1/23/23 "The Indian Clerk" Book

libraries you're linking in linux come after

you can do cc ./(function) -lm

↑

import math library

Don't use math lib for lab2

Numerical Methods

multiplication - shift + add

division - shift + subtract

addition - exclusive or

to compute sin, cos, etc

- use taylor series
  - more terms give you more precision

todo $x^{(\text{floating point})}$, use log

floating point modulus

- for centering trig functions

(ex: sin 400π = sin 2π

Pade Approximate

- generates a couple polynomials that don't need loops
- can put more terms to be more accurate

Harmonics series

- extremely slow

log is slow

$^{0}$

- do it by inverting $e^x$ series

for $\sqrt{x}$ or $\log(x)$ you can do a binary search
bc no Taylor series for $\sqrt{x}$
- meh speed ~750 operations for 64 bits

Binary search
- cut value in half, square it, check if it is less than x, if difference within some error, then it doesn't matter, exit

Can do better

$\sqrt{x^2} = x$

- invert square function
- use newton, iterate
- close approximation

log:
- works with euler's number, gets weird after larger values

in our algorithms, there are other operations worth doing to save time exs

$\sqrt{4x} = 2\sqrt{x}$

exs $\log(xe^f) = \log(x) + \log(e^f)$

$\log x + f$

Can use newton's method w/ inverse trig functions

Round off errors
- happens w/ almost every float operation
- trying to write real #s as floating points give you weird decimals

adding & subtracting floats of different magnitudes lose precision

$10^{-14}$ good epsilon generally (error value)

Relative error vs absolute error
relative error is relative to the values, very critical