Activity 11-2 Rules about working on the project

Grading

Your grade on the project is evaluated as follows. First, there are three items on which you receive "individual" grades.

1. You can ask to *lead* a feature. This means that you'll own this feature and bring it to completion. 5% of the final grade is a "participation grade" for leading the estimated number of features. For example if there are 18 features and 12 students, every student is expected to lead at least one feature. Some students will end up leading 2 features, but they don't get more points for that.
2. You are expected to *pair up* with others on their issues. Again this is a "participation grade" expecting you to pair up with someone the expected number of times. This is another 5% of the final grade.
3. You will each evaluate every other member of your team at the end of the project, with regards to the teamwork they exhibited. We will in particular ask you about the following:

   - How communicative were your teammates when you partnered with them?
   - Did they look for and respect your input and your ideas?
   - Did they offer you their ideas and in general contribute to the progress of the project?

   These assessments, along with our reflections on them, will account for another 5% of your final grade.

Next, there are two items on which you will receive "group" grades.

1. An evaluation of the quality of the entire codebase, considering elements such as proper naming, complete and proper testing, simple design, use of design patterns where suitable, etc. This will count for 20% of the final grade (a bit less than half the overall project grade), to emphasize its importance. *It is the responsibility of every member of the team to care for the health of the codebase.*
2. An evaluation of the degree to which the project accomplished its deliverables (i.e. features completed). This will take into account the difficulty level of the project and it will not simply be "number of features completed", and it will count for another 10% of the final grade. It is everyone's responsibility to help bring as many features as possible to completion.

The takeaways are:

- Working on your own projects is just as valuable as helping others with their projects.
- Keeping your codebase clean is at least as important as completing features, probably even more important.

Mechanics (Fall)

- We will be working on 2-week iterations. This means that we have 2 weeks to complete features and produce something to show to our customer, however small.
- You will be putting in 4 hours of work a week, split into two 2-hour sessions. You will work with a different partner for each session.
- At any given point half of the team members will be leading their features, and the other half will be pairing up with them.
- We will be working in 25-minute intervals of focused work, called "tomatoes" because of a tomato-shaped timer that the inventor of this technique used. During those 25 minutes you *protect the tomato* and don't let any distractions get in the way.
- A tomato is followed by a 5-minute break, during which you should get up and walk around, and clear your mind in preparation of the next tomato.
- We can carry out roughly 4 tomatoes on each 2-hour session.
- When you pair up, one of you will be at the keyboard while the other will be observing and offering suggestions, and making sure that the "peripheral tools" we will discuss in a minute are properly maintained.
- You should be logged in the system with the account of the owner of the feature you are working on.
- You should switch roles with your partner after each tomato.
- Peripherals: You should keep with you and maintain the following:
  - The red-green-refactor pyramid. Make sure to advance it as you go through the items.
  - A sheet holding your TODO list of things that you need to do. For example, tests that you need to write, refactorings that you need to do when you pass the green phase, other classes that you might need to create in the future, questions you have for the rest of the team, etc. *This sheet should be kept on the team's board.*
  - One or more sheets where you can draw basic diagrams to communicate.
- Working with the repository:
  - Check out the newest version of the code at the beginning of your session, via the menu VCS -> Update Project. Make sure to select the options Rebase and Using Stash.
  - As you do your work, create commits by using the VCS -> Commit option. Make sure only the changes you have made are included; IntelliJ will try to add some other files of its own.

Mechanics (Spring)

- We will be working on 2-day iterations. This means that we have 2 afternoons to complete features and produce something to show to our customer, however small.
- We will be starting each afternoon session at 12:45. The time in the afternoon will be divided into roughly half-hour intervals.
- At any given point half of the team members will be leading their features, and the other half will be pairing up with them.
- We will be working in 25-minute intervals of focused work, called "tomatoes" because of a tomato-shaped timer that the inventor of this technique used. During those 25 minutes you *protect the tomato* and don't let any distractions get in the way.
- A tomato is followed by a 5-minute break, during which you should get up and walk around, and clear your mind in preparation of the next tomato.
- We can carry out roughly 8 tomatoes each day. You will work with the same partner for 4 tomatoes, then switch partners for the remaining of the day.
- When you pair up, one of you will be at the keyboard while the other will be observing and offering suggestions, and making sure that the "peripheral tools" we will discuss in a minute are properly maintained.
- You should be logged in the system with the account of the owner of the feature you are working on.
- You should switch roles with your partner after each tomato.
- Peripherals: You should keep with you and maintain the following:
  - The red-green-refactor pyramid. Make sure to advance it as you go through the items.
  - A sheet holding your TODO list of things that you need to do. For example, tests that you need to write, refactorings that you need to do when you pass the green phase, other classes that you might need to create in the future, questions you have for the rest of the team, etc.
  - One or more sheets where you can draw basic diagrams to communicate.