Assignment 5 :: Performance Measurement on Single and Multiple Threads

Code 1: Matrix Addition Single Thread

For this task, create two 2D random arrays called A and B. Create a 3rd 2D random array called C of the same size which performs the operation:

$$\mathbf{C} = \mathbf{A} + \mathbf{B} \quad (1)$$

The size of the arrays should be controlled using a for loop and should range from 100 to 500 with 100 units of jumps. In each iteration, measure the time before and after the execution of Equation (1). Pseudo-code is roughly as follows:

```
for (n = 100; n <= 500; n+=100) {
        A = Random Array (n x n);
        B = Random Array (n x n);
        Start = Get System Time
        C = A + B
        End = Get System Time
        Print (n, End-Start)
}
```

When the code finishes execution, you should have a table as follows:

Matrix size [n x n]	Execution Time
100	
200	
300	
400	
500	

Code 2: Matrix Addition Multiple Threads

Extend Code 1 as the following pseudo-code

```
for (n = 100; n <= 500; n+=100) {
        A = Random Array (n x n);
        B = Random Array (n x n);
        Print (n);
        for (c = 10; c <= n; c+=10 {
                Threads[c]
                Start = Get System Time
                C = A + B /* Multi-Threaded Version */
                End = Get System Time
                Print (c, End-Start)
        }
}
```

Note: Please note that you need to evenly distribute the operations across all threads. There will be some problems which would not be easily done, e.g., a 100 x 100 with 3 threads would not be doable. But a 100 x 100 with 2, 4, 5, 10, 20, 25, 50 threads would be possible. Extra marks may be given if you improvise around this, otherwise you can ignore these cases.

When the code finishes execution, you should have a table as follows:

Matrix size [n x n]	Execution Time				
	C = 10	C = 20	C= 30	…	C = N
100					
200					
300					
400					
500					