Ruchit Patel
CSE13S, Spring 2021
Prof. Darrell Long
8 April 2021

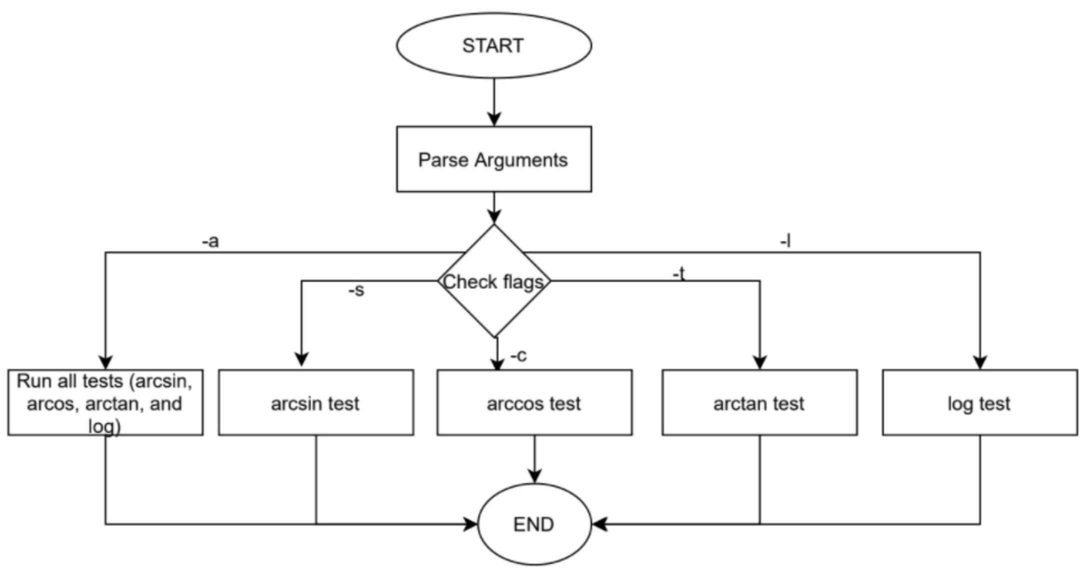
Assignment 2: A Small Numerical Library

Brief Description:

- In this lab, we implement four mathematical functions namely arcsin(x), arccos(x), arctan(x), and ln(x).
- All of the functions are implemented using Newton's method.

Flowchart:

FLOWCHART



Note: Any of the flags could be used together but the output should not repeat

Descriptions/Implementation:

Argument Parsing

- Arguments are parsed using getopt GNU utility.
- If a valid argument is encountered, an integer is bitwise ORed with predefined flag values to indicate that the argument is encountered.
- For example, if -a is encountered, inside the case for -a, the integer flag is ORed with 0x00001.
- After the while loop for parsing the arguments, any non-option arguments are checked (by comparing the optind with argc).
- After that, the integer flag is bitwise ANDed with original flag values to check if the flag is encountered. The -a flag is checked first. If it is encountered then all print functions are called and the program returns with 0. Else other respective flags are checked one by one and print functions for corresponding flags are called.
- If the program has insufficient or incorrect arguments (either encountered during or after parsing) the usage message is printed and the program returns with -1.

Pseudocode:

```
define flags
while (getopt returns valid)
        switch to the argument case if valid
                set the corresponding flag with OR
        if invalid print usage message and return
if non-option arguments then print usage message and return
if -a flag
        print and run all functions and return
else check other flags (with AND)
        run respective print function if flag encountered
```

Tests (Note: The tests stop when the next value is less than ε (10^{-10}).

1. arcSin(x): [Newton-Raphson Method]
    - The arcsin(x) function is calculated using the inverse method.
    - The inverse of arcsin(x) is sin(x). Thus, in order to calculate arcsin(z) = y, the root of z = sin(y) or sin(y) -z = 0 needs to be calculated.

- The root is calculated using Newton method. The following formula is used: $x_{k+1} = x_k - \frac{(sin(x_k) - z))}{cos(x_k)}$ [simplified: $x_{k+1} = x_k + \frac{(z - sin(x_k))}{cos(x_k)}$], where z is the input value, x_k is the initial guess, and x_{k+1} is the output/root value after $\frac{(z - sin(x_k))}{cos(x_k)} < \varepsilon$.

