# COMP 201 - Fall 2021

# Assignment-3:
# The Game of Life

# The Game of Life

A cellular automaton devised by the British mathematician John Horton Conway (1937–2020) in 1970.

- Best-known example of a cellular automaton.
- Zero-player game -> its evolution is determined by its initial state
- One interacts with the Game of Life by creating an initial configuration and observing how it evolves.

# The Game of Life

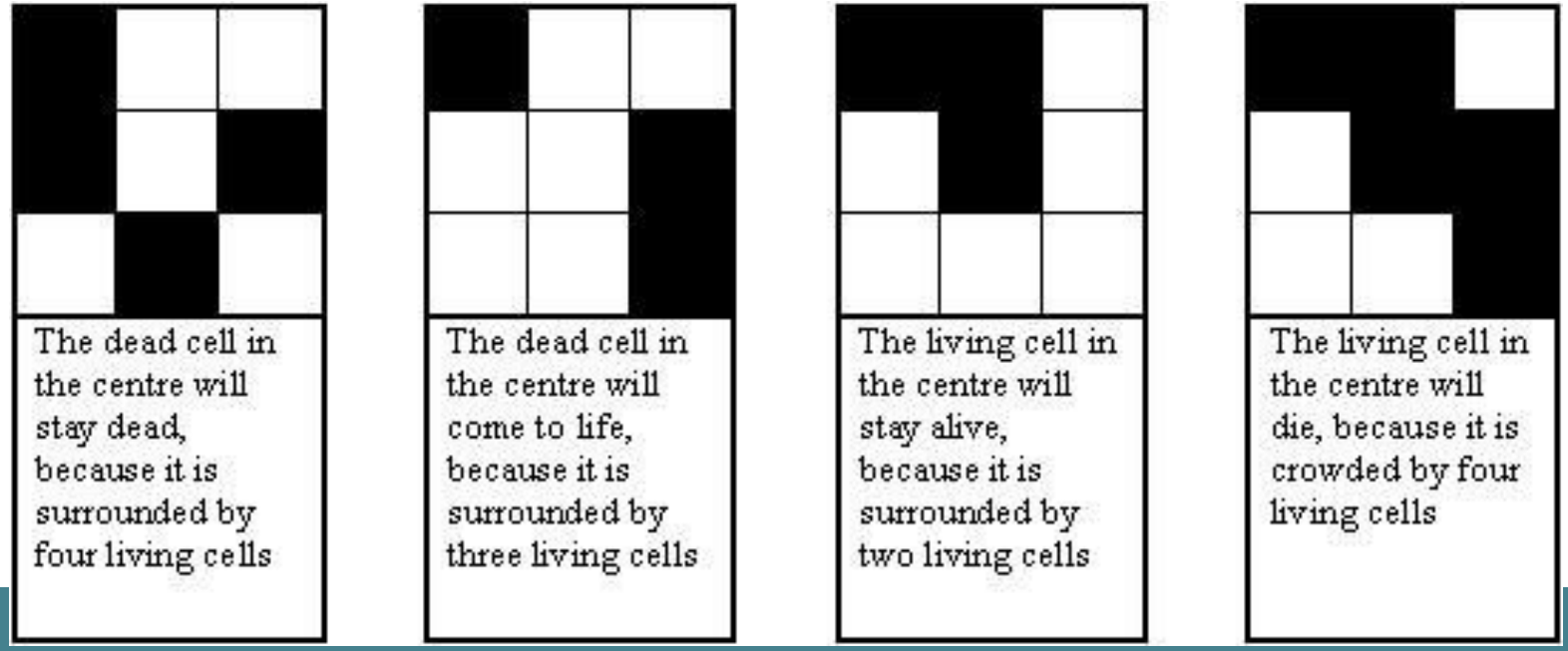
- The universe is an infinite, two-dimensional orthogonal grid of square cells
- Two possible states: live (X) or dead.
- Every cell interacts with its eight neighbours, horizontally, vertically, or diagonally adjacent. (A square on the border of the grid has fewer than eight neighbors)

In the following example, the red colored middle location has four "live" neighbors:

| | | |
|---|---|---|
| | X | |
| X | | |
| X | X | |

At each step in time, the following transitions occur:

- Any live cell with fewer than two live neighbours dies, as if by needs caused by under
- population.
- Any live cell with more than three live neighbours dies, as if by overcrowding.
- Any live cell with two or three live neighbours lives, unchanged, to the next generation.
- Any dead cell with exactly three live neighbours cells will come to life.



To illustrate these two patterns below should alternate forever (*a blinker*):

| | | |
|---|---|---|
| X | X | X |
| | | |

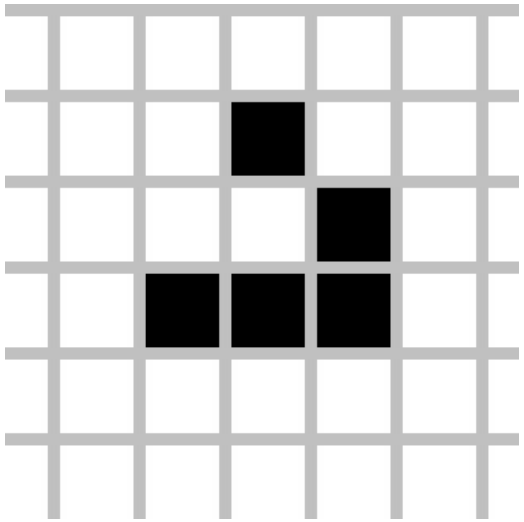
| | | |
|---|---|---|
| | X | |
| | X | |
| | X | |

There are also stationary patterns called still lifes which do not change from a generation to the next. An example still life, *block*:

