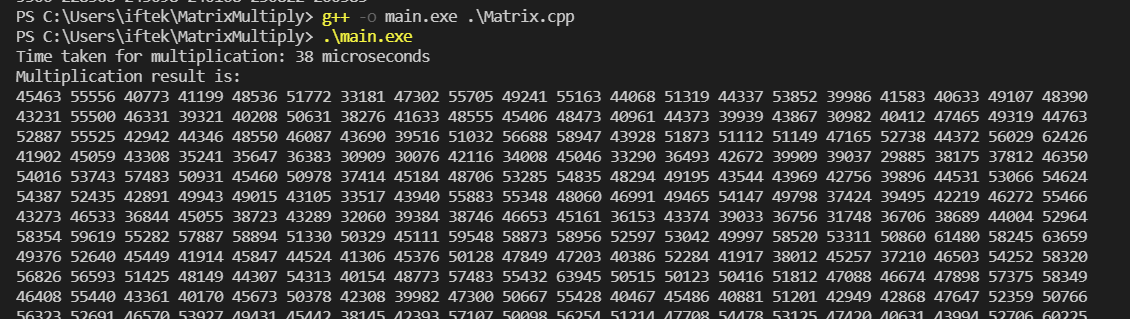
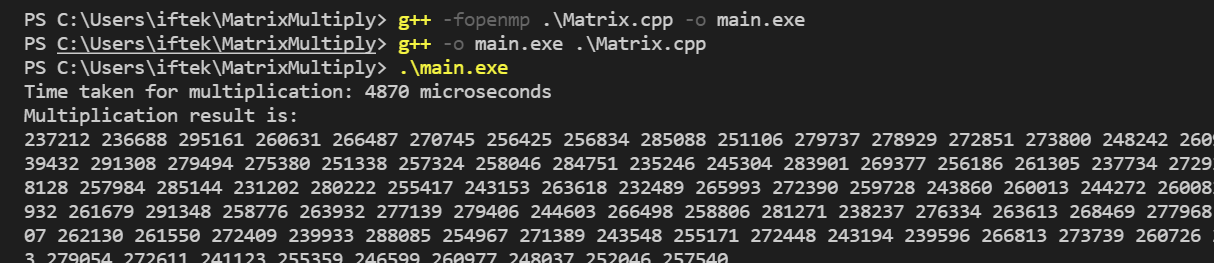
TaskM2.T1P: Parallel Matrix Multiplication

Sequential program

Runtime of sequential program of Matrix Multiplication with size of 20:



Runtime of sequential program of Matrix Multiplication with size of 100:



Increased execution time for larger matrix size. Time can be reduced by parallelising the program using Threads. We can use PTThread library or OpenMP library for threading purposes.

How to parallelise

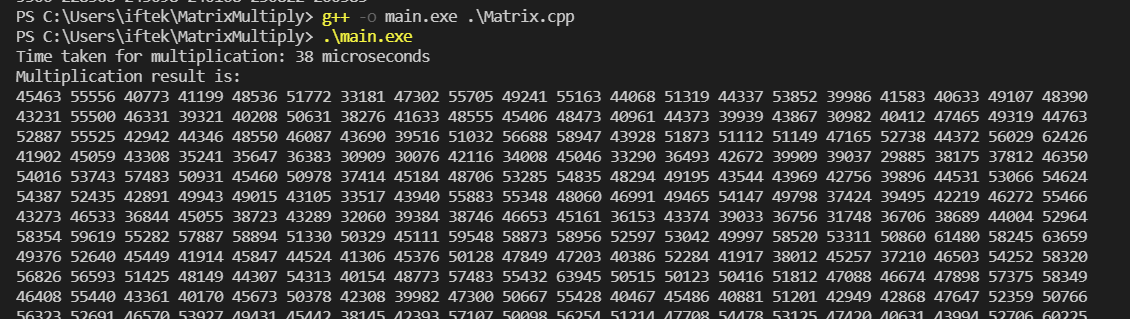
First step is to choose what parts of the program to parallelise. Separate threads can be used to do a specific task in the program and then threads can be joined together for getting the final input.

In the occasion that I used PTThread in my program-

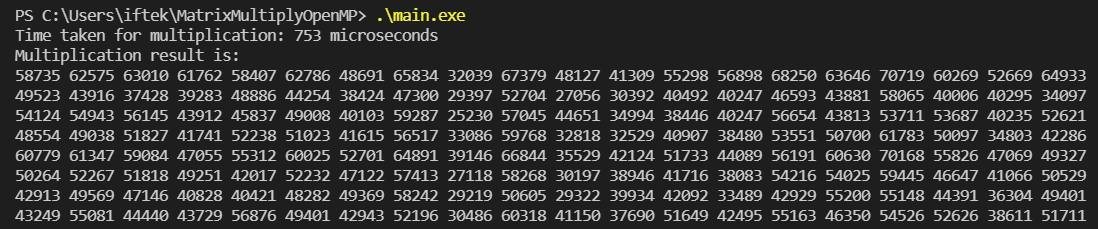
* I am populating 2 of my matrices (for multiplication) with random numbers by dividing the number of threads into 2 equal halves. For instance, if number of threads= 12, 6 of the threads will be used for randomising array A, and 6 for randomising array B.
* I am multiplying the arrays using threads and here I am diving the threads according to the partition size. Partition size = size of matrix / total number of threads.

Comparison of Sequential vs PTThread vs OpenMP

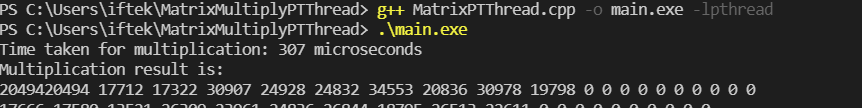
**For Matrix size of 20**

****

Sequential

****

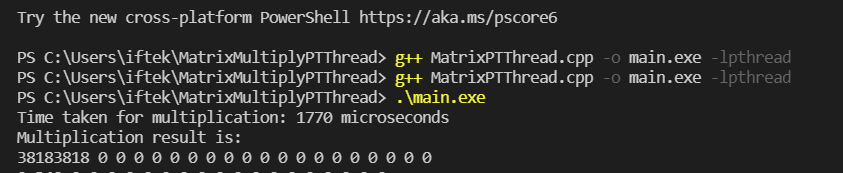
OpenMP (2 threads)

****

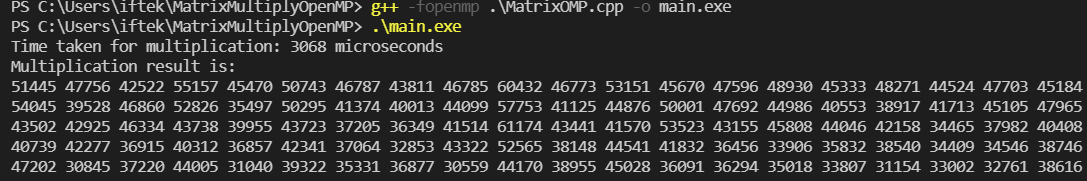
PT Thread (2 threads)

Above is a comparison of execution time of multiplication of size 20 matrix. As it can be seen, **sequential is performing the** best with a **small matrix size.** OpenMP is the slowest amongst them.

Increasing threads also did not decrease the multiplication time for size 20 matrices. It significantly worsened for both PT Thread and OMP as seen below.

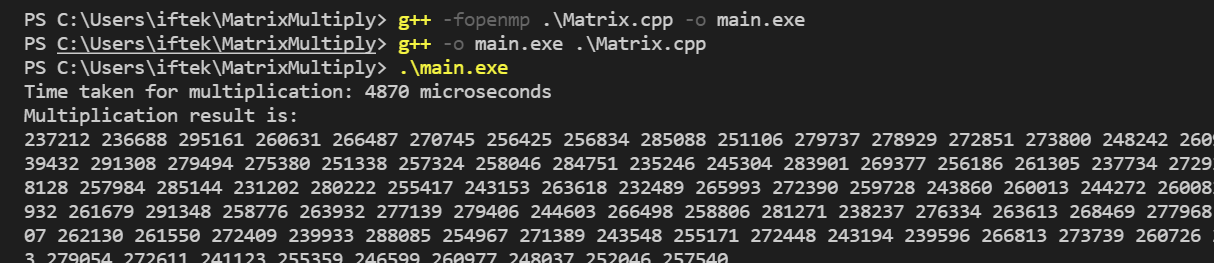


PT Thread (12 threads)

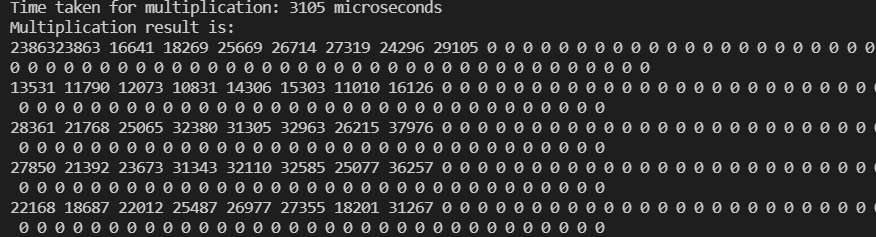


OMP(12 threads)

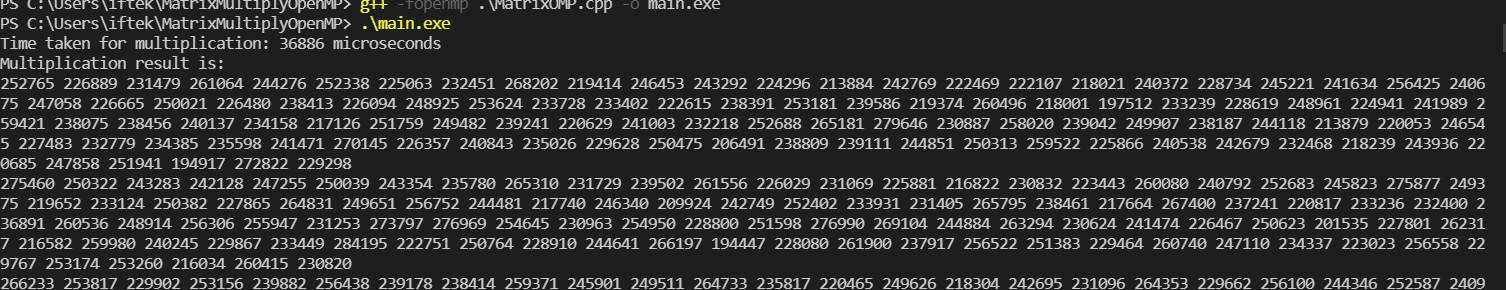
**For Matrix size of 100**



Sequential



PTThread (12 threads)



OMP (12 Threads)

For a larger matrix size of 100, sequential is performing badly. It’s taking 4870ms while PT Thread is completing in 3105ms. However, OMP is again performing the worst amongst them with a significantly larger time of 36886ms. Both parallelisation was done using 12 threads.

**Conclusion**

Sequential program is more efficient for small scale programs. When doing large scale calculation, like matrix multiplication of 100 sized matrices, task can be performed quickly with appropriate parallelisation of the program.