Name: Brijesh Mavani
CWID: A20406960
University: Illinois Institute of Technology
Course: Parallel and Distributed Processing
Assignment: 2

Shared memory programming (Pthreads & openMP):

In this exercise we need to implement the parallel code for the given sequential code of Gaussian Elimination without pivoting. The Gaussian elimination stage of the algorithm comprises (n-1) steps. In the algorithm, the ith step eliminates nonzero subdiagonal elements in column i by subtracting the ith row from row j in the range [i+1,n], in each case scaling the ith row by the factor A ji / A ii so as to make the element Aji zero.

Here, we want to parallelize the operation of zeroing out the elements in lower half of the matrix. The zeroing out of each element (and the associated updates to the row elements to the right of it) are all independent. Thus, when doing the first (element A[1,1]) we will be able to operate on all the remaining rows A[2..N,*] in parallel. Since each row contains n elements this results in (N-1)*n independent operations to parallelize. Further, the operations performed on each element are simple (a multiply and subtraction). This suggests that the speedup from parallelism will likely be limited, especially for small matrices and/or many threads. To address synchronization overhead we may want to group potentially parallel operations into larger groups and execute them serially. Hence, the function name Inner was built which takes care of making lower half of the matrix zero. After the computation thread will free the memory allocated to it and exit the function.
The synchronization was handled by making another for loop for joining all the threads. This loop will execute for same no of times as of the outer loop. This assures that all created threads are joined.

- Pthread:

The first variation was made to implement Pthread by defining inner loop as a function and also by creating the chunk. The structure was created to hold the normalizing row and id variable. This was executing without error but wasn't giving any value in X as shown in below screen shot.

```
bmavani@Brijesh-HP-Pavilion-dv6-Notebook-PC:/media/bmavani/01D33C14DD6B0FF0/STUDY/MS/PDP/Assignments/Assignment2$ gcc -o Gauss_Pthread_Chunk Ga
uss_Pthread_Chunk.c -lpthread
bmavani@Brijesh-HP-Pavilion-dv6-Notebook-PC:/media/bmavani/01D33C14DD6B0FF0/STUDY/MS/PDP/Assignments/Assignment2$ ./Gauss_Pthread_Chunk 8 100
Random seed = 100

Matrix dimension N = 8.

Initializing...

A =
        20683.02, 28149.61, 17326.79, 30246.88, 51816.14, 4591.98, 9608.34, 56357.20;
        18674.05, 26746.22, 40733.46, 42631.96, 1344.17, 29355.54, 29809.72, 9809.90;
        15768.05, 9839.23, 43957.80, 50621.17, 26421.44, 36044.86, 14709.94, 19091.57;
        31727.72, 38440.23, 23464.04, 32528.63, 20058.85, 61714.84, 1976.80, 19208.40;
        24627.99, 41394.83, 13654.79, 63314.98, 29493.78, 46682.33, 60056.60, 61626.04;
        3519.46, 40229.90, 35007.70, 3759.22, 53167.66, 11242.32, 57341.90, 20435.73;
        37373.47, 26957.53, 25199.35, 48296.68, 29898.08, 40136.64, 52597.97, 45629.84;
        63570.23, 7018.38, 55348.83, 29506.70, 2398.01, 4610.37, 27049.22, 16148.89;

B = [33778.71; 57122.85; 57904.46; 28387.21; 29026.49; 24897.11; 55120.88; 49929.51]

Starting clock.
Computing parallelly using Pthreads.
Stopped clock.

X = [ -nan;  -nan;  -nan;  -nan;  -nan;  -nan;  -nan;  -nan]

Elapsed time = 3.001 ms.
(CPU times are accurate to the nearest 0.001 ms)
My total CPU time for parent = 0 ms.
My system CPU time for parent = 0 ms.
My total CPU time for child processes = 0 ms.
----------------------------------------
bmavani@Brijesh-HP-Pavilion-dv6-Notebook-PC:/media/bmavani/01D33C14DD6B0FF0/STUDY/MS/PDP/Assignments/Assignment2$
```

