Faculty of Engineering, Alexandria University

Computer and Systems Engineering Department

CS 333 | Operating Systems | Fall 2017

Name: Aya Aly Saad Aly Abouzeid

SID: 2

A) Code organization:

The code is divided into six .c files,

1. main.c:

-Determine which files the program will work with.

-call the multiplying functions.

-Compute execution time.

-Call the printing functions.

2. read_from_files.c:

-Read the given matrices files.

-Fill the matrices with the given input.

3. matMult.c:

-Compute the matrices multiplication using no threads.

4. Method1.c:

-Compute the multiplication using threads for each row in the output matrix.

5. Method2.c:

-Compute the multiplication using threads for each element in the output matrix.

6. output_to_file.c:

-Prints the output of multiplication to the given output files.

B) Main functions:

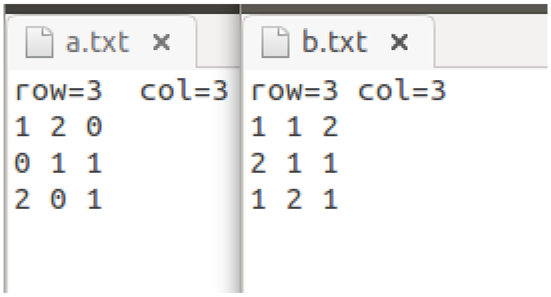
- **readMatrices()**
    - reads marices files.
- **multiplyMatrices()**
    - calls the three multipling functions.
- **getDimentions()**
    - reads the matrices dimentions in the first line.
- **MultiplyNoThreads()**
    - multiply the matrices using no threads.
- **useMethod1()**
    - creates threads for computing each row.
- **Multiply1()**
    - multiplies the 2 matrices using a thread for each row.
- **useMethod2()**
    - creates threads for computing each element.
- **Multiply2()**

- multiplies the 2 matrices using a thread for each element.

- **writeOutput1() , writeOutput2()**
  - prints the output to external files.

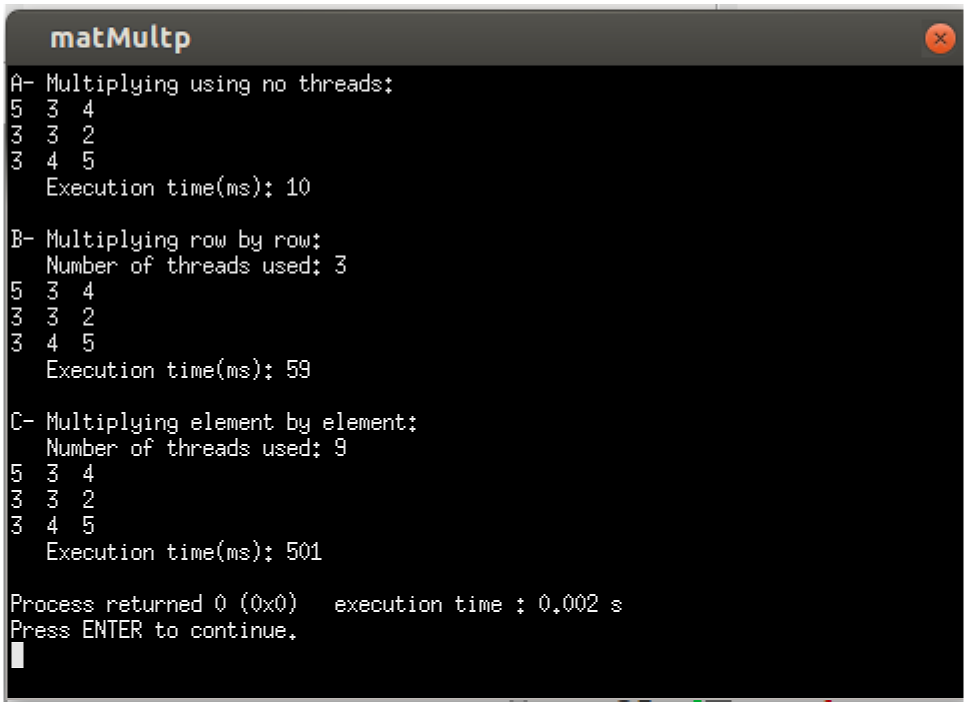
C) Sample runs:

