Assignment 1: Matrix Multiplication using Posix Threads

Note: You must use Linux for this task !! Your task is to write a code for matrix multiplication using Posix Threads. Given two matrices A and B, compute the third matrix C as C = A * B. For simplicity, consider all matrices to be *n* x *n.* A pseudo-code implementation will be along the following lines:

```
for i = 1 to n
  for j = 1 to n
    for k = 1 to n
      C(i,j) = C(i,j) + A(i,k) * B(k,j)
    end
  end
end
```

Instructions

1. **Skeleton File**
   A skeleton file for code development (array-demo.c) has been provided for you. Use this to start your work.

2. **Size of the Matrices A, B, and C:**

   The code must take the size of the matrix from the SIZE preprocessor variable. For e.g:
   `#define SIZE 10`
   will create a 10x10 matrix. Later on, when your code is ready, you must **DELETE** this line from the code.

3. **Input Data for matrices A and B:**
   Input data is not important for our implementation. But for testing your code, it would be important to look at the final result. As an example:

```
If A =    1 1 1 1, and   B =  1 1 1 1, then C =   4 4 4 4
          1 1 1 1            1 1 1 1             4 4 4 4
          1 1 1 1            1 1 1 1             4 4 4 4
          1 1 1 1            1 1 1 1             4 4 4 4.
```

   For this reason, you may use the following code to populate your matrix:

```
int rc, t, u;
for (t = 0; t < SIZE; t++) {
     for (u = 0; u < SIZE; u++) {
          A[t][u] = B[u][t] = 1;
     }
}
```

   and the following code to display the final matrix C:

```
for (t = 0; t < SIZE; t++) {
     for (u = 0; u < SIZE; u++) {
          printf("%.0f ", C[t][u]);
     }
     printf("\n");
}
```

4. **Data Type of matrices A, B, and C:**
   Data type of all matrices MUST be double.

5. **Measuring Speed of Execution:**
   You have to find out the execution time of only those parts of the code that are involved in thread related activities (For example, creating threads, execution of threads, joining of threads, etc.). All other activities such as printing matrices, providing input data to matrices, etc. must not be recorded. To compute the execution time, enclose "your code" with the following:

```
struct timespec start, finish;
double elapsed;
clock_gettime(CLOCK_MONOTONIC, &start);
     /* Multi-Threading Activity */
clock_gettime(CLOCK_MONOTONIC, &finish);
elapsed  = (finish.tv_sec – start.tv_sec);
elapsed += (finish.tv_nsec - start.tv_nsec) / 1000000000.0;
fprintf(stderr, "Time: %f seconds\n", elapsed);
```

   Do understand what the above code is doing !!! If the above requires a library, find it from google and use it!

Deliverable

Your deliverable should contain the following:

1. Your Name (Roll Number)
2. Your Source Code (with Comments)
3. The output of the Shell Script ./assignment1.sh. Run the shell script as:
   ./assignment1.sh sourceCode.c
   and it will generate some results for you as follows (demo)

   Name: omar (123)
   Processor Specs: Intel(R) Core(TM) i5-2430M CPU @ 2.40GHz (4)
   Memory: 3.84002 GB
   Operating System: Linux 4.0.5-gentoo

N	Output 1	Output 2	Output 3	Output 4	Output 5	Average
20	0.014044	0.010022	0.012130	0.011077	0.008566	0.0111678
40	0.043181	0.038821	0.042012	0.037849	0.037842	0.039941

   ...

4. The image file <your roll number>.eps

Generate a PDF of all the above and submit it on slate.

Marking:

80 % marks will be for your assignment deliverable.

20 % marks will be awarded to the top 20% students in class who have the fastest results (based on average execution time).

So make sure you have a fast PC and optimized code to get the 20%.