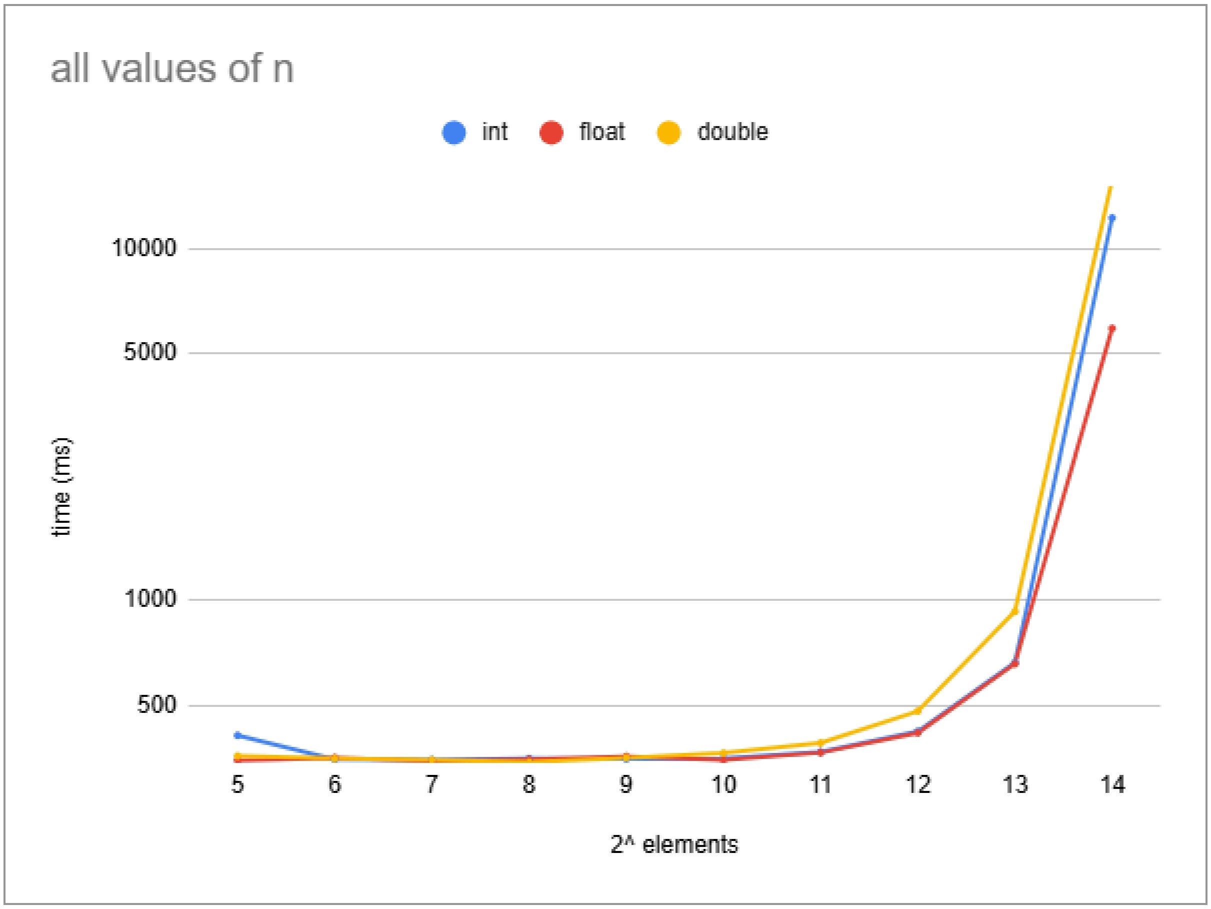
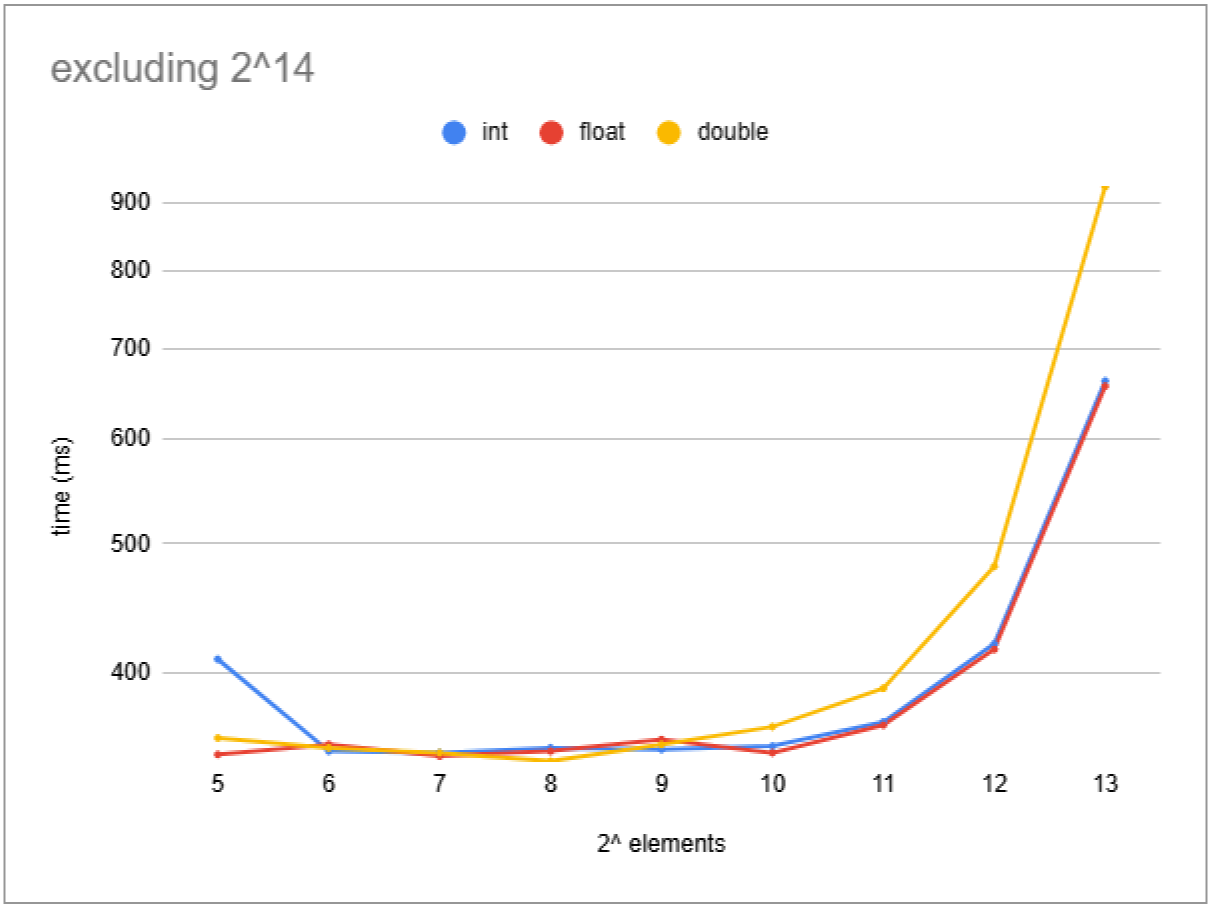
Leah Blasczyk
HW07
GitHub Username: **Lebrra**
(Assignment 7 Path: https://github.com/Lebrra/repo759/tree/main/HW07)