After the incorrect implementation, I corrected the code by removing the structure. I simply created the function for inner loop and called it from the outer loop in Gauss function. This was working without issue and also provided the correct output as shown in below screen shot.

```
bmavani@Brijesh-HP-Pavilion-dv6-Notebook-PC:/media/bmavani/01D33C14DD6B0FF0/STUDY/MS/PDP/Assignments/Assignment2$ gcc -o Gauss_Pthread Gauss_Pt
hread.c -lpthread
bmavani@Brijesh-HP-Pavilion-dv6-Notebook-PC:/media/bmavani/01D33C14DD6B0FF0/STUDY/MS/PDP/Assignments/Assignment2$ ./Gauss_Pthread 8 100
Random seed = 100

Matrix dimension N = 8.

Initializing...

A =
        20683.02, 28149.61, 17326.79, 30246.88, 51816.14, 4591.98, 9608.34, 56357.20;
        18674.05, 26746.22, 40733.46, 42631.96, 1344.17, 29355.54, 29809.72, 9809.90;
        15768.05, 9839.23, 43957.80, 50621.17, 26421.44, 36044.86, 14709.94, 19091.57;
        31727.72, 38440.23, 23464.04, 32528.63, 20058.85, 61714.84, 1976.80, 19208.40;
        24627.99, 41394.83, 13654.79, 63314.98, 29493.78, 46682.33, 60056.60, 61626.04;
        3519.46, 40229.90, 35007.70, 3759.22, 53167.66, 11242.32, 57341.90, 20435.73;
        37373.47, 26957.53, 25199.35, 48296.68, 29898.08, 40136.64, 52597.97, 45629.84;
        63570.23, 7018.38, 55348.83, 29506.70, 2398.01, 4610.37, 27049.22, 16148.89;

B = [33778.71; 57122.85; 57904.46; 28387.21; 29026.49; 24897.11; 55120.88; 49929.51]

Starting clock.
Computing parallelly using Pthreads.
Stopped clock.

X = [ 9.13;  5.59; -13.10; 16.06; 11.87; -8.71;  0.90; -20.49]

Elapsed time = 2.359 ms.
(CPU times are accurate to the nearest 0.001 ms)
My total CPU time for parent = 0 ms.
My system CPU time for parent = 0 ms.
My total CPU time for child processes = 0 ms.
----------------------------------------
bmavani@Brijesh-HP-Pavilion-dv6-Notebook-PC:/media/bmavani/01D33C14DD6B0FF0/STUDY/MS/PDP/Assignments/Assignment2$
```

- OpenMP:

We can parallelize the code for OpenMP by providing directives. We will use #pragma omp parallel for directive to parallelize our code. This directive divides loop iterations between the spawned threads. We will use this directive just inside the outer loop. We can use it inside inner loops as well but it would create an overhead for threads to maintain a track of where exactly the partitioning of the innermost loops has occurred. Also, the outer loops would have to wait up until all threads for that iteration has finished its work, which reduces the performance. I have created the multiple versions of code and evaluated the results. The various versions and description is provided in below section:

Variations tried:

1) Share and Private variable:

   #pragma omp parallel for shared(A, B) private(multiplier,row,col)

