COSC2430: Programming and Data Structures
HW2: Evaluate Arithmetic Expressions using stack

1 Introduction

You will create a C++ program that can evaluate arithmetic expressions with integer numbers having any number of digits. These numbers are an alternative to fixed size integers or floating point numbers that always have a maximum number of accurate digits (dependent on size of CPU register).

2 Input and Output

The input is a regular text file, where each line is terminated with an end-of-line character(s). Each line will contain an arithmetic expression. Operators $+ - *$ is mandatory. 10% extra credits for operator /. The program should display the input expression and the results, separated with =. You need to check whether the expression is valid or not. If one expression is not valid, skip it and continue to process next expression.

Input example:

```
0*00000000000000000+0000000000000001
(1+2)*(1000+2000)
(+1+2)*(1000+2000)
((1+2)*(1000+2000))*(1+10000)
(1+2)*(1000+2000))*(1+10000)
(1000000000000000-1)
99999999999999999999/999999999999999999999
(-100000+10000)*8
((-1))-(-1)
```

Output example:

```
0*00000000000000000+0000000000000001=1
(1+2)*(1000+2000)=9000
(+1+2)*(1000+2000)=9000
((1+2)*(1000+2000))*(1+10000)=90009000
(1000000000000000-1)=999999999999999
99999999999999999999/999999999999999999999=1
(-100000+10000)*8=-720000
((-1))-(-1)=0
```

3 Program input and output specification

The main program should be called infinitearithmetic. Call syntax at the OS prompt:

```
infinitearithmetic input=input1.txt
```

Assumptions:

- The file is a small plain text file (say < 10000 expressions); no need to handle binary files.

- Only integer numbers as input (no decimals!). Input numbers *may* have leading zeroes. Output number will be written without leading zeroes (i.e. eliminate them).
- Operators: $+ - */$. 10% extra credits for operator /.
- Keep in mind a single + or can be a sign instead of an operator.
- You can assume the input are only integers and the ouptut is only an integer. Therefore, you can truncate the decimal part when evaluating division.
- do not break an arithmetic expression into multiple lines as it will mess testing.

4 Requirements

- Homework is **individual**. Your homework will be automatically screened for code plagiarism against code from the other students and code from external sources. If you copy/download source code from the Internet or a book it is better you acknowledge it in your comments, instead of the TAs detecting it. Code that is detected to be copied from another student (for instance, renaming variables, changing for and while loops, changing indentation, etc) will result in "Fail" in the course and being reported to UH upper administration.
- std::stack is not allowed. Please implement your own stack.
- Remove leading zeroes from the result (e.g. 1, instead of 00001; 0 instead of 00000).
- timeout is set to 10s.