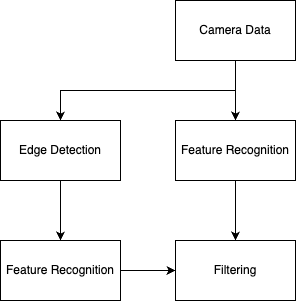
Problem:

You work for a company that produces software for cameras. The current version of the software allows users to capture an image with a connected camera and display it on a GUI. The company is planning to add a flexible post-processing framework that enables users to operate on the data received from connected cameras. Users will be able to create several post-processing steps, chain them together, and see their results displayed in the GUI. The diagram below shows one example:



To begin the design cycle, you have been asked to make a prototype of the post-processing system that uses scalar floating-point data instead of image data. Your system will accept a floating-point input and execute a series of arithmetic post-processing steps in the order of dependency. Once the system is complete it should be able to correctly determine the value of the expression: ((x + 1) \* 2 – x / 3) / x. Where “x” is the input, and each post-processing step computes the result of at most one mathematical operation (i.e. +, -, /, \*).

Requirements:

* A program, written in C++, which accepts the input value x and computes the value of the expression ((x + 1) \* 2 – x / 3) / x.
* Please compile an executable for either x64 Ubuntu Linux, or x64 Windows, or provide the necessary machinery to build your executable along with detailed steps on how to build. Note that if we cannot build your code on our own, we will not evaluate your project.
* The code in the form of a public github repository, or a zipped folder
* Treat post-processing steps as units of computation with designated inputs and outputs. Each post-processing step may only compute the result of one mathematical operator (e.g. 1 + 2, 5 / 6, or 3 – 1)
* Proper error handling; the program should be able to handle x = 0, and other corner cases, and inform the user before exiting
* The code you write should be extendible—imagine a client who would like to extend your code to include a ‘%’ operator.
* The code you write should adhere to clean object-oriented design principles with a focus on re-use for when the company implements the production version of your design to operate on image data

Bonus:

Bonus points will be given if you can extend your program to accept any mathematical expression which contains only the (\*, +, -, /) operators, floating point numbers, the input “x”, and “()” braces for logical grouping.