1-



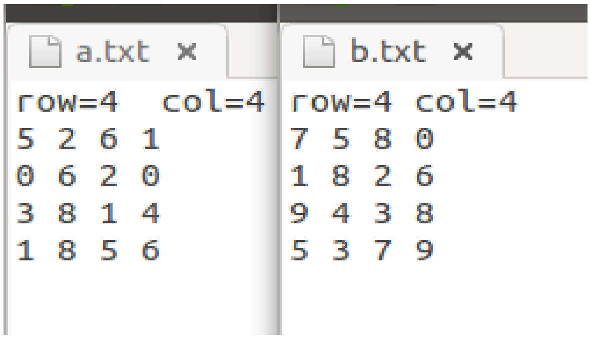
```
row=3   col=3
1 2 0
0 1 1
2 0 1
```

```
row=3 col=3
1 1 2
2 1 1
1 2 1
```



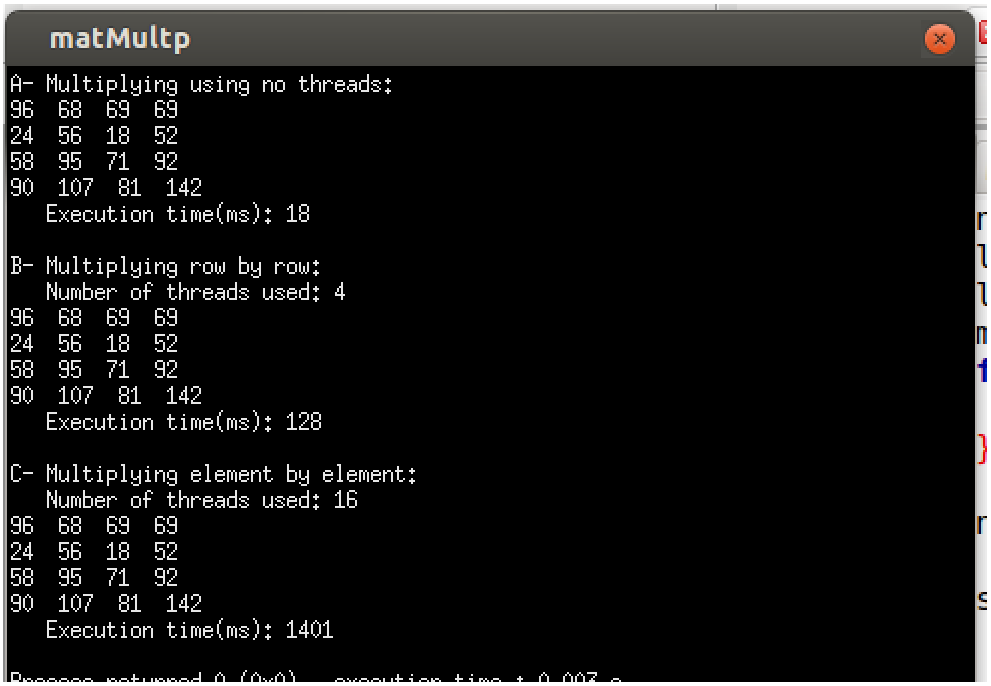
```
A- Multiplying using no threads:
5  3  4
3  3  2
3  4  5
   Execution time(ms): 10

B- Multiplying row by row:
   Number of threads used: 3
5  3  4
3  3  2
3  4  5
   Execution time(ms): 59

C- Multiplying element by element:
   Number of threads used: 9
5  3  4
3  3  2
3  4  5
   Execution time(ms): 501

Process returned 0 (0x0)   execution time : 0.002 s
Press ENTER to continue.
```

