'Calculator' is a simple C++ console application for this coding exercise.

As the application could be a console application, web service, Windows application (WPF or Windows Forms) and extended by other members of the team, A 3-Tier architecture design was implemented for encapsulating the valuable business logic/rules and then being able to reuse them in a variety of applications.

'Main.cpp' file contains the main function and serves as user interface of the console application. On a console, user can type "Calculator/?" for printing usage information. User can also provide input string as a optional command-line argument. Otherwise, the program asks user to type in a string.

'InputStringParser' and 'CalcLogic' are working classes in Business Logic Layer. They don't have any data members. They currently work with v1.1 format input strings, a string of operators (+,-,\*, and / only) and operands (integers only). They can be extended to work with new format of input strings. The 'InputStringParser' class parses input string and generates a queue of parsed algebraic expression(s). The operator characters ('+', '-', '\*', '/') are converted to 'enum' type for being able to be stored in the same list along with integer type operands. It's more efficient to work with primitive data type than objects in a list. The parser works on the original input string, no copy is made. The 'CalcLogic' class evaluates parsed algebraic expression(s) and generates a queue of output number(s).

The output number can be int or float type. 'OutputNumber' super-class and 'TypedOutputNumber ' template sub-class are created for storing different type of output number in one list, without adopting third party C++ library. The solution is functional but could be tweaked to be a smarter one.

The 'CalcController' class is controller class of Presentation Layer. It receives the input string from user interface and passes it to 'InputStringParser' class. It then passes 'algebraicExpressions' list to 'CalcLogic' class. The 'outputNumberList' data member contains the calculated output number(s). I use 'deque' container class instead of 'vector' for all lists. It's more efficient as the program is adding an unknown quantity of objects/integers to the lists.

There is no Data Access Layer for this simple application. Please see the Class Diagram below.

CalcController

InputStringParser

TypedOutputNumber

OutputNumber

CalcLogic

1..\*

1

1

1

1

1

Class Diagram