mathlib design doc

goal: run mathlib_test and parse command line options to see which tests to print, then prints difference between my math library approximations and the ones from math.h for the following approximations: e, pi using bbp, euler, madhava, and viete methods, and square root using the newton method.

sudo:

mathlib_test.c:

main:
take command line options and set which tests will be run, display help message if needed
for each test that has been set to run:
print data from function, data from math.h function, and the difference between
if s was in command line options, print terms function

e.c:

e:
set terms to 0
set next term to 1
while next term > epsilon
add next term to sum
add 1 to terms
set denominator to 1
for every k above 0 //factorial loop
multiply denominator by loop count
set next term to 1 / denominator
add next term to sum
add 1 to terms
return sum

e_terms:
return terms

bbp.c:

pi_bbp:
set terms to 0

```
            infinite while
                set temp to 16
                if k is 0, set temp to 1
                else for k above 1, multiply temp by 16
                set next term to (1/temp) x ((4/(8k+1)) - (2/(8k+4)) - (1/(8k+5)) - (1/(8k+6)))
                if next term is < epsilon, add term to sum, break
                add term to sum
            return sum

        pi_bbp_terms:
            return terms

euler.c:
        pi_euler:
            set terms to 0
            set next term to 1
            while next term is > epsilon
                add next term to sum
                add 1 to terms
                set next term to 1 / k^2
            add next term to sum
            add 1 to terms
            return sqrt of 6 times sum

        pi_euler_terms:
            return terms

madhava.c:
        pi_madhava:
            set terms to 0
            set next term to 1
            while abs of next term is > epsilon
                add next term to sum
                add 1 to terms
                set temp to -3
                for k above 1 multiply temp by -3
                if k is > 0 set temp to reciprocal of temp
                else set temp to 1
                set next term to temp x (1/2k + 1)
            add next term to sum
            add 1 to terms
            return sum times sqrt of 12

        pi_madhava_terms:
```

```
                return terms

newton.c:
        sqrt_newton:
                set iters to 0
                if input is 0 return 0
                set x to input / 2
                while abs of x-y > epsilon
                        set y to x
                        set x to (x/y + y )/ 2
                        add 1 to iters
                return x

        sqrt_newton_iters:
                return iters

viete.c:
        pi_viete:
                set terms to 0
                set ak-1 to sqrt of 2
                set sum to 1
                set next term to sqrt 2 / 2
                while abs of next term - 1 > epsilon
                        multiply sum by next term
                        add 1 to terms
                        set ak to sqrt 2 + ak-1
                        set ak-1 to ak
                        set next term to ak / 2
                return 2 / sum

        pi_viete_terms:
                return terms
```