2-



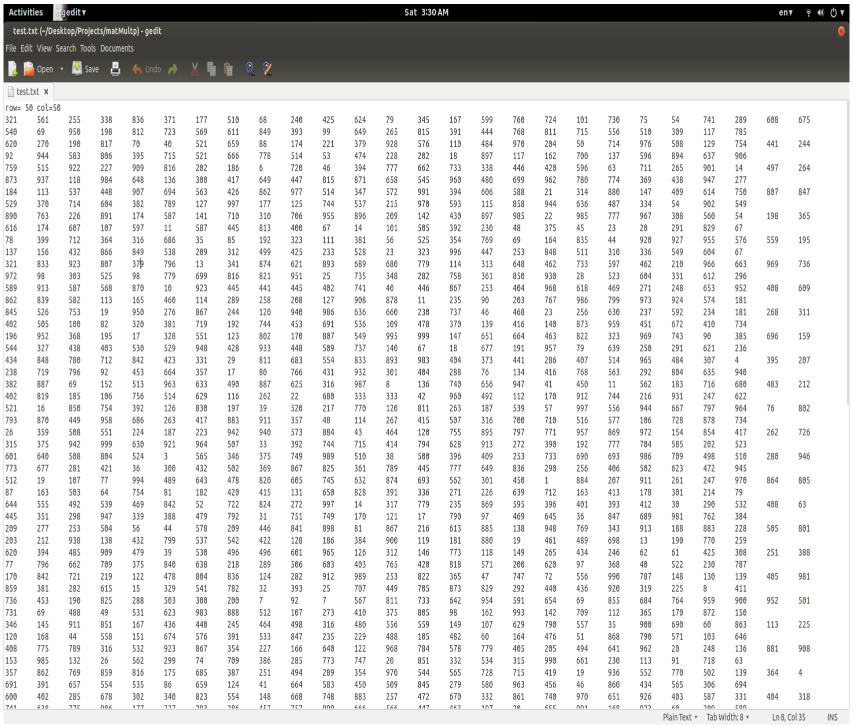
```
row=4   col=4
5 2 6 1
0 6 2 0
3 8 1 4
1 8 5 6
```

```
row=4 col=4
7 5 8 0
1 8 2 6
9 4 3 8
5 3 7 9
```

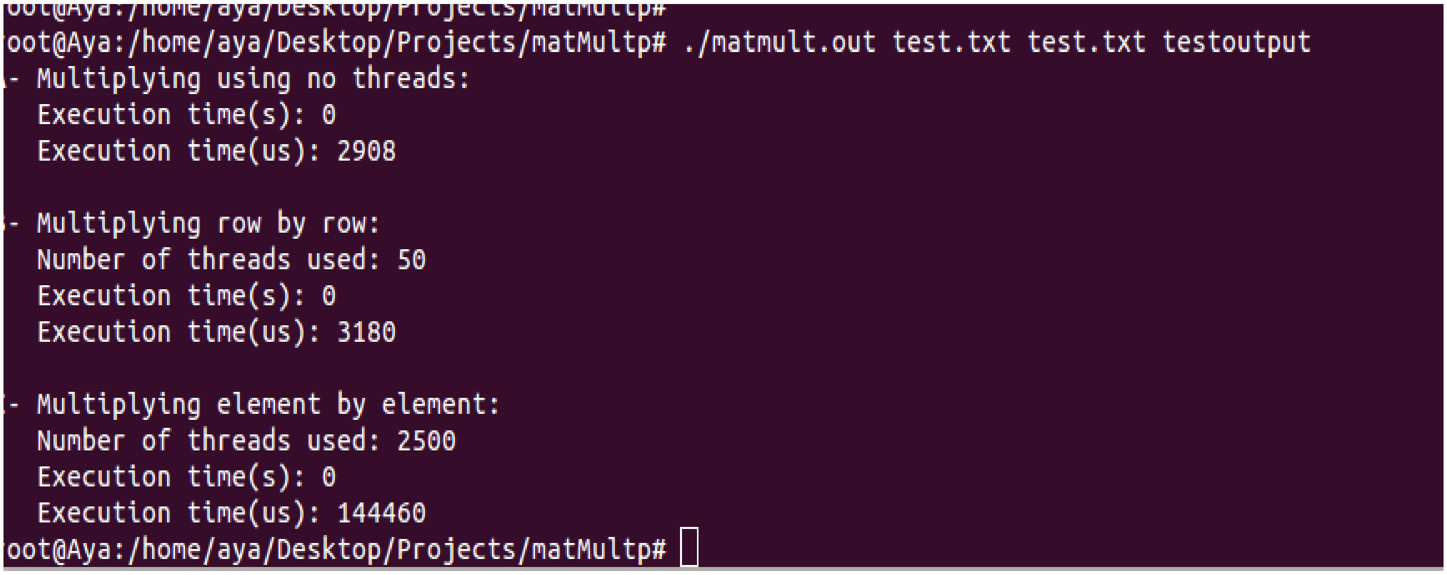


```
A- Multiplying using no threads:
96  68  69  69
24  56  18  52
58  95  71  92
90  107  81  142
   Execution time(ms): 18

B- Multiplying row by row:
   Number of threads used: 4
96  68  69  69
24  56  18  52
58  95  71  92
90  107  81  142
   Execution time(ms): 128

C- Multiplying element by element:
   Number of threads used: 16
96  68  69  69
24  56  18  52
58  95  71  92
90  107  81  142
   Execution time(ms): 1401

Process returned 0 (0x0)   execution time : 0.003 s
```

