# CSE 102 Programming Assignment 7

## DUE

December 29, 2024, 23:55

## Description

- This is an individual assignment. Please do not collaborate
- If you think that this document does not clearly describes the assignment, ask questions before its too late.

You are going to write a complete C program which implements the following functionality:

- your program reads:
  - `input.txt`
- your program creates:
  - `output.txt`

### `input.txt`

- Holds data for `N x M` grid
- `N` and `M` are between `1` and `1024`
- The first line is `N` and `M` values separated by whitespace.
- Each cell in a grid is either `1` or `0`
- The cell data is separated with whitespace.
- Example:
- A `7x10` grid of values is given.

```
7 10
1 1 1 0 0 0 0 1 1 0
1 1 1 0 0 0 0 1 0 0
1 1 0 0 0 0 0 1 0 0
0 0 0 0 1 1 1 0 0 0
0 0 0 1 1 1 1 0 0 0
0 0 0 1 1 1 0 0 0 0
0 0 0 1 1 1 1 0 0 0
```

### `output.txt`

- Holds data for the same size grid given in `input.txt`
- Each cell in a grid is an integer between `1` and `128`
- The cell data is separated with whitespace.
- Example:
- A `7x10` grid of values.
- Each different value in the output is about a group of cells.

```
1 1 1 0 0 0 0 2 2 0
1 1 1 0 0 0 0 2 0 0
1 1 0 0 0 0 0 2 0 0
0 0 0 0 3 3 3 0 0 0
0 0 0 3 3 3 3 0 0 0
0 0 0 3 3 3 0 0 0 0
0 0 0 3 3 3 3 0 0 0
```

### `Group` of cells

- A `Group` is a connected group of cells.

### `Connected` cells

- Given two cells `Grid[y1][x1]` and `Grid[y2][x2]`:
- These are `connected` if `(|y1-y2| == 1 and x1==x2)` or `(|x1-x2| == 1 and y1==y2)`

## Find groups in the grid

- The connected cells (group) will be labeled with the same number.
- Every group will have a different number.
- Find different groups in the grid and label each cell in the same group with the same number.
- There can be at most `127` groups.

## BONUS (+100pts)

- Learn about PPM file format. (It is a text-based image format)
- Implement file read/write to support `.ppm` files.
- `input.ppm`:
  - image with black and white pixels.
  - black pixels are `0`s. white pixels are `1`s
- `output.ppm`:
  - Each group of pixel is a different color.

## Remarks

- **You have to dynamically allocate the space for grid data**.
- Don't forget to `free` the dynamic allocation.
- Write comments in your code.
- If your code does not compile you will get `0`
- Do not share your code with your classmates.
- Do not print anything other than the expected output.
- You cannot use anything which is not covered in class.
- Do not submit any of the files you used for testing.
- One possible struct to represent a gate can be as follows:

## Turn in:

- Source code of a complete C program. Name of the file should be in this format: `<full_name>_PA7.c`.
- Example: `hoshen_kopelman_PA7.c`. Please do not use any Turkish special characters.
- You don't need to use an IDE for this assignment. Your code will be compiled and run in a command window.
- Your code will be compiled and tested on a Linux machine(Ubuntu). GCC will be used.
- Make sure you don't get compile errors when you issue this command : `gcc <full_name>_PA7.c`.
- A script will be used in order to check the correctness of your results. So, be careful not to violate the expected output format.
- Provide comments unless you are not interested in partial credit. (If I cannot easily understand your design, you may loose points.)
- You may not get full credit if your implementation contradicts with the statements in this document.

## Late Submission

- Late submission is NOT accepted.

## Grading (Tentative)

- `Max Grade : 100`.

All of the followings are possible deductions from `Max Grade`.

- No submission: `-100`.
- Compile errors: `-100`.
- Irrelevant code: `-100`.
- Major parts are missing: `-100`.
- Unnecessarily long code: `-30`.
- Using language elements and libraries which are not allowed: `-100`.
- Not caring about the structure and efficiency: `-100`. (avoid using hard-coded values).
- Significant number of compiler warnings: `-10`.
- Not commented enough: `-10`. (Comments are in English).
- Source code encoding is not `UTF-8` and characters are not properly displayed: `-5`. (You can use 'Visual Studio Code', 'Sublime Text', 'Atom' etc... Check the character encoding of your text editor and set it to `UTF-8` ).
- Fails at reading `circuit.txt`: `-150`.
- Fails at reading `input.txt`: `-150`.
- The result is wrong: `-100`.
- Output format is wrong: `-30`. (Be careful with spacing)
- Infinite loop or recursion: `-150`.
- Prints anything extra: `-30`.
- Unwanted chars and spaces in the output: `-30`.
- Submission includes files other than the expected: `-10`.
- Submission does not follow the file naming convention: `-10`.
- Sharing or inheriting code: `-400`.
- IF YOU DON'T FOLLOW THE FILE NAMING CONVENTIONS YOU WILL GET `0`.

Note: Some of these items are not independent. So, you cannot expect isolation of many of them. For example, if you cannot read input file correctly, you will fail to produce the correct output file. Partial grading is not guaranteed.