CS F372: Operating Systems Assignment 2

Details Of Group Members

Name	ID NO
Raj Srivastava	2019B1A71426H
Abhishek Jalan	2019B1A71547H
Ashish Avs	2019B2A71435H
Shashwat Singh	2019B2A81481H
Ayush Tiwari	2019AAPS0242H
Vaibhav Mishra	2019A3PS1350H

PROBLEM STATEMENT B:

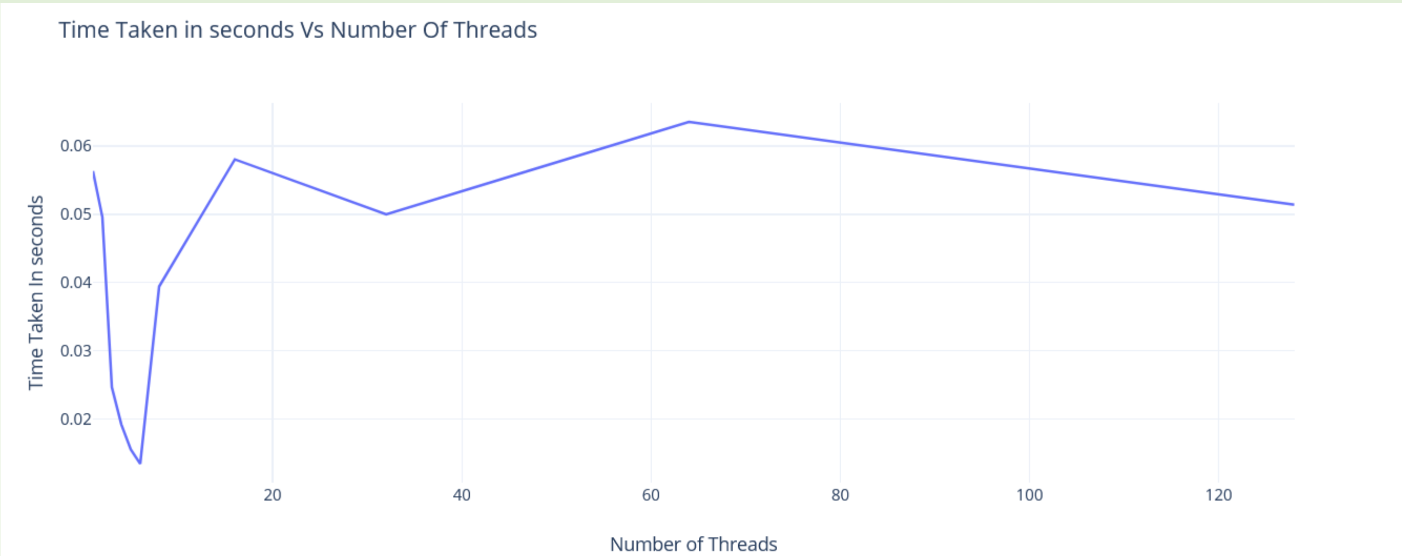
Number of Threads vs time taken by P1 with those given number of threads

About P1: Pre-processing Input matrices given in the format of matrix and matrix transpose respectively for finding the places of new line characters present in the matrix text files then using multiple threads for spawning and reading row and column data

The below result is for reading two 100x100 matrices

No of threads	P1 time taken in seconds
1	0.056327
2	0.049585
3	0.024697
4	0.019263
5	0.015594
6	0.013474
8	0.039435
16	0.058038
32	0.050001
64	0.063518
128	0.051421

Figure 1 depicts the plot of Time vs Number of threads for P1



Analysis:

We can see that for increasing the threads count to six, we are getting a uniform decrease in time, while on increasing the number of lines beyond 6, the time is growing in an almost constant manner this is since First off, adding more threads than we have cores won't improve speed since our CPU has a hardware restriction on the number of threads it can run simultaneously (for example, 4 for a quad-core).

Adding more threads makes things worse because there is more work to be done in scheduling and synchronisation while the amount of work done per unit of time stays the same.

