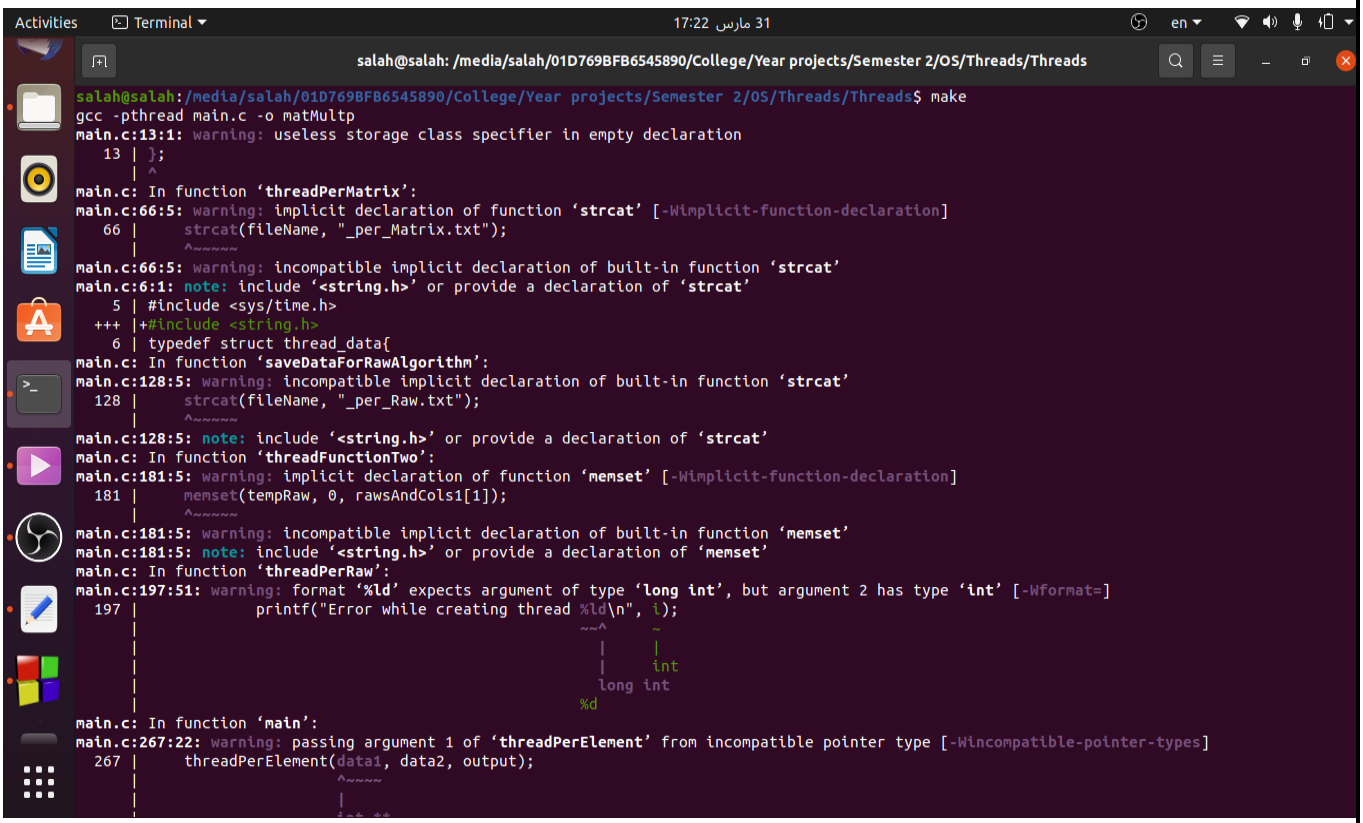
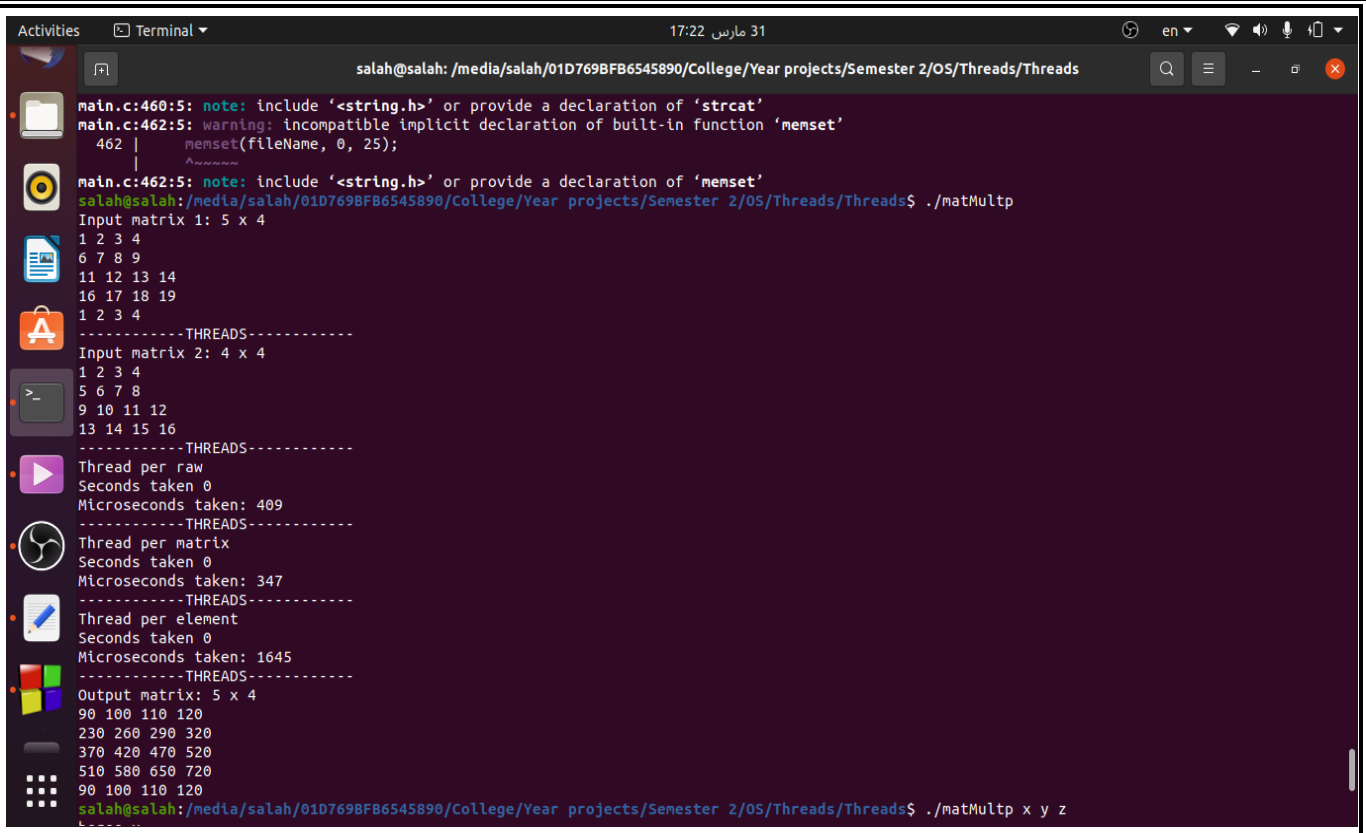
THREADS

