SE2890: Software Engineering Practices
Lab 1: Java Refresher and Introduction to Software Process
Due: 11:59 PM CDT March 13, 2013

1. Objectives

- Review the usage of the Java Programming language through the construction of a simple Java program using multiple classes
- Perform rudimentary effort estimating and project tracking techniques to a personal project
- Analyze the accuracy of effort estimates

2. Activities

This is an individual lab; each student is to submit a separate report. However, you are encouraged to discuss your work with others and to consult with the instructor as necessary.

In this lab, you will:

- Create a small java program using the java development environment
- Track your time by development phase
- Report on your results

3. Lab Requirements

For this lab, you are to construct a Java program which is capable of storing a list of numbers and calculating statistical information about those numbers. The exact design of the program is up to you, but the program must consist of at least two classes.

In specific, your program must perform the following:

- Start from the command line prompt
- Accept parameters on the command line prompt, which list the names of text files that contain lists of numbers.
- For each file read in, the program shall be capable of calculating the average, the maximum, the minimum, and the standard deviation for the list of numbers read in, as well as a count of the numbers read in.
- The numbers shall be stored into an appropriate data structure.
- For all files read in, the program shall display the average, the maximum, the minimum, and the standard deviation as well as a count of all numbers read in.
- All floating point numbers shall be formatted to have two decimal places displayed.

An example of this program running is shown in the Figure 1, and the sample files are given in Figure 2 and 3. A second execution example is given in Figure 4 with the sample files shown in Figures 5, 6, and 7.

```
D:\ClassPreps\se2890\Labs\Lab1a>java -classpath bin MainClass file1.txt file2.txt
Program Report:
################################################################
file1.txt
Average: 5.5
Max: 10.0
Min: 1.0
Standard Deviation: 3.02
Count: 10
################################################################
################################################################
file2.txt
Average: 550.0
Max: 1000.0
Min: 100.0
Standard Deviation: 302.76
Count: 10
################################################################

################################################################
All Files
Average: 277.75
Max: 1000.0
Min: 1.0
Standard Deviation: 348.49
Count: 20
################################################################


D:\ClassPreps\se2890\Labs\Lab1a>
```

Figure 1 Sample program execution.

```
1
2
3
4
5
6
7
8
9
10
```

Figure 2 file1.txt.

```
100
200
300
400
500
600
700
800
900
1000
```

Figure 3 file2.txt.

```
D:\ClassPreps\se2890\Labs\Lab1a>java -classpath bin MainClass file3.txt file4.txt file5.txt
Program Report:
################################################################
file3.txt
Average: 5.0
Max: 9.0
Min: 1.0
Standard Deviation: 2.74
Count: 9
################################################################
################################################################
file4.txt
Average: 5.0
Max: 9.0
Min: 1.0
Standard Deviation: 2.74
Count: 9
################################################################
################################################################
file5.txt
Average: 5.0
Max: 9.0
Min: 1.0
Standard Deviation: 2.74
Count: 9
################################################################

################################################################
All Files
Average: 5.0
Max: 9.0
Min: 1.0
Standard Deviation: 2.63
Count: 27
################################################################


D:\ClassPreps\se2890\Labs\Lab1a>
```

Figure 4 Sample program execution.

```
1
2
3
4
5
6
7
8
9
```

Figure 5 file3.txt.

```
9
8
7
6
5
4
3
2
1
```

Figure 6 file4.txt.

```
1
9
2
8
3
7
4
6
5
```

Figure 7 file5.txt.

4. Detailed Process

As this lab is tailored towards the imparting of discipline in your software development process, it will begin with an emphasis on following instructions and detailed, disciplined development approaches. FAILURE TO FOLLOW THESE INSTRUCTIUONS WHILE STILL DELIVERING A FINISHED PRODUCT WILL IMPACT YOUR GRADE.

Throughout the process, you will be recording times spent on each task. It is important that you keep accurate track of your time spent on the project, as well as any time when you are distracted from the work. (For example, if you start testing at 9:15 and finish at 10:05, but 35 minutes of that time was spent on the phone with your girlfriend, the actual time spent on task was only 15 minutes, not 50.)

4.1. Planning

Before digging into this project, we have a planning phase. In the planning phase, you must accomplish the following. For time tracking purposes, the planning phase begins as soon as you start working on this project.

1. Download the process spreadsheet, which will help you through your estimations and time tracking in each phase.
2. Estimate how many classes you will need for your development and how many method each classes will have. In doing this, you are starting to come up with a concept of how complex the software will be.
3. Review the Java language if you are out of practice. Do you need to look up how to do file I/O or any of the other operations you have not yet seen?
4. Estimate how long you believe this project will take to complete. In your estimation, include time that you have spent in the planning phase.