Provide share and private both clauses. This will not alter the results as Matrices A and B are public and other variables are defined as private.

```
bmavani@Brijesh-HP-Pavilion-dv6-Notebook-PC:/media/bmavani/01D33C14DD6B0FF0/STUDY/MS/PDP/Assignments/Assignment2$ gcc -o Gauss_OpenMP_General G
auss_OpenMP_General.c -fopenmp
bmavani@Brijesh-HP-Pavilion-dv6-Notebook-PC:/media/bmavani/01D33C14DD6B0FF0/STUDY/MS/PDP/Assignments/Assignment2$ ./Gauss_OpenMP_General 8 100
Random seed = 100

Matrix dimension N = 8.

Initializing...

A =
        20683.02, 28149.61, 17326.79, 30246.88, 51816.14, 4591.98, 9608.34, 56357.20;
        18674.05, 26746.22, 40733.46, 42631.96, 1344.17, 29355.54, 29809.72, 9809.90;
        15768.05, 9839.23, 43957.80, 50621.17, 26421.44, 36044.86, 14709.94, 19091.57;
        31727.72, 38440.23, 23464.04, 32528.63, 20058.85, 61714.84, 1976.80, 19208.40;
        24627.99, 41394.83, 13654.79, 63314.98, 29493.78, 46682.33, 60056.60, 61626.04;
        3519.46, 40229.90, 35007.70, 3759.22, 53167.66, 11242.32, 57341.90, 20435.73;
        37373.47, 26957.53, 25199.35, 48296.68, 29898.08, 40136.64, 52597.97, 45629.84;
        63570.23, 7018.38, 55348.83, 29506.70, 2398.01, 4610.37, 27049.22, 16148.89;

B = [33778.71; 57122.85; 57904.46; 28387.21; 29026.49; 24897.11; 55120.88; 49929.51]

Starting clock.
Computing parallel via OpenMP.
Stopped clock.

X = [ 9.13;  5.59; -13.10; 16.06; 11.87; -8.71;  0.90; -20.49]

Elapsed time = 3.078 ms.
(CPU times are accurate to the nearest 0.001 ms)
My total CPU time for parent = 0.001 ms.
My system CPU time for parent = 0 ms.
My total CPU time for child processes = 0 ms.
------------------------------------------
bmavani@Brijesh-HP-Pavilion-dv6-Notebook-PC:/media/bmavani/01D33C14DD6B0FF0/STUDY/MS/PDP/Assignments/Assignment2$
```

2) Only shared variables
   #pragma omp parallel for shared(A, B)

Provide only shared variables. This will still execute without error but it won't give correct output as variables multiplier,row,col are accessed by all threads and they should be private for computation.

```
bmavani@Brijesh-HP-Pavilion-dv6-Notebook-PC:/media/bmavani/01D33C14DD6B0FF0/STUDY/MS/PDP/Assignments/Assignment2$ gcc -o Gauss_OpenMP_Share Gau
ss_OpenMP_Share.c -fopenmp
bmavani@Brijesh-HP-Pavilion-dv6-Notebook-PC:/media/bmavani/01D33C14DD6B0FF0/STUDY/MS/PDP/Assignments/Assignment2$ ./Gauss_OpenMP_Share 8 100
Random seed = 100

Matrix dimension N = 8.

Initializing...

A =
        20683.02, 28149.61, 17326.79, 30246.88, 51816.14, 4591.98, 9608.34, 56357.20;
        18674.05, 26746.22, 40733.46, 42631.96, 1344.17, 29355.54, 29809.72, 9809.90;
        15768.05, 9839.23, 43957.80, 50621.17, 26421.44, 36044.86, 14709.94, 19091.57;
        31727.72, 38440.23, 23464.04, 32528.63, 20058.85, 61714.84, 1976.80, 19208.40;
        24627.99, 41394.83, 13654.79, 63314.98, 29493.78, 46682.33, 60056.60, 61626.04;
        3519.46, 40229.90, 35007.70, 3759.22, 53167.66, 11242.32, 57341.90, 20435.73;
        37373.47, 26957.53, 25199.35, 48296.68, 29898.08, 40136.64, 52597.97, 45629.84;
        63570.23, 7018.38, 55348.83, 29506.70, 2398.01, 4610.37, 27049.22, 16148.89;

B = [33778.71; 57122.85; 57904.46; 28387.21; 29026.49; 24897.11; 55120.88; 49929.51]

Starting clock.
Computing parallel via OpenMP.
Stopped clock.

X = [-18.41; 15.63;  0.53; -0.58;  0.14; -0.89;  0.55; -0.46]

Elapsed time = 3.234 ms.
(CPU times are accurate to the nearest 0.001 ms)
My total CPU time for parent = 0 ms.
My system CPU time for parent = 0 ms.
My total CPU time for child processes = 0 ms.
------------------------------------------
bmavani@Brijesh-HP-Pavilion-dv6-Notebook-PC:/media/bmavani/01D33C14DD6B0FF0/STUDY/MS/PDP/Assignments/Assignment2$
```

