CSCE 156 – Lab 1: Java Introduction

Summer2018

0. Prior to the Laboratory

- Review the user documentation at Eclipse's website http://help.eclipse.org/

1. Objectives

Following the lab, you should be able to:

- Log into the network using a Windows machine and your CSE account.
- Write a simple program in a selected IDE, compile, and execute that program.
- Learn how to use CSE *webhandin* and *webgrader*.
- Get a basic idea of source-code control.

2. Topics Covered

- CSE Account Management
- Eclipse IDE
- Data types and Variables
- CSE webgrader
- Console I/O
- Mathematical Expressions
- CSE webhandin
- Source-code Control

Activity 1: Administrative Stuff

Note: Skip thisactivityif you already have a CSE account. If you do not have your password, you can get it reset at the Student Resource Center (Avery 12).

1. **Setup your CSE Account:**If you do not have a CSE account, you will receive an account sheet from the from one of the system administrators at the start of the lab.
2. **Login:** Log into the network using a Windows machine by entering your CSE account name (as indicated on account sheet) and by entering the password exactly as typed on the account sheet.
3. **Change Password:** Click on the "options" button. Click on "Change password." Follow instructions to create new password.
4. **Fill Consent Form:** Using the web browser of your choice, login to your accountat https://cse.unl.edu/account. Click on the "Consent Form" and sign it by accepting the terms and conditions. This form covers the ethical use of the resources made available to you by the department. You have one week to electronically sign this after which time, if still unsigned, your account will expire.
5. For further information, you can visit the CSE FAQ at http://cse.unl.edu/faq

Activity 2: Editing, Compiling, and Running a Simple Java Program in an IDE

Eclipse is a popular, industry-standard Integrated Development Environment (IDE). IDEs make development easier by providing code mark-up, code completion, and other useful features in a GUI environment. This activity will familiarize you with the Eclipse environment. Eclipse is free and open source so you can/should download it (from http://eclipse.org) and install it on your own machine(s).

Instructions

1. Download all the *.java* files from Canvas to your desktop.
2. Start Eclipse and select a directory to use as your workspace on your *CSE Z: drive*
3. Create a new project:
   - Right-click the Project Explorer area at the left
   - Select New -> Project… -> Java -> Java Project -> Next
   - Name your project Lab01, click Finish
4. Create a new package. Java classes are stored in a hierarchy of packages. This is done for better logical organization and for package visibility reasons (to be discussed later).
   - Right-click the src (source) folder and select New -> Package
   - Name your package `unl.cse.labs.lab01`
   - Click Finish
5. Create a new class. All Java code must be contained in a class. This is in contrast to other paradigms that may allow global variables or treat functions as “first-class citizens” (functions can exist without an object or a class).
   - Right-click your new package and select New -> Class
   - Name your class `Statistics`
6. Open the downloaded `Statistics.java` file and cut and paste its contents into your new `Statistics` class.
7. We can also directly import an external Java file. Drag and drop the StatisticsDemo.java file from your desktop to the unl.cse.labs.lab01 package. Make sure that “copy files” is selected and click Ok.
8. The `StatisticsDemo` class contains a main method,
   `public static void main(String args[])`
   In Java, classes can only be executed if the main method is defined. Classes without a main method can be used by other classes, but they cannot be run by themselves as an *entry point* for the Java Virtual Machine. Run the `StatisticsDemo` as follows.
   - Open The `StatisticsDemo` class file and then click on the “play” button (a green circle with a “play” arrow on it).
   - The output for this program will appear in the lower part of Eclipse in the “console” tab.
   - Click on the console tab and enter the input as specified.

Activity 2A: Completing the Statistics Program

Though the program runs, it does not output correct answers. You will need to modify these classes to complete the program.

1. Implement the `getMax` method in the Statistics class. Use the `getMin` method for directions on syntax.
2. Implement the `getSum` method in the Statistics class. Use the other methods for direction on syntax.

Activity 2B: Modifying the Statistics Program

The program you've completed is *interactive* in that it prompts the user for input. You will now change the program to instead use *command line arguments* to read in the list of numbers directly from the command line.

Command line arguments are available to your main method through the `args` array of Strings. The size of this array can be obtained by using `args.length` which returns an integer. Modify your code to iterate through this array and convert the arguments to integers using the following snippet of code:

```
for(...) {
array[i] = Integer.parseInt(args[i]);
}
```

Of course you will need to write the for-loop above yourself. Use the previous activity (2A) as a guide.

The "command line" may not be apparent as you are using an IDE. However, it is still available to you. Instead of clicking the "Play" button to run your program, click the down arrow right next to it. Then select "Run Configurations". This brings up a dialog box with which you can run custom configurations. Click the Arguments tab and enter a space-delimited list of numbers under "Program Arguments". And click Run.

Activity 2C: Submitting and Grading Your Program

Now that you've completed your statistics program, you will hand it in and grade yourself using the *webhandin* and *webgrader*applications.

1. Open a web browser to the following URL:
   https://cse.unl.edu/handin/
2. Login with your CSE credentials and select the appropriate course (CSE156). From the list of open assignments listed, click on the "*0a. Lab01-Activity2C-Statistics Program*" assignment.
3. Follow the instructions and submit your source files (`Statistics.java` and `StatisticsDemo.java`). To find these files on your lab computer, you can right-click them in Eclipse and go to "Properties" to find the file locations on the file system.
4. Grade your program by going to the following URL:
   https://cse.unl.edu/~cse156/grade/

Enter your CSE credentials and select this lab assignment from the Assignment drop-down menu.

5. Click on "Grade Me!"
6. Observe the result: the grader runs several test cases and presents your program's output along with the expected output.
7. Click on the "Help" button for some frequently asked questions.

The *webhandin* and *webgrader* will be used to grade all of your assignments so you should get used to using them. In general, assignments can be handed in and graded as many times as you need prior to the due dates. *Take full advantage of this tool!*

Demonstrate the *webgrader* output to a lab instructor and have them *sign* off on your worksheet.

Activity 3: Introduction to Source Code Control

As you develop software and make changes, add features, fix bugs, it is very useful to have a mechanism to keep track of changes and to ensure that your code base and artifacts are well-protected by being stored on a reliable server (or multiple servers). This allows you access to historic versions of your application's code in case something breaks or to "roll-back" to a previous version if a critical bug is found. Such a management system is called a version control system.

You may be already familiar with a few online (or "cloud") storage systems such as Google Drive or Dropbox that allow you to share and even collaborate on documents and other files. However, a version control system is a lot more. It essentially keeps track of all changes made to a project and allows users to work in large teams on very complex projects while minimizing the conflicts between changes. Each team member can have their own working copy of the project code without interfering with other developer's copies or the main trunk. There are several widely used version control systems including CVS (Concurrent Versions System), SVN (Apache Subversion), and GIT. GIT, however, is a decentralized system; multiple servers can act as repositories, but each copy on each developer's own machine is also a complete revision copy. Code commits are committed to the local repository. Merging a branch into another requires a push/pull request. Decentralizing the system means that anyone's machine can act as a code repository and can lead to wider collaboration and independence since different parties are no longer dependent on one master repository.

(Adapted from GIT Tutorial at http://cse.unl.edu/~cbourke/)

You would be using GIT for version control and collaborating with your team members in future labs, assignments and projects. Look up more about source-code control on internet and answer the questions on the worksheet.