COMP 210 **Object-Oriented Programming**

Lab 1 & Homework 1

January 16 *Spring 2019*

Abstract

For your first assignment you'll use Java to write the kind of procedural code you wrote in C++. This is not how we plan to actually use Java throughout the course. The point of this exercise is to get your basic tools up and running (Eclipse, JUnit, and Git) and to get used to some differences between key statements in Java and C++.

1 The Program

Between lab 1 and homework 1 you'll design, implement, and test the following procedures:

- A function named *sumTo* that computes the sum off all the integers from 0 to $n-1$ when given n. In mathematical terms it computes this: $\sum_{i=0}^{n-1} i$
- An output procedure named *fizzbuzz* that carries out the classic fizzbuzz problem[1]. This procedure takes the PrintStream and a positive integer n and for each number i from 1 until n it will print:
  - *Fizz* if the number is divisible by 3 but not 5
  - *Buzz* if the number is divisible by 5 but not 3
  - *FizzBuzz* if the number is divisible by 3 and 5
  - i otherwise (its value, not the letter i)

  Output should be comma separated and on one line. There should be no comma after the final output.
- A mutator named *rmvAll* which takes a StringBuilder (basically a mutable string) and a character and remove all occurrences of the character forming the string.
- An input procedure named *readUntil* which takes a Scanner, a string and a StringBuilder and reads all the strings from the Scanner until the string argument is encountered or until there are no more strings, whichever comes first. The strings read are appended to the StringBuilder without any separation.

2 Lab 1

Your goal in lab is to get your project setup, main stubbed, the procedures declared, documented, and stubbed, and the JUnit tests stubbed (letting Eclipse do the work). Once this is done, be sure you can run your stubbed out code. You can leave the tests and *main* blank for the lab portion — in the homework you will move on to writing tests and completing *main*.

2.1 Getting the Starter Code

- Assuming you've signed up for Github and sent me your username, you should have received an email about this lab. I have made a repository at `https://github.com/MC-comp210-s19/lab1-starter`, though there's not much there. When you follow the instructions in the email, Github will "fork" this starter repository into a new repository just for you.
- It will be helpful to have Eclipse show you some information about Git repositories, so in Eclipse click `Window > Show View > Other... > Git > Git Repositories`. You should also try the "Git Staging" view.
- In the "Git Repositories" click the button for "Clone a Git Repository and add the clone to this view. (Hover over the buttons to find the right one).

[1] This is a common interview question for entry-level programming jobs.

- Enter the URL of *your* repository, and your Github username and password. Hit "Next" twice.
- At the "Local Destination" screen you can change the directory for this. I highly suggest changing it to a new folder in your existing Eclipse workspace. Click "Finish".
- Now create a new project just like we have discussed in class (and in the notes), but *make sure* the location is the same as your Git repository.
- Right-click the repository's name in the "Git Repositories" view and click `Switch To... > New Branch`. Call the new branch "submission".

2.2 Submitting Your Code

At the end of lab or when you finish the lab work, whichever comes first, you'll need to submit your code. The first step is "pushing" your changes to your Github repository:

- In the "Git Repositories" view, click the button with two pluses (or the single plus if there are some files you don't want to push).
- Type in a descriptive message in the "Commit Message" box, e.g., "stubbed out procedures and unit tests".
- Click the "Commit and Push" button.

Almost done. Now we need to create a "pull request" with the "New Pull Request" button. Make sure the request goes between your new branch ("submission") and the "master" branch, NOT between your repository and the starter code repository. This makes sure no one else can see your pull request. Review the changes and make the pull request. From here I will look at your pull request, grade the code and give comments directly on the pull request.

3 Homework 1

Implement all the procedures and use them in *main*. You should use all the above procedures in *main*, but beyond that it doesn't matter. Debug as needed.

When you have finished, create another pull request with your changes. Due before class time of the next lab.