HW 01

Part I : Matrix

a. Write a class Column_Major_Matrix that has a member all_column which is of type vector<vector<T>>
b. Write a class Row_Major_Matrix that has a member all_row which is of type vector<vector<T>>
c. Provide a constructor for each class that takes arguments to specify the dimensions (e.g., Column_Major_Matrix<int> cc1 (1000, 1000);), and fills up all elements by randomly generated values of type T.
d. Provide getter/setter function to access each column and row by an index.
e. Overload copy/assignment and move copy/assignment operator to allow the following in the main function:

```
Column_Major_Matrix<int> cc1 (1000, 1000);
Row_Major_Matrix<int> rr1( 1000, 1000);
Column_Major_Matrix<int> cc2 (cc1);
Row_Major_Matrix<int> rr2 = (rr1);
Column_Major_Matrix<int> cc3 = std::move( cc2 );
Row_Major_Matrix<int> rr3 = std::move( rr2 );
```

f. Overload operator* in Row_Major_Matrix to allow calculation of the product of a Row_Major_Matrix instance to a Column_Major_Matrix instance, and return the resultant product as a Row_Major_Matrix.
g. Overload operator* in Column_Major_Matrix to allow calculation of the product of a Column_Major_Matrix instance to a Row_Major_Matrix instance, and return the resultant product as a Column_Major_Matrix.

h. Write type conversion operators (i.e., operator Row_Major_Matrix() and operator Column_Major_Matrix()) to allow implicit type conversion between Row_Major_Matrix and Column_Major_Matrix. Show it works by:

```
Column_Major_Matrix<int> cc (55, 1000);
Row_Major_Matrix<int> rr (1000, 66);
Row_Major_Matrix<int> rr = cc*rr;
```

i. Overload operator% to use exactly 10 threads to multiplex the multiplication, and use std::chrono to show the speedup w/ and w/o multithreading.