York University

EECS 2311

Calculator Testing

Team 3

Brooks Dulla
Andrew Ferreira
Tedi Papajorgji
Dan Sheng
Yari Yousefian

February 24, 2016

Blackbox Testing

In this section we outline our methods for testing our calculator with regards only to input and output arguments.

Simple Cases

Table 1: Testing of Simple Inputs and Results

Test	Input	Actual Output	Expected Output
Simple Operations			
Addition	5, Enter, 2, Enter, +	7	7
Subtraction	5, Enter, 2, Enter, -	3	3
Multiplication	5, Enter, 2, Enter, ×	10	10
Division	5, Enter, 2, Enter, /	2.50	2.50
Operation foregoing 2^{nd} Enter	5, Enter, 2, +	7	7
Decimal Point Operations			
Addition	5.25, Enter, 4.43, +	9.68	9.68
Subtraction	5.25, Enter, 4.43, -	0.82	0.82
Multiplication	5.25, Enter, 4.43, ×	23.26	23.26
Division	5.25, Enter, 4.43, /	1.19	1.19
Input over 2 decimal places	2.5555, Enter	2.56	2.56
Clear	Clear	No output, items in stack cleared	No output, items in stack cleared
Pi Button	Pi	3.14	3.14
Factorial	5!	120	120
Zero Factorial	0!	1	1
Negative Factorial	-5!	1	Error Message
Fraction Factorial	1.5!	Error Message	Error Message
Sine and Cosine			
Sin function	sin, 30, Enter	0.50	0.50
	sin, 0, Enter	0	0
	sin, 90, Enter	1	1
Negative argument	sin, 90, (-), Enter	-1	-1
Cos function	cos, 60, Enter	0.50	0.50
	cos, 0, Enter	1	1
	cos, 90, Enter	0	0
Negative argument	cos, 90, (-), Enter	0	0
Change Sign	5, Enter, 3, (-) +	2	2
Undo			
Undo while user is in the middle of typing	5, Enter, 6, Enter, Undo	6 is removed, and 5 remains	6 is removed and 5 remains

*Tests colored red indicate a failed test

Simple Bugs

- Negative Factorial: A negative argument passed to the factorial function should print an error message to the calculator's screen. Instead we see that the calculator simply outputs 1 in this case.

Table 1 includes various tests for cases that one would most likely encounter when using the calculator for everyday calculation. These tests are designed to indicate the presence of bugs in the simplest of cases.

The above tests are sufficient because they handle each case for the operators. More rigorous black box tests which aim to break the calculator to reveal more specific bugs are found in the next section.

Boundary Cases

This section aims to break the calculator by using invalid inputs in order to determine whether our calculator handles exceptions appropriately. The boundary cases we've tested, along with the corresponding bugs are listed:

- Any operation (+, -, ×, /, !, sin, cos) with very large operands causes undocumented behavior, throws exceptions.
- Factorial does not properly calculate numbers above 16, causes overflow errors because output is occasionally negative.
- Division by zero only throws exception in Eclipse, however no error message is displayed on calculator screen.
- Using the "Change sign" button, it is possible to change the sign of 0 to -0, which is non-sensical.
- Entering very large number (i.e. 9999999999999...) stores a misrepresented value of 10, which does not actually behave like the number 10 in operations.
- Some operation conditions produce an output with 3 decimal places, despite current design choice of 2 decimal places.

Fixed Bugs

- Spamming "undo" makes "Start new computation — value = 0" appear on screen. Each subsequent "undo" press removes a character from the aforementioned string, until all characters have been removed and the string is again displayed on screen.

Other Bugs

- The calculator view dimensions seem to vary across different platforms. The prism computers display the calculator's components at all of the proper sizes, however when the app is used on different computers with differing aspect ratios/resolutions, the calculator's screen display sometimes overlaps with the calculator's boundary. More testing on different systems is in order to find the root of the problem.