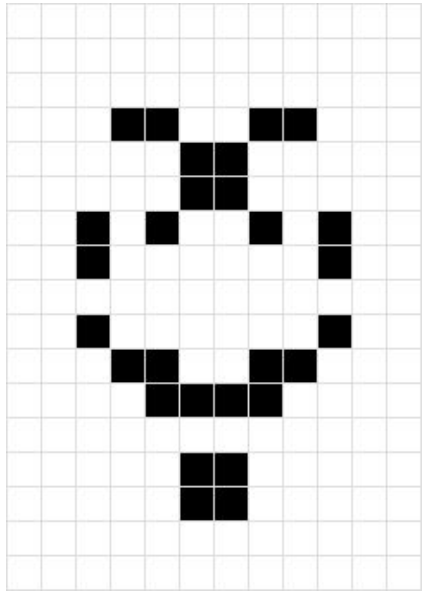
| | | | |
|---|---|---|---|
| | | | |
| | x | x | |
| | x | x | |
| | | | |

# Some Patterns

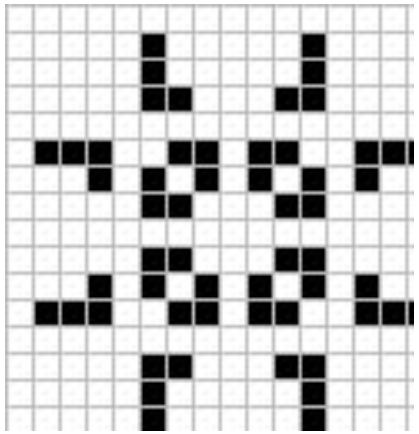
Glider Gun



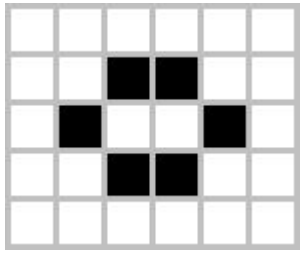
Glider



Spaceship



Oscillator



Still life

# Programming Task

## 1. How to Start:

- Accept the GitHub Classroom assignment using the link:

  https://classroom.github.com/a/MQI-5jvG

- Clone the GitHub repository created for you to a Linux machine in which you plan to do your work (Using Linux servers [**linuxpool.ku.edu.tr**].

$ git clone https://github.com/COMP201-Fall2021/assignment-3-USER.git

(Replace USER with your GitHub username that you accept the assignment).

# Programming Task

## 2. Your Task:

You are going to implement a version of the game of life. The checkerboard in Conway's game of life can be represented as a two-dimensional matrix. Given an initial pattern, the game simulates the birth and death of future generations using simple rules.

# Input Parameters:

Below you can see a sample input:

```
1    // number of iterations
8    // number of rows
10   // number of columns
- - - x x - - - - -
- - - - x - - - - -
- - - - - - - - - -
- - - - - - - - - -
- - - x x - - - - -
- - x x - - - - - -
- - - - - x - - - -
- - - - x - - - - -
```

# Output Parameters:

Below you can see a sample input:

```
3   // dead count
4   // born count

-  -  -  x  x  -  -  -  -  -
-  -  -  x  x  -  -  -  -  -
-  -  -  -  -  -  -  -  -  -
-  -  -  -  -  -  -  -  -  -
-  -  x  x  x  -  -  -  -  -
-  -  x  x  -  -  -  -  -  -
-  -  -  x  x  -  -  -  -  -
-  -  -  -  -  -  -  -  -  -
```

# How to run:

Your could should take two command line arguments:

- input file path
- output folder path in which you should generate output files

```
$ ./game_of_life diamond/diamond.txt output/
```

# Bonus Task:

There are some initial patterns called "*oscillators*" which returns to its original state, in the same orientation and position, after a finite number of generations. The bonus task is to implement a boolean function, *IsOscillator(),* given the initial pattern and maximum number of steps to check, determines whether the pattern is an oscillating object.

**Hint:** You can simulate the game for given maximum steps and collect the results in a 3D dynamically allocated array structure. Using this "game history", it is possible to check for duplicate game states.

# Evaluation

You are not allowed to use **static arrays**. Your implementation must utilize **dynamic memory allocation**.

Your score will be computed out of a maximum of 100 points based on the following distribution:

- **80** Correctness points
- **10** Effective use of version control points
- **10** Style points
- **20** Bonus task

# Evaluation

- Correctness points:

  Your code will be evaluated based on a set of given input parameters and should result in correct numbers for:

  1. Death count at each step
  2. Born count at each step.
  3. *.txt files that demonstrate each step's status using '-' and 'X' characters.

# Evaluation

- Effective use of version control points:

  You are required to push your changes to the repository frequently. If you only push the final version, even if it is implemented 100% correctly, you will lose a fraction of the grade because you are expected to learn to use Version Control Systems effectively. You do not have to push every small piece of change to GitHub, but every meaningful change should be pushed. For example, each of the functions coded and tested can be one commit. **For each function, there should be at least one commit (with proper commit message) that includes just modifications on that function.**

# Evaluation

- Style points:

  Finally, we've reserved 10 points for a subjective evaluation of the style of your solutions and your commenting. Your solutions should be as clean and straightforward as possible. Your comments should be informative, but they need not be extensive.

# Late Submission Policy

**You may use up to 7 grace days (in total) over the course of the semester for the assignments.**

That is, you can submit your solutions without any penalty if you have free grace days left. Any additional unapproved late submission will be punished (1 day late: 20 % off, 2 days late: 40 % off) and **no submission after 2 days will be accepted.**

# Resources

- https://playgameoflife.com/
  - Check Lexicon for interesting patterns.
- Digital clock in Game of life
  - Game of Life is Turing complete. Anything that can be computed algorithmically can be computed within the Game of Life.

# THANK YOU