3- 50 x 50 and 50 x 50

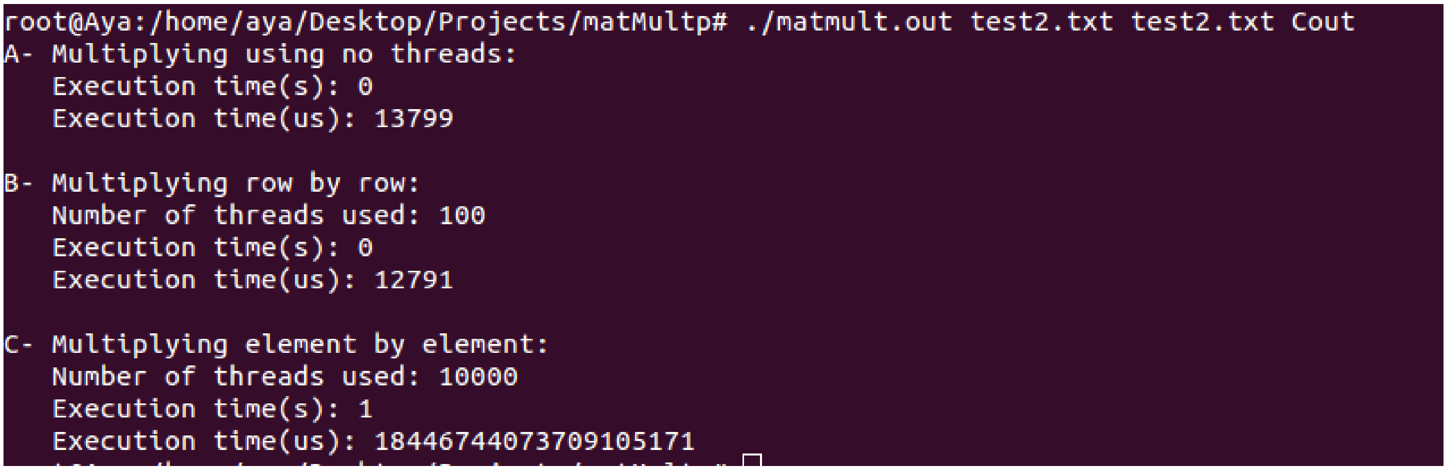


```
row= 50 col=50
321     561     255     338     836     371     177     510     68      240     425     624     79      345     167     599     760     724     101     730     75      54      741     289     608     675
540     69      950     198     812     723     569     611     849     393     99      649     265     815     391     444     768     811     715     556     510     309     117     785
620     270     190     817     70      40      521     659     88      174     221     379     928     576     110     484     970     204     50      714     976     508     129     754     441     244
92      944     583     806     395     715     521     666     778     514     53      474     228     202     18      897     117     162     700     137     596     894     637     906
759     515     922     227     909     816     202     186     6       720     46      394     777     662     733     338     446     420     596     63      711     265     901     14      497     264
873     937     118     984     648     136     300     417     649     447     815     871     658     545     960     480     699     962     780     774     369     438     947     277
184     113     537     448     907     694     563     426     862     977     514     347     572     991     394     606     588     21      314     880     147     409     614     750     807     847
529     370     714     604     382     789     127     997     177     125     744     537     215     970     593     115     858     944     636     487     334     54      902     549
890     763     226     891     174     587     141     710     310     706     955     896     209     142     430     897     985     22      985     777     967     308     560     54      198     365
616     174     607     107     597     11      587     445     813     400     67      14      101     505     392     230     48      375     45      23      20      291     829     67
78      399     712     364     316     686     35      85      192     323     111     381     56      525     354     769     69      164     835     44      920     927     955     576     559     195
137     156     432     866     849     538     209     312     499     425     233     528     23      323     996     447     253     848     511     310     336     549     604     67
321     833     923     807     379     796     13      341     874     621     893     689     680     779     114     313     648     462     733     597     462     210     966     663     969     736
972     98      303     525     98      779     699     816     821     951     25      735     348     282     758     361     850     930     28      523     604     331     612     296
589     913     587     568     870     10      923     445     441     445     402     741     40      446     867     253     404     968     618     469     271     248     653     952     408     609
862     839     582     113     165     460     114     289     258     208     127     908     878     11      235     90      203     767     986     799     973     924     574     181
845     526     753     19      950     276     867     244     120     940     986     636     660     230     737     46      468     23      256     630     237     592     234     181     268     311
402     505     100     82      320     381     719     192     744     453     691     536     109     478     370     139     416     140     873     959     451     672     410     734
196     952     368     195     17      328     551     123     802     170     807     549     995     999     147     651     664     463     822     323     969     743     90      385     696     159
544     327     438     403     530     529     948     428     933     448     509     737     140     67      18      677     191     957     79      639     250     291     621     236
434     848     780     712     842     423     331     29      811     683     554     833     893     983     404     373     441     286     407     514     965     484     307     4       395     207
238     719     796     92      453     664     357     17      80      766     431     932     301     404     288     76      134     416     768     563     292     804     635     940
382     887     69      152     513     963     633     490     887     625     316     987     8       136     740     656     947     41      450     11      562     183     716     680     483     212
402     819     185     106     756     514     629     116     262     22      680     333     333     42      960     492     112     170     912     744     216     931     247     622
521     16      850     754     392     126     830     197     39      520     217     770     120     811     263     187     539     57      997     556     944     667     797     964     76      802
793     870     449     958     686     263     417     883     911     357     48      114     267     415     507     316     700     710     516     577     106     728     878     734
26      359     508     551     224     187     223     942     940     573     884     43      464     120     755     895     797     771     957     869     972     154     854     417     262     726
315     375     942     999     630     921     964     507     33      392     744     715     414     794     628     913     272     390     192     777     704     585     202     523
601     640     508     804     524     3       565     346     375     749     989     510     38      500     396     409     253     733     690     693     986     709     498     510     280     946
773     677     281     421     36      300     432     502     369     867     825     361     789     445     777     649     836     290     256     406     502     623     472     945
512     19      107     77      994     489     643     478     820     605     745     632     874     693     562     301     450     1       884     207     911     261     247     970     864     805
87      163     503     64      754     81      182     420     415     131     650     828     391     336     271     226     639     712     163     413     178     301     214     79
644     555     492     539     469     842     52      722     824     272     997     14      317     779     235     869     595     396     401     393     412     30      290     532     408     63
445     351     298     947     339     388     479     792     31      751     749     170     121     17      790     97      469     645     36      847     689     981     762     384
209     277     253     504     56      44      578     209     446     841     898     81      867     216     613     885     138     948     769     343     913     188     883     228     505     801
203     212     938     138     432     799     537     542     422     128     186     384     900     119     181     880     19      461     489     698     13      190     770     259
620     394     485     909     479     39      530     496     496     601     965     126     312     146     773     118     149     265     434     246     62      61      425     308     251     388
77      796     662     709     375     840     638     218     289     506     603     403     765     420     818     571     200     620     97      368     40      522     230     787
170     842     721     219     122     478     804     836     124     282     912     989     253     822     365     47      747     72      556     990     787     148     130     139     405     981
859     381     282     615     15      329     541     782     32      393     25      707     449     705     873     829     292     440     436     920     319     225     8       411
736     453     190     825     288     503     300     200     7       92      7       567     811     733     642     954     591     654     69      855     684     764     959     900     952     501
731     69      488     49      531     623     983     888     512     107     273     410     375     805     98      162     993     142     709     112     365     170     872     150
346     145     911     851     167     436     440     245     464     498     316     480     556     559     149     107     629     790     557     35      900     690     60      863     113     225
120     168     44      558     151     674     576     391     533     847     235     229     488     105     482     60      164     476     51      868     790     571     103     646
408     775     789     316     532     923     867     354     227     166     640     122     968     784     578     779     405     205     494     641     962     20      248     136     881     908
153     985     132     26      562     299     74      709     386     285     773     747     20      851     332     534     315     990     661     230     113     91      718     63
357     862     769     859     816     175     685     387     251     494     289     354     970     544     565     728     715     419     19      936     552     770     502     139     364     4
691     391     657     554     535     86      659     124     41      664     583     450     509     845     279     580     963     456     46      860     434     565     306     694
600     402     285     678     302     340     823     554     148     668     748     883     257     472     670     332     861     740     970     651     926     403     587     331     404     318
```



