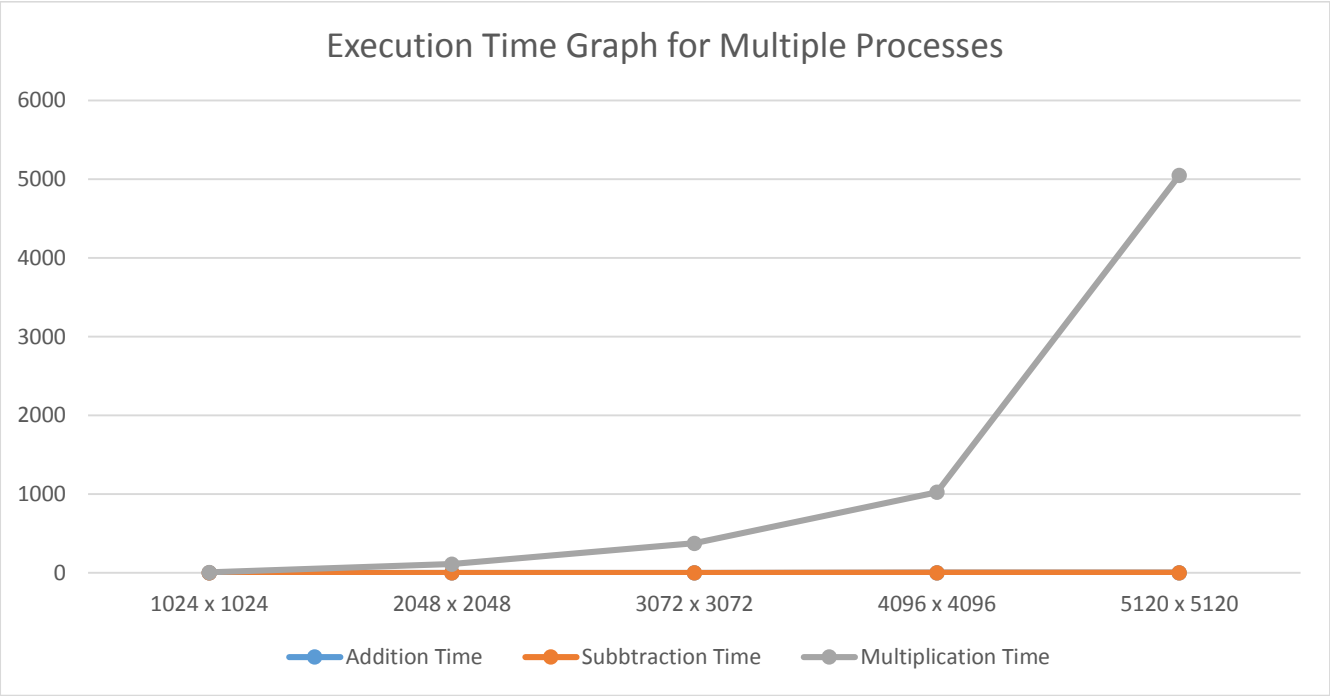
Task 1 Matrix Arithmetic:

Program 1:

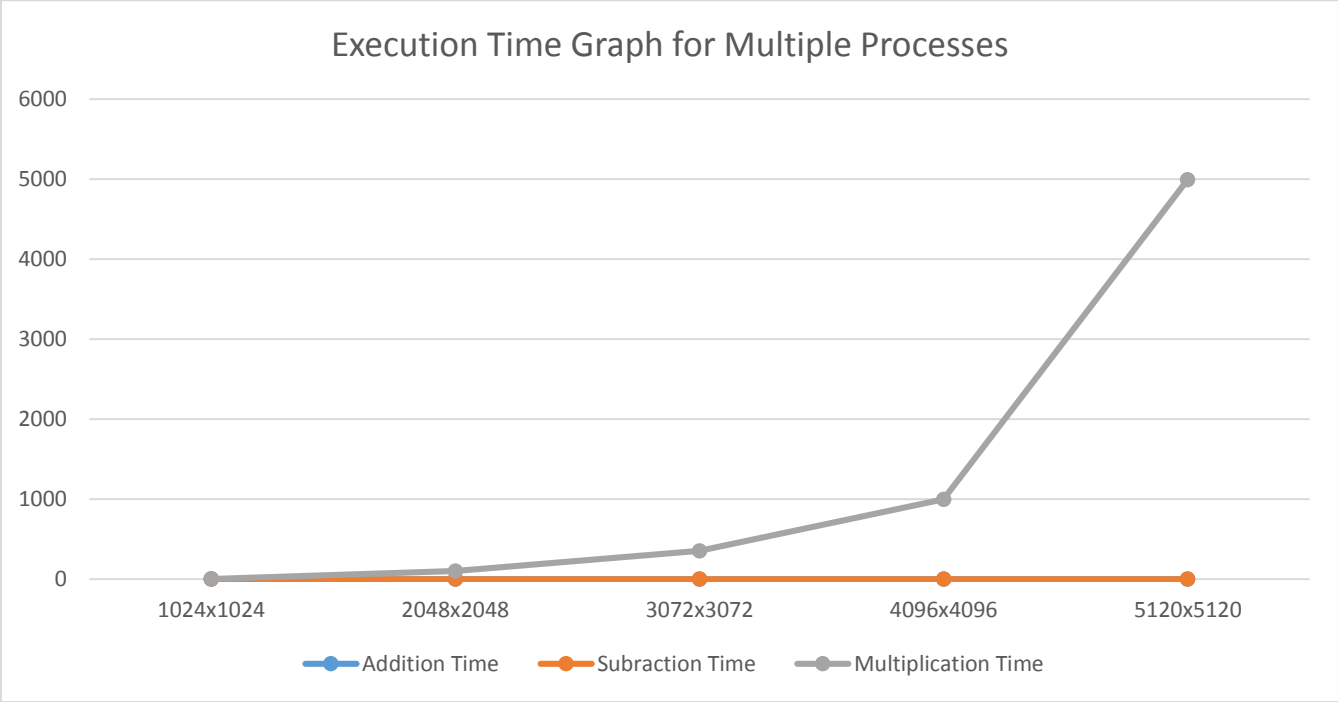
In first program a matrix of 1024x1204 is Added, Subtracted and multiplied by another matrix of same size, five times in a loop and each time the size of array is incremented. Below is a chart comparing Array Size vs. Execution Time:




```
compro@ubuntu: ~/assignment-1-fall-2019-102539-62722_62462/Code
compro@ubuntu:~/assignment-1-fall-2019-102539-62722_62462/Code$ ./SingleProcess
1024
Mat Size: 1024x1024 Execution Time(Multiplication)  : 4.898832
Mat Size: 1024x1024 Execution Time(Addition)        : 0.037863
Mat Size: 1024x1024 Execution Time(Subraction)      : 0.036605
Mat Size: 2048x2048 Execution Time(Multiplication)  : 110.893478
Mat Size: 2048x2048 Execution Time(Addition)        : 0.131219
Mat Size: 2048x2048 Execution Time(Subraction)      : 0.131650
Mat Size: 3072x3072 Execution Time(Multiplication)  : 375.376636
Mat Size: 3072x3072 Execution Time(Addition)        : 0.302144
Mat Size: 3072x3072 Execution Time(Subraction)      : 0.299324
Mat Size: 4096x4096 Execution Time(Multiplication)  : 1022.932980
Mat Size: 4096x4096 Execution Time(Addition)        : 0.544225
Mat Size: 4096x4096 Execution Time(Subraction)      : 0.566521
Mat Size: 5120x5120 Execution Time(Multiplication)  : 5050.837474
Mat Size: 5120x5120 Execution Time(Addition)        : 0.731054
Mat Size: 5120x5120 Execution Time(Subraction)      : 0.729244
compro@ubuntu:~/assignment-1-fall-2019-102539-62722_62462/Code$
```

Program 2:

In the second program a matrix of 1024x1204 is Added, Subtracted and multiplied by another matrix of same size, five times in a loop and each time the size of array is incremented. But this time each arithmetic operation has its own process. Below is a chart comparing Array Size vs. Execution Time:




```
compro@ubuntu: ~/assignment-1-fall-2019-102539-62722_62462/Code
compro@ubuntu:~/assignment-1-fall-2019-102539-62722_62462/Code$ ./MultipleProcess
1024
Mat Size: 1024x1024 Execution Time(Addition)       : 0.030516
Mat Size: 1024x1024 Execution Time(Subraction)     : 0.031093
Mat Size: 2048x2048 Execution Time(Addition)       : 0.122493
Mat Size: 2048x2048 Execution Time(Subraction)     : 0.122493s
Mat Size: 3072x3072 Execution Time(Addition)       : 0.288133
Mat Size: 3072x3072 Execution Time(Subraction)     : 0.282680s
Mat Size: 4096x4096 Execution Time(Subraction)     : 0.556438
Mat Size: 4096x4096 Execution Time(Addition)       : 0.591824
Mat Size: 5120x5120 Execution Time(Subraction)     : 0.724122
Mat Size: 5120x5120 Execution Time(Addition)       : 0.796388
Mat Size: 1024x1024 Execution Time(Multiplication) : 4.898832
Mat Size: 2048x2048 Execution Time(Multiplication) : 101.893478
Mat Size: 3072x3072 Execution Time(Multiplication) : 354.376636
Mat Size: 4096x4096 Execution Time(Multiplication) : 998.932980
Mat Size: 5120x5120 Execution Time(Multiplication) : 4991.837474
compro@ubuntu:~/assignment-1-fall-2019-102539-62722_62462/Code$
```

Comparison between Program 1 and Program 2:

In first program the whole process is done together as a single process and in the second program each arithmetic operation is done in a separate process. As a result the in second program all of the multiplication iterations are noticeably faster. This is because in a single process all three arithmetic operations are occupying the memory till the end. While in multiple process each operation has its own process and when the process is finished it is no longer occupying any memory so that is why multiplication is easy to do because most of the multiplication is happening after addition and subtraction(because multiplication takes more time due to complexity of n^3).