Clinton Hawkes
hawkesc@oregonstate.edu
CS475-400
Project #0

Introduction to Parallel Programming

1. This was run on my laptop with the following specs:
   Lenovo X1 Extreme
   Intel Core i7 9750H (6 cores/12 threads)
   32GB Ram
   Nvidia GTX 1650
   Running Linux kernel 5.5.9

2. Results using SIZE = 32768 and NUMTRIES = 100000

   Using 1 thread:
   Peak Performance = 590.16 MegaMults/Sec
   Avg Performance = 573.20 MegaMults/Sec

   Using 4 threads:
   Peak Performance = 2116.66 MegaMults/Sec
   Avg Performance = 1874.43 MegaMults/Sec

   I did not use the -O3 flag when compiling my program, because it seemed to throw off the results. I was getting an increase of about 2 when using it. The MegaMults/Sec were also much higher when using it, which is good, but the figures I was seeing did not match what I was expecting. I was expecting an increase of about 4, due to using 4 threads rather than 1. Peak performance for 1 thread was about 5000 MM/S and peak performance for 4 threads when using the flag was about 10000 MM/S. I saw a post on Slack that mentioned results were closer to the expected results when omitting the flag, so I dropped it and got the results presented above.

3. S = 2116.66/590.16 = 3.59

4. We may be using 4 times the number of threads when implementing parallel computing, but there is a certain amount of overhead involved to setup and take down the multi-threading. This may include creating the thread pool, thread wait time, memory management, and deleting the threads when task has been completed. This may be why the speedup is below 4.

5. Parallel Fraction = (4/3)*(1-(1/3.59)) = .9619
   I don't know what this means, but I am thinking it is the portion of code that can be effectively "parallelized".

   I hope that is enough commentary. This was a pretty simple assignment with most of the code provided, so I am at a loss for words. I'm sure the next projects will involve more analysis.