General Coding Exercise

Description:

You will be building a 2-D cellular automata: a grid of “cells” that exhibits behavior over the course of iterative generations. Each generation is computed based on the array of cells in the previous generation. If you are familiar with Conway’s Game of Life, this is a very similar concept but with a different ruleset.

We want to see how you think about programming and problem solving, independently from any particular framework, language, or development environment. So, this exercise requires no libraries, does not interact with the web or a database, and can be done in any language you like.

Evaluation Criteria (in order):

1. Code readability, cleanliness and architecture (are you organizing your code in a logical, readable, and maintainable format)
2. Solution Completeness (you found the right answer)

Process:

* Write in any language you like
* When you begin, first create a git repository with an initial commit
* Then commit incrementally as you complete the assignment
* Your project directory should include
* A README.md with instructions for executing the code
* A REVISIONS.md file. It should answer the following questions:
* Obviously, this was done under time constraints that may have forced you
* down a path or pattern that ended up with some code that is non ideal.
* Are there any sections that you would have done differently with hindsight? Anything that you'd like to refactor given more time?
* You have 24 hours (timeboxed) to finish all of the code (not counting packaging and REVISIONS.md write up)

Submission:

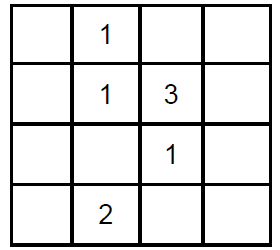
* Please submit everything bundled together in a zip file. To avoid security issues with gmail please upload the zip to a google drive or dropbox and share the link.
* Submit back to recruiter, or sender’s email
* To recap, the submission bundle should include **code**, a **README.md file detailing execution**, and a **REVISIONS.md file detailing any updates you would make with more time**.

Project Specifications:

A cellular automata is a 2D grid of cells. Cells may be either empty or alive. You may represent

empty cells with a zero or null value. If cells are alive, they have an “age”: an integer value that is 1 (newborn), 2 (adult), or 3 (senior).

Below is an example 4 x 4 grid containing three newborns, one adult, and one senior.



Neighbors

The “neighbors” of a cell are the eight cells immediately surrounding it orthogonally or

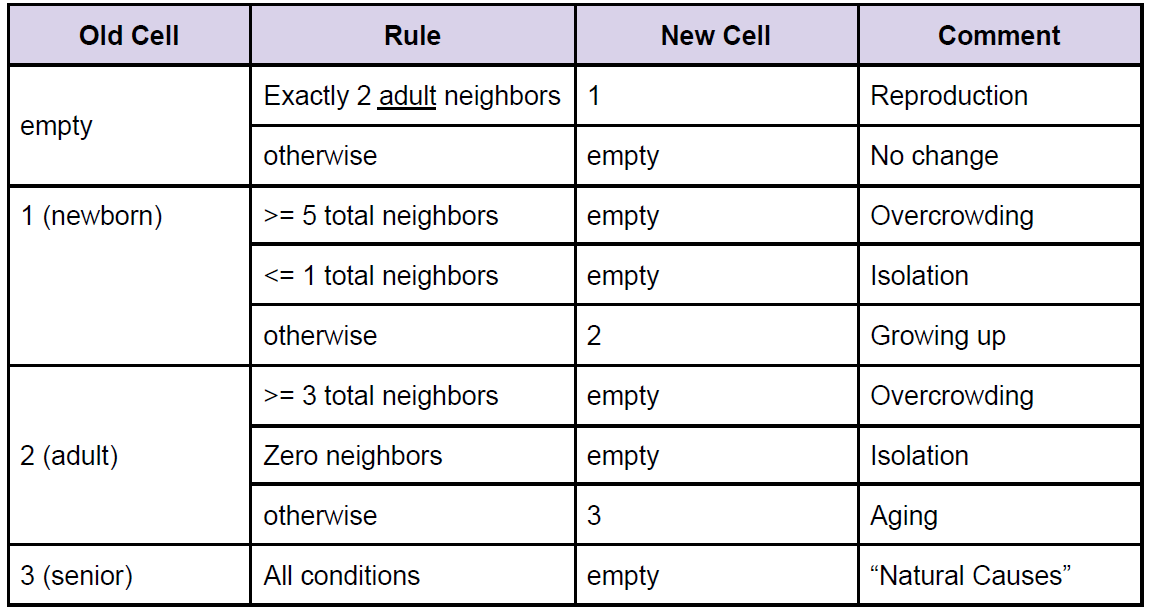
diagonally. For example, the senior in the example above has three neighbors, all of them

newborns. The adult in the example above has only one neighbor, also a newborn.

