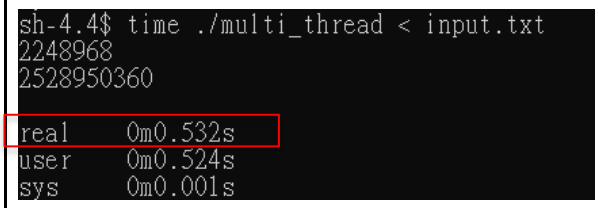
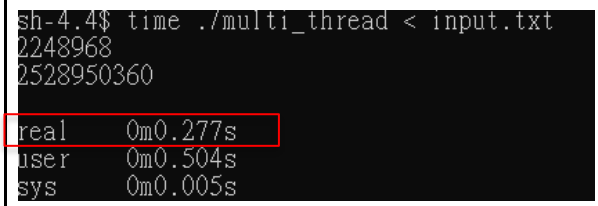
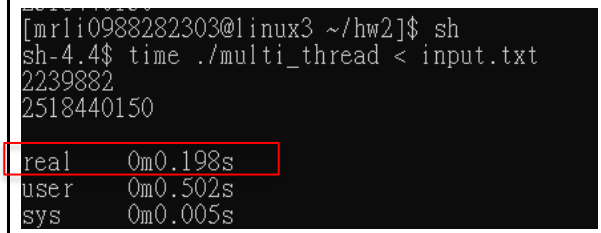
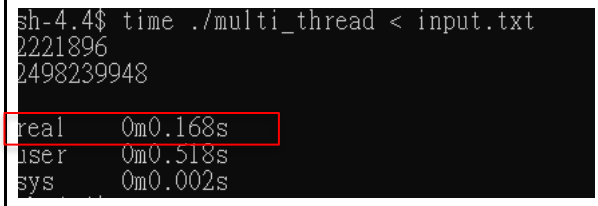
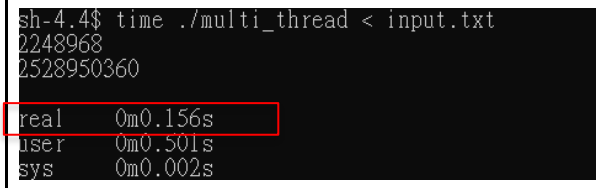
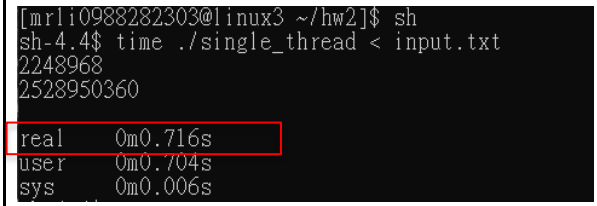
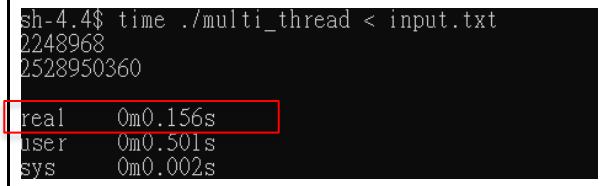
2022 NYCU OS HW2 report

Question	Answer
Q1. (5pts) Briefly describe your design for the add, multiple function of matrix, the thread management. Also, describe the number of threads in the Multi-thread program.	In order to increase speedup, I combine the add function and the multiple function in a function. Next, create a struct “Range” including four variables: start, end, addans, and mulans. The variables start and end are the index of rows this thread need to handle. The addans is the answer for addition in this thread and the mulans is the answer for multiplication in this thread. At first, there is only one thread which handling 500 rows. However, in the multi-thread program, I create twenty threads. Every thread just needs to handle 25 rows. I use pthread_create to create thread to calculate the answer for matrix addition and matrix multiplication. Moreover, I also use pthread_join to ensure every thread is finished. Finally, sum up the answer of every thread and output the answer.
Q2. (15pts) Try at least 3 kinds of number of threads, and compare the difference in time. (Take screenshots of the time of each case) Also, explain the results.	 Thread Num = 1  Thread Num = 2  Thread Num = 3

	 Thread Num = 13  Thread Num = 20 When thread num = 1, the time is less than single thread because I put matrix addition and matrix multiplication in same function. When thread num = 2, the time decreases about 50%. When thread num = 3, the time decreases more. So, we can conclude that multi-thread can speed up the program. When thread num > 13, the time is roughly the same. I think that is related to the size of data that one thread needs to handle. For example, every thread needs to handle about 38 rows for thread num = 13 and 25 rows for thread num = 20. The gap of total calculation is rather small which reflects on the result. However, for thread num = 1, the thread needs to handle 500 rows, and for thread num = 2, every thread needs to handle 250 rows. The gap of total calculation is huge. Thus, I think if the size of matrix is bigger than 500*500. The time for thread num = 13 and the time for thread num = 20 must be different.
Q3. (10pts) Show the best speedup between multi-thread and single-thread. (Take screenshots of the time of single-thread and multi-thread) Also, explain why multi-thread is faster.	 single-thread

	 Multi-thread: 20 threads Speedup = 0.716/0.156 = 4.589 In single-thread, the thread needs to handle 500 rows. However, in multi-thread, I use 20 threads to handle 500 rows, that is, every thread just needs to handle 25 rows. The size of data is the key to decide the time. So, multi-thread is obviously faster than single-thread.