: Check if the requirements are met as stated in the use case or the user stories. For instance, the goals of the stakeholders must be met. You will be graded on the quality of your checklist. The examples above give an idea of what we are looking for.

- You must read the code given by MAS before the review meeting. You need to spend at least an hour reading the code for the component that MAS has selected. When reading the code, please annotate problems that you discover and bring them up during the review. A successful review requires the RR team to be familiar with the code beforehand.
- You can request additional artifacts from MAS if necessary. MAS must provide them in a timely manner.
- If MAS does not provide the code to be reviewed in a timely manner, please e-mail the TA to report this.

During the Review

MAS pair

- Someone should play the role of an author/scribe and the other student(s) should play the role of a moderator. The author will start off by talking briefly about the component that will be reviewed. It's a good idea to ensure that everyone can see the code. So either print a copy of the code for everyone or use a projector to project the code.

  - The moderator will lead the meeting. The moderator should lead the reviewers through separate classes/tests. The moderator should also ensure that the meeting progresses, and the discussion on each class is not too long. It might be a good idea to run the code/tests to show that they actually work. Code that doesn't work is also something to watch out for. A moderator plays an important role in the review session. Some tips for the moderator:
    1. Be decisive – If a discussion doesn't seem to be getting anywhere, it might be time to move onto a new topic.
    2. Be assertive – Ask the reviewers to tell you what problems they see. There are no silent reviewers – everyone has to say something.
    3. Be in control – When necessary, let the author know that he/she needs to stop justifying the design.
    4. Ask, don't tell – Don't waste too much time explaining the code to the reviewers. The reviewers are supposed to read the code beforehand. Instead ask the reviewers what they saw in their initial reading of the code. Remember the purpose of this meeting is to find as many defects as possible and not to discuss solutions with the reviewers.

- The scribe will write down the issues that the reviewers bring up during the meeting. As a pair, you will be graded on the quality of the meeting notes/transcripts.
- If RR appear to have not read the code, please e-mail the TA to report this. It's better to report this early than to have points taken off later.

RR pair

- The reviewers (who have read the code beforehand) will bring up the issues they had after reading the code. They can ask the author for clarifications about certain parts of the code. But in general, the author (who is acting as the scribe) should remain quiet through the meeting.
- It is important to bring up serious problems that are discovered during the review even if RR might have missed them during the initial code reading.
- If MAS appears to be not prepared, please e-mail the TA to report this. It's better to report this early than to have points taken off later.

After the Review

Use the checklist on page 491 of Code Complete 2 to help determine if the code review meeting was productive.

MAS pair

- MAS should compile a transcript of the review meeting. Make sure that each issue that was brought up by the reviewers is recorded.
- MAS should augment the transcript with a discussion of the severity of each problem and what actions they are going to take to solve the problem. MAS can also justify why they did things in that manner and why they might not want to change their implementation, if that
is the case.

**RR pair**
- RR pair should provide to the MAS pair a copy of their checklist for inclusion in the submission.

**Switch Roles**

MAS and RR will now switch roles and perform the same exercise as above with the switched roles.

**Deliverables**

MAS will be responsible for the submission of a report. The submission should have:

1. a description of the user story being reviewed.
2. a copy of the checklist from RR.
3. a copy of the code that was reviewed. Please mark/circle the areas that were addressed/discussed during the meeting (use the marking / annotation features of PDF tools). Doing so provides traceability – so we know which part of the code caused an issue.
4. a copy of the meeting transcript together with a discussion on the severity of each problem identified and actions they plan to take to remedy them.

**Submission**

Each pair will participate in two submissions. However, each pair will be responsible for sending in only one single submission (this is the submission in which they play the MAS role). To submit, email your report to your TA by the homework deadline. **Your submission must be in PDF**. You are responsible for making sure that you can convert or export your work to PDF.

**Scores**

This homework is purely optional. Students can however get some extra credit for doing HW4. We have decided to assign 2 points for HW4 (out of 100 points final grade for the whole course).