Operating Systems

Lab 2

Threads

Name: Nada Mostafa Rashad

ID: 60

Problem Statement:

It is required to implement two popular algorithms as multi-threaded ones

1) Matrix Multiplication:

It is required to implement two variations of this algorithm:

a. The computation of each element of the output matrix happens in a thread.

b. The computation of each row of the output matrix happens in a thread.

2) Merge Sort algorithm where:

in the algorithm, each time the list is divided; two threads are created to do merge-sort on each half separately. This step is repeated recursively until each sub-list has only one element.

Program Design, implementation:

- Both programs are implemented using C++ programming language.
- For the first requirement:
  - The project is divided into three classes; the reading class which reads the input matrices from the text file and passes them to the main which checks if the input is valid then passes it to the to the Variation1 class and Variation2 class which implement the multiplication process using a different variation each;
  - Variation2 → The computation of each element of the output matrix happens in a thread.
  - Variation1 → The computation of each row of the output matrix happens in a thread.
  - The main then receives the output of each class and prints them along with the time taken to compute each output.
- For the second requirement:
  - The project is divided into two classes; The reading class which reads the input array from a text file then passes it to the main which checks that its length is bigger than 1 then passes it to mergeSort class which sorts the array using merge sort algorithm where each recursive call is done in a new thread.

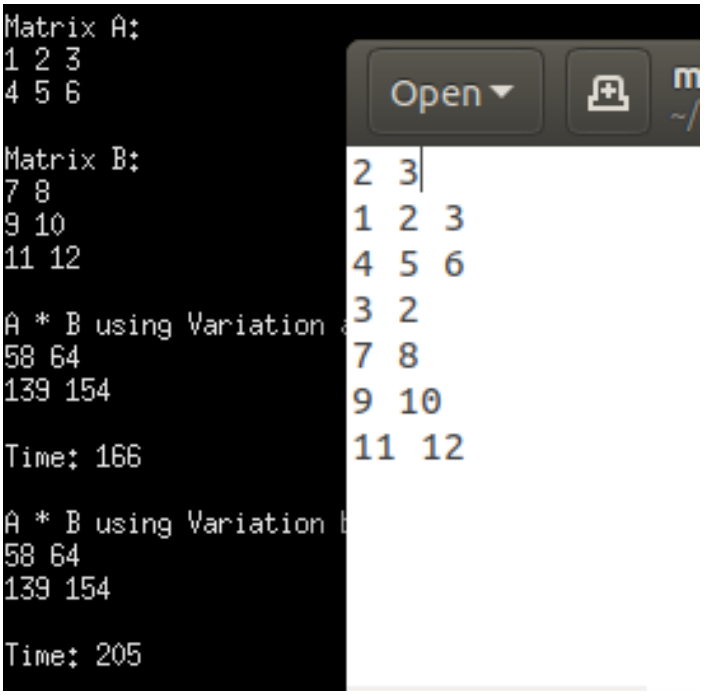
- The mergeSort class then passes the output array to the main which prints it.

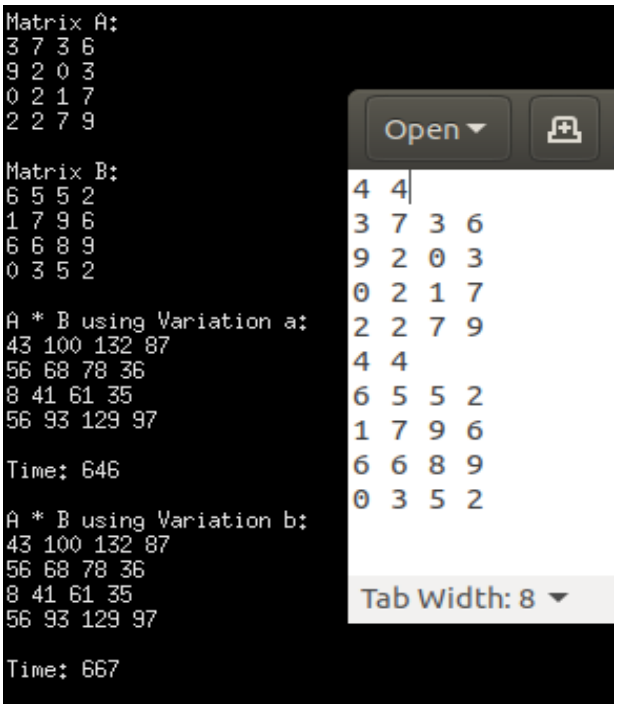
Assumptions:

- For the matrix multiplication part, it is assumed that the input is given in a text file where a line contains the number of rows of a matrix (r) and the number of columns (c) then the next r lines contain the elements of the matrix then the same is repeated for the second matrix.
- For the merge sort part, it is assumed that the first line contains the length of the array to be sorted and the following line contains the elements of the array.

