Homework #2

You probably will want to plan out these 2 programs carefully before you start writing code.

A. Arithmetic sequences

You may do error checking with your program if you want, but it's not required.

This program involves 5 numbers that are relevant in a (finite) arithmetic series.

Suppose we are considering the series

The 5 values we're interested in are

F (the first term): 1.1

I (the increment from one term to the next): 0.1

L (the last term): 1.3

N (the number of terms): 3

T (the total of the terms): 3.6

Let's say that as far as this program is concerned, I has to be nonzero and N has to be > 1.

(Then we don't have to worry about trivial cases.)

The idea here is that the user will input any 3 of these 5 values, and your program outputs the other 2. The user's input is a letter followed by a number, a letter, a number, a letter, and a number. The 3 letters are case insensitive, all different, from FILNT; the numbers give the corresponding values.

For example, still considering the example , if the user types

**t 3.6 f 1.1 L 1.3**

then the user is saying to your program, "I'm thinking of an arithmetic sequence, whose total is 3.6, whose first term is 1.1, and whose last term is 1.3. You tell me the other 2 things." Your program should then output

I 0.1 N 3

The user can pick any 3 of the 5 values to input, and can input them in any order.

Your program can output the two other values in either order, labeled with the appropriate letters.

You may recall that doubles can be problematic in that their answers are off by a tiny amount at hard-to-predict time. For instance, it would be a problem to say

unsigned n = unsigned((first - last)/inc) + 1;

even though that equations is mathematically correct.

One way to get around that problem is to say

double n = (first - last)/inc + 1;

instead.

B. Geometric sequences

Same as program A, except now you're working with geometric sequences instead of arithmetic sequences.