  Pseudocode:

  variables to store $\frac{(z - sin(x_k))}{cos(x_k)}$, x_{k+1}, initial guess x_0

  while $\frac{(z - sin(x_k))}{cos(x_k)}$ is greater than epsilon:

      calculate and store $\frac{(z - sin(x_k))}{cos(x_k)}$

      add calculated result to x_{k+1}

  return x_{k+1}

2. arcCos(x): [Newton-Raphson Method]
   - Implementation of arccos(x) is similar to that of arcsin(x).
   - The arccos(x) function is also calculated using the inverse method.
   - The inverse of arccos(x) is cos(x). Thus, in order to calculate arccos(z) = y, the root of z = cos(y) or cos(y) - z = 0 needs to be calculated.
   - The root is calculated using Newton method. The following formula is used: $x_{k+1} = x_k - \frac{(cos(x_k) - z))}{-sin(x_k)}$ [$simplified$: $x_{k+1} = x_k + \frac{(cos(x_k) - z))}{sin(x_k)}$], where z is the input value, x_k is the initial guess, and x_{k+1} is the output/root value after $\frac{(z - cos(x_k))}{-sin(x_k)} < \varepsilon$.

     Pseudocode:

     variables to store $\frac{(cos(x_k) - z))}{sin(x_k)}$, x_{k+1}, initial guess x_0(can not be zero since sin(0) = 0)

     while $\frac{(cos(x_k) - z))}{sin(x_k)}$ is greater than epsilon:

         calculate and store $\frac{(cos(x_k) - z))}{sin(x_k)}$

         add calculated result to x_{k+1}

return x_{k+1}

3. arcTan(x):
    - The arcTan(x) is calculated using the formula arcCos($\frac{1}{\sqrt{1+x^2}}$), x > 0.
    - The arcCos(x) function's pseudocode and design is the same as mentioned above.
    - The square root function is implemented using the inverse method.
    - The inverse of √x is x^2. Thus, in order to calculate √z = y , the root of z = y^2 or y^2 - z = 0 needs to be calculated.
    - The root is calculated using Newton method. The following formula is used:

    $x_{k+1} = x_k - \frac{(x_k^2 - z)}{2x_k}$ [$simplified$: $x_{k+1} = x_k + \frac{(z - x_k^2)}{2x_k}$], where z is the input value, x_k is the initial guess, and x_{k+1} is the output/root value after $\frac{(z - x_k^2)}{2x_k} < \varepsilon$.

    Pseudocode:

    variables to store $\frac{(z-x_k^2)}{2x_k}$, x_{k+1}, initial guess x_0

    call square root of $\frac{1}{\sqrt{1+x^2}}$:

        while $\frac{(z-x_k^2)}{2x_k}$ is greater than epsilon:

            calculate and store $\frac{(z-x_k^2)}{2x_k}$

            add calculated result to x_{k+1}

        return x_{k+1}

    call arcCos(x) function with x = return value from square root function

    Return the return value from arcCos(x) function

4. Log(x): [Newton-Raphson Method]
    - The natural log function is calculated using the Newton method. The following formula is used:

$x_{k+1} = x_k + \frac{(y - e^{x_k})}{e^{x_k}}$, where y is the input value, and x_{k+1} is the output value after k rounds.

- The ln(x) function is calculated using the inverse method.
- The inverse of ln(x) is e^x. Thus, in order to calculate ln(z) = y, the root of z = e^y or e^y -z = 0 needs to be calculated. e^x is calculated using Taylor series $\frac{x^k}{k!}$(credits given in the source file).
- The root is calculated using Newton method. The following formula is used:

$x_{k+1} = x_k - \frac{(e^{x_{k-z}})}{e^{x_k}}$ [simplified: $x_{k+1} = x_k + \frac{(z - e^{x_k})}{e^{x_k}}$], where z is the input value, x_k is the initial guess, and x_{k+1} is the output/root value after $\frac{(z - e^{x_k})}{e^{x_k}} < \varepsilon$.

Pseudocode:

variables to track x_k (1 to start), e^{x_k}, final

while x_k is greater than epsilon:

    calculate x_{k+1} using the formula

    assign x_{k+1} to x_k

    add x_k to final