Programming Exercises

All programs should:

- be production quality
- run on Linux
- come with unit tests and instructions on how to run them
- come with appropriate design documentation and api documentation
- compile into one library or executable
- compile with gcc
- take about 4 hours of your time to write

We will be looking at the quality of the design, quality of the code and completeness of the unit tests.

Calculation Engines

Develop a program that performs calculations on a series of integers. The program will run from the command line. It has two versions:

calc <engine_name> <file_list>
calc <engine_name> <list of integers>

Files will contain one or more lines of integers. All file names have an implicit ".txt" on the end of them – e.g. "input" on the command line maps to the file "input.txt". All files are in the current working directory (i.e. don't worry about directory names).

The program should supply two engines:

- Multiplier – multiplies all the integers together; it should only work with a file list as input data
- Divider – divides all the integers, one after another; it should work with both file list and integer lists as input data

The program should read the input data, calculate a result and display the result on *stdout*.

Supply an API so that a user can write his own Engine, say to calculate standard deviation. The user should be able to easily add the engine to the library by recompiling all your source code.

As part of the API, you should also supply a factory to generate Engines. At minimum, a user of the factory would specify a calculation and an allowable input type.

Only do unit tests to test the Divider calculations.