Testing Plan:
JKK Consultant QuadSolver

Paul R Phillips, Brian Greener

2018–03–14

Introduction

Purpose of this project is to create a program which can calculate the solutions for a standard quadratic equation. The program takes input values from the command line, and finds a solution for such inputs using the quadratic equation. Output should be rounded to seven decimal places. If given a prompt, the program should return output from a machine's command line. Output does not need to be stored, rather program is treated as a calculator. Tests are needed to make sure inputs produce only real solutions (with the exception of double solutions), and are 32 bit single precision values (output should be the same precision as well). Inputs should also be able to take values in scientific notation. Operating system intended for application is Hubunto 1604 (Linux), using C, and the GCC compiler.

Stories

Engineer is able to provide a quadratic equation (assuming in standard form) as input, and have output validated (error messages returned) Values Evaluated for: Non-numeric, +/- Inf, NaN, NA, a = zero.

Note: Input is successfully converted to IEEEfp (single precision), and has scientific notation compatibility.

Engineer's input calculates the standard quadratic equation's determinant. Solver should flag complex and double solutions (returning value of double solution).

Engineer is able to execute the Quadratic solver from the command line (i.e. does not have to open application) and is able to receive desired output (output does not need to be stored).

"Developer should be able to have a list of unit tests which its purpose to test and validate the functionality of the individual components which build the quadsolver".

Create a .tar file so that program can be distributed to third part users.

Developer is able to check output from automated tests ran (instead of having to manually run tests, developer will only have to maintain results of tests).

Use Cases

Testing Plan

Unit Testing

It is our plan to conduct unit tests in order to validate that our quadsolver program will work in the following possible situations:

- Input is successfully converted from string to the IEEE floating point format

- Inputted values do not contain NA's, character values, or NaN's

- Inputted values also do not contain positive or negative Inf's, and are not zeros

- The determinant should flag an error if the equation's determinant is less than zero
- If the determinant equals zero, a warning message should display along with the double solution for the equation

The framework we'll be using for Unit Testing is JUnit for our C version of the program and RUnit for the R version.

Coverage Testing

Coverage testing is used to determine the amount of source code actually being used in a program. This can be determined by reviewing the code by hand since this is such a small program.

Usability Testing

A third party member of our class, Austin Ragotzy, will be volunteering to operate in our blackbox testing and usability testing.The usability test should be completed within a ten minute time frame.

Performance Testing

We will use basic timers in C in order to test the runtimes of various functions as well as the total runtime from start to finish. From this we can chart out various runtimes from different numbers of inputs to see the time complexity of the program. However we will probably be seeing a constant runtime complexity due to the nature of this program.