Ryan Hueckel
jrhuecke@ucsc.edu
4/18/2021

CSE 13s Spring 2021
Assignment 2: A Small Numerical Library

Program Function:

This program compares the outputs of functions created by me to those provided by the math.h library. The functions created by me do mathematical approximations in order to get values as close as possible to the ones outputted by the math.h library functions. The functions in the program are inverse sine, inverse cosine, inverse tangent, and log. The user can call for any combination of the functions to be outputted, but each function will only be outputted once.

When the program is run it will output a table with the columns being the following: the input, the output of my function, the output of math.h function, and the difference between the two functions. Inverse sine and inverse cosine will be tested using inputs from -1 to 1, with 0.1 increases. Inverse tangent and log will be tested using inputs from 1 to 10, with 0.1 increases.

Program Structure:

The program consists of three files: ‘mathlib.c’, ‘mathlib.h’, and ‘mathlib-test.c’. The first file, ‘matlib.c’, contains the functions that I created. The functions in ‘mathlib.c’ do not print anything when ran and just return a value given an x value. The second file ‘mathlib.h’ is a header file that contains function calls for the functions I implemented in ‘mathlib.c’. It is the file that is included in the main file ‘mathlib-test.c’. The last file ‘mathlib-test.c’ contains the main function and is the file that outputs the tables. It is the file that is initially run and takes user inputs to decide what functions need to be called to produce the tables for the output.

mathlib-test.c:

This file contains the main function, takes command-line options as input, and prints out the tables by calling the functions contained in the other files.

Pseudocode:

```
//All the table functions are very similar to this one
arc_sin_table Function()
    print first two rows of table
    for (i from -1 to 1)
        print row of numbers using my arcSin and math.h’s asin
    return

main
//To make sure the tables are printed in the correct order, whether or not the tables need to be //printed are saved into a variable first, and then checked in the later section.
for (all of the command-line options = i)
    switch i:
    case ‘a’
        run_all = true
        break
    case ‘s’
        arc_sin_ran = true
        break
    case ‘c’
        arc_cos_ran = true
        break
    case ‘t’
        arc_tan_ran = true
        break
    case ‘l’
        log_ran = true
        break

//Now regardless of the order the command-line options were inputted, it will always output the //tables in
the same order. Also if the all option is entered then the program ends early to //prevent duplicate tables
from being printed.
if run_all == true
    run all the table printing functions
    end program
if arc_sin_ran == true
    run arc_sin_ran table function
if arc_cos_ran == true
    run arc_cos_ran table function
if arc_tan_ran == true
    run arc_tan_ran table function
if log_ran == true
    run log_ran table function
end program
```

end of pseudocode

arcSin Function:

This function uses a Taylor Series expansion in order to get a close approximation of the value outputted by the math.h function. The function consists of a while loop that will run until the most recent term calculated is less than a defined EPSILON value at which point it will return the approximation. At values of x close to 1 or -1, x is redefined by using this trig identity:

$$\sin^{-1}(x) = \cos^{-1}(\sqrt{1-x^2}).$$

This is done because the approximation loses accuracy when it gets close to 1 or -1, so the x is redefined before the Talyor Series Expansion is done. If the x input is close to -1, then the function inverts the sign of the sum at the end due to the trig identity that redefines x only outputting positive values.

Pseudocode:

```
//The function starts by checking if the x needs to be
//redefined, and if so then it also checks if it is negative.
//x is then redefined
if Abs(x) > .8
    if x < 0
        negative = true
    x_redefined = true
    redefines x using trig identity

//This for loop does the Taylor Series Expansion. It does not
//use factorial or power functions and instead stores the value
//of the current term's factorial and power to use in the calculation
// of the next term.
for (i=3; Abs(pterm) > EPISOLON; i += 2)
    calculates next term in taylor series and stores in pterm
    sum += pterm

//If the x was redefined then the sum is fixed, since the value gotten
//from the expansion is that of arcCos. Then if the x was negative the
//sum is corrected.
if x_redefined == true
    (pi/2) - sum
    if negative == true
        sum = -sum
return sum
```

end of pseudocode

arcCos Function:

This function utilizes the arcSin function in order to get its approximation. It simply does (pi/2) - arcSin(x) in order to find the approximation for arcCos.

arcTan Function:

This function also uses arcSin in order to get its approximation. It uses this equation that was provided by the assignment doc in order to find the approximation: arcTan = arcSin(x / sqrt(x^2 + 1)).

Log Function:

This function uses Newton's Method in order to find an approximation for ln(x). Its initial guess starts at 1 because that is somewhat in the middle of the expected range of outputs. The function runs until the previous guess is less than an EPSILON different from the current guess.

Pseudocode:

```
while guess - previous_guess > EPSILON
    previous_guess = guess
    guess = previous_guess - ((Exp(previous_guess) - x) / Exp(previous_guess))
return guess
```

Abs Function:

This function returns the absolute value of the given integer. All it does is check if the number is negative, and if so then it makes it positive.

Exp Function:

This function takes a value 'x' as input and returns the value of 'e^x'. This function is from Piazza.

Sqrt Function:

This function takes a value 'x' as input and returns the value of square root x. This function was also provided by Piazza.