CS 375 - UNIX System Programming
Fall 2018 – Programming Project 7

Assignment
Write two separate functions that multiply two square matrices. (If you don't remember, or don't know, how to do this, you'll have to look it up.) Function A should multiply the two matrices using a single thread while function B uses multiple threads to obtain the same result. The prototype for each function is:

void funcA(float *MATA, float *MATB, float *MATC, int N);

MATA and MATB are pointers to an area of memory containing the two input matrices and MATC points to an area of memory that will contain the result. N is the dimension of each of the square matrices, that is, each matrix has dimension N x N. Note that the matrices are laid out with the elements contiguous in memory. Assume that the matrices are in row order, that is, the first row is followed by the second row, and then the third row, etc. The total number of elements is, of course, N x N. The formula for converting two dimensional indexing (i.e. row, col) into one dimension is not difficult.

You are to develop your own algorithm for deciding on how many threads to use in the multithreaded algorithm. (Generally, the number of threads should be somehow proportional to N but you probably don't want more than 5? or 10? threads. Feel free to experiment.)

Write a driver program (named **`matmultiply`**) to test your functions. The driver should take two arguments. The first argument should be either the letter '**`s`**' or the letter '**`m`**' to indicate whether the single-threaded **`funcA()`** should be used or the multithreaded **`funcB()`**. The second argument should be an integer number that indicates the dimension of the matrix multiplication. The driver should allocate space for all three matrices on the heap (use **`new`**) and generate random input matrices for multiplication using **`rand()`**.

Use the UNIX "**`time`**" command to time how long it takes your program to run. To time a program use it like this:

```
time ./program program_arguments
```

It reports the real (wall clock), user (CPU time spent running user code) and system (CPU time spent running kernel code – exec, fork, open, etc) times. The wall clock time will depend on the number of other users and processes running on the system, so you may want to run the program multiple times and report the lowest wall clock time. Observe times for the single-threaded and multi-threaded multiplications for matrices of size (i.e. N) equal 10, 100, and 1000.

In a PDF document, briefly describe your algorithm for choosing the number of threads and dividing the work among the threads for the multithreaded function. Also report the times you observed in a table format. Discuss whether the results were what you were expecting or not.

Grading
Grades for this project will be awarded based on the following criteria:

80%	correctness
10%	result reporting
10%	elegance of design in using threads

What to submit

- Provide a makefile named **`Makefile`** that will make an executable named **`matmultiply`**.
- Create a tarfile or zipfile containing your program source file(s), makefile, and documentation.
- Submit your archive using the submission system (http://submission.evansville.edu). The grading script only will make the project and check that executable named **`matmultiply`** is produced. It will not run anything.

This assignment was created by Dr. Hwang.