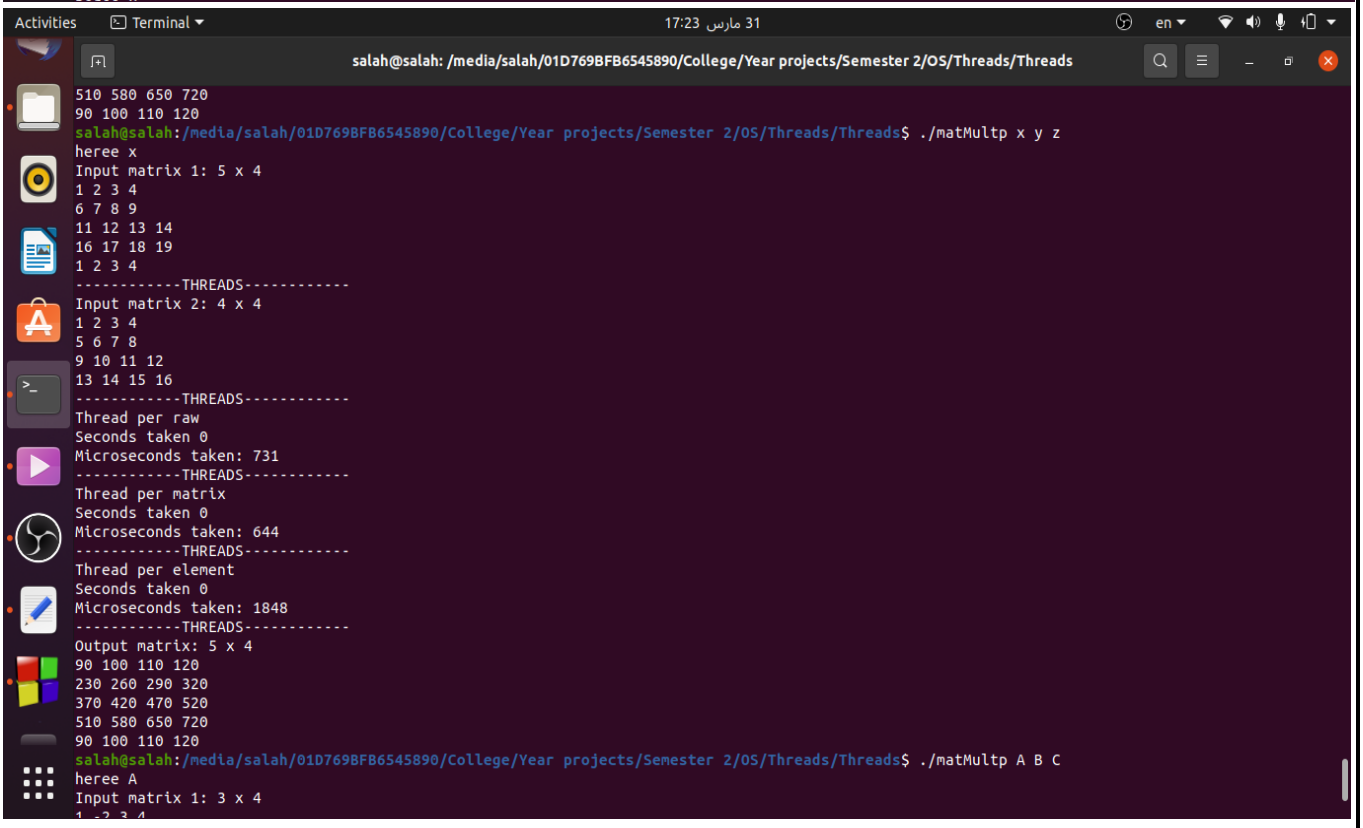
NAME	Salah El-Din Ahmed Muhammed El-Sayed Eltenehy.
ID	**19015854.**