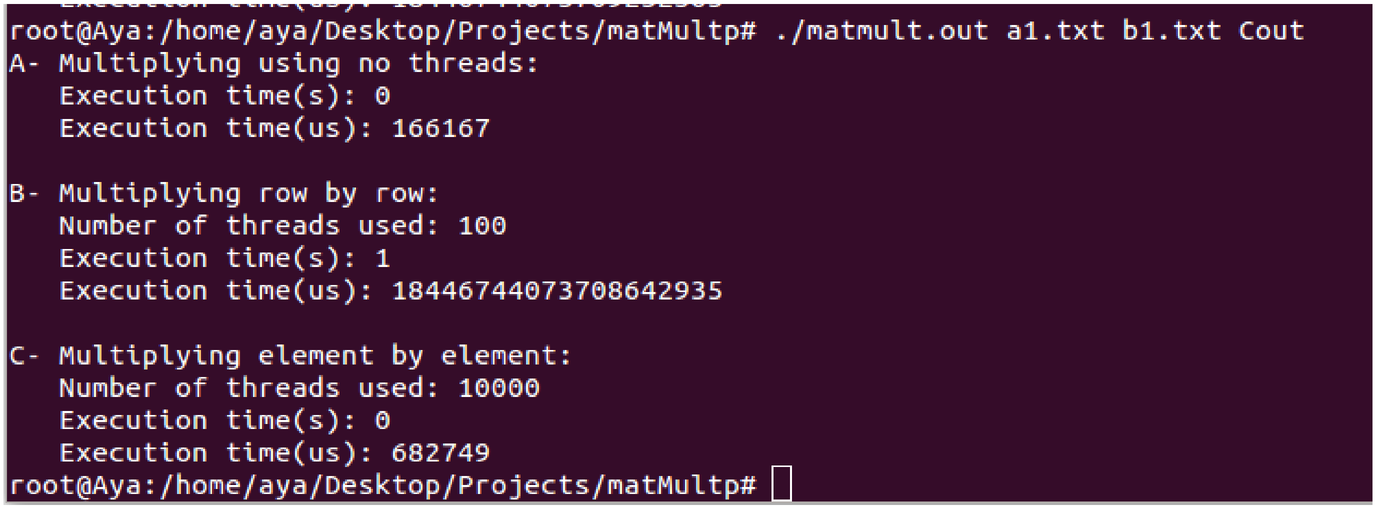
```
oot@Aya:/home/aya/Desktop/Projects/matMultp#
oot@Aya:/home/aya/Desktop/Projects/matMultp# ./matmult.out test.txt test.txt testoutput
- Multiplying using no threads:
  Execution time(s): 0
  Execution time(us): 2908

- Multiplying row by row:
  Number of threads used: 50
  Execution time(s): 0
  Execution time(us): 3180

- Multiplying element by element:
  Number of threads used: 2500
  Execution time(s): 0
  Execution time(us): 144460
oot@Aya:/home/aya/Desktop/Projects/matMultp#
```

