**CS 4500; HW4**

HW1, HW2, and HW3 told you what to do, and sometimes how to do it. HW4 gives you some more latitude.

You are to design and implement an experiment or experiments using your software that simulates the strange games previously described. You have many choices of exactly what to explore in your experiment, but there are clearly several different ways to go with the data you are already collecting, such as the number of turns or moves it takes to end the game, the size of the grid, the number of times a given game is executed, minimum grid cell touches, and maximum grid cell touches.

But what are you to do with these numbers? Fundamentally, I want you to explore game behavior using an independent variable (something you change multiple times) and a dependent variable (something you measure for each different value of the independent variable). Since we are dealing with pseudo-random events, use many repetitions each time you set your independent variable.

The results of this kind of experiment are almost always reported with a graph, and often with a table as well. The independent variable is measured on the x-axis, and the dependent variable on the y axis. For a very quick refresher on the idea of independent and dependent variables, and an example graph, please see <http://chemistry.about.com/od/chemistryterminology/a/What-Is-The-Difference-Between-Independent-And-Dependent-Variables.htm>.

Please note that a histogram you did is NOT the same as the dependent / independent graph you need for HW4. A histogram only displays the values of ONE variable, not two variables. Be careful that you don’t confuse these.

Feel free to get creative about HW4. Perhaps you could examine how the game changes when you have one, two, three, or four game pieces. (Be careful to document the rules your simulation enforces for these games, because the behavior of your simulation will depend heavily on the rules you make.) For example, you could fix a single grid size, and fix the number of repetitions, and run the simulation with 1, 2, 3, and 4 game pieces. Then you could make your dependent variable the average number of turns, or the perhaps the maximum number of touches. You could then change the size of the grid, rerun your experiment, and see if that gives more interesting results.

Perhaps you could make up a different rule for what happens when a piece “jumps off the board;” one possible rule would be to instead of skipping a turn (that is, staying in the same place), you could instead jump to the other side of the board, “wrapping around.” If you jump off the top of the grid, you appear at the bottom of the grid in the same column; if you jump off the right, you reappear on the left, on the same row. Does that wrap around change the average game behavior? Here you might end up having TWO dependent variables and one independent variable: the independent variable could be the length of the side of a square grid, and the two dependent variables are (1) the average number of turns to completion for the HW1 definition (skip a turn if you would fall off) and (2) the average number of turns to complete using your new wraparound scheme for piece movement. Each of those “average” points will require many simulations, so your program will be running many times.

Whatever you decide to do, it must be implemented as a Javascript program that I can run in the W3 school tutorial environment. In your documentation, and on the screen for your interactive user, start with an explanation of the experiment, and then show the results. Both your screen output AND your code documentation should give details of the experiment. The code documentation should also include information about how you implemented the game, and the people you talked to about your software. The screen output should NOT include implementation details.

Let your imagination roam a bit before you commit to a HW4 plan. Be creative, and surprise yourself and others with an interesting result. Perhaps you can fix the number of cells in a rectangular grid (say 100 cells), and experiment with different shapes by varying the width of the grid from 1 to 100. Perhaps you can implement a triangular, rectangular, pentagonal, and hexagonal grid, and see if that changes your results. The sky is the limit, so go a little crazy (in a mathematical, programmable way).