

CSCI-20 Lab #3

Please work in groups of two or three on these programs. The infix and postfix algorithms are given in exercises at the back of chapter 2 in the text, starting on page 47.

1. Implement a `STACK` class to hold a stack of digits. Use the `STACK` class to implement the algorithm from class to convert a string in the format of a floating-point number into a floating-point value held in a variable.

The string format is as follows:

- an optional sign, either '+' or '-'
- zero or more digits for the integer part of the number
- a decimal point '.' (which may be omitted only if there are no fractional part digits)
- zero or more digits for the fractional part of the number

For example, these are legal floating-point strings:

```
7
-3
+2.5
-12345.6789
.125
6.
```

2. Modify your `STACK` class to hold operators also, and write a program to convert infix to postfix. Assume all integer operands. Support these operators `(,)`, `+`, `-`, `*`, `/` and `%`.
3. Modify your `STACK` class to hold integers and write a program to solve postfix expressions, using all integer arithmetic.

These programs are easily found on the Internet. Please, don't look for them, solve them yourselves! After the stack class, they are each about a page or so long.

Test your programs with 20 or 30 expressions each. Use direct file I/O, and pass the file names into your program on the command line. As usual, email me the source, data and output files.

Hint: if you keep the digits as characters not integers, the conversion for part 2 is trivial...