4- 100 x 100 and 100 x 100



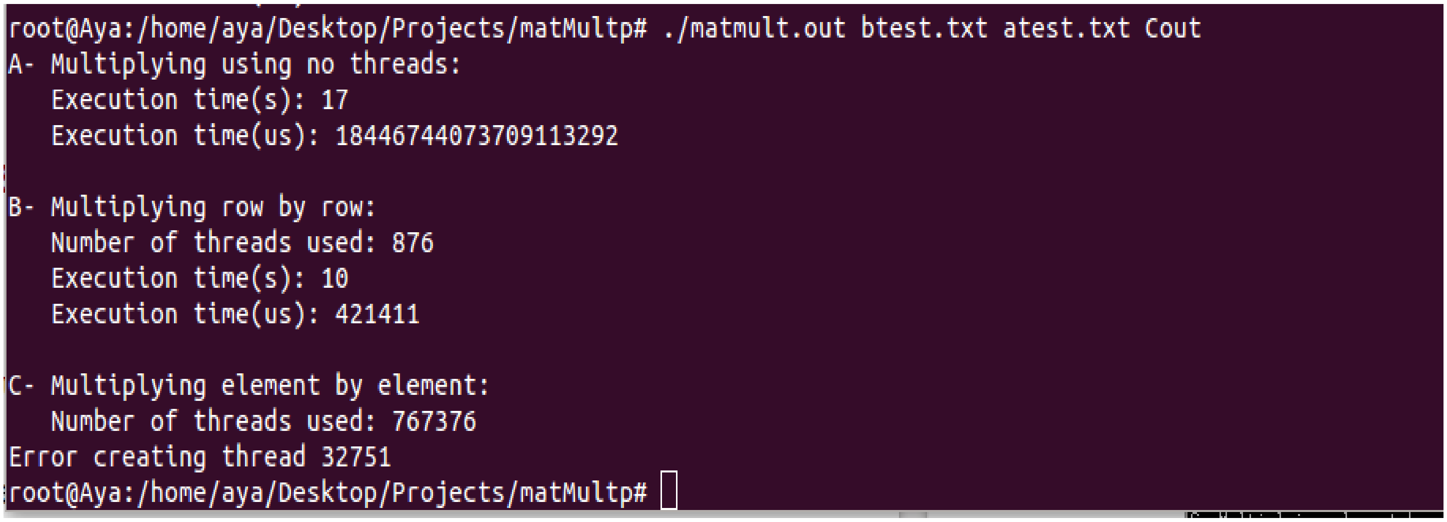
```
root@Aya:/home/aya/Desktop/Projects/matMultp# ./matmult.out test2.txt test2.txt Cout
A- Multiplying using no threads:
   Execution time(s): 0
   Execution time(us): 13799

B- Multiplying row by row:
   Number of threads used: 100
   Execution time(s): 0
   Execution time(us): 12791

C- Multiplying element by element:
   Number of threads used: 10000
   Execution time(s): 1
   Execution time(us): 18446744073709105171
```

