Object-Oriented Programming
Assignment 2 – Game of Life

March 28, 2017

Objectives

Practice and get familiar with arrays and classes in C++ language. In this assignment you will make use of the subject matters about Arrays (ch. 5), Classes (ch. 6), and Constructors and Other Tools (ch. 7).

Problem Description

Write a class `GameOfLife` for Conway's "Game of Life"[1].

- LIFE is an organism that lives in a discrete, two-dimensional world. While this world is actually unlimited, we don't have that luxury, so we restrict *by default* the array to 80 characters wide by 23 character positions high.
- This world is an array with each *cell* capable of holding one LIFE cell. Generations mark the passing of time. Each generation brings births and deaths to the LIFE community. The births and deaths follow this set of rules:
  1. We define each cell to have eight neighbor cells. The neighbors of a cell are the cells directly above, below, to the right, to the left, diagonally above to the right and left, and diagonally below, to the right and left.
  2. If an occupied cell has zero or one neighbor, it dies of loneliness. If an occupied cell has more than three neighbors, it dies of overcrowding.
  3. If an empty cell has exactly three occupied neighbor cells, there is a birth of a new cell to replace the empty cell.
  4. Births and deaths are instantaneous and occur at the changes of generation. A cell dying for whatever reason may help cause birth, but a newborn cell cannot resurrect a cell that is dying, nor will a cell's death prevent the death of another, say, by reducing the local population.
- The world initializes with one of the four patterns: (1) glider; (2) lightweight spaceship; (3) pulsar; (4) random, where the input pattern number is the percentage of live cells around the 80×23 area.

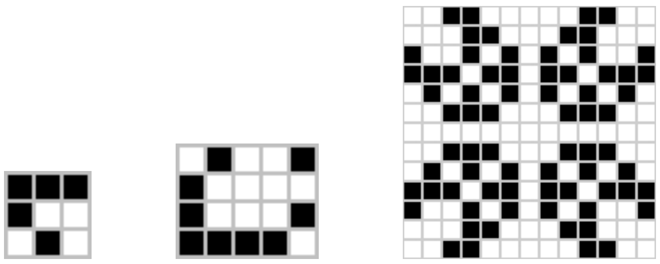
(a) Glider (b) Lightweight Spaceship (c) Pulsar

Figure 1: Initialization patterns

[1]See program project 12 of Chap 5 for more details.

Requirements

In class `GameOfLife`, you MUST write

1. Constructors for indicating width $w \in \{16, \dots, 256\}$ and height $h \in \{16, \dots, 256\}$ of the map.
    - Default constructor: $w = 80$, $h = 23$.
    - Constructor for two parameters (w, h).
2. A *private* two-dimensional array `cell` with two possible states (*live*, *dead*)
3. Public member functions
    (a) `initialize(int p)`: The value $p \in \{1, 2, 3, \dots, 100\}$ denotes initialization pattern (1: glide; 2: lightweight spaceship, 3: pulsar, 4–100: percentage of live cells for random initialization). The patterns 1–3 should be located in the center of map.
    (b) `proceed(int t)`: Proceed t generations (default: 1).
    (c) `display()`: Display the world.
4. Some private member functions
    (a) At least one uses 2-dimensional array parameter (PFA).
    (b) Any other you need.
5. No public member variables.
6. Remember to describe pre/post-conditions and make your class and functions robust!
7. Write in `main()` to test your class with all the four patterns in 100 generations.

Evaluation

- Correctness: 90%
- Styling: 10%

Submission

- **Due: 2017/04/10 (degrade by 10 points for each day delay)**
- Source code (*.cpp)
    - Show your information (Name, Student ID, Dept, Year) as comments in the beginning of your code.
    - Name your file as "Hw2_(#Student ID).cpp".
    - Upload the file to eCourse.