task1 plot(s)





(1c) The best performing block_dim when n = 2^14 seems to be around □

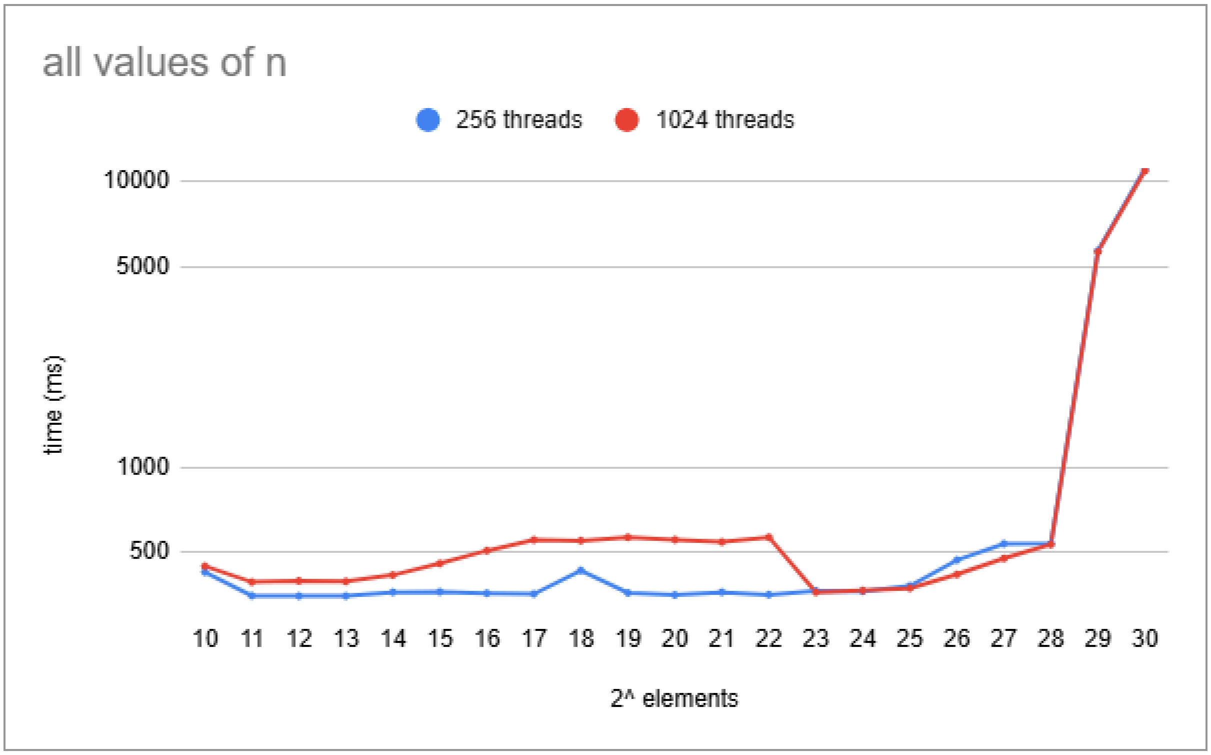
(1d) Floats and ints seem to perform about the same, however doubles are definitely slower. I would assume that since doubles are 8 bits and floats and ints are (usually) 4 bits, this is causing a slow down with larger values of n.