5- 100 x 1000 and 1000 x 100



```
root@Aya:/home/aya/Desktop/Projects/matMultp# ./matmult.out a1.txt b1.txt Cout
A- Multiplying using no threads:
   Execution time(s): 0
   Execution time(us): 166167

B- Multiplying row by row:
   Number of threads used: 100
   Execution time(s): 1
   Execution time(us): 18446744073708642935

C- Multiplying element by element:
   Number of threads used: 10000
   Execution time(s): 0
   Execution time(us): 682749
root@Aya:/home/aya/Desktop/Projects/matMultp#
```

6- 876 x 876 and 876 x 876



```
root@Aya:/home/aya/Desktop/Projects/matMultp# ./matmult.out btest.txt atest.txt Cout
A- Multiplying using no threads:
   Execution time(s): 17
   Execution time(us): 18446744073709113292

B- Multiplying row by row:
   Number of threads used: 876
   Execution time(s): 10
   Execution time(us): 421411

C- Multiplying element by element:
   Number of threads used: 767376
Error creating thread 32751
root@Aya:/home/aya/Desktop/Projects/matMultp#
```

D) Compiling and running :

1. Change the directory to the project's directory.

2. Open a terminal.

3. Type “make” command.

4. Execute by typing “**./matmult.out**” either by specifying input and output files such as “**./matmult.out x.txt y.txt z.out**” or just “**./matmult.out**” and the the program would consider the default values which are a.txt , b.txt for input files and c1.out , c2.out for output files.

E) comparing the two methods of matrix multiplication:

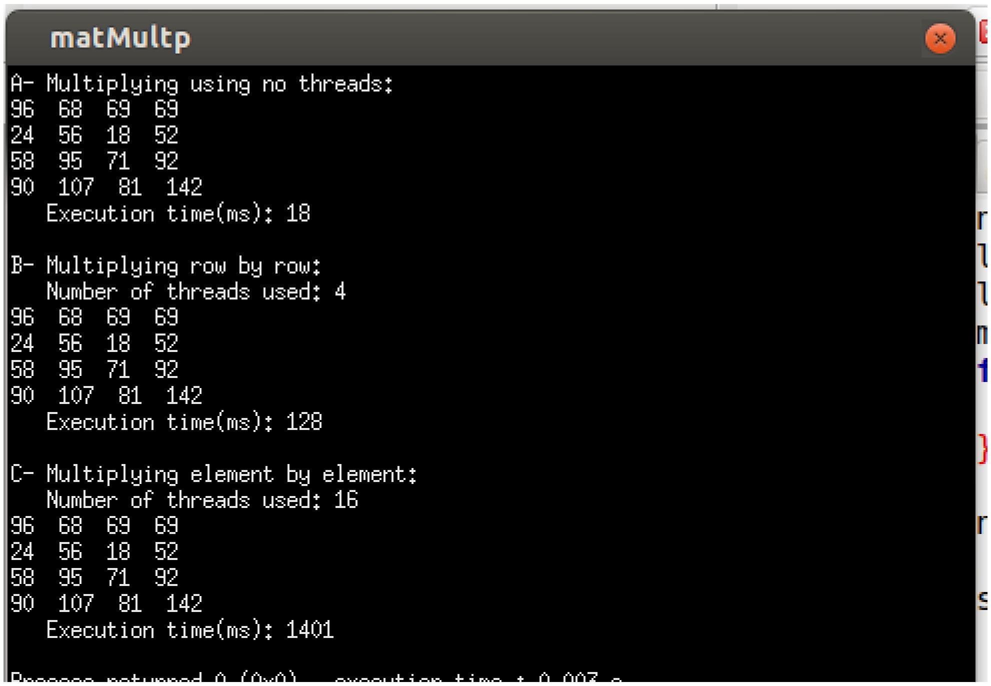
- After several runs for the program with significantly small matrix size, it appears that the execution time is:

  T(No threads) < T(Thread for each row) < T(Thread for each element).

