Project Proposal

Student Names: Tommy Avetisyan, Kyle Felkel
Proposed System Under Test (SUT): C++ Math Solver
Link to SUT Source Code: https://github.com/Tomavetisyan/C--Math-Solver
SUT Size: 2000

SUT Description:

This is a console application written in C++ that takes the user's inputs to solve geometric shapes and give results such as parameter, area, base heights, angle measurements ext. This project emphasizes the depth of use class protection and inheritance in the C++ language.

State of SUT:

This project was developed about 5 years ago and hasn't been touched since. The code is all separated neatly into a class file system and has no miscellaneous comments to the code.

Attributes:

Fast: Gives results quickly.
User Friendly: The code guides you through everything
Interactive: Just plug and the computer will chug

Components:

Input based GUI: User will use basic inputs to guide through screens
Error detection: The code will detect if something is wrong with the inputs for a given shape
Wide coverage: This will solve from basic rectangles to 3 dimensional pyramids
Solutions: the program will output the shapes solution with clear labeling

Capabilities:

The input based GUI is user friendly where the program guides you through your inputs.
The solutions are fast where you only need to go through a few screens for results.
The inputs are interactive where you just plug your numbers as they ask, and you get your solution

Unit Tested Capabilities(s):

Unit tests will be written on all the different cpp files and try to see if the inheritance and private variables are handled properly. All the different shapes can also be tested to see if the outputs give correct results. The operator overloads inside the programs can be tested to see if it handles the objects correctly.

Automatically Tested Capabilities(s):

We can randomly generate integer values to represent the various lengths of shapes. These values can then be fed into various objects to calculate values such as perimeter, area, and volume. Expected correct values can be calculated using official math formulas. We can generate values, calculate correct values, receive actual values, and compare using Assert statements in a loop.