As we can see that time taken for program is almost similar to previous version. But values for X are totally different and incorrect. This is due to threads accessing same copy of variables multiplier, row, col.

3) Final version: Only private variable.

As stated earlier, we will use #pragma omp parallel for directive for parallelizing. We will pass the clause for private and shared variable. As, Matrices A and B are defined globally, they are already accessible and public for all functions. Hence, we do not need to provide them as shared explicitly. We will only pass variables multiplier, row, col as private: #pragma omp parallel for private (multiplier, row, col)

We can also provide the schedule as dynamic instead of default static. Dynamic scheduling is better when the iterations may take very different amounts of time. Here, we can say each iteration is taking approximately same

time as all of them are executing same arithmetic operations. Also, there is some overhead to dynamic scheduling. After each iteration, the threads must stop and receive a new value of the loop variable to use for its next iteration. Due to overhead of dynamic scheduling in this case we won't use it. Instead we will use default static schedule.

```
bmavani@Brijesh-HP-Pavilion-dv6-Notebook-PC:/media/bmavani/01D33C14DD6B0FF0/STUDY/MS/PDP/Assignments/Assignment2$ gcc -o Gauss_OpenMP_Final Gau
ss_OpenMP_Final.c -fopenmp
bmavani@Brijesh-HP-Pavilion-dv6-Notebook-PC:/media/bmavani/01D33C14DD6B0FF0/STUDY/MS/PDP/Assignments/Assignment2$ ./Gauss_OpenMP_Final 8 100
Random seed = 100

Matrix dimension N = 8.

Initializing...

A =
        20683.02, 28149.61, 17326.79, 30246.88, 51816.14, 4591.98, 9608.34, 56357.20;
        18674.05, 26746.22, 40733.46, 42631.96, 1344.17, 29355.54, 29809.72, 9809.90;
        15768.05, 9839.23, 43957.80, 50621.17, 26421.44, 36044.86, 14709.94, 19091.57;
        31727.72, 38440.23, 23464.04, 32528.63, 20058.85, 61714.84, 1976.80, 19208.40;
        24627.99, 41394.83, 13654.79, 63314.98, 29493.78, 46682.33, 60056.60, 61626.04;
        3519.46, 40229.90, 35007.70, 3759.22, 53167.66, 11242.32, 57341.90, 20435.73;
        37373.47, 26957.53, 25199.35, 48296.68, 29898.08, 40136.64, 52597.97, 45629.84;
        63570.23, 7018.38, 55348.83, 29506.70, 2398.01, 4610.37, 27049.22, 16148.89;

B = [33778.71; 57122.85; 57904.46; 28387.21; 29026.49; 24897.11; 55120.88; 49929.51]

Starting clock.
Computing parallel via OpenMP.
Stopped clock.

X = [ 9.13;  5.59; -13.10; 16.06; 11.87; -8.71;  0.90; -20.49]

Elapsed time = 2.965 ms.
(CPU times are accurate to the nearest 0.001 ms)
My total CPU time for parent = 0.002 ms.
My system CPU time for parent = 0 ms.
My total CPU time for child processes = 0 ms.
------------------------------------------
bmavani@Brijesh-HP-Pavilion-dv6-Notebook-PC:/media/bmavani/01D33C14DD6B0FF0/STUDY/MS/PDP/Assignments/Assignment2$
```