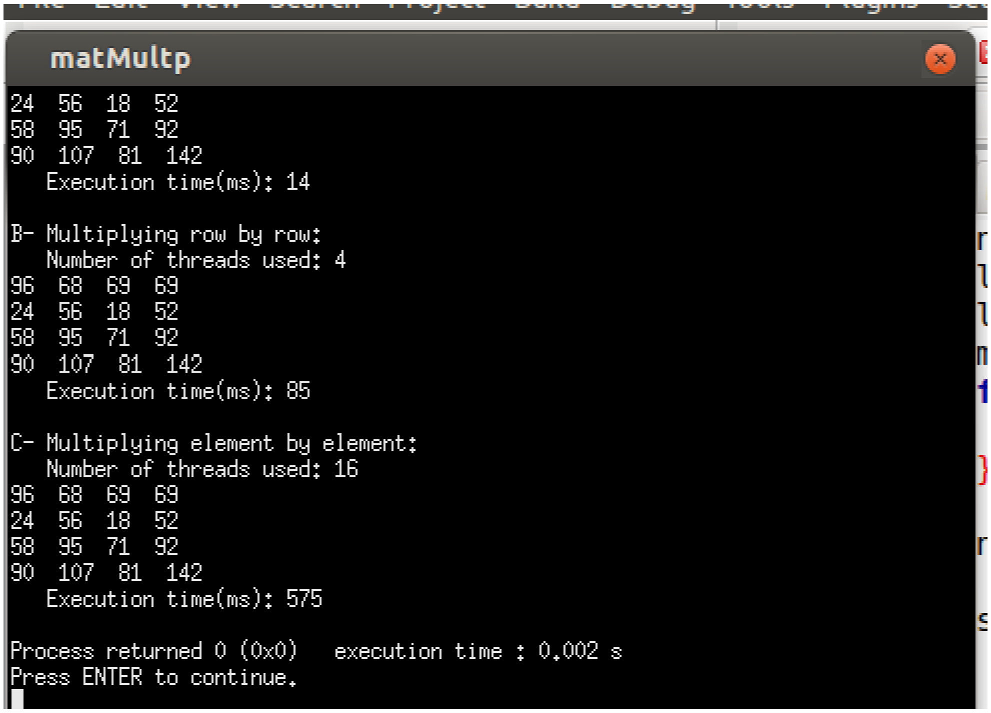
And number of threads:

N(Thread for each row) < N(Thread for each element).

Here are several runs for the same input showing the execution time:



```
A- Multiplying using no threads:
96  68  69  69
24  56  18  52
58  95  71  92
90  107  81  142
   Execution time(ms): 18

B- Multiplying row by row:
   Number of threads used: 4
96  68  69  69
24  56  18  52
58  95  71  92
90  107  81  142
   Execution time(ms): 128

C- Multiplying element by element:
   Number of threads used: 16
96  68  69  69
24  56  18  52
58  95  71  92
90  107  81  142
   Execution time(ms): 1401

Process returned 0 (0x0)   execution time : 0.003 s
```



```
24  56  18  52
58  95  71  92
90  107  81  142
   Execution time(ms): 14

B- Multiplying row by row:
   Number of threads used: 4
96  68  69  69
24  56  18  52
58  95  71  92
90  107  81  142
   Execution time(ms): 85

C- Multiplying element by element:
   Number of threads used: 16
96  68  69  69
24  56  18  52
58  95  71  92
90  107  81  142
   Execution time(ms): 575

Process returned 0 (0x0)   execution time : 0.002 s
Press ENTER to continue.
```

- Where as when using larger size matrices such as 100 x 100, it appears that:

  T(Thread for each row) < T(No Threads) < T(Thread for each element).

- While going for a larger sized matrices (100 x 1000) and as the number of rows increase, it led to a completely different result:

  T(Thread for each row) > T(Thread for each element).