

Name: Muhammed Essam Khamis

ID: 64

1. How Code is Organized:

Firstly, I managed to divide the code into modules, each module handles a part in the process of reading, writing and making multiplications.

a- Main:

This is the main module that responsible for calling the other modules by order to get the wanted result.

b- FileHandler:

Used to connect the program with the outside environment, it's responsible for reading the input files and writing the result to the output file.

c- Executer:

It's responsible for calculating the matrix multiplication.

d- TimeCalculator:

It's used for calculating the time for the first and the second method.

e- ThreadMaker:

Responsible for generating the threads for each multiplication according to the used method, if it's thread per cell or thread by row.

f- memoryManager:

manger of the memory which is responsible for creating and freeing the memory after using it.

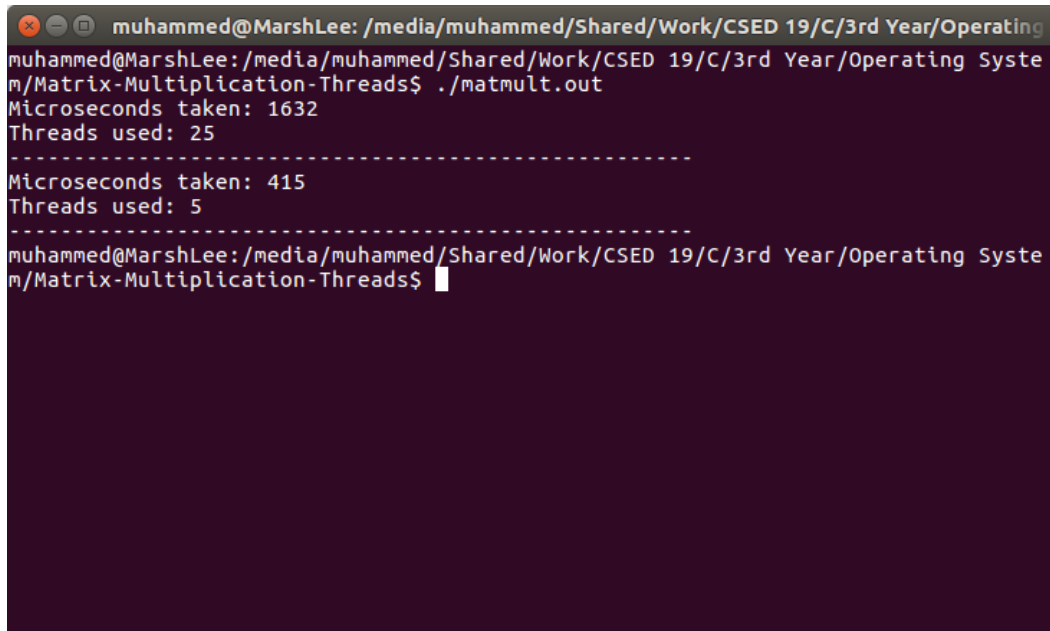
2- Main Functions:

- a. Main:
 - main function to launch the program, and run the two methods sequentially.
- b. calculateTime:
 - used to calculate the time for each method.
- c. freeThread_info:
 - it free the variables and the arrays used by this struct.
- d. computeCell:
 - It's a function in the executers module that responsible for calculating the value of a cell in the matrix given the row and the column for the multiplication.
- e. computeRow:
 - the same as the computeCell, but it computes the whole row, by calling the computeCell for every cell in the row.
- f. computeMatrix:
 - used for computing the whole matrix by calling the computeRow for each row in the matrix.

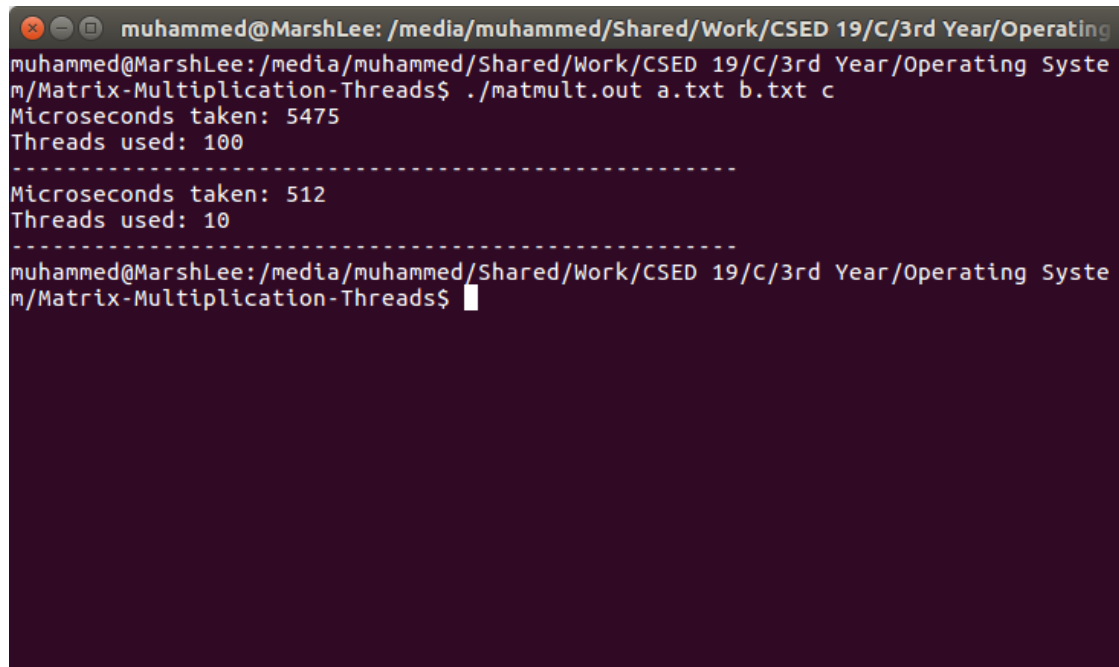
3-How to compile Code:

- a-You can compile the code by, firstly running the Makefile by writing make in the Linux shell, to compile the code and create the object file.
- b- you can run the program with no argument, but this will cause that the program must find 2 files a.txt and b.txt to read the input.
- c- You can run the program with arguments, the first argument will be considered and the first input, the second one as the second input, and the third will be the name of the output file.

4-Screen Shots:



```
muhammed@MarshLee: /media/muhammed/Shared/Work/CSED 19/C/3rd Year/Operating
muhammed@MarshLee:/media/muhammed/Shared/Work/CSED 19/C/3rd Year/Operating Syste
m/Matrix-Multiplication-Threads$ ./matmult.out
Microseconds taken: 1632
Threads used: 25
-----
Microseconds taken: 415
Threads used: 5
-----
muhammed@MarshLee:/media/muhammed/Shared/Work/CSED 19/C/3rd Year/Operating Syste
m/Matrix-Multiplication-Threads$
```



```
muhammed@MarshLee: /media/muhammed/Shared/Work/CSED 19/C/3rd Year/Operating
muhammed@MarshLee:/media/muhammed/Shared/Work/CSED 19/C/3rd Year/Operating Syste
m/Matrix-Multiplication-Threads$ ./matmult.out a.txt b.txt c
Microseconds taken: 5475
Threads used: 100
-----
Microseconds taken: 512
Threads used: 10
-----
muhammed@MarshLee:/media/muhammed/Shared/Work/CSED 19/C/3rd Year/Operating Syste
m/Matrix-Multiplication-Threads$
```

```
muhammed@MarshLee: /media/muhammed/Shared/Work/CSED 19/C/3rd Year/Operating
muhammed@MarshLee:/media/muhammed/Shared/Work/CSED 19/C/3rd Year/Operating Syste
m/Matrix-Multiplication-Threads$ ./matmult.out
Microseconds taken: 62918
Threads used: 2500
-----
Microseconds taken: 2407
Threads used: 50
-----
muhammed@MarshLee:/media/muhammed/Shared/Work/CSED 19/C/3rd Year/Operating Syste
m/Matrix-Multiplication-Threads$
```

```
muhammed@MarshLee: /media/muhammed/Shared/Work/CSED 19/C/3rd Year/Operating
muhammed@MarshLee:/media/muhammed/Shared/Work/CSED 19/C/3rd Year/Operating Syste
m/Matrix-Multiplication-Threads$ ./matmult.out
Can't generate that number of threads for this method
-----
Microseconds taken: 781380
Seconds taken 7
Threads used: 1000
-----
muhammed@MarshLee:/media/muhammed/Shared/Work/CSED 19/C/3rd Year/Operating Syste
m/Matrix-Multiplication-Threads$
```

```
muhammed@MarshLee: /media/muhammed/Shared/Work/CSED 19/C/3rd Year/Operating
muhammed@MarshLee:/media/muhammed/Shared/Work/CSED 19/C/3rd Year/Operating Syste
m/Matrix-Multiplication-Threads$ ./matmult.out
Microseconds taken: 68976
Seconds taken 0
Threads used: 2500
-----
Microseconds taken: 1474
Seconds taken 0
Threads used: 50
-----
muhammed@MarshLee:/media/muhammed/Shared/Work/CSED 19/C/3rd Year/Operating Syste
m/Matrix-Multiplication-Threads$
```

5-Comparison between 2 methods:

1- Method 1 (thread for each cell):

- i. Has more number of threads.
- ii. Use more memory.
- iii. Takes more time.

2- Method 2 (thread for each row):

- i. Has less number of threads.
- ii. Use less memory.
- iii. Takes less time.