Summary:

Team TBD has revamped our testing script in order to work per specifications. We have also cleaned it up to produce easier to read results. The script now calls the test driver specified in the test cases. We have also made more test cases bringing our total up to the required 25. Our test cases are more thorough and fully test each method. We developed our drivers as well.

Testing Framework:

/scripts/ holds our runAllTests bash file.
/testCases/ holds the collection of text cases in format explained below.
/project/* holds the **bin/** and **src/** folders containing the calculator files themselves.
/testCaseExecutables/ holds the drivers used for each type of test.
/reports is where the report of the tests run is placed, and contains an external stylesheet for its formatting.

Running Instructions:

1. Clone the project from the Team TBD repository.
2. Navigate to the TBD-master folder.
3. Run *./TestAutomation/scripts/runAllTests.sh*
4. The tests will run and print out each result in the terminal.
5. After completion of all the test an HTML report of the test results will be created and opened.

Note on test cases:

We have written our test cases in the following format:

<Parameter> + [TAB] + <Value>

An example test case file would appear as the following:

ID	1
Req	Calculating the square of the input
Comp	Calculator.java
Method	public Double calculateMono(MonoOperatorModes square, Double num)
Inputs	2.0
Expect	4.0
Driver	testDriverSquare

Results:

Test #	Method	Input	Expected	Result
Test #1	public Double calculateMono(MonoOperatorModes square, Double num)	2.0	4.0	4.0
Test #2	public Double calculateMono(MonoOperatorModes square, Double num)	9.0	81.0	81.0
Test #3	public Double calculateMono(MonoOperatorModes squareRoot, Double num)	9.0	3.0	3.0
Test #4	public Double calculateMono(MonoOperatorModes squareRoot, Double num)	0.0	0.0	0.0
Test #5	public Double calculateMono(MonoOperatorModes oneDividedBy, Double num)	1.0	1.0	1.0
Test #6	public Double calculateMono(MonoOperatorModes oneDividedBy, Double num)	10.0	0.1	0.1
Test #7	public Double calculateMono(MonoOperatorModes cos, Double num)	1.0	0.5403023058681398	0.5403023058681398
Test #8	public Double calculateMono(MonoOperatorModes cos, Double num)	0.0	1.0	1.0
Test #9	public Double calculateMono(MonoOperatorModes sin, Double num)	1.0	0.8414709848078965	0.8414709848078965
Test #10	public Double calculateMono(MonoOperatorModes sin, Double num)	0.0	0.0	0.0
Test #11	public Double calculateMono(MonoOperatorModes tan, Double num)	1.0	1.5574077246549023	1.5574077246549023
Test #12	public Double calculateMono(MonoOperatorModes tan, Double num)	0.0	0.0	0.0
Test #13	public Double calculateMono(MonoOperatorModes tan, Double num)	-1.0	-1.5574077246549023	-1.5574077246549023
Test #14	public Double calculateMono(MonoOperatorModes sin, Double num)	-1.0	-0.8414709848078965	-0.8414709848078965
Test #15	public Double calculateMono(MonoOperatorModes cos, Double num)	-1.0	0.5403023058681398	0.5403023058681398
Test #16	public Double calculateMono(MonoOperatorModes oneDividedBy, Double num)	42.0	0.023809523809523808	0.023809523809523808
Test #17	public Double calculateMono(MonoOperatorModes squareRoot, Double num)	5.0	2.23606797749979	2.23606797749979
Test #18	public Double calculateMono(MonoOperatorModes squareRoot, Double num)	-5.0	NaN	NaN
Test #19	public Double calculateMono(MonoOperatorModes squareRoot, Double num)	100.0	10.0	10.0
Test #20	public Double calculateMono(MonoOperatorModes squareRoot, Double num)	10.5	3.24037034920393	3.24037034920393
Test #21	public Double calculateMono(MonoOperatorModes square, Double num)	-2.0	4.0	4.0
Test #22	public Double calculateMono(MonoOperatorModes square, Double num)	-9.0	81.0	81.0
Test #23	public Double calculateMono(MonoOperatorModes square, Double num)	73.5	5402.25	5402.25
Test #24	public Double calculateMono(MonoOperatorModes square, Double num)	-73.5	5402.25	5402.25
Test #25	public Double calculateMono(MonoOperatorModes oneDividedBy, Double num)	-15.5	-0.06451612903225806	-0.06451612903225806