Comp. Arch. HW #5 Due: 10/18/14 (Sat.) at 11:59 pm

Matrix Multiplication is a frequently used operation that takes two matrices A (m x q) and matrix B (q x n) and produces matrix C (m x n), where c_{ij} is the dot product of the i^{th} row of A with the j^{th} column of B. In other words,

$$c_{ij} = \sum_{k=0}^{q-1} a_{ik} * b_{kj}$$

For example: A × B = C

$$\begin{pmatrix} 2 & 3 & 1 \\ 0 & 2 & 1 \\ 2 & 2 & 1 \\ 0 & 3 & 2 \end{pmatrix} \times \begin{pmatrix} 2 & 2 \\ 1 & 0 \\ 2 & 1 \end{pmatrix} = \begin{pmatrix} 9 & 5 \\ 4 & 1 \\ 8 & 5 \\ 7 & 2 \end{pmatrix}$$

The sequential code for matrix multiplication is:

```
for i = 0 to m-1 do
    for j = 0 to n-1 do
        cij = 0
            for k = 0 to q-1 do
                cij = cij + aik * bkj
            end for k
     end for j
end for i
```

Your assignment is to implement matrix multiplication serially using C/C++ on student.cs.uni.edu to compile and time your code.

1) Download and extract the starter code `hw5.zip` which is available at:
http://www.cs.uni.edu/~fienup/cs2420f14/homework/

2) For this activity I want you to:

- use FileZilla, WinSCP, scp, ... to copy the starter code `hw5` directory to student.cs.uni.edu
- use an editor (emacs or nano) to complete the `mmult_sequential.c` program that prompts the user for the size (n) of the matrices
- compile the C to an executable file using: `gcc -o mmult mmult_sequential.c`
- when its working capture the interactive running of the program using: `script out.txt` to start the capture, `./mmult 4` to run the program, and `<Ctrl>+d` to end the capture
- display the contents of the `out.txt` to the screen using the `less out.txt` command (q-to exit `less`)

3) Use a secure ftp client (e.g., FileZilla, WinSCP, scp, etc.) to copy your `hw5` directory back to your local computer
(On a MAC you can probably use: `scp -r localDir userName@student.cs.uni.edu:/hw5`)

4) On your local computer zip the `hw5` directory and submit as Homework #5 at:
http://www.cs.uni.edu/~fienup/cs2420f14/homework/submissionDirections.htm

```
#include <stdio.h>
#include <math.h>
#include <time.h>
#include <stdlib.h>

// Prototype of functions
void printMatrix(double ** matrix, int rows, int columns);

int main(int argc, char * argv[]) {

  int n;
  double ** A;  // pointer to matrix A that's n x n
  double ** B;  // pointer to matrix B that's n x n
  double ** C;  // pointer to matrix C that's n x n

  int i, j, r, c, k;

  long startTime, endTime,seqTime;

  if (argc != 2) {
    printf("usage: %s <integer size of matrix>\n", argv[0]);
    exit(1);
  } // end if

  sscanf(argv[1], "%d", &n);

  srand(5);  // initialize random number generator
  //  srand((long)time(NULL)); /* initialize rand() */

  // dynamically allocate matrices A, B, and C.

  // I'LL DO A, BUT YOU NEED TO ALLOCATE B AND C.

  A = (double **) malloc(sizeof(double *)*n);
  for (r = 0; r < n; r++) {
    A[r] = (double *) malloc(sizeof(double)*n);
  } // end for r

  printf("after allocating matrices\n");

  /* initialize array A and B */
  for( r=0; r<n ; r++ ){
    for( c=0; c<n ; c++ ){
      A[r][c] = rand() / (double) RAND_MAX;
      // B[r][c] = rand() / (double) RAND_MAX;
    } // for c
    //    printf("%d", );
  } // end for r

  printf("after initializing matrices\n");

  time(&startTime);

  /* Calculate AND time sequential matrix-multiplication results */
  // ADD CODE HERE TO PERFORM C = A X B MATRIX MULTIPLICATION WITH n X n ARRAYS

  time(&endTime);
  seqTime = endTime-startTime;
  printf("Seq time = %ld\n",seqTime);
} // end main

void printMatrix(double ** matrix, int rows, int columns) {
  int r, c;

  for( r=0; r<rows ; r++ ){
    for( c=0; c<columns ; c++ ){
      printf("%10.5f   ",matrix[r][c]);
    } // for c
    printf("\n" );
  } // end for r
} // end printMatrix
```