**Game of Life – 2D Array Project**

Overview – In this project, you’ll be creating a program called **GameOfLife**, which allows the user to run a console based population growth simulation using a 2D array. Here are the project details:

* The user is given the choice to choose the dimensions of the game board.
* The user is also given the ability to choose how many living organisms he/she wants to start with.
* The user also gets to place these organisms anywhere on the board following these rules:
  + The location specified cannot already contain an organism.
  + The location must be a valid location on the board.
  + Invalid input will prompt the user to re-enter valid data.
* Finally, the user is also given the ability to choose how many turns the simulation will run.

On each turn, each location on the board will follow rules to determine if it dies, spawns a new organism, or stays the same. There are four rules:

1. A living organism with less than 2 adjacent organisms will die (due to under population)
2. A living organism with 2-3 adjacent organisms lives on to the next turn.
3. A living organism with more than 3 adjacent organisms dies (due to overpopulation)
4. An empty cell with exactly 3 adjacent organisms becomes a living organism.

Your game will be expressed in the console. You can pick the format, but it may look something like:

|  |  |  |  |
| --- | --- | --- | --- |
| . | . | . | . |
| . | 0 | 0 | . |
| 0 | . | . | . |
| . | 0 | . | 0 |

where periods (.) represents an empty cell and zeroes (0) represents a living organism.

Sample simulation:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 |  | 0 |  |  | 0 |  |  | 0 | 0 | 0 |  | 0 |  | 0 |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 0 |  |  | 0 | 0 | 0 |  | 0 | 0 | 0 |  |  |  |  |  | 0 |  | 0 |  |  | 0 |  |  |  |  |  |
|  | 0 |  |  |  |  |  |  |  | 0 |  |  | 0 | 0 | 0 |  |  | 0 |  |  |  | 0 |  |  |  |  |  |

Start Turn 1 Turn 2 Turn 3 Turn 4 Turn 5 Turn 6 - Death

**Turn Specifics**

* Each turn your program starts by outputting the Turn number. (See sample run on page 2)
* Following the turn number, the new state of the board should be calculated and outputted to the console. (Yours will appear vertically, not side by side as shown in the example)
* Your program will continue to run until one of the following happens:
  + The number of turns specified by the user is reached.
  + The board produces no living organisms. It will show one final board with no organisms before stopping.
  + The board state didn’t change from the last turn. It will show 2 consecutive identical boards before stopping the simulation.

An appropriate message should be displayed as to the reason the simulation ended:

* Completed all turns
* Death! No living organisms remain.
* Still life (unchanging) board state achieved.

Sample Run:

Enter number of rows: 4

Enter number of columns: 4

Enter number of starting organisms: 5

How many turns would you like your simulation to go? 6

Placing organism # 1

Enter row and column

2 2

Placing organism # 2

Enter row and column

2 2

Organism already placed there.

Placing organism # 2

Enter row and column

3 2

Placing organism # 3

Enter row and column

1 5

Invalid column value. Try again.

Placing organism # 3

Enter row and column

2 1

Placing organism # 4

Enter row and column

3 1

Placing organism # 5

Enter row and column

2 0

Turn 1

#####

....

.0..

0.0.

0.0.

Turn 2

#####

....

.0..

0.0.

....

Turn 3

#####

....

.0..

.0..

....

Turn 4

#####

....

....

....

....

Start:

#####

....

....

000.

.00.

Death! No living organisms remain.

\*Again, yours will not appear side by side, but rather vertically, one turn after another.

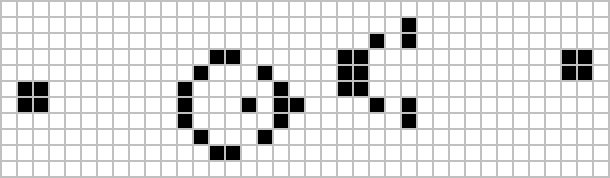
\*Note - Your program should allow the user to enter all of the organisms’ locations in a list of integers. In the previous example, above, when prompted for the first organism’s row and column, we could enter all 5 organisms’ info. (minus the invalid inputs, of course). My test cases provide this so you don’t have to key in all the values.

2 2 3 2 2 1 3 1 2 0

The different patterns that could results include:

|  |  |  |
| --- | --- | --- |
| Name | Description | Example |
| Still Life | The pattern doesn’t change |  |
| Oscillator | The image will repeat itself over a consistent period of turns. The patterns to the right will alternate. (period of 2). Some oscillators have been found to repeats with periods of 30 or more. |  |
| Spaceship | The image will repeat itself, like an oscillator, but will move across the board while doing so. |  |
| Eventual oscillator  Eventual still life | Some patterns will alter the original pattern before becoming a still life or an oscillator. |  |
| Death | A pattern that results in the eventual death of all organisms, though it may take many turns before this happens. | See example on Page 1 |

Conway originally conjectured that no pattern can grow indefinitely. He offered $50 to anyone that found a pattern that proved him wrong. An MIT student named Bill Gosper took Conway’s money with the “Gosper Glider Gun”. It is essentially a spaceship that leaves behind debris as it move forever in a specific direction.



Here are some test inputs for your program:

**Test # 1** – Death pattern after 12 turns

Rows: 5 Columns: 5 Organisms: 13 Turns: 20

Locations: 0 0 0 1 0 2 0 4 1 0 2 3 2 4 3 1 3 2 3 4 4 0 4 2 4 4

**Test # 2** – 15 period oscillator

Rows: 18 Columns: 11 Organisms: 18 Turns: 30

Locations: 3 5 4 4 4 5 4 6 5 3 5 4 5 5 5 6 5 7 12 3 12 4 12 5 12 6 12 7 13 4 13 5 13 6 14 5

**Test # 3** – 2 period oscillator develops starting in turn 6

Rows: 9 Columns: 9 Organisms: 5 Turns: 9

Locations: 3 4 4 3 4 4 4 5 5 4

**Test # 4** – Spaceship that moves down and left with 4 period oscillation

Rows: 20 Columns: 20 Organisms: 5 Turns: 68

Locations: 0 1 1 2 2 0 2 1 2 2

Verification Checks for Tests

**Test # 1 Test # 2**

Turn # 30

#########

...........

...........

...........

.....0.....

....000....

...00000...

...........

...........

...........

...........

...........

...........

...00000...

....000....

.....0.....

...........

...........

...........

Turn # 29

#########

...........

...........

...........

...........

....000....

...0...0...

..0.....0..

...........

.0.......0.

.0.......0.

...........

..0.....0..

...0...0...

....000....

...........

...........

...........

...........

Turn # 12

#########

.....

.....

.....

.....

.....

Turn # 11

#########

...0.

...0.

.....

.....

.....

Death! No living organisms remain.

Completed all turns!

**Test # 3 Test # 4**

Turn # 68

#########

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

..................0.

...................0

.................000

Starting state:

#############

.0..................

..0.................

000.................

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

....................

Turn # 9

#########

....0....

....0....

....0....

.........

000...000

.........

....0....

....0....

....0....

Turn # 8

#########

.........

...000...

.........

.0.....0.

.0.....0.

.0.....0.

.........

...000...

.........

Completed all turns!

**Extra Challenges**

1. All your pattern to “wrap”. In other words, the first column is considered adjacent to the last column (and vice-versa). Same with your rows. This one makes your spaceship never end.
2. Instead of cropping your board, add to it as your organisms demand. This will requiring rebuilding the array as a new size. This one is tough!