1 Description

This goal of this project is to demonstrate the ability to produce and implement an API adhering to JavaDoc standards. This will be accomplished by producing a specified API and then implementing that API.

2 Requirements

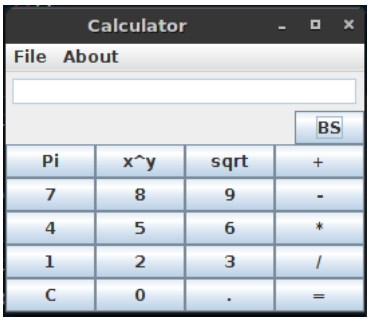

Figure 1: It should look similar to this

You will create an API with JavaDoc which describes a GUI calculator as shown above. This calculator must be capable of performing addition, subtraction, division, multiplication, and exponential functions on the Real numbers (no Complex values). The methods for mathematical operations may be used by another program as some later date and should not truncate or round. Any truncating or rounding must be done after values are passed back to the GUI.

Moreover, the final program should exhibit the following behaviors.

1. It should compile.
2. The API should be written in such a way that code reuse is simple.
3. The API should be well-documented (if you fill out the JavaDoc correctly you will have done this).
4. Code should be self-documenting, do not enter comments saying what a switch statement is doing. Commenting explains blocks of code. If you have to read snippets from multiple classes to understand what a section of code is doing put a comment.
5. The implementation should take advantage of Java library methods. You may only use the Java standard library.

3 Part 1: Designing the API

Design the MathMethods API with which your calculator will interface. Essentially, `MathMethods.java` is the container for all mathematical operations that the calculator will perform. It consists entirely of static methods and has no global variables. Furthermore, each of the methods in MathMethods must return a double. MathMethods JavaDoc must make mention of when it will throw exceptions due to the operation requested, e.g square root of a negative number.

4 Part 2: Implementation

Implement `MathMethods.java`. Furthermore, you will implement a GUI that will interface with `MathMethods.java`. This GUI must consist of at least one class, namely `CalcGUI.java`. You may spread the GUI functions across as many classes as you wish. The GUI will have a menubar with a File menu and an About menu. The File menu consists of one item, an exit button. The About menu consist of one item, an about entry which creates a new window listing the names of the members of the team. All buttons indicated in Figure 1 above must be present on the calculator. The calculator may only perform binary operations. E.g. "`2+2=`" is a valid operation, and "`2+2+2=`" is not. If a user enters the following "`2+2=`" and follows this with a "`+2=`" the calculator should display "`6`". The "`C`" button should clear all input. The "`x^y`" button means to compute x^y. The display should display a "`^`" for both exponentials or roots. E.g. "`5^.5`" is $\sqrt{5}$, and "`5^5`" is 5^5.

5 Grading

The primary purpose of this assignment is to demonstrate mastery of writing an API, therefore 60% of the total grade consists of an evaluation of the API. The remaining 40% is in producing a functioning calculator. The API (skeleton `MathMethods.java` with JavaDoc, A.K.A. Part 1) **must** be committed to GitHub no later than 11:59pm November 21, 2015. The source code for Calculator (Part 2) must be committed before class on November 24th. Deviations from the submitted API must be documented in the JavaDoc of the final code. Be prepared to see your code displayed, if you write spaghetti everyone will see it. If your code is completely left justified everyone will see it (and you will lose points on code readability).

6 Extra Credit

Implement an algorithm to serve as a proof of concept that $P = NP$. Add capacity for CalcGUI.java to receive keyboard input.