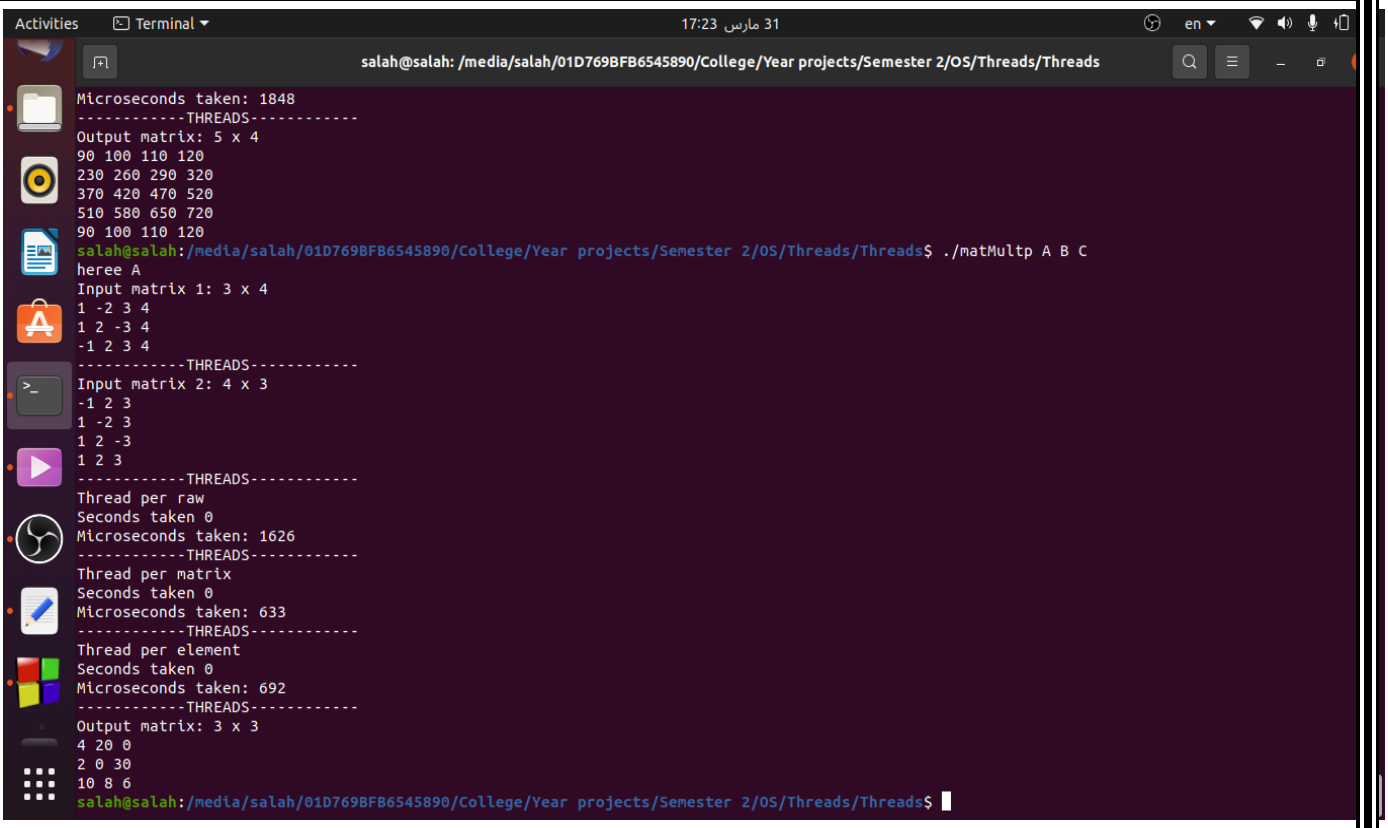
- Code organization:
  - Code mainly consists of three parts.
  - Each part for one of algorithms.
  - Thread per matrix:
    - It is forward calculate the value and store it directly in the file.
  - Thread per Raw:
    - Calculate each raw and save it within the right index in a global matrix to save it later in the file.
  - Thread per element:
    - Calculate the element and send the indexes to store it correctly in the global matrix and save it later in the file.
- Main functions:
  - Thread per matrix:
    - Takes two arguments “array1 & array2”.
    - Loop iterates for all elements.
    - Call another function “getColumn” and send array2 to it to return the required column.
    - Store the value in the file and repeat the previous steps.
  - Thread per raw:
    - It creates n threads “where n the first dimension of first array”
    - Each thread takes raw from the first array and the second array.
    - Calculate the result raw and save it in the global array.
    - After finishes all calculations save the global array.
  - Thread per element:
    - Same as thread per raw except number of threads.
    - It creates n threads “n →first dimn * second dimn”