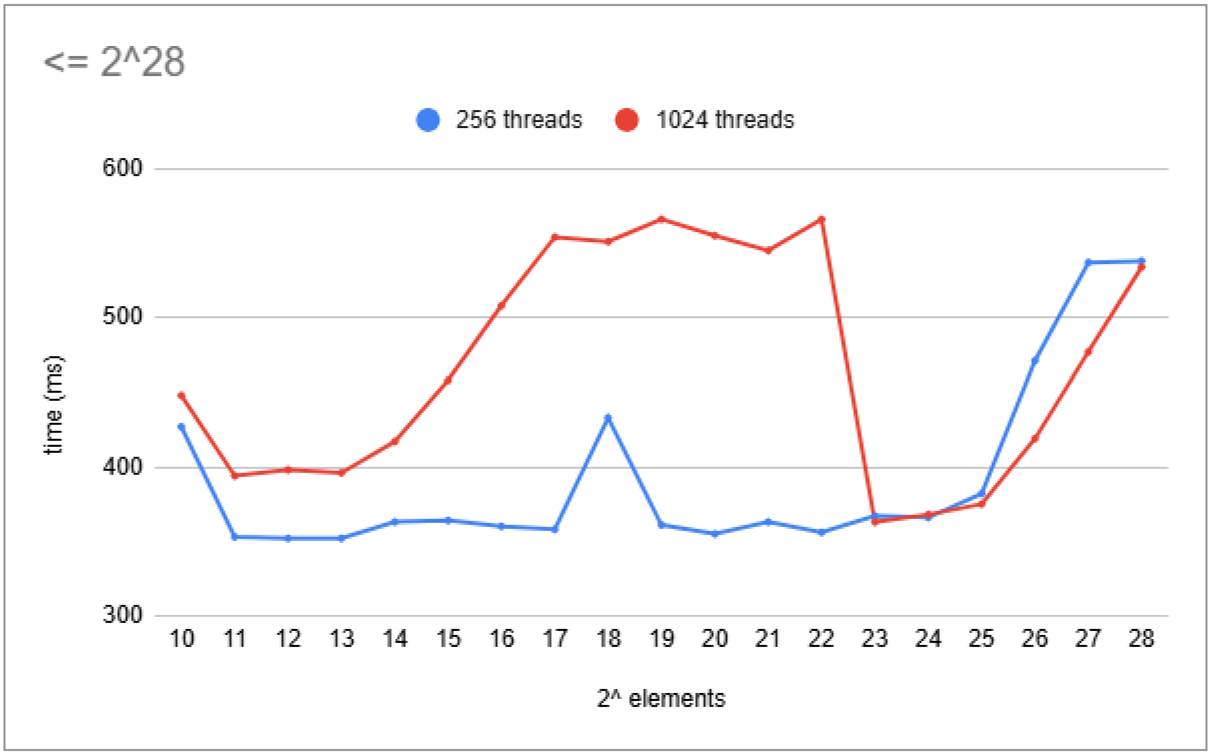
(1e) My HW06 with an n = 2^14 performed much worse than the shared/tiled implementation here. At 1024 threads HW06 ran in 58346ms and with 64 threads it finished in 58208ms. I would expect the shared space to perform much better than non-shared space because of class, however the tiled structure would also help with more consistent accessing of data that is placed near each other.

(1f) I gave a few attempts to run my mmul implementation from HW02 with n = 2^14 but was met with several timeouts, so I'm going to assume that it must take longer than 10 minutes. Clearly the GPU implementations work better than the CPU for matmul, which makes sense since the GPU is specialized for compute-intensive, highly data-parallel computation (taken from the slides).

task2 I unfortunately ran out of time to fully finish, but here are my results:
I didn't get to

- multiple kernel calls for bigger values of N
- at n >= 65536 my reduce result is 0 (probably due to too many blocks)


all values of n
● 256 threads
● 1024 threads
time (ms)
2^ elements


<= 2^28
● 256 threads
● 1024 threads
time (ms)
2^ elements