PROBLEM STATEMENT C:

Number of threads vs Time Taken by P2 to calculate matrix multiplication in seconds

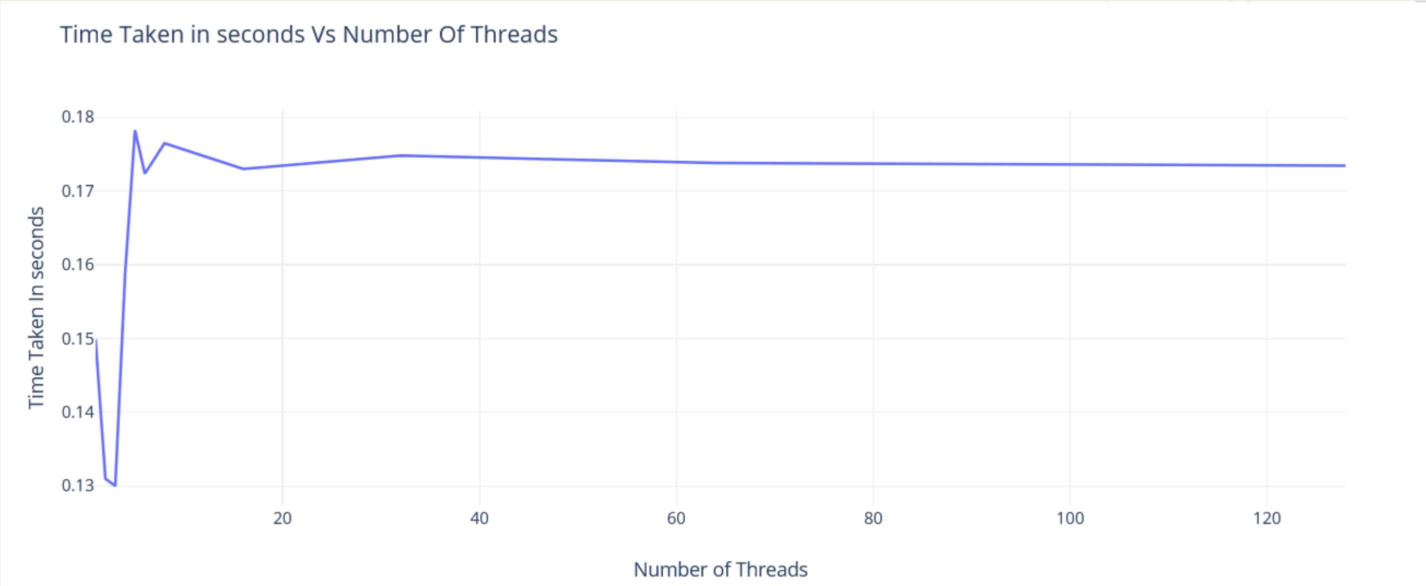
About P2:

P2 uses matrices provided by P1 and then multiplies them to produce the final output.txt file containing the product of the matrices provided using multithreading, the time taken for which against the number of threads is provided below.

No of threads	P2 time taken in seconds
1	0.149873
2	0.130979
3	0.130002
4	0.158787
5	0.178232
6	0.172332
8	0.176453
16	0.172987
32	0.174783
64	0.173786
128	0.173426

The data that has been produced is for the product of two 50x50 matrices i.e., total number of individual multiplications is 50x50x50

The figure below represents the tabular data above in terms of the line plot.



Analysis of the plot:

We can see that on increasing threads used for multiplication till three, the time taken is reduced significantly, after which it remains almost constant on the increment of lines.

Reason :

Operating over four threads results in markedly elevated switching overheads, which prolong the total amount of time. Also, adding more threads increases the CPU used for synchronisation and scheduling internally.

