Lab#2: Matrix Multiplication (Multi-Threading)

Student Name: Mohamed Ayman Saaed Mahmoud

ID: 19016250

* File Organization:

I’ve one folder has files:

1. Source code “matMultp.c”
2. Bash script to build source code “build.sh”
3. Five text files for input and output default “a.txt, b.txt, c\_per\_matrix.txt, c\_per\_row.txt, c\_per\_element.txt”

* Main Functions
* thread\_per\_matrix():
  + It doesn’t take arguments
  + It performs multiplication of matrix A \* B
  + It put the result in matrix CPerMatrix
* thread\_per\_row():
  + It takes row number as argument
  + Perform multiplication of row of A \* all columns of matrix B
  + It put the result in matrix CPerRow
* thread\_per\_element()
  + It takes struct has row number and column number as argument
  + Perform multiplication of row of A \* a column of matrix B
  + It put the result in matrix CPerElement
* parse\_argv():
  + It parse arguments that passes to main function
  + Set files variables name
* read\_from(matrix, file):
  + It takes matrix and file name and
  + It serialize input from file to matrix
* write\_file(matrix, file, det):
  + It takes matrix, file name and det variable
  + Det variable determine which file postfix
  + It write element of matrix to file name
* Construct():
  + It takes matrix reference, row and columns size
  + It allocate matrix in heap
* Code Organization
  + I’ve divided my code into function each does separate take
  + When you run the argument you entered taken by function parse\_argv and it assign files name variables
  + Then I declare thread and path threads functions to each thread
  + Matrices allocated in Matrix struct has row and columns size and a reference to matrix elements in heap
* How to compile and run

1. Open terminal in the folder that has source code “matMultp.c”
2. You will find file “build.sh” in that folder
3. Run the following command “bash build.sh”

Graphical user interface, text

Description automatically generated

1. You will notice that file “matMultp” created.
2. Execute command “./matMultp ‘arg1’ ‘arg2’ ‘arg3’ to run the file

Text

Description automatically generated with medium confidence

Text

Description automatically generated

* Sample Runs

A picture containing text, device, meter, gauge

Description automatically generated

A picture containing graphical user interface

Description automatically generatedA picture containing graphical user interface

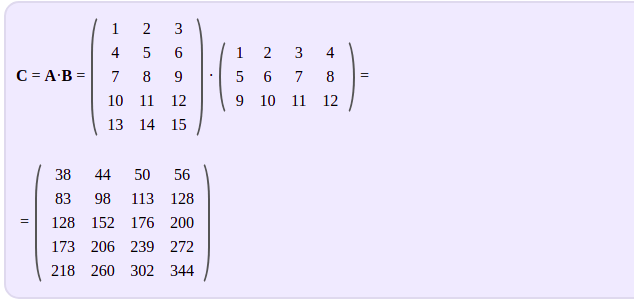
Description automatically generated A picture containing calendar

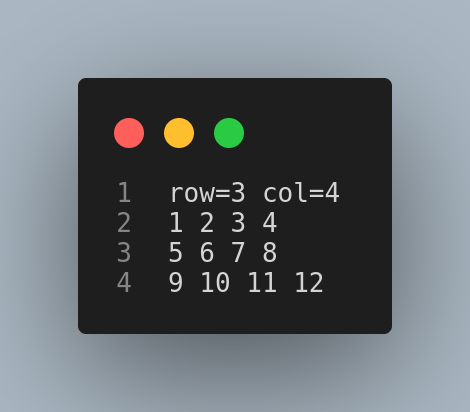
Description automatically generated

output

a.txt

b.txt



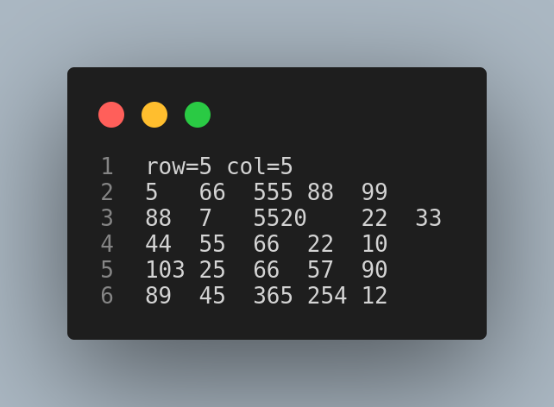
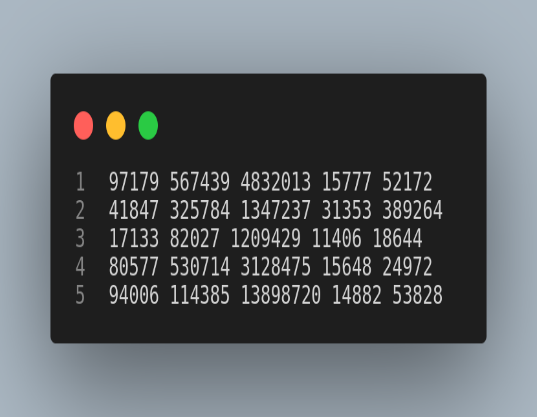
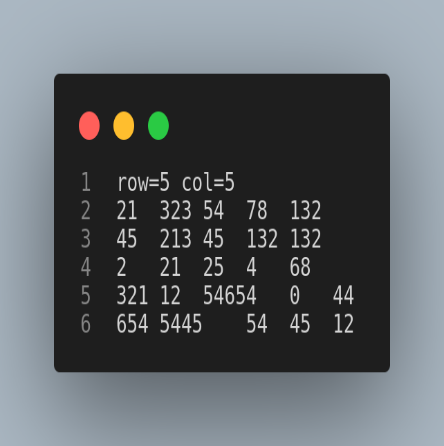
output

a.txt

b.txt

Table

Description automatically generated

output

a.txt

b.txt

* Comparison between methods

1. Thread per matrix
   1. It needs one thread.
   2. Time it takes.

Graphical user interface, text

Description automatically generated with medium confidence

1. Thread per row
   1. It needs threads equal to number of rows of matrix A.
   2. Time it takes.

Graphical user interface, text

Description automatically generated with medium confidence

1. Thread per element
   1. It needs threads equal to the dimension of the result array.
   2. Time it takes.

Graphical user interface, application

Description automatically generated