Sample Runs:

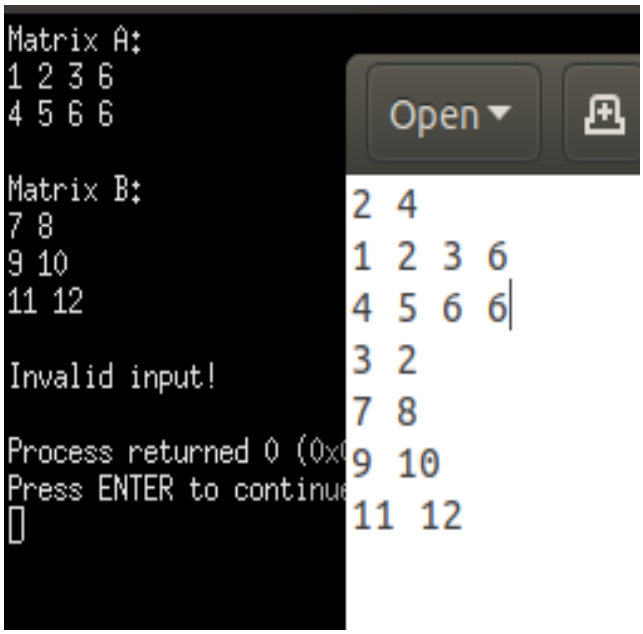
- **Matrix Multiplication:**





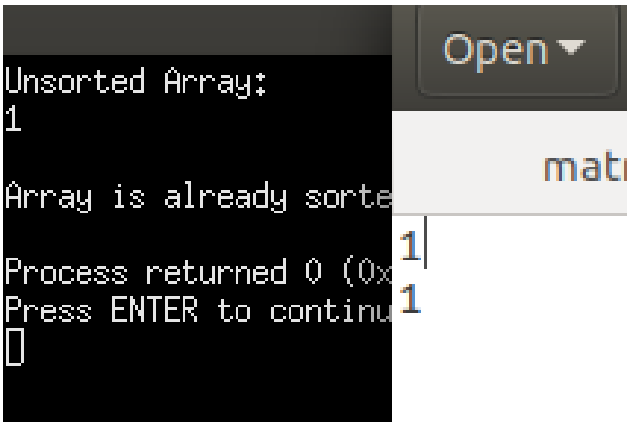
Valid input

→ The second variation is faster than the first because each thread has a nested for loop instead of having a a thread for each loop so the waiting time of each thread will decrease.

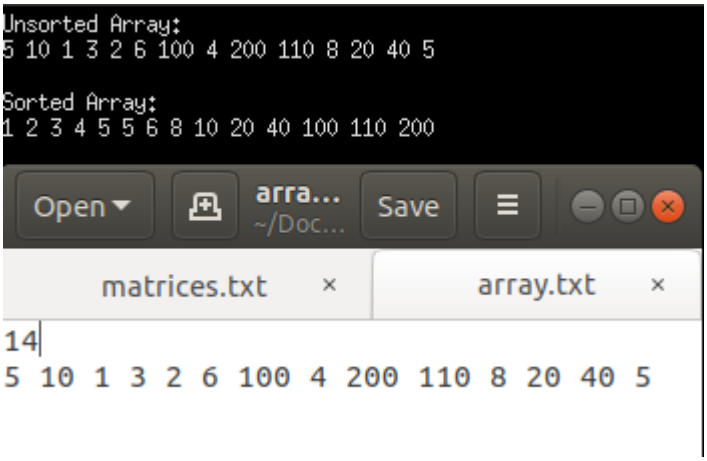


Invalid Input

- **Merge Sort:**



Array with length 1



Array with many elements