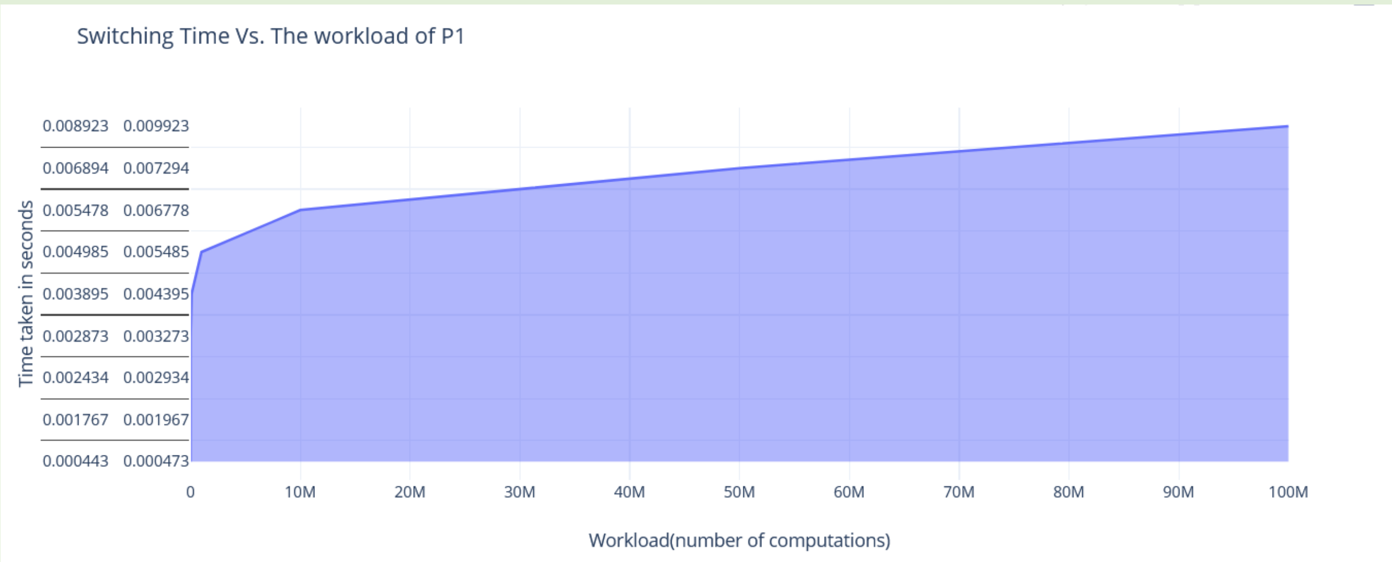
PROBLEM STATEMENT D:

ROUND ROBIN SCHEDULER USED FOR SCHEDULING P1 AND P2

Switching Time:

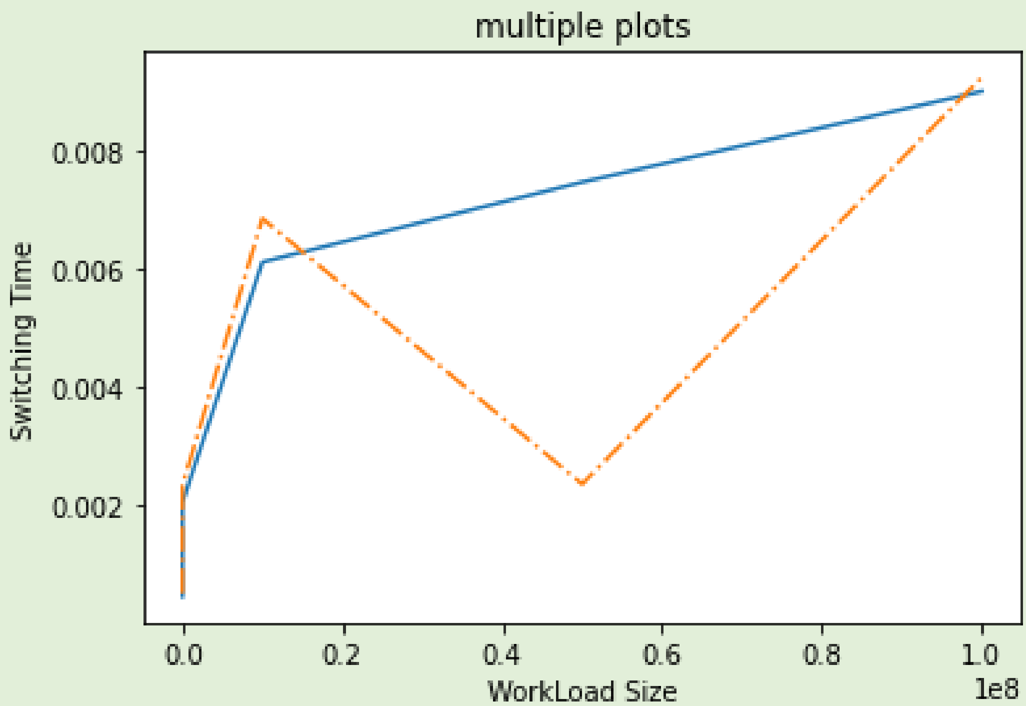
Table and Line Plot for P1:

Workload P1	Switching time P1 TQ 1ms	Switching time P1 TQ 2ms
100	0.000443	0.000473
500	0.001767	0.001967
1000	0.002434	0.002934
10000	0.002873	0.003273
100000	0.003895	0.004395
1000000	0.004985	0.005485
10000000	0.005478	0.006778
50000000	0.006894	0.007294
100000000	0.008923	0.009923



For P2:

Workload P2	Switching time P2 TQ 1ms	Switching time P2 TQ 2ms
100	0.000447	0.000499
500	0.002039	0.002333
10000000	0.006109	0.006849
50000000	0.007464	0.002349
100000000	0.009001	0.009234



Analysis of plots:

Switching overhead is more for time quanta 1ms than 2ms as number of context switches for 1ms would be more as in 2ms most of the program is executed in P1 and P2 is only left for multiplication. Also, as workload increases the number of context switches increased resulting in more switching time.

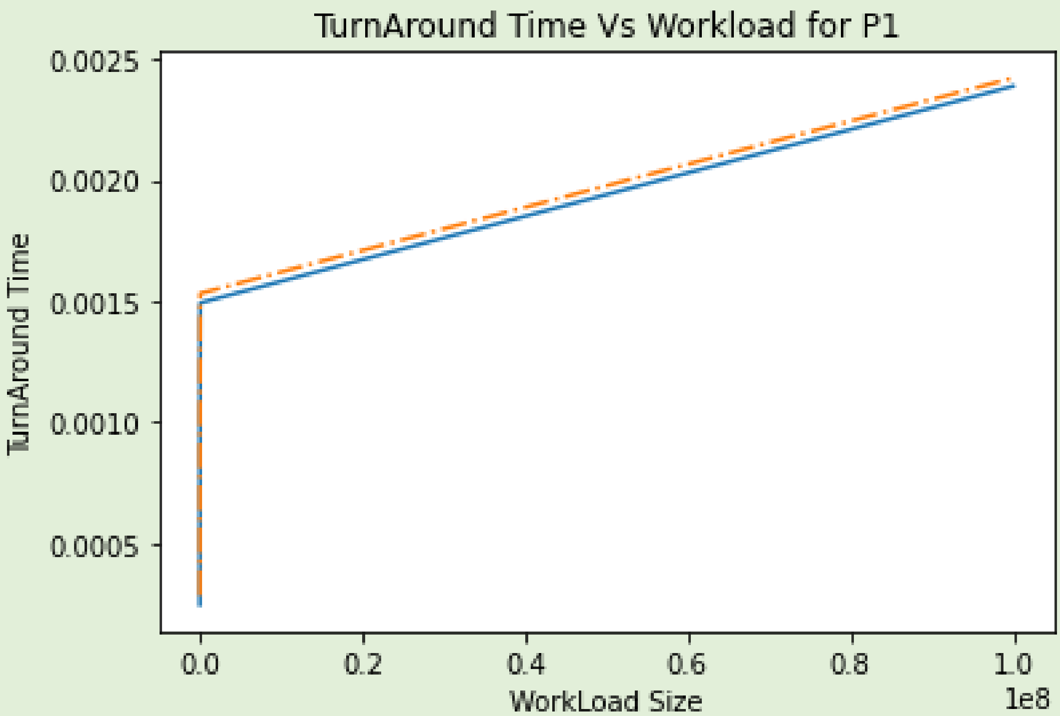
Turnaround time, Table and Plots:

For P1:

Table:

Workload For P1	Turnaround Time for TQ 1ms	Turnaround Time for TQ 2ms
100	0.000245	0.000284
500	0.000476	0.000532
10000	0.000891	0.000941
50000	0.001493	0.001532
100000000	0.002389	0.002423

Figure:

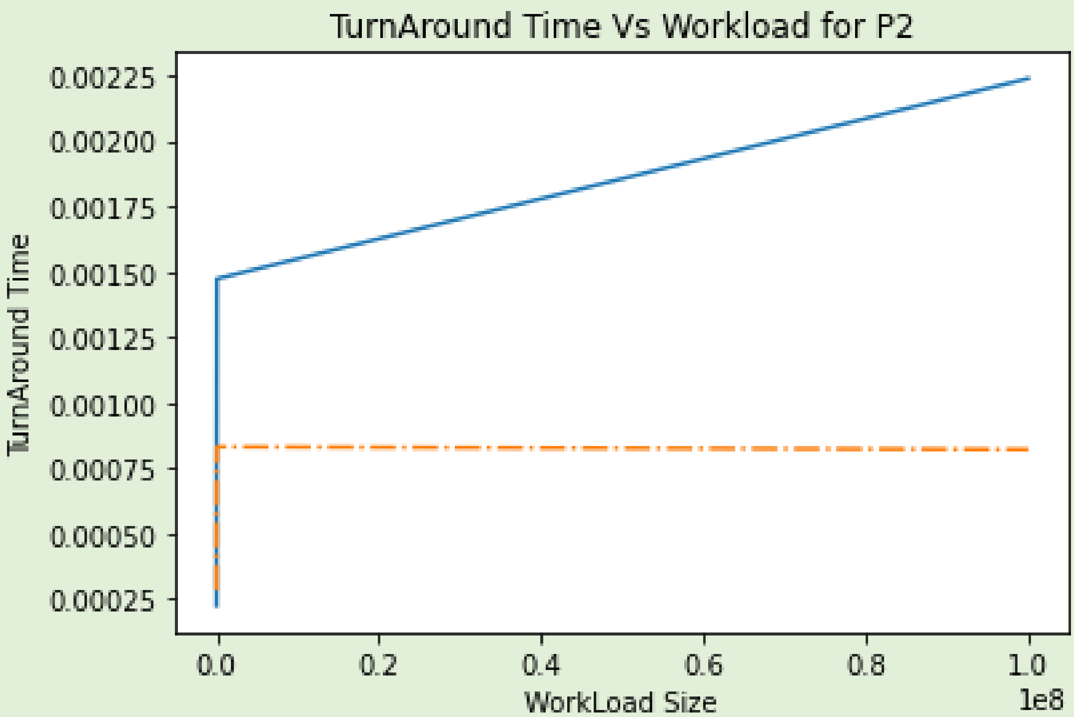


For P2:

Table:

Workload For P1	Turnaround Time for TQ 1ms	Turnaround Time for TQ 2ms
100	0.000225	0.000284
500	0.000446	0.000532
10000	0.000791	0.000621
50000	0.001473	0.000832
100000000	0.002238	0.000822

Figure:



Analysis of Turnaround time vs Workload Graphs:

It can be observed that turnaround time is very similar in both cases and a major thing to be noticed in the P2 graphs is that for p2 when time quanta are 2 ms, the turnaround time difference is massive. This is because the time between context switching is way less than for time quanta of 1 ms.

