Objectives

In this practical you will analyze various Java programs and modify them to fix the errors and to implement small new features to the program. This practical assignment is designed to help you review the latest material that was covered in the course.

Reading Assignment

If you have not done so already, please read all of the relevant “GitHub Guides”, available at https://guides.github.com/, that explain how to use many of GitHub’s features. In particular, please make sure that you have read guides such as “Mastering Markdown” and “Documenting Your Projects on GitHub”; each of them will help you to understand how to use both GitHub and GitHub Classroom. To learn more about the concepts associated with writing static methods, using different types of loops, and utilizing switch statements and arrays, please study the content in Chapters 6, 7 and 8.

Analyzing and Modifying Given Programs

After you have accepted the practical assignment and cloned your repository for it, locate three Java programs inside the “src” directory in your practical’s repository. Study each program and make sure you understand what each program is attempting to do. Pay a close attention to the programs, where the methods are written in and called from the same class. For example, printOne method in the CallingMethodsInSameClass class is called from CallingMethodsInSameClass class. printOne is a static method, which allows us to call it without creating an instance of the object first. To call the methods from within the same class, you need to specify the method name and parameters if any (as is done with the printOne method); if you call static methods from another class, you need to specify the class name then use a dot operator followed by the method name (e.g., Math.pow(4,8)).

All of the given programs contain one or more errors, which may be compile-time errors found when you compile the program or run-time errors found when you run the program. Locate all of the errors and fix them to get all of the programs to compile and run.

Next, find // TO DO: comment lines in each program and add Java statements to satisfy the tasks specified in the // TO DO:. Compile and run each modified program and save the output.

You should test each program one at a time by running the Gradle’s run command. It is currently configured to run the CallingMethodsInSameClass class. To change the Gradle set up to run a different class, you should open build.gradle file and uncomment the appropriate line that starts with “//mainClassName = 'practicalten.’”.

Checking the Correctness of Your Program and Writing

As in the past assignments, you are provided with an automated tool for checking the quality of your source code. Please note that the practical assignments do not require you to produce a writing document as you do in the laboratory assignments. However, to check your Java source code you can started with the use of GatorGrader, type the command "gradle grade" in your terminal window. If you do have mistakes in your assignment, then you will need to review GatorGrader’s output, find the mistake, and try to fix it. Specifically, don’t forget to add in the required writing! If you are having trouble running GatorGrader locally, don’t forget to ensure that you still transfer all of your source code to GitHub. Please see the course instructor if you have questions about this step.

Once your program is building correctly, fulfilling at least some of the assignment’s requirements, you should commit and push your files to GitHub. When you use the “git push” command to transfer your source code to your GitHub repository, Travis CI will initialize a “build” of your assignment, checking to see if it meets all of the requirements.

This assignment invites you to submit, using GitHub, the following deliverables.

1. A properly documented and completed version of the `CallingMethodsInSameClass` program.
2. A properly documented and completed version of the `LoopStyles` program.
3. A properly documented and completed version of the `SwitchDay` program.

Evaluation of Your Practical Assignment

Practical assignments are graded on a completion — or “checkmark” — basis. If your GitHub repository has a ✓ for the last commit before the deadline then you will receive the highest possible grade for the assignment. Please see the course instructor if you do not understand how practical assignments are graded or you do not know how to complete one of the specific tasks in this assignment. Finally, remember that, in adherence to the Honor Code, students should complete this practical assignment on an individual basis. Please see the course instructor if you have any questions about this course policy.