High-level design Pseudo Code (Before I completed any code)

In the c program:

- e.c
- madhava.c
- euler.c
- bbp.c
- viete.c
- newton.c
- mathlib-test.c

e.c and its pseudo code:

- Have a variable called epsilon that is preset to a very small number that will control the maximum number of iterations
- Use a while loop to simulate the addition of an infinite series
- Use a parameter K that represents the last added value of the series until it is smaller than Epsilon
- In the for loop, add 1/K! to the previous value or total value variable in the loop and set it into a next value variable
- The result is then added to the total variable
- Iteraiate the appropriate variables and repeat loop
- Return total var

madhava.c and its pseudo code:

- Have a variable called epsilon that is preset to a very small number that will control the maximum number of iterations
- Use a while loop to simulate the addition of an infinite series
- Use a parameter a “last” variable that represents the last added value of the series until it is smaller than Epsilon
- Have a k variable that represents the current element of the series
- In the main loop the previous value will be added to ((-3)^-k)/(2k+1)
- (-3)^-k will be represented as 1/(-3)^k and I will use a for loop to to multiply -3 bytiself k times
- Iteraiate the appropriate variables and repeat loop
- Exit the loop
- Once the loop is done take square root of 12 using the function provided
- Multiply the last two values(total and) together and return it
- Return the total after multiplying

euler.c

- Have a variable called epsilon that is preset to a very small number that will control the maximum number of iterations
- Use a for loop to simulate the addition of an infinite series

- Use a parameter K that represents the last added value of the series until it is smaller than Epsilon
- In the for loop, add 1/(k*k) to the previously held value or to the total value variab,e
- Once the loop is done multiply the total variable value by 6
- Then use provided square root function
- Return the result

bbp.c (super high level sudo code since the formula is quite complexe and hard to write down):

- Have a variable called epsilon that is preset to a very small number that will control the maximum number of iterations
- Use a for loop to simulate the addition of an infinite series
- Use a parameter K that represents the last added value of the series until it is smaller than Epsilon
- Have a total value variable to keep track of total value
- Each iteration of loop added value of K when it has gone through the equation add it to total value
- Iteraiate the appropriate variables and repeat loop
- Return the total value when done

viete.c :

- Have a variable called epsilon that is preset to a very small number that will control the maximum number of iterations
- Use a for loop to simulate the product infinite series
- Use a parameter K that represents the last added value of the series until it is smaller than Epsilon
- Have a total value variable to keep track of the total
- Have a variable that represents the numerator variable which will start of as sqrt(2)
- In the loop make the numerator equal to the previous value added 2 and then square rooted
- Then divide the numerator by two and set it to k
- Multiply k with the total var
- Iteraiate the appropriate variables and repeat loop

newton.c:

- Follow provided code

mathlib-test.c:

- Use functions for all the test case comparisons
- Use getopt to parse options
- Use switch statements for command options controls for the test file
- For -a option call all test functions