DESIGN asgn2

Githika Annapureddy

January 2023

mathlib-test.c

include "mathlib.h"
include the necessary libraries to print and use the math library
we are parsing user input using getopt() so we must define the options as aebmrvnsh'
for main function, accept int argc, char **argv as arguments
use a while loop to read from the user. use while getopt() is not empty to read
use a switch to code for each letter of the options
if the user enters a, print all of the test cases
if e, print test case for e.c
if b, print test case for bbp.c
if m, print test case for madhava.c
if r, print test case for euler.c
if v, print test case for viete.c
if n, print test case for newton.c
if s, print the number of iterations or terms calculated for each test case specified
if h, print the help menu which shows how to use the program (what to enter to see what tests)

as for the tests,
for the newton.c test, test the function sqrt_newton() in the range 0 to 10 in increments of 0.1
for each other test, run the first function in each program with an output like this
e() = [output], M_E= [math library's e], diff = [math library's e minus output]
if the program estimates pi, use M_PI instead

e.c

create global variable terms
for e()
create a variable k = 1
create a variable for the sum, initialize it to 0

enter while loop. while absolute value of the current term is less than Epsilon (the value for which is given in mathlib.h).. do the following

Instead of calculating 1/k! each time, I can multiply 1/k with 1/(k-1)! by saving that value.

At each k, I need to
calculate 1/k
multiply it by the previous value which is 1/(k-1)!
make the previous variable this new value
add this new value to the total sum
iterate k

outside of loop: terms = k
return sum

for e_terms(): return the global variable terms

madhava.c

create global variable terms
for pi_madhava()
create a variable k = 1
create a variable for the sum, initialize it to 0
enter while loop. while absolute value of the current term is less than Epsilon (the value for which is given in mathlib.h).. do the following To do the madhava series, at each iteration k I need to
calculate -3 to the -k divided by 2 times k plus one
add this value to the sum
iterate k

outside of loop: terms = k
return sum * square root of 12 (call square root function in newton.c)

for pi_madhava_terms(): return the global variable terms

euler.c

create global variable terms
for pi_euler()
create a variable k = 1
create a variable for the sum, initialize it to 0
enter while loop. while absolute value of the current term is less than Epsilon

(the value for which is given in mathlib.h).. do the following

I need to calculate the sum of the series, then multiply it by 6. Then I need to take the square root of the whole thing.

To find the sum of the series: for each iteration of k, I will
sum equals sum plus 1 over k squared
iterate k
outside of loop: terms = k
return square root of the sum (get square root using the function from newton.c)

for pi_euler_terms(): return the global variable terms

bbp.c

create global variable terms
for pi_bbp()
create a variable k = 1
create a variable for the sum, initialize it to 0
enter while loop. while absolute value of the current term is less than Epsilon (the value for which is given in mathlib.h).. do the following At each k, I need to
add the term (16 to the -k power times k times (120 times k plus 151) plus 47) divided by (k times (k times (k times (512 times k plus 1024) + 712) + 194) + 15) to the sum
iterate k

outside of loop: terms = k
return sum

for pi_bbp_terms(): return the global variable terms

viete.c

create global variable factors
for pi_viete()
create a variable k = 1
create a variable for the sum, initialize it to 0
enter while loop. while absolute value of the current term is less than Epsilon (the value for which is given in mathlib.h).. do the following At each k, I need to
store previous value of loop. initialize to 0.
variable current is the square of 1/2 plus (1/2 times the previous value)
then update previous to be current
then add current to the total sum iterate k

outside of loop: factors = k
divide the sum by 2 and then make it equal 1 over itself
return sum

for pi_viete_factors(): return the global variable factors

newton.c

create global variable iterations
for sqrt_newton(x) that takes x as the number to find the square root of
make a variable z = 0
make a variable root = 1
enter while loop (while absolute value of root minus z is greater than some epsilon
z equals root
root equals .5 times (z plus x over z)
add one to iterations
outside of loop: iterations = k
return root

for sqrt_newton_iters(): return the global variable iterations

Makefile

makefile only creates an executable for mathlib-test.c only, but you must compile the other prorams for the test program
use -lm on the line that crates the mathlib-test executable in order to allow usage of the math library