TCSS 333 Spring 2015 HW1

Due: by midnight Friday April 10 (or with 10% late penalty, by midnight Saturday April 11).

Submit a single .c file to Canvas
(Don't turn in anything else. No Eclipse projects. Don't use multiple source files.)

Write an interactive program that does addition, subtraction, multiplication, and division of binary numbers entered by the user. The binary numbers will always have a binary point, so they may either be integers (such as 111.0 = 7) or they may be fractions (like 0.011 = 3/8 = 0.375) or they may be combinations (11.01 = 3.25). All numbers entered by the user are guaranteed to have a binary point and at least one digit on each side. The user will only use +,-,*,/ as operators. The operator will be separated from the binary numbers by one or more blanks. Your program can assume that user input is error-free. Here is a sample run of the calculator program (user's typing is in **bold**):

```
> 0.11100 * 10.0
1.11
> 0.111 * 100.0
11.1
> 0.111 * 1000.0
111.0
> 1000.0 * 1000.0
1000000.0
> 11.0 * 11.0
1001.0
> 0.0000000001 * 0.0000000001
0.0000000000000000001
> 1111.1111 * 0.0001
0.11111111
> 1111.1111 * 10000.0
11111111.0
> 11.11 * 11.11
1110.0001
> Q
```

Notice that the calculator stops when the user enters Q. You may implement negative numbers if you like, but your program will not be tested with them.

Your calculator must convert the user's binary numbers into doubles, do the required calculation using doubles, then print out the answer (also a double) as a binary number. These conversions can be done without arrays.

To read the next non-whitespace character, you need `scanf(" %c", &myChar);` (notice the blank before %c)

Write your program all in one file (it will require about 100 lines of code). In addition to main, you should have separate functions to read in and write out a binary number. (And additional functions if you wish.)

Global variables are variables declared above or in between functions. <u>You may not use global variables!</u>
If you want to move data from one function to another, use parameters and/or return values.

<u>Binary conversion algorithms (from binary to float)</u>
Suppose we're working with the binary number 110.011 Start with value = 0.
Until you run into the binary point, repeat this procedure:
- double the value
- add the next bit (0 or 1)

Doing this with 110 should give a result of 6.

Step over the binary point.
Starting with 0.5 as the current power of 2, repeat this procedure:
- if the next bit is a 1, add in the current power of 2 (for 0, don't)
- divide the current power of 2 by 2 (0.5, 0.25, 0.125,...)

Doing this to .011 would add .375 (0.25 + 0.125) to the value calculated above for a total of 6.375

Binary conversion algorithms (from float to binary)
Starting with 1, calculate a power of 2 that is greater than the value you're trying to convert.
In the case of the double 30.25, that would be 32.
Now divide that power by 2 (16 in this case).
Repeat this procedure:
- If you can subtract the current power of 2 from the double, do that and output a 1
- Otherwise output a 0.
- Divide the current power of 2 by 2

Do this as long as the current power of 2 is at least 1. (But print at least one digit before the binary point.)
(For the value 30.25, we would now have printed out 11110 while substracting 16,8,4,2)
Print out the binary point.
Then start the same repetitive procedure again. (The current power of 2 starts at 0.5)
This will lead to more 1's and 0's based on whether you can subtract the current power of 2.
You must print at least one digit after the binary point.
You can stop printing binary fraction digits when either the remaining value of the double is 0
or you have printed out 20 fractional digits.

Basic Code Quality Requirements (these apply to future assignments as well)

Your code must include your name in a comment at the top of the program!

Write multiple functions that each do one simply described thing. If it's not possible to see all the code of a function at once (without scrolling), the function is probably too long.

C requires you to define a function before you call it. If function A calls another function B, B should appear above A. (main naturally ends up at the bottom.) Or add prototypes to the top of your .c file if you wish.

The names you choose for variables, functions, etc. should be meaningful and descriptive.

Redundant code is code that is clearly repetitive and doesn't need to be. If you find yourself copying and pasting while coding, you very likely are writing some redundant code. Redunant code should be cleaned up before submission. Use functions and loops to eliminate redundancy.

Your code must be neatly and correctly indented. Don't use the tab key to indent. Tab doesn't have the same effect in every editor, so code that is beautifully indented in one editor can look bad in another. The text book is a good model for indenting.