Once you have completed planning, you CAN NOT return to this activity. Record the exit time from this phase and calculate the amount of time spent in this phase.

4.2. Design Phase

When you start designing, notate the time that you entered this phase.

In the design phase, you are to design your implementation. Using UML, sketch the design for your software system as a set of class diagrams. Make certain all necessary methods are included on your diagrams, and that the parameters passed and return values are shown properly. This can be done by hand on a CLEAN 8 ½ x 11 sheet of paper.

When you are done with this phase, save your design file (if doing things electronically).

Mark the completion time for the design phase, and record the time spent in this phase.

4.3. Implementation Phase

When you enter this phase, record the time.

Now that you have a completed design, start implementing based off of that design. Begin by locating the "Build Project Automatically tab" in Eclipse and turning it off by un-checking it (assuming you are working in Eclipse). Implement the code fully in Java, but DO NOT compile or build your code. If Eclipse gives you a warning, fix it in this phase, but DO NOT compile your code.

Your code shall be implemented in the package <your csd id> where <your csd id> is the id you use to log into E-mail, etc.

If, in implementing your code, you discover a missing method that you need from your design, add it to your design, but make sure it is added in a. a different color or b. the file is saved with a different filename so we can tell how many things were changed after exiting the design phase.

Keep a simple tally of how many changes you had to make to your design phase during the implementation phase.

Once the implementation of your code is completed, save the code, but DO NOT COMPILE IT.

Record the time at which your implementation is complete and exit the phase.

4.4. Review Phase

Start by recording the time when you start reviewing your code.

Before compiling your code, read through your code again. Are there any mistakes you see? Are there any problems that have occurred when you translated your design into source code that you recognize by "proofreading" your code? If so, fix them here and keep a simple tally of the count of the errors fixed in this phase.

Record the time when you exit this phase, as well as the number of minutes spent in this phase.

4.5. Compilation Phase

Start by recording the time upon entering this phase.

In Eclipse, build your source code by clicking on the Project menu and Build. If there is a compilation error, fix it in this phase, all while keeping a tally of the number of errors fixed in this phase.

Record the time when your code compiles correctly, and calculate the time spent in this phase.

4.6. Testing Phase

Upon entry into this phase, record the time.

Once your code compiles, test it by running a set of tests to verify that your program is working properly. If there is a problem that requires a fix, fix it while keeping a tally of the number of errors fixed in this phase.

When your program operates properly, record the time and exit this phase.

4.7. Post Mortem Phase

In this phase, you are to take a look at the performance on your project. In specific, you need to be able to answer the following questions:

- How long did you spend in each of the phases (planning, design, implementation, review, compiling, testing, postmortem), and what percentage of total project time does this represent?
- How many problems did you resolve in each of the phases?
- How many lines of code is your project? (A tool is available for download that will help with this task.)
- What was your productivity (Lines of code / hour) for the entire project? (This is calculated by dividing the LOC count of your project by the total project time.)
- How close were the actual to your initial estimate?
- Assuming a labor cost of $100.00 an hour (cheap by commercial standards), how expensive was your software?

5. Deliverables

Deliverables shall be submitted electronically through the online web submission system on the course webpage.

5.1. Source code

Submit your completed source code. This should only include the java files, and all files should be placed into a single .zip archive. No other files should be submitted.

5.2. Excel File

Submit your Excel file used for planning and keeping track of time.

5.3. Lab Report

The lab report should consist of the following and shall be submitted in pdf format:

- Title page.
  - This should contain your name, section, instructor's name, Date, and the assignment number / description.
- Design.
  - Submit copies of your design. This should include both your initial design (made in the design phase) as well as the final design that you used for implementation. (Note: If you hand drew your designs, please either scan them using a scanner (preferred) or submit them in hard copy form during the start of lab.
  - Explain how you went about doing your design. What thought processes did you go through as you designed your software.
- Time Logs
  - Paste a copy of the Excel spreadsheet you completed with your time logs.
  - Defect counts List the count of defects uncovered in each activity.
- Things Gone Right
  - What things went right in performing this lab?
- Things Gone Wrong
  - What problems did you have?
  - What mistakes did you make?
  - What material did you not remember from SE1011 and SE1021 that you needed for this lab?
- Analysis of your time logs.
  - Where did you spend the most time on this project, and where did you discover the most problems with your system.
  - If you wanted to improve your performance, which phase would be a good starting point and why?
- Conclusions. State what you learned from this experience.
- Source Code
  - Attach your source code as an appendix to your report.

If you have any questions, consult your instructor.