Data Structures
Tiveron
Game of Life

Assignment adapted from: ***https://www2.cs.arizona.edu/~mercer/Projects/GameOfLife.pdf***

Simulation: **https://bitstorm.org/gameoflife/**

The Game of Life was invented by John Conway to simulate the birth and death of cells in a society. The following rules govern the birth and/or death of cells between two consecutive time periods. At time T + 1

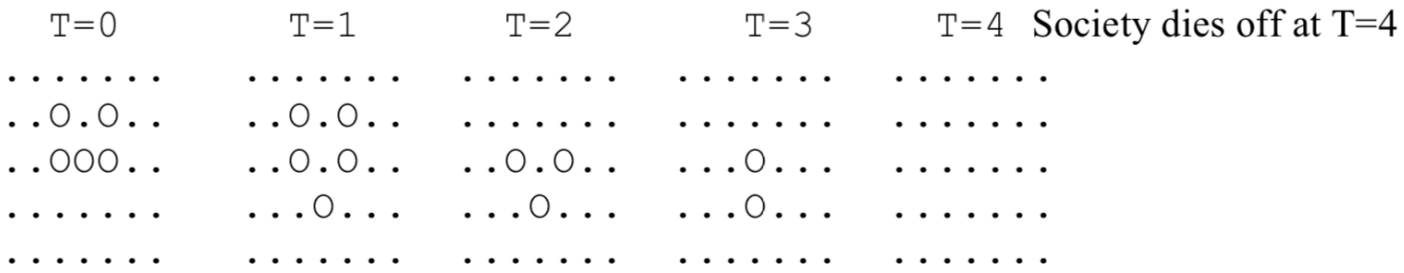
- A cell is born if the cell had exactly three alive neighbors at time T
- An existing cell remains alive if at time T it has exactly two or three neighbors
- A cell dies from isolation if at time T there were fewer than two neighbors
- A cell dies from overpopulation if a time T there were more than three neighbors

A neighborhood consists of the eight elements around any element (cell).

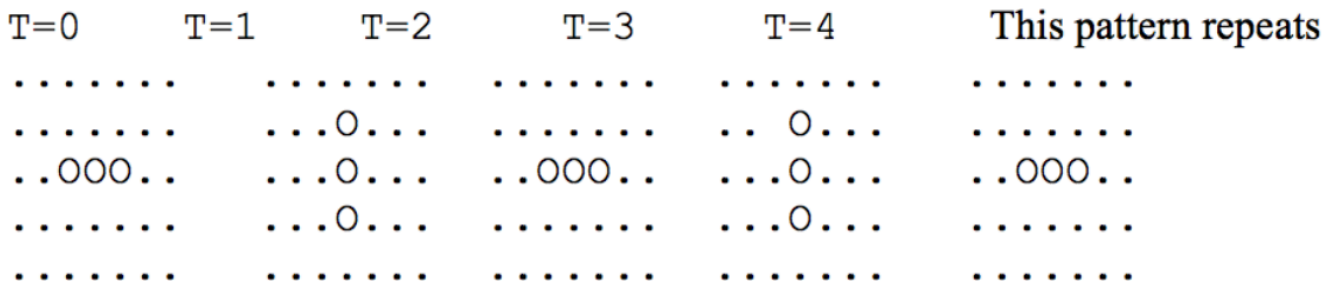
N N N
N **C** N
N N N

***C** represents a given cell*
**N represents one neighbor*

The neighborhood can extend to the other side of society. For example, a location in the first row has a neighborhood that includes three locations in the last row. The following patterns would occur when T ranges from 1 to 5, with the initial society shown at T = 0. O represents a live cell a black indicates that no cell exists at the particular location in the society.



Other societies may stabilize like this:



Description

Your task is to implement `class GameOfLife` along with unit test `class GameOfLifeTest`. Once you have completed both of these classes you will be given a GUI class to run your simulation (or you can make your own!). For GameOfLifeTest make sure to write careful assertions, especially for wraparound and corners. For each of the eight required methods (except toString()). Please follow this recommended approach for developing well-tested software:

1. Add a test method first with calls to non-existent methods
2. Make the test compile (add the method heading and return some dummy return value if necessary)
3. See the test fail (get the red bar)
4. Write the code necessary to make your tests pass
5. Revise your code to make it more efficient

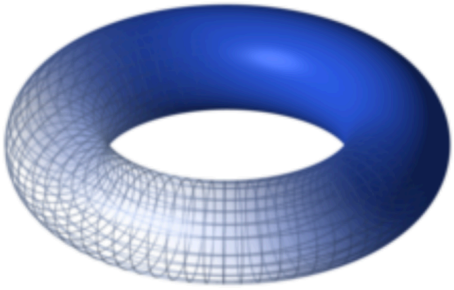
Project

1. Download the *GameOfLife* starter project from canvas.
2. Spend some time reviewing the provided class subs and tester methods.
3. Your task is to implement each of the 8 methods provided and create tester methods for each of them. Please implement one method at a time and then write tests for each method to make sure each works properly. Some test methods have been provided to help you get started.
4. You should have a large number of `@Test` methods (more than 10 for `neighborCount` and `update`.
5. Note for neighborCount:

The O has 8 neighbors labeled a..h. Wraparound is needed for d..h. The labels are repeated to show where they need to be checked.

	f	g	h			
e	O	a				e
d	c	b				d
	g	h				f

Try to imagine the cells covering a torus:



6. Once you have completed all of the methods and testers have your instructor check in your project. You will then be given a GUI class to run your simulation.