Final Project - OOP Calculator

Comp 3141- Object-Oriented Design and Programming
April 24th, 2024

Faculty of Computer Science
Thompson Rivers University

Submitted By:

Martin Atanacio - T00684924

Class Structure

This project consists of creating a calculator program to demonstrate some of the concepts I learnt about object-oriented design. Two types of calculators will be implemented in this program: Basic and Scientific. The Basic Calculator consists of simple functions like addition, subtraction, multiplication, division, square, and square root. On the other hand, the Scientific Calculator consists of functions like sin, cos, tan, log, and log10. In order to achieve this, my program will consists of 3 classes:

1. The base class “Calculator” that will contain the function to retrieve a number from the user and store it in a private member.
2. A child class “BasicCalculator” that will use their parent’s function as well as defining the unique functions according to a Basic Calculator.
3. Another child class “ScientificCalculator” that will do the same as BasicCalculator, except it will define its own unique set of functions according to a Scientific Calculator.

In order to test the functionality of these classes, I have created a “driver” program where the main function can be found. When run, the program will display a menu to the output console, where the user can select which calculator to use, or quit the program. Once a type of calculator is selected, a new menu will appear with the corresponding functionalities. When a user selects a functionality, the process of that functionality starts. For example, if the user selects “Basic” in the main menu and then “ + ”, they will be prompted to enter 2 numbers to be added and the result will be output to the console. Otherwise, the user can select to quit the calculator and this will return them to the main menu, where they choose a calculator type again. The user is able to swap between calculator types and use functionalities as long as they want until they quit.

File Structure

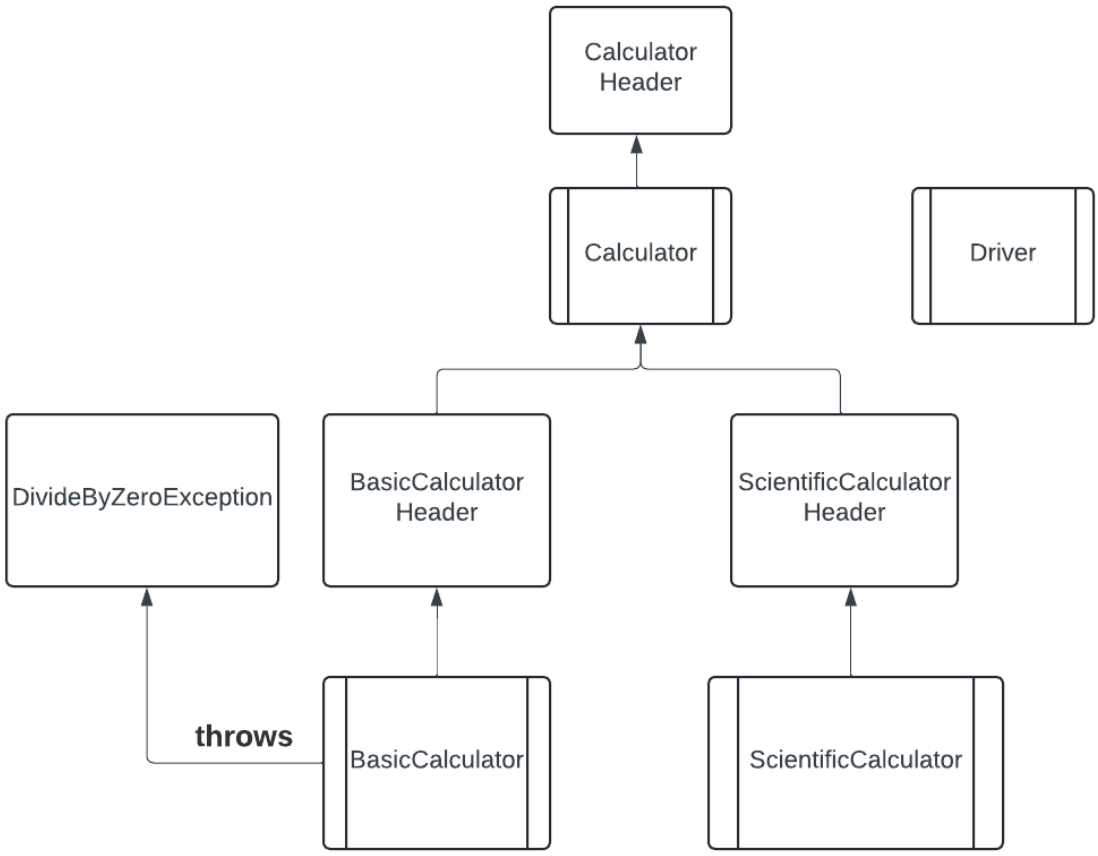
In order to achieve a well structured object-oriented approach, I have created 8 files:

1. Calculator.h - header file that contains the blueprint for a Calculator object
2. Calculator.cpp - defines the members and functions of its header file
3. BasicCalculator.h - header file that contains the blueprint for a BasicCalculator object, also inherits from Calculator
4. BasicCalculator.cpp - defines the members and functions of its header file
5. DivideByZeroException.h - runtime error header file, used by BasicCalculator to throw an error when attempting to divide by 0
6. ScientificCalculator.h - header file that contains the blueprint for a ScientificCalculator object, also inherits from Calculator
7. ScientificCalculator.cpp - defines the members and functions of its header file
8. Driver.cpp - driver program that displays menus for the user to interact with the calculator objects

Martin Atanacio - T00684924

Hierarchy Structure

Here is an overview of the structure of the program:



Outputs

The following screenshots demonstrate how the program works.

1. When you run the program, the first menu will display a selection of calculator types:

```
SELECT MODE:
        1. Basic
        2. Scientific
        3. Quit
Enter an option:
```

2. The user types to a console their selection. Invalid input will produce an error, and 3 will quit the program:

```
SELECT MODE:
        1. Basic
        2. Scientific
        3. Quit
Enter an option:
4

Incorrect value. Try again!
SELECT MODE:
        1. Basic
        2. Scientific
        3. Quit
Enter an option:
3
```

3. When the user selects 1 for Basic, the appropriate menu will show up:

```
Basic Calculator Menu:
        1. +
        2. -
        3. *
        4. /
        5. sqrt(x)
        6. x^2
        7. Quit to main menu
Enter an option:
```

4. The user can select any of the functionalities, for example 2 for subtraction. This will prompt the user to enter 2 numbers and the output will be displayed:

```
Enter an option: 2
Enter first number: 10
Enter second number: 7
10 - 7 = 3
```

5. The Basic calculator menu will automatically be displayed again, giving the user chance to either select another functionality or quitting to the main menu:

```
Basic Calculator Menu:
        1. +
        2. -
        3. *
        4. /
        5. sqrt(x)
        6. x^2
        7. Quit to main menu
Enter an option: 7

Exiting Basic Calculator
SELECT MODE:
        1. Basic
        2. Scientific
        3. Quit
Enter an option:
```

6. The user can then select any of the calculators again. Here's the example for the Scientific calculator:

```
SELECT MODE:
        1. Basic
        2. Scientific
        3. Quit
Enter an option:
2

Scientific Calculator Menu:
        1. sin
        2. cos
        3. tan
        4. log
        5. log10
        6. Quit to main menu
Enter an option:
```

7. Again, the user can select one of the functionalities and test it:

```
Enter an option: 5
Enter a number: 100
log10(100) = 2
```

8. Wrong values will produce errors:

```
Scientific Calculator Menu:
        1. sin
        2. cos
        3. tan
        4. log
        5. log10
        6. Quit to main menu
Enter an option: 8
Incorrect value. Try again!
```

9. A division by zero in the Basic calculator will throw a DivisionByZeroException exception: