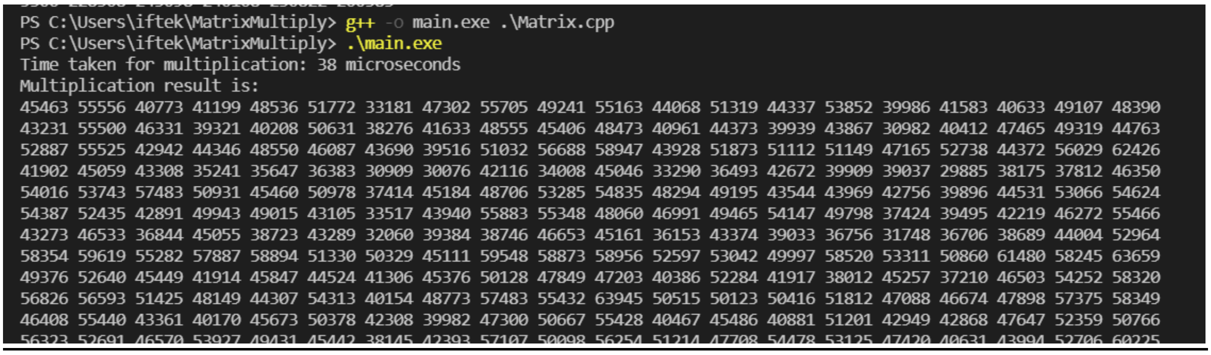
TaskM2.T1P: Parallel Matrix Multiplication

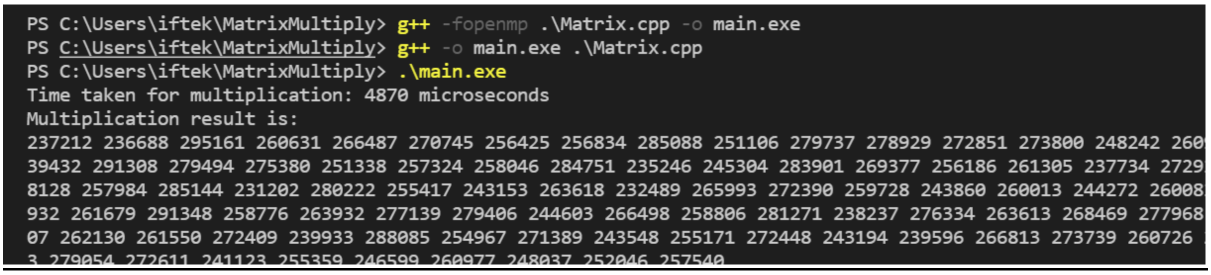
Sequential program

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



```
PS C:\Users\iftek\MatrixMultiply> g++ -o main.exe .\Matrix.cpp
PS C:\Users\iftek\MatrixMultiply> .\main.exe
Time taken for multiplication: 38 microseconds
Multiplication result is:
45463 55556 40773 41199 48536 51772 33181 47302 55705 49241 55163 44068 51319 44337 53852 39986 41583 40633 49107 48390
43231 55500 46331 39321 40208 50631 38276 41633 48555 45406 48473 40961 44373 39939 43867 30982 40412 47465 49319 44763
52887 55525 42942 44346 48550 46087 43690 39516 51032 56688 58947 43928 51873 51112 51149 47165 52738 44372 56029 62426
41902 45059 43308 35241 35647 36383 30909 30076 42116 34008 45046 33290 36493 42672 39909 39037 29885 38175 37812 46350
54016 53743 57483 50931 45460 50978 37414 45184 48706 53285 54835 48294 49195 43544 43969 42756 39896 44531 53066 54624
54387 52435 42891 49943 49015 43105 33517 43940 55883 55348 48060 46991 49465 54147 49798 37424 39495 42219 46272 55466
43273 46533 36844 45055 38723 43289 32060 39384 38746 46653 45161 36153 43374 39033 36756 31748 36706 38689 44004 52964
58354 59619 55282 57887 58894 51330 50329 45111 59548 58873 58956 52597 53042 49997 58520 53311 50860 61480 58245 63659
49376 52640 45449 41914 45847 44524 41306 45376 50128 47849 47203 40386 52284 41917 38012 45257 37210 46503 54252 58320
56826 56593 51425 48149 44307 54313 40154 48773 57483 55432 63945 50515 50123 50416 51812 47088 46674 47898 57375 58349
46408 55440 43361 40170 45673 50378 42308 39982 47300 50667 55428 40467 45486 40881 51201 42949 42868 47647 52359 50766
56323 52691 46570 53927 49431 45442 38145 42393 57107 50098 56254 51214 47708 54478 53125 47420 40631 43994 52706 60225
```

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



```
PS C:\Users\iftek\MatrixMultiply> g++ -fopenmp .\Matrix.cpp -o main.exe
PS C:\Users\iftek\MatrixMultiply> g++ -o main.exe .\Matrix.cpp
PS C:\Users\iftek\MatrixMultiply> .\main.exe
Time taken for multiplication: 4870 microseconds
Multiplication result is:
237212 236688 295161 260631 266487 270745 256425 256834 285088 251106 279737 278929 272851 273800 248242 260
39432 291308 279494 275380 251338 257324 258046 284751 235246 245304 283901 269377 256186 261305 237734 2729
8128 257984 285144 231202 280222 255417 243153 263618 232489 265993 272390 259728 243860 260013 244272 26008
932 261679 291348 258776 263932 277139 279406 244603 266498 258806 281271 238237 276334 263613 268469 277968
07 262130 261550 272409 239933 288085 254967 271389 243548 255171 272448 243194 239596 266813 273739 260726
3 279054 272611 241123 255359 246599 260977 248037 252046 257540
```

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

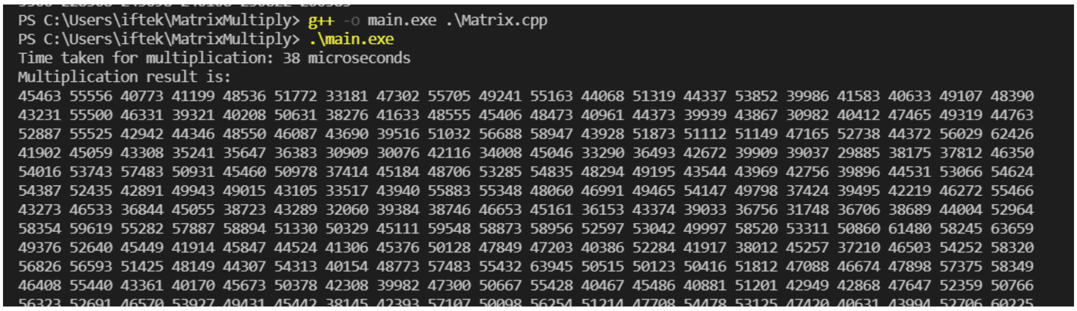
First step is to choose what parts of the program to parallelise. Separate threads can be used to do a specific task in the program and then threads can be joined together for getting the final input.

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.

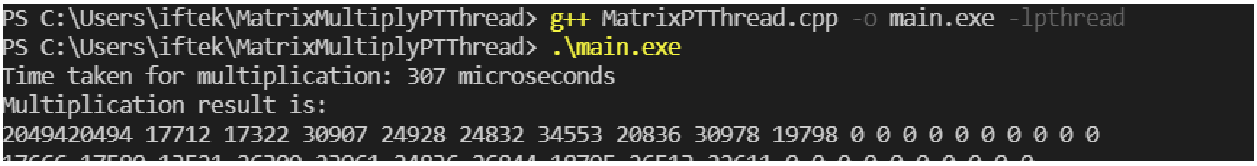
Comparison of Sequential vs PTThread vs OpenMP

For Matrix size of 20



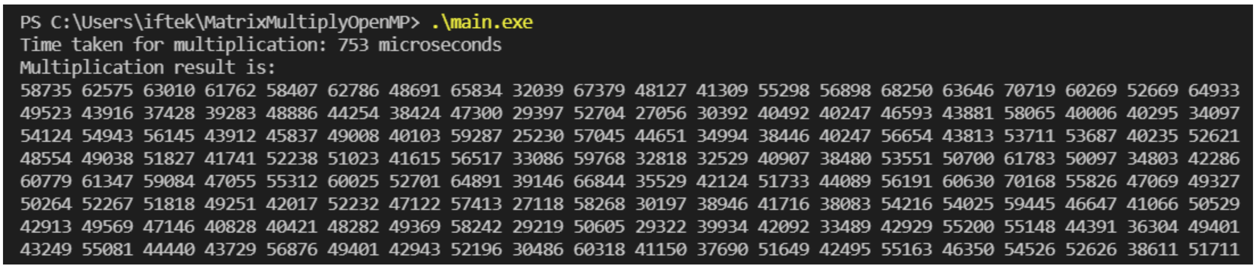
```
PS C:\Users\iftek\MatrixMultiply> g++ -o main.exe .\Matrix.cpp
PS C:\Users\iftek\MatrixMultiply> .\main.exe
Time taken for multiplication: 38 microseconds
Multiplication result is:
45463 55556 40773 41199 48536 51772 33181 47302 55705 49241 55163 44068 51319 44337 53852 39986 41583 40633 49107 48390
43231 55500 46331 39321 40208 50631 38276 41633 48555 45406 48473 40961 44373 39939 43867 30982 40412 47465 49319 44763
52887 55525 42942 44346 48550 46087 43690 39516 51032 56688 58947 43928 51873 51112 51149 47165 52738 44372 56029 62426
41902 45059 43308 35241 35647 36383 30909 30076 42116 34008 45046 33290 36493 42672 39909 39037 29885 38175 37812 46350
54016 53743 57483 50931 45460 50978 37414 45184 48706 53285 54835 48294 49195 43544 43969 42756 39896 44531 53066 54624
54387 52435 42891 49943 49015 43105 33517 43940 55883 55348 48060 46991 49465 54147 49798 37424 39495 42219 46272 55466
43273 46533 36844 45055 38723 43289 32060 39384 38746 46653 45161 36153 43374 39033 36756 31748 36706 38689 44004 52964
58354 59619 55282 57887 58894 51330 50329 45111 59548 58873 58956 52597 53042 49997 58520 53311 50860 61480 58245 63659
49376 52640 45449 41914 45847 44524 41306 45376 50128 47849 47203 40386 52284 41917 38012 45257 37210 46503 54252 58320
56826 56593 51425 48149 44307 54313 40154 48773 57483 55432 63945 50515 50123 50416 51812 47088 46674 47898 57375 58349
46408 55440 43361 40170 45673 50378 42308 39982 47300 50667 55428 40467 45486 40881 51201 42949 42868 47647 52359 50766
56323 52691 46570 53927 49431 45442 38145 42393 57107 50098 56254 51214 47708 54478 53125 47420 40631 43994 52706 60225
```

Sequential



```
PS C:\Users\iftek\MatrixMultiplyPTThread> g++ MatrixPTThread.cpp -o main.exe -lpthread
PS C:\Users\iftek\MatrixMultiplyPTThread> .\main.exe
Time taken for multiplication: 307 microseconds
Multiplication result is:
2049420494 17712 17322 30907 24928 24832 34553 20836 30978 19798 0 0 0 0 0 0 0 0 0 0
```

PT Thread (2 threads)

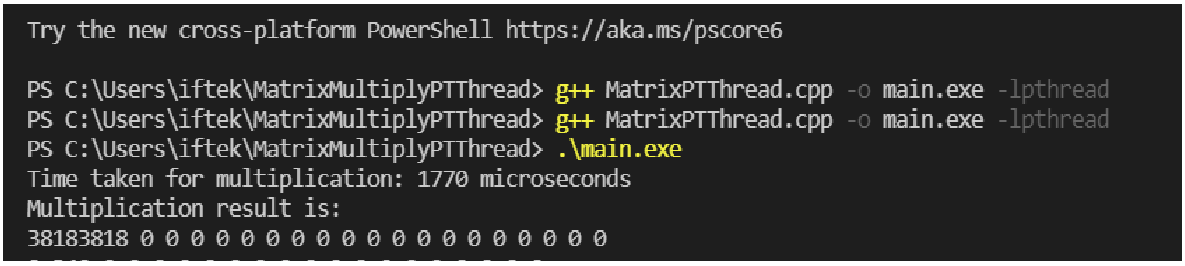


```
PS C:\Users\iftek\MatrixMultiplyOpenMP> .\main.exe
Time taken for multiplication: 753 microseconds
Multiplication result is:
58735 62575 63010 61762 58407 62786 48691 65834 32039 67379 48127 41309 55298 56898 68250 63646 70719 60269 52669 64933
49523 43916 37428 39283 48886 44254 38424 47300 29397 52704 27056 30392 40492 40247 46593 43881 58065 40006 40295 34097
54124 54943 56145 43912 45837 49008 40103 59287 25230 57045 44651 34994 38446 40247 56654 43813 53711 53687 40235 52621
48554 49038 51827 41741 52238 51023 41615 56517 33086 59768 32818 32529 40907 38480 53551 50700 61783 50097 34803 42286
60779 61347 59084 47055 55312 60025 52701 64891 39146 66844 35529 42124 51733 44089 56191 60630 70168 55826 47069 49327
50264 52267 51818 49251 42017 52232 47122 57413 27118 58268 30197 38946 41716 38083 54216 54025 59445 46647 41066 50529
42913 49569 47146 40828 40421 48282 49369 58242 29219 50605 29322 39934 42092 33489 42929 55200 55148 44391 36304 49401
43249 55081 44440 43729 56876 49401 42943 52196 30486 60318 41150 37690 51649 42495 55163 46350 54526 52626 38611 51711
```

OpenMP (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.

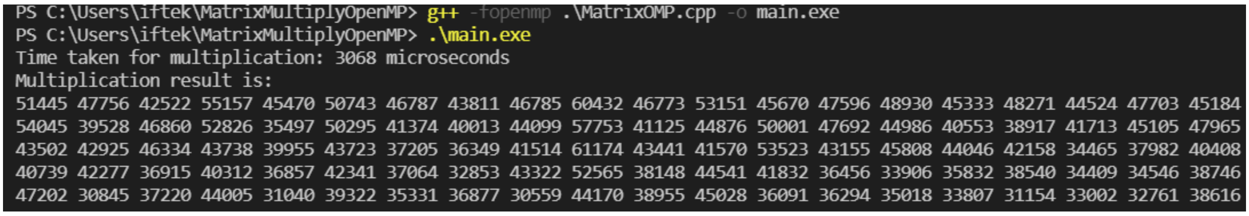
Increasing threads also did not decrease the multiplication time for size 20 matrices. It significantly worsened for both PT Thread and OMP as seen below.



```
Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\iftek\MatrixMultiplyPTThread> g++ MatrixPTThread.cpp -o main.exe -lpthread
PS C:\Users\iftek\MatrixMultiplyPTThread> g++ MatrixPTThread.cpp -o main.exe -lpthread
PS C:\Users\iftek\MatrixMultiplyPTThread> .\main.exe
Time taken for multiplication: 1770 microseconds
Multiplication result is:
38183818 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

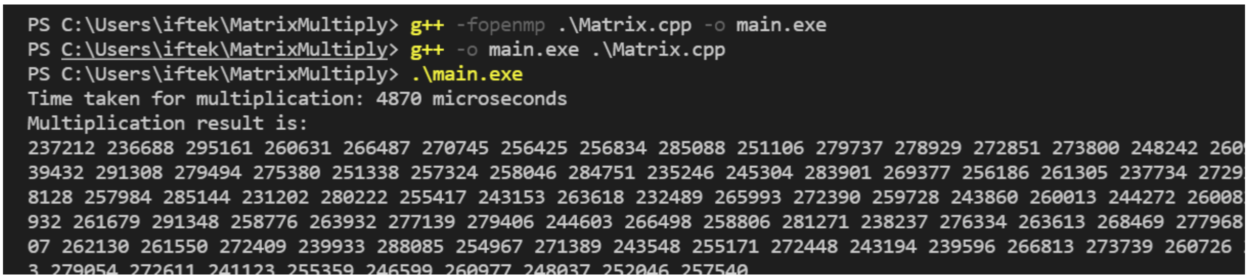
PT Thread (12 threads)



```
PS C:\Users\iftek\MatrixMultiplyOpenMP> g++ -fopenmp .\MatrixOMP.cpp -o main.exe
PS C:\Users\iftek\MatrixMultiplyOpenMP> .\main.exe
Time taken for multiplication: 3068 microseconds
Multiplication result is:
51445 47756 42522 55157 45470 50743 46787 43811 46785 60432 46773 53151 45670 47596 48930 45333 48271 44524 47703 45184
54045 39528 46860 52826 35497 50295 41374 40013 44099 57753 41125 44876 50001 47692 44986 40553 38917 41713 45105 47965
43502 42925 46334 43738 39955 43723 37205 36349 41514 61174 43441 41570 53523 43155 45808 44046 42158 34465 37982 40408
40739 42277 36915 40312 36857 42341 37064 32853 43322 52565 38148 44541 41832 36456 33906 35832 38540 34409 34546 38746
47202 30845 37220 44005 31040 39322 35331 36877 30559 44170 38955 45028 36091 36294 35018 33807 31154 33002 32761 38616
```

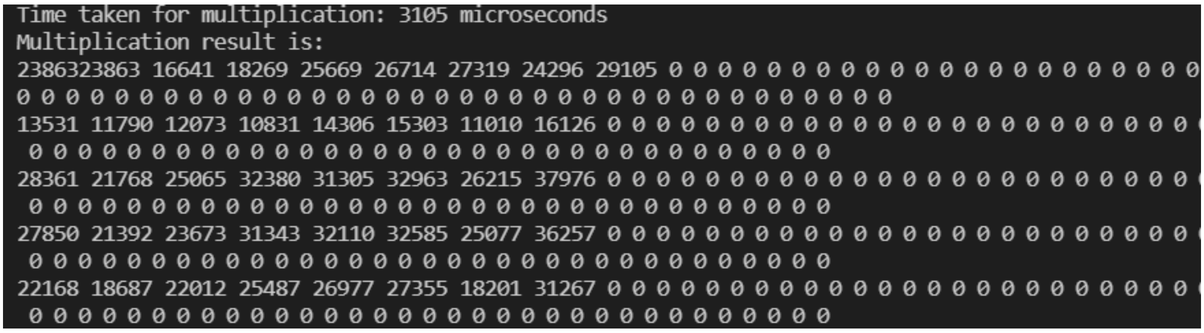
OMP(12 threads)

For Matrix size of 100



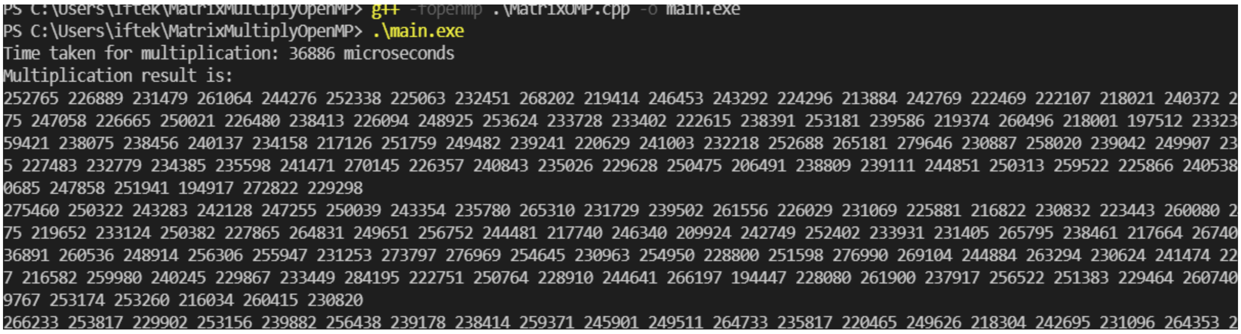
```
PS C:\Users\iftek\MatrixMultiply> g++ -fopenmp .\Matrix.cpp -o main.exe
PS C:\Users\iftek\MatrixMultiply> g++ -o main.exe .\Matrix.cpp
PS C:\Users\iftek\MatrixMultiply> .\main.exe
Time taken for multiplication: 4870 microseconds
Multiplication result is:
237212 236688 295161 260631 266487 270745 256425 256834 285088 251106 279737 278929 272851 273800 248242 260
39432 291308 279494 275380 251338 257324 258046 284751 235246 245304 283901 269377 256186 261305 237734 2729
8128 257984 285144 231202 280222 255417 243153 263618 232489 265993 272390 259728 243860 260013 244272 26008
932 261679 291348 258776 263932 277139 279406 244603 266498 258806 281271 238237 276334 263613 268469 277968
07 262130 261550 272409 239933 288085 254967 271389 243548 255171 272448 243194 239596 266813 273739 260726
3 279054 272611 241123 255359 246599 260977 248037 252046 257540
```

Sequential



```
Time taken for multiplication: 3105 microseconds
Multiplication result is:
2386323863 16641 18269 25669 26714 27319 24296 29105 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
13531 11790 12073 10831 14306 15303 11010 16126 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
28361 21768 25065 32380 31305 32963 26215 37976 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
27850 21392 23673 31343 32110 32585 25077 36257 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
22168 18687 22012 25487 26977 27355 18201 31267 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

PTThread (12 threads)



```
PS C:\Users\iftek\MatrixMultiplyOpenMP> g++ -fopenmp .\MatrixOMP.cpp -o main.exe
PS C:\Users\iftek\MatrixMultiplyOpenMP> .\main.exe
Time taken for multiplication: 36886 microseconds
Multiplication result is:
252765 226889 231479 261064 244276 252338 225063 232451 268202 219414 246453 243292 224296 213884 242769 222469 222107 218021 240372 2
75 247058 226665 250021 226480 238413 226094 248925 253624 233728 233402 222615 238391 253181 239586 219374 260496 218001 197512 23323
59421 238075 238456 240137 234158 217126 251759 249482 239241 220629 241003 232218 252688 265181 279646 230887 258020 239042 249907 23
5 227483 232779 234385 235598 241471 270145 226357 240843 235026 229628 250475 206491 238809 239111 244851 250313 259522 225866 240538
0685 247858 251941 194917 272822 229298
275460 250322 243283 242128 247255 250039 243354 235780 265310 231729 239502 261556 226029 231069 225881 216822 230832 223443 260080 2
75 219652 233124 250382 227865 264831 249651 256752 244481 217740 246340 209924 242749 252402 233931 231405 265795 238461 217664 26740
36891 260536 248914 256306 255947 231253 273797 276969 254645 230963 254950 228800 251598 276990 269104 244884 263294 230624 241474 22
7 216582 259980 240245 229867 233449 284195 222751 250764 228910 244641 266197 194447 228080 261900 237917 256522 251383 229464 260740
9767 253174 253260 216034 260415 230820
266233 253817 229902 253156 239882 256438 239178 238414 259371 245901 249511 264733 235817 220465 249626 218304 242695 231096 264353 2
```

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.