Intro to Git (with Eclipse)

Part 1: Creating and initializing a repository

1. Open Eclipse and import the example project: File -> Import -> General -> Existing Projects into Workspace.
2. Set the root directory to the location of the sample Java project (e.g., /Users/seanbayley/Desktop/gittutorial/example). Click *Finish*.
3. Go to github.com and click *New repository*. Give the repository a name, **ensure it is set to public**, and click *Create repository.* Make note of the repository URI (e.g., https://github.com/smbayley/gittutorial.git)
4. In Eclipse, open the Git perspective: Window -> Perspective -> Open perspective -> Other -> Git.
5. Click *Create a new local git repository.* Set the repository directory to the location of the sample Java project. Click *Finish*.
6. In the *Git Repositories* view, click on the repository you just created. You should now see unstaged changes in the *Git Staging* view. Select all of the unstaged files, right click, and select *Add to index*. All of the files should now be listed as staged changes.
7. Add a commit message (e.g., "initial commit") and click *Commit and push*.
8. Leave the remote name as "origin" and copy/paste the GitHub repository URI into the *URI* field. Leave everything else as default. Optionally you can add your GitHub username and password, otherwise it will prompt you for this information every time you try to do a GitHub operation. Click *Next.*
9. Leave everything as default and click *Next.* **Depending on your version of Eclipse, you might need to select a "source ref":** click the source ref dropdown and select "master [branch]." Click *Add Spec.*
10. Click *Finish*.
11. Once complete, click *OK.*
12. Head back over to github.com and verify that the project has been successfully pushed to the repository (you'll need to refresh the page if you left it open from before).

Part 2: Creating a new branch

Using git **branches** is useful for parallel development, and is a good way to make sure that there is always a "stable" copy of the project. Often times, new features are implemented in a separate branch and only once they've been fully implemented and tested are they **merged** back into master.

1. In Eclipse, switch to the Git perspective and click the drop-down arrow on your repository. Right click on "Branches." Select Switch to -> New Branch. Name the branch "feature" and click Finish.

Part 3: Committing changes

1. Switch to the Java perspective and open SimpleCalculator.java.
2. Implement the stubbed methods and save your changes.
3. Switch back to the Git perspective. You should see that SimpleCalculator is now listed in unstaged changes. Right click the file and select *Add to Index*.
4. Note: before you commit, it's a good idea to **pull**: someone else might have made changes to the same file that you were just editing. In this case, it isn't necessary because you're the only one editing anything. You can pull by right clicking the repository in the Git

perspective and clicking *Pull*. This will update your local copy of the project with any changes that have been pushed by other collaborators.
5. Add a commit message. The message should be meaningful and explain what was changed: this will make it easier to find bugs later on.
6. Click *Commit and push*.
7. If prompted: copy and paste the repository URI and click *Next*. Leave the branch name as "feature", enter in your GitHub username and password and click *Next*.
8. Click *Finish*.
9. Go to the repository on github.com and verify that you have two branches: feature (with the changes to SimpleCalculator.java) and master (with no changes).

Part 4: Merging branches
Now that you've finished implementing the new feature, it's time to merge the changes back into the master branch.

1. Open Eclipse and go to the Git perspective.
2. Right click on *Branches* and select *Switch To* -> master
3. Click on the Branches dropdown arrow.
4. Click on the Local dropdown arrow.
5. Right click on the feature branch and select *Merge*.
6. Click *OK*.
7. Verify that SimpleCalculator.java is updated with the new changes.
8. Right click on the master branch and select *Push Branch.*
9. Click *Next*.
10. Click *Finish*.
11. Verify that the changes have been pushed by checking the repository on github.com.

Part 5: Group Exercise
1. Find 3-4 people and select one person's repository to work on. **If it is your own repository, skip steps 2-5.**
2. In Eclipse, delete your project from the Package Explorer and delete the repository from the Git perspective.
3. Select File -> Import -> Git -> Projects from Git. Click *Next*.
4. Select *Clone URI* and click *Next*.
5. Copy the URI of the repository you selected. Fill in your username and password and click *Next*.
6. Click *Next*.
7. Select a location to save the project and click *Next*.
8. Click *Next*.
9. Click *Finish*.
10. The project should now be accessible in the Eclipse Package Explorer view.
11. Start making changes to SimpleCalculator.java: add new methods, change the implementation of existing ones, etc.
12. Commit your changes. Remember to pull before you push!
13. If you run into problems (hopefully you do) and you can't sort them out, ask questions. The goal is to make sure everyone's changes are finally reflected on the master branch.