CS111 Project Description

With special thanks to Frank McCown @ the Harding University

In this project you will be using the knowledge you gained from lectures and homeworks and applying them to a lengthier programming assignment. The answers to some of your questions do not strictly reside in this document. You are encouraged to look at other sources for inspiration. This semester's project will involve a biological simulation. This is an *individual* project, and as such you will be held to the University's academic integrity policies. You are welcome to discuss strategies and ideas with other students (and will be encouraged to in recitation), ***but you must code the project on your own***.

The project will be divided into Weekly Milestones. You will have deliverables due every week as described by the milestone document. Below is a general overview of the project, as well as Milestone 1.

This project is what you make of it. There are points assigned to each milestone which are further broken down into tasks. However, you should not be limited by what we suggest the minimum work required here is. To get the most out of this project, I urge you all to push yourselves to find your limits. You might surprise yourself! Good luck!

Description

You have just discovered a new type of cell in a species of fish found in the deep waters of the Mariana Trench. Under the microscope, you notice that in some specimens, the fish's cells are in a constant state of flux, and in others, cell movement has been stabilized. After consultation with many of your colleagues, you come to the conclusion that cell movement is governed by a very specific set of criteria, which appear below. You decide to build a computer simulation of cell movement. This project will guide you through the process of doing so.

You discover that cell movement is based on the satisfaction of individual cells in a tissue sample. For now, assume there are only two types of cells: X cells and O cells. In computer science parlance, we will refer to each type of cell as an *agent type,* and individual cells as *agents*.

Two populations of the two agent types are initially placed into random locations of a tissue sample represented by a grid. After placing all the agents in the grid, each cell is either occupied by an agent or is empty as shown below.

Agents placed randomly in grid

X	X	O	X	O
	O	O	O	O
X	X			
X	O	X	X	X
X	O	O		O

Now we must determine if each agent is *satisfied* with its current location. A **satisfied** agent is one that is surrounded by at least *t* percent of agents that are like itself. This threshold *t* is one that will apply to all agents in the model, even though in reality everyone might have a different threshold they are satisfied with. Note that the higher the threshold, the higher the likelihood the agents will not be satisfied with their current location.

For example, if *t* = 30%, agent X is satisfied if at least 30% of its neighbors are also X. If fewer than 30% are X, then the agent is not satisfied, and it will want to change its location in the grid. For the remainder of this explanation, let's assume a threshold *t* of 30%. This means every agent is *fine with being in the minority* as long as there are at least 30% of similar agents in adjacent cells.

The picture below (left) shows a satisfied agent because 50% of X's neighbors are also X (50% > *t*). The next X (right) is not satisfied because only 25% of its neighbors are X (25% < *t*). Notice that in this example empty cells are not counted when calculating similarity.

Satisfied because 1/2 (50%) of neighbors are X

X	X	O	X	O
	O	O	O	O
X	X			
X	O	X	X	X
X	O	O		O

Dissatisfied because only 1/4 (25%) of neighbors are X

X	X	O	X	O
	O	O	O	O
X	X			
X	O	X	X	X
X	O	O		O

When an agent is not satisfied, it can be moved to any vacant location in the grid. Any algorithm can be used to choose this new location. For example, a randomly selected cell may be chosen, or the agent could move to the nearest available location.

In the image below (left), all dissatisfied agents have an asterisk next to them. The image on the right shows the new configuration after all the dissatisfied agents have been moved to unoccupied cells at random. Note that the new configuration may cause some agents which were previously satisfied to become dissatisfied!

Dissatisfied agents marked with *

X	X*	O	X*	O
	O	O	O	O
X	X			
X	O*	X	X	X
X	O	O		O*

All dissatisfied agents relocated

X		O		O
O	O	O	O	O
X	X	X		X
X		X	X	X
X	O	O	O	

All dissatisfied agents must be moved in the same *round*. After the round is complete, a new round begins, and dissatisfied agents are once again moved to new locations in the grid. These rounds continue until all agents in the neighborhood are satisfied with their location.

Movement within the tissue sample stops when all cells are satisfied.

Milestone 1 *10 pts* (Due Monday, Oct 27, 2014)

In this milestone, you will complete the design phase of your project. In a text document, describe the following items. It should be clear to the reader of your document how you intend to implement these items.

Assume you are given a tissue sample represented by a *n* x *n* grid. Each cell in the grid is addressable by a coordinate system, e.g. the upper leftmost cell can be referred to by the coordinate (0,0) and the lower rightmost can be referred by (n-1, n-1). Your simulation should continually run "rounds" moving unsatisfied cells until all cells have been satisfied.

a) Describe algorithm to determine if a cell of type X located at (i, j) is "satisfied". Determine the inputs and outputs of your implementation.

b) Describe how given a solution to (a), how you would determine if an entire tissue sample has been stabilized (i.e. every cell has been satisfied).

c) Describe an algorithm that would move an unsatisfied X cell to a vacant location. What are your inputs and outputs for your algorithm?

d) Describe the other operations you'll need to complete the simulation.

e) What is the bound on the number of rounds your algorithm will have to run? What is it dependent on? Can you make some predictions about how the inputs to your simulation will affect how long your simulation will run?