1. General Remarks

This assignment is an introduction to openMP programming. In the assignment you will permute an integer array `A` in two different ways.

2. Permutations

Way 1. Let $A \in \mathbb{Z}^{m \times n}$ where $\mathbb{Z}$ denotes the set of all integers.

In step i consider the submatrix $A[i:m, i:n]$ (this is a matrix consiting of rows i to m and columns i to n) of the current working matrix.

Find the maximal element in column $A[i:m, i]$, let it be $a[k, i]$. Swap rows i and k. Increase i until the working matrix is empty.

Call the final working matrix $Final\_A$

A pseudo algorithm is:

```
for i = 0 to n-1
    find max over k of abs(A[k,i]) in A[i:n,i]
    swap A[i,:] with A[k,:]
```

For example

$$\left(\begin{array}{r|rrr} 12 & 3 & -7 & 5 \\ 14 & -1 & 20 & 4 \\ 2 & 21 & 6 & 9 \\ -20 & 30 & 25 & 11 \end{array}\right) \xrightarrow{column(1)} \left(\begin{array}{r|r|rr} -20 & 30 & 25 & 11 \\ \hline 14 & -1 & 20 & 4 \\ 2 & 21 & 6 & 9 \\ 12 & 3 & -7 & 5 \end{array}\right) \xrightarrow{column(2)} \left(\begin{array}{rr|r|r} -20 & 30 & 25 & 11 \\ 2 & 21 & 6 & 9 \\ \hline 14 & -1 & 20 & 4 \\ 12 & 3 & -7 & 5 \end{array}\right) \quad \text{done}$$

Record all pairwise permutations $p(i, k)$ applied to A. Given A and $p(i, k)$, how would you find $Final\_A$ without repeating the just described proces?

(Note that A does not have to be a square matrix.)

Way 2. In step i find the maximal **in absolute value** element in the current submatrix $A[i : m, i : n]$, let it be $a[p, q]$. Swap rows i and p and columns i and q. Increase i until the working matrix becomes empty.

Record pairwise row and column permutations performed in step i for all i. Given A and all pairwise row and columnpermutations, how would you find $Final\_A$ without repeating the just described proces?

3. Codes format

There are two OpenMP codes to be written.

- Your codes must be written in standard C language and compiled by the `gcc` compiler (use the `-O3` optimization flag).
- 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 codes must show how the code should be compiled. The codes must be verified on the Linux machines in Phillips 314.
- Codes must be saved in separate files named
  `your_net_id_hw2_openmp_sort_column.c`,
  `your_net_id_hw2_openmp_sort_block.c`,
  respectively.
- Your codes must be described in a file `your_net_id_hw2_codes.pdf`. For clarity and to save space you can refer to relevant lines in the codes.
- Your findings and discussion should be described in a file `your_net_id_hw2_writeup.pdf`
- All files need to be archived with the `tar` or `gzip` facilities. The packed file must have the name `your_net_id_hw2_2.suffix` where `suffix` is either `tar` or `zip`.

  Please do not include any unnecesary subdirectories in your archive.
- Submit your work on Blackboard
- For OpenMP directives consult

  `https://computing.llnl.gov/tutorials/openMP/`

  or any other convenient source. Try to use only the simplest OpenMP constructs.
- If you relay on resources outside lecture notes but publically available, you need to cite sources in your write-up.

4. Benchmarking.

The assignment is about finding fast methods for

- finding a maximal element in a 1D array
- finding a maximal in absolute value elements element in a 2D array
- finding the best data decomposition amomg
  - block row or block column
  - cyclic by row or cyclic by column
  - tiled

so data movements can be accomplished efficiently and workload distributed evenly.

- decide for what sizes of `A` creating multiple threads is beneficial (you may want to vary the number of threads within the code so the execution is as fast as possible)

Parameters:

(a) To assess the quality of your codes you need to banchmark your code for speed-ups ove sequential execution over a range of threads from 2 to twice the number of cores of the systems you will use. (Do not create a single thread but rather write a sequential code.) Do not increase the number of threads if it results in slowdown (you need to provide an evidence of slowdown).

(b) It may be beneficial to decrease the number of threads when your submatrices become smaller and smaller.

(c) Please banchmark your code over a range of dimension of the matrix `A`. Start with $m = n = 256$, Then

- Increase m by a factor of 2 to at least $m = 2^{10}$.
- Increase n by a factor of 2 to at least $n = 2^{10}$.
- Increase both m and n by a factor of 2 to at least $m = n = 2^{10}$. (You may want to test your code on a small matrix, say $n = 8$, to check for corretness of your code.)

(d) You are asked to graph and tabulate your results and discuss your findings in the file `your_net_id_hw2_writeup.pdf`

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.

5. Remarks.

(i) To generate `A` use standard `gcc` number generator `rand()` but set the seed first to the last digit of your net id.

(ii) Your code will consist of parallel, sequential and mixed sections. You may want to time sections to see which ones take most time and then work on speeding them up.

6. Timing C codes.

You will need to use a fine resolution clock to time your codes. You can find examples how to do it at

`https://www.cs.rutgers.edu/~pxk/416/notes/c-tutorials/gettime.html`

Check also

`http://linux.die.net/man/3/clock_gettime`