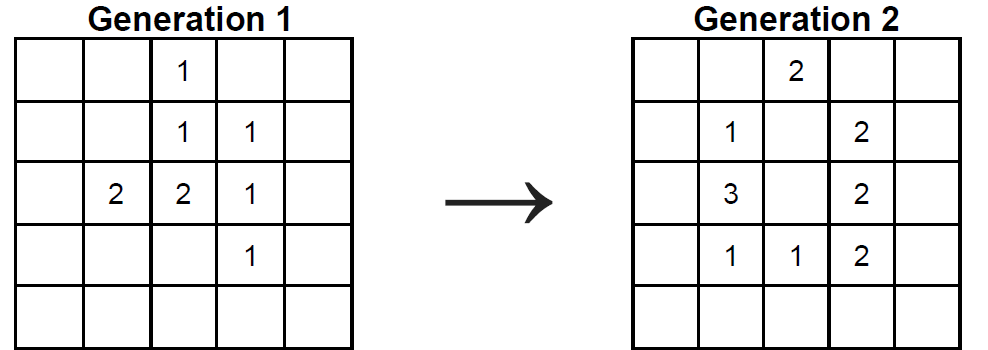
Note that edge cells have only 5 potential neighbors, and corners have only 3.

Ruleset

In each generation, compute the new value for the grid by the following rules:

****

Generation Example

****

● Three newborns are born near the pair of adults at (1,2) and (2,2).

● One of the adults has 2 neighbors and ages. The other has 5 and dies of overcrowding.

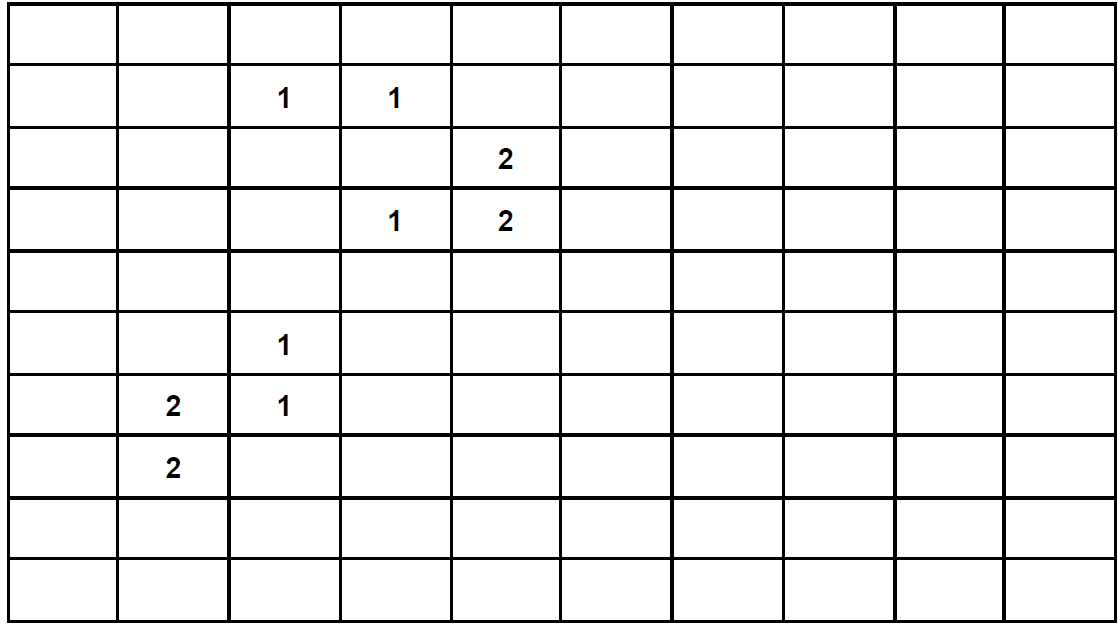
● The newborn at (2,1) dies of overcrowding

● All the other newborns grow up to become adults.

Question

If the following 10x10 grid is Generation 1, what is Generation 20? Provide your answer in any

format you like so long as it’s readable.



Datasets

The attached datasets.md file provides the initial data you can copy/paste into your code. Additionally, it contains three test cases with first generation results to validate your solution.