1. General Remarks

This assignment is an introduction to openMP programming. You are asked to write an openMP code which inverts a matrix via Gaussian elimination with partial pivoting. It is bassicaly finding solutions to multiple systems of linear equations but with the same system matrix and specially chosen right hand vectors. More precisely for an $n \times n$ matrix A we want to solve

$$Ax^{(i)} = e^{(i)},\ i = 1, 2, ..., n,$$

where

$$e^{(i)}_j = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases}$$

Or in other words we want to solve a matrix equation

$$AX = I_n$$

where I_n is an $n \times n$ identity matrix and X is a matrix which the ith column is the vector $x^{(i)}$,

$$X = [x^{(1)}, x^{(2)}, ..., x^{(n)}]$$

2. A pseudo code

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.

(We use Matlab notation `A(i:n,j:m)` to denote a submatrix of `A` consisting of rows `i` to `n` and columns `j` to `m`. A shorthand `A(i:n,:)` means that all columns are included. Similarly for `A(:,j:m)`.)

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

```
/* Step 1: triangularization                          */
/* Matrices A and I 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 rows i and m of the current I
/* update A(i+1:n,i+1:n) and I(i+1:n,:)                   */
   for j = i+1 to n {
     p = A(j,i)/A(i,i)
     A(j,i:n) = A(j,i:n) - p*A(i,i:n)
     I(j,:) = I(j,:) - p*I(i,:)
  }
```

```
}

/* Step 2: backsubbstitution                              */
for i = n to 1 {
  X(i,:) = I(i,:)/A(i,i)
  for j = i - 1 to 1 {
    I(1:i-1,:) = I(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

$$R = AX - I, \ \ nR = \sqrt{\sum_{i=1,j=1}^{n,n} R_{ij}^2}, \ nA = \sqrt{\sum_{i=1,j=1}^{n,n} a_{ij}^2}, \ nX = \sqrt{\sum_{i=1}^{n} X_{ij}^2}, \ err = \frac{nr}{nA \cdot nX}$$

where R is called the residual matrix and nR, nA, nX are norms of R, A, X, respectively (norms measure the length of vectors and matrices). The quantity `err` should be around 10^{-12} or less.

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 maximum number of cores. Measure the achieved speed-ups versus a sequential code (not an openMP code with a single thread). (Do not include the time for numerical verification of your results).

Write a document that describes how your programs work.

- Sketch the key elements of your parallelization strategy.
- Explain how your program partitions the data and work among threads.
- How synchronization is achieved.
- Explain whether the workload is distributed evenly among threads.
- Justify your implementation choices (selection of openMP directives).

Your findings, a discussion of results, graphs and tables should be saved in a file `your_net_id_hw3_writeup.pd`

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.

Your code (with instructions how to compile and execute it) should be saved in a file `your_net_id_hw3_code.c` The first two lines of the code should tell how your code needs to be compiled and how it shoule be executed.

Both files should be archived and named `your_net_id_hw3.x` where `x` stands for `zip` or `tar`.

NOTE: When we unzip your submission we want to see only the files

- `your_net_id_hw3_writeup.pdf` and
- `your_net_id_hw3_code.c`.

Please NO other files or subdirectories.