Problem #1

Matrix Multiplication

In this task you need to tackle the problem of multiplying 2 matrices in a multi-threaded fashion. First matrix (A) is given in row major order. The second matrix (B) is given in transpose form (column major order). The multiplication is guaranteed to be possible. Parallelize the task as follows:

- Create a set of worker threads. How many threads should be created - this will be a parameter given by user to your program. Let's call this set W. It is guaranteed that row(A) * column(B) is a multiple of |W|.
- Write a class called WorkItem. Each WorkItem object holds i^{th} row of A (A_i) and j^{th} column of B (B^T_j) for all 0 <= i, j <= row size of A. A WorkItem also stores the indices i and j. So there will be row(A) * column(B) work items. All these work items are put in a queue. This can be done by your main thread.
- A worker thread of W1 extracts a work item from the queue and performs the summation of multiplication between the A_i and B^T_j. Each worker performs exactly row(A) * column(B) / |W| number of work. The worker writes the result to the appropriate cell of the resultant matrix.
- Main thread writes the resultant matrix to the console.

Be sure to consider the following:

- Be prudent in using synchronization. Do not use it unnecessarily. Also, use enough synchronization to avoid data corruption.
- Perform wait, notify and joining, wherever you find it appropriate.
- Analyze your code to detect bugs. Ensure good parallelism and integrity of data
- Double check your results with a regular (single threaded) matrix multiplication routine.