Experience report

How to build

The following applies if you are running the program with Windows command line.

In the directory containing pom.xml type

```
mvn package
```

To run the program

In the target/ directory containing calculator-0.0.1-SNAPSHOT.jar type:

```
java –jar calculator-0.0.1-SNAPSHOT.jar <expression to calculate> <verbosity>
```

Example:

java -jar calculator-0.0.1-SNAPSHOT.jar add(2,2)

or

java -jar calculator-0.0.1-SNAPSHOT.jar mult(2,2) –INFO

or

java -jar calculator-0.0.1-SNAPSHOT.jar mult(2,2) –DEBUG

or

java -jar calculator-0.0.1-SNAPSHOT.jar mult(2,2) -ERROR

Details about issues I have encountered

Note: This section is not intended to be excuses for the state of the project I am handing in. It is intended to simply explain some background.

The first issue I encountered was the timing of the project. The weekend of October 10-12 is a national holiday in Canada and I had already made plans which I could not change. Making this worse is the location I was traveling was a 9 hour drive away. The commute time basically ate up two days where I was unable to work on the project. In addition while this project was important to me I also had things I had promised to do that weekend further limiting my time. As such I worked on the project when I was able.

Another challenge was the lack of internet access. Internet access was very limited and sporadic and not dependable throughout the weekend. As such I did not have the liberty of searching, downloading or using third party frameworks. Instead I had to write the code for anything I used.

My second issue was the technologies I was asked to use. While I like to believe I am very proficient in java and having a good understanding of git I did not know Maven and had never implemented a logging feature such as it was stated in the project description.

Do to these constraints and knowing I had promised to hand in the project on Wednesday I knew I had to approach this project as an agile project. In agile development you are not allowed to change your deadline and instead you adjust scope to meet that deadline. As such I decided that the basic functionality (actually being able to take the input and print a result) was the most important part to complete. I decided that I would exclude most error checking and omit the logging functionality unless time permitted. (I was able to add logging).

My IDE of choice is Eclipse so that aided in creating the Maven configuration.

When I was actually building the program I found coming up with a parsing solution was tedious. I had not worked with parsing math equations (keeping track of order of operations and dealing with brackets). As stated above I could not use an already made parser because I could not access it nor its documentation. The solution I envisioned was to split the equation into individual parts and then use a stack to keep track of the operations. I had a partial solution in mind but did when I did have access I consulted the internet and found a very similar solution to the one I was thinking of and used it as a basis for my solution. This saved valuable time.

My last main obstacle for this project was actually running a program that was compiled with Maven. I needed to add a couple plugins to the pom.xml file in order to properly set the java version and the location of the class containing main() to compile and run the project.

<u>Assumptions</u>

- For the time constraints placed on this project my solution does minimal error checking.
- My solution also assumes that the expression/equation entered is valid. My solution does not check to see if the expression/equation is valid.

- For the Let operation I assume that the variable associated with it is only available within the let statement (the variable acts as a local variable to the let it was declared in).
    - For example: let(b, let(a, 10, add(a,2)), add(b,a)) Will not compute properly as a in add(b,a) would be undefined.
    - My solution will still compute the above example (do to the lack of expression validity checking) but instead it will use the value of b for a but the answer will be incorrect.

<u>Time spent on the project</u>

Learning Maven – Before this project I had not worked with Maven. I spent around 2 hours learning what Maven is and how to use it. This time also would include time taken creating a maven project.

Understanding the Problem – This was the quickest part of the project. I identified what needed to be done and of the solution I would present. I spent ~ 1 hour on this.

Writing the program – This would be the time spent in front of a computer writing all the source code for the project. This does not including testing but does include changes made based on results of testing. This stage took ~ 3-4 hours.

Testing the program – This was testing the program. Testing involved manual testing and tracing the execution of the program. This stage took ~2-3 hours.

Writing the report – This took ~ 1 hour.

<u>Optional Jenkins Setup</u>

I have never used Jenkins and it is something I was unable to fir in due to the time restrictions for this project.

Final Notes

Generally I would never hand in a project with such little error checking / handling nor a project without a basic set of test cases and I want to apologise for that. My solution would be considered delivering something fast and not polished.

Synopses Coverity have been very respectful and understanding of my situation and made several changes to their interview process to accommodate me. I want to thank all of the

Synopses Coverity team for the opportunity and I hope I get the chance to work with you in the near future.