**מחשוב מקבילי ומבוזר**

תרגיל #4

**The purpose of this exercise is to practice the MPI Cartesian Topology, Scatter and Gather and to write a parallel program to implement Shearsort Algorithm**

Structure **Pixel** is defined as follows:

struct Pixel {

int id;

int x;

int y;

float rgb[3];

};

The given file of **pixel.txt** contains a set of few such structures. The first line of this file contains **N** - a number of pixels to sort. Following **N** lines contains data of structures Pixel, each structure in one line, values of each members of structure are separated by blanks:

**id x y rgb[0] rgb[1] rgb[2]**

For example,

**16**

**0 1000 146 0.1 0.13 0.9**

**1 450 55 0.3 0.12 0.61**

**2 -12 500 0. 0. 0.**

**3 107 -203 0.9 0.1 0.45**

**…**

**15 -12 600 0. 0. 0.**

The purpose of the program is to write a parallel program that sorts these structures according to specified criteria:

If both pixels are black (**rgb[] = {0.0, 0.0, 0.0}**) – compare distances from the pixel to the coordinate origin **(0, 0).**

Otherwise compare their intensity - sum of elements of arrays **rgb[]** of the pixels.

**Requirements**:

* One of the processes reads all pixels from the file. This process will display the pixels before and after the sort. This process uses **Scatter** to send pixels to other processes, **Gather** to collect results after sort.
* Use Cartesian Topology to define "position" of the processes during sorting.
* At the end of the program the pixels are displayed in descending order.
* Use **Odd Even Sort** to sort rows and columns
* Suppose that you have **n2** processes to sort **n2** pixels and **n** is an even number.

**Note:** You need **(2log(n)+1)** row/column phases to sort  **n2** pixels.

**Grading Policy**:

* **10 points** for code quality:
  1. The code has to be divided into small functions (not more than 40 lines of code).
  2. Use meaningful names for variables, functions, files, constants.
  3. Place enough comments to understand the code
  4. No unused lines of code. Don't repeat the code – use functions!
  5. Write README.TXT file if special instructions are needed to run the solution. The file has to be in the root folder of the solution.
* **90 points** – for proper implementation of the requirements.
* The Homework has to be delivered in time. No delay will be accepted.

**Important:**

The homework may be performed in pairs. Only one member of pair submits the solution through the Moodle. The whole project must be zipped and named as

**111111111\_222222222.zip**

where **111111111** is ID of the one student and **222222222** is ID of another student

בהצלחה