1. General Remarks

This assignment is an introduction to pthread programming. You are asked to write a pthread code which solves a system of linear equtions via Gaussian elimination with partial pivoting. (Partial pivoting is needed for numerical stability).

Your codes must compile by a standard `gcc` compiler and benchmarked on `ecelinux.ece.cornell.edu`.

- Your code must be well documented so any person with limited knowledge of C could understand what the code is doing.
- The first line in the code must show your name and net id, and specify how the code should be compiled and executed.
- The code must be saved in a text file named `your_net_id_hw2_code.c`.
- Results from the benchmarks need to be described in a file `your_net_id_hw2_writeup.pdf` (please DO NOT submit `*.docx` files)
- All files need to be archive with `tar` or `zip`. The archive must have the name `your_net_id_hw2.suffix` where `suffix` is either `tar` or `zip`. Please submit your work to Canvas.

2. A pseudo code

Say we want to solve for x

$$Ax = b$$

where $A = (a_{i,j})$ is an $n \times n$ matrix and $b = (b_i)$ is an $n \times 1$ vector, both real. We will use Matlab notation $a_{:,j}$ to denote column j of A, $a_{i:j,k}$ to denote elements from i to j in column k of A, etc.

The Gaussian elimination method first transforms a matrix to an upper triangulat form and next solves a an upper triangular system of linear equations by backsubstitution.

The following is an outline of a pseudocode written for a sequential implementation of Gaussian elimination with partial pivoting.

```
/* Step 1: triangularization                      */
/* Matrix A and vector b are updated as follows */
for i = 1 to n -1 {
   find the largest in magnitude element in a(i:n,i), say it is a(m,i)
   swap rows i and m of the current A
   swap b(i) with b(m)
/* update a(i+1:n,i+1:n) and b(i:n)              */
   for j = i to n {
```

```
      p = a(j,i)/a(i,i)
      a(j,i:n) = a(j,i:n) - p*a(i,i:n)
      b(j) = b(j) - p*b(i)
  }
}

/* Step 2: backsubbstitution                    */
for i = n to 1 {
  x(i) = b(i)/a(i,i)
  for j = i - 1 to 1 {
    b(1:i-1) = b(1:i-1) - a(1:i-1,i)*x(i)
  }
)
```

3. Requirements

Your algorithm should accept two arguments:

- n - the size of a matrix, followed by
- p - the number of threads.

Please populate the $n \times n$ matrix A and $n \times 1$ vector b with random numbers using `drand48` random numbers generator.

Apply your algorithm to compute x. To check your answer, compute

$$r = Ax - b, \quad \rho = \sqrt{\sum_{i=1}^{n} r_i^2}$$

where r is called the residual vector and ρ is its norm (norms measure the length of a vector).

Measure the execution time for Step 1 and Step 2 for various values of (n, p). Start with `n = 512` and increase it by a factor of 2 till `n = 4096` (or more if there is enough memory on the `ecelinux` machines). Start with `p = 1` and increase to twice the number of cores. Measure the achieved speed-ups versus a sequential implementation of Gaussian elimination (the case of a single therad compared to a sequential implementation will tell you how much of an overhead is introduced by using pthreads). Do not include the time for numerical verification of your results.

Write a document that describes how your program works. Sketch the key elements of your parallelization strategy. Explain how your program partitions the data and work among threads and how they are synchronized. Explain whether the workload is distributed evenly among threads. Justify your implementation choices.

Please present your tables and graphs in a way so they are easily readable. For example, if certain combinations of (number o threads, matrix dimension) do not bring any new information, you may omit them from your graphs. But then explain why you are omitting them.

For timings use the same directives as used in Assignment 1.