Coding Exercise

The following problem should take from 2 to 4 hours to complete the full solution. The submitted code should be of the quality we can expect during a normal working week.

Using maven or ant is not a requirement; however you are welcome to do so. Please, provide instructions on how to run the application from the command line.

There is no requirement to cater for invalid input. As you will not have the opportunity to clarify requirements, feel free to note any assumptions in your submission.

You are welcome to use any external libraries you want. Include them in your submission. *hint: there are libraries that do file I/O.

It is important to insist on Test Driven Development, which means proceed first with tests and then implementation. You are required to show your ability to provide reasonable production quality code. Complete lack of test coverage will disqualify you immediately.

There are no tricks or gotchas: only implement the functionality described below. It is a simple problem to solve, but code quality is as important as solving the problem itself.

Problem: Calculator

Write some code to calculate a result from a set of instructions. Instructions comprise of a keyword and a number that are separated by a space per line. Instructions are loaded from file and results are output to the screen. Any number of Instructions can be specified. Instructions can be any binary operators of your choice (e.g., add, divide, subtract, multiply etc). The instructions will ignore mathematical precedence. The last instruction should be "apply" and a number (e.g., "apply 3"). The calculator is then initialised with that number and the previous instructions are applied to that number.

Examples of the calculator lifecycle might be:

 Example 1.

```
[Input from file]

add 2
multiply 3
apply 4

[Output to screen]
18
```

```
[Explanation]

(4 + 2) * 3 = 18
```

Example 2.

```
[Input from file]
multiply 9
apply 5

[Output to screen]
45

[Explanation]

5 * 9 = 45
```

Example 3.

```
[Input from file]
apply 1

[Output to screen]
1
```