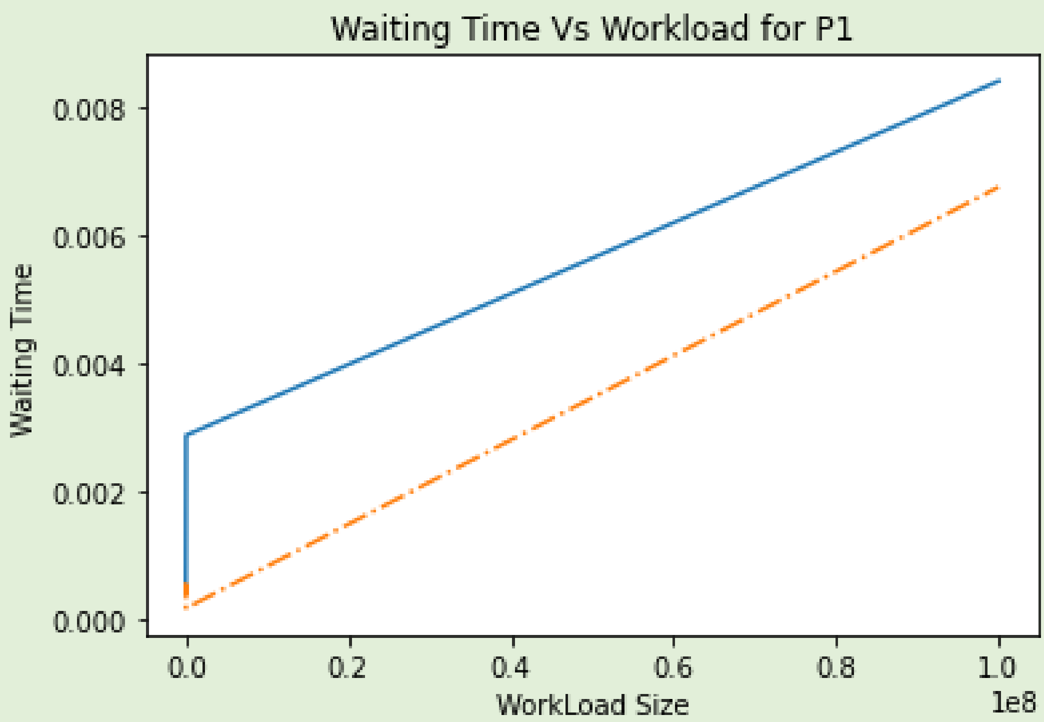
Waiting time vs Workload

For P1:

Table:

Workload for P1	Waiting time for TQ 1ms	Waiting time for TQ 2ms
100	0.000444	0.000399
500	0.000455	0.000402
10000	0.000676	0.000567
50000	0.002883	0.0001893
100000000	0.008399	0.006747

Figure:

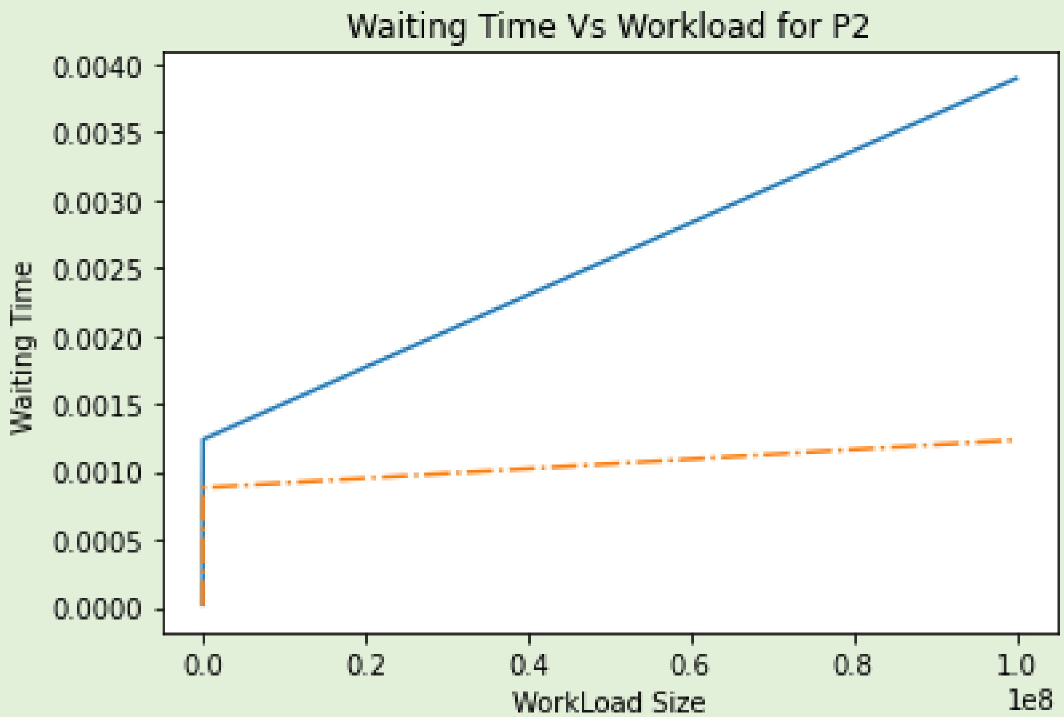


For P2:

Table:

Workload for P2	Waiting time for TQ 1ms	Waiting time for TQ 2ms
100	0.000022	0.000012
500	0.000088	0.000044
10000	0.000923	0.000642
50000	0.001239	0.000882
100000000	0.003897	0.001234

Figure:



Analysis of Waiting Time Plots:

We can make similar observations as turnaround time here that waiting time is more for TQ of 1ms than 2ms due to frequent switching of the programs in the uniprocessor round-robin scheduler.

Conclusion:

Program S is used to schedule between P1 and P2 using round robin algorithm with time quanta 1ms and 2ms. P1 is used for reading of the matrices and P2 is used for Multiplication and output of the resultant matrix in file.