        - Each threads takes raw and column and calculate the result.
- Compile and run the code:
    - Open the terminal and change directory to the project file.
    - Write “make” command to build the code.
    - Write “./matMultp” command to run the code.
    - The default names for files “a b c”.
    - You can change them. e.g.”./matMultp x y z”
    - Where:
        - X the name of first file.
        - Y the name of second file.
        - Z the name of output files.
- Sample run:
-




Activities
Terminal
17:22 31 مارس
en
salah@salah: /media/salah/01D769BFB6545890/College/Year projects/Semester 2/OS/Threads/Threads
main.c:460:5: note: include '<string.h>' or provide a declaration of 'strcat'
main.c:462:5: warning: incompatible implicit declaration of built-in function 'memset'
462 | memset(fileName, 0, 25);
| ^~~~~~
main.c:462:5: note: include '<string.h>' or provide a declaration of 'memset'
salah@salah:/media/salah/01D769BFB6545890/College/Year projects/Semester 2/OS/Threads/Threads$ ./matMultp
Input matrix 1: 5 x 4
1 2 3 4
6 7 8 9
11 12 13 14
16 17 18 19
1 2 3 4
------------THREADS------------
Input matrix 2: 4 x 4
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
------------THREADS------------
Thread per raw
Seconds taken 0
Microseconds taken: 409
------------THREADS------------
Thread per matrix
Seconds taken 0
Microseconds taken: 347
------------THREADS------------
Thread per element
Seconds taken 0
Microseconds taken: 1645
------------THREADS------------
Output matrix: 5 x 4
90 100 110 120
230 260 290 320
370 420 470 520
510 580 650 720
90 100 110 120
salah@salah:/media/salah/01D769BFB6545890/College/Year projects/Semester 2/OS/Threads/Threads$ ./matMultp x y z


Activities
Terminal
17:23 31 مارس
en
salah@salah: /media/salah/01D769BFB6545890/College/Year projects/Semester 2/OS/Threads/Threads
510 580 650 720
90 100 110 120
salah@salah:/media/salah/01D769BFB6545890/College/Year projects/Semester 2/OS/Threads/Threads$ ./matMultp x y z
heree x
Input matrix 1: 5 x 4
1 2 3 4
6 7 8 9
11 12 13 14
16 17 18 19
1 2 3 4
------------THREADS------------
Input matrix 2: 4 x 4
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
------------THREADS------------
Thread per raw
Seconds taken 0
Microseconds taken: 731
------------THREADS------------
Thread per matrix
Seconds taken 0
Microseconds taken: 644
------------THREADS------------
Thread per element
Seconds taken 0
Microseconds taken: 1848
------------THREADS------------
Output matrix: 5 x 4
90 100 110 120
230 260 290 320
370 420 470 520
510 580 650 720
90 100 110 120
salah@salah:/media/salah/01D769BFB6545890/College/Year projects/Semester 2/OS/Threads/Threads$ ./matMultp A B C
heree A
Input matrix 1: 3 x 4

- 

- Comparison:
  - As shown in the pictures the thread per matrix takes the minimum time.
  - It means that when we create many threads they take long time.
  - As a result the thread per element is slowest algorithm.