Assignment 1 – Program Design and Testing

Goals-

Review of programming with arrays

Convert requirements (i.e. the rules) to a software design

Think about testing process, before and after writing the program.

Implement Conway's *Game of Life*

You will design, implement, and test a program that runs the game of life. Conway's *Game of Life* is a standard example of a cellular automaton. This means that you have an array or matrix of cells. Each turn or generation the value of each cell may change based upon its neighbors. To get more background look online, such as: http://en.wikipedia.org/wiki/Conway's_Game_of_Life Please do the research so you understand the terms and how the rules are applied.

The rules are: Each cell has eight neighbor cells. The neighbors 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.

1. If an occupied cell has zero or one neighbor, it dies of loneliness.
2. 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.

NOTE- Think this through carefully before you start coding. If you change a cell it can impact those around it. You should determine the changes required for all cells before changing any cell!

There are significant initial questions that you must address in your design. We will limit the visible "world" to 40 x 20 cells. That's a lot of characters to type in from the keyboard so don't make the user do it. There are numerous ways to address this problem. That's why it's called design! :-) You must give the user the option to start with one of three starting shapes in the grid; a fixed oscillator, a glider, and a glider cannon.

Probably the most difficult part of the design is what you will need to do to handle the cells on the edges. Remember this is a window on an infinite grid. The patterns go on forever. You are just showing them while they are visible in this small window. You may not be able to just stop the patterns at the edge as that may change the behavior. Specifically, any pattern should not change as the object moves out of the visible grid.

You will create your design document BEFORE you start coding. The TA will grade, in part, on how well your implementation matched your design. Remember, in your reflections document you can explain how you had to change the design because your first idea, just didn't work. Just explain what

you learned. Simply stated, program design is identifying the problem to be solved, identify the inputs, specify the desired output, and then develop the algorithm(s) to convert the input into the output.

You must also describe how you tested your program. In this case you must demonstrate that the rules work for all cells. The edges will be a special case. You have 2 types of horizontal edges and two types of vertical edges. Each corner involves a different combination of these edges. Your reflections must describe how you planned to test these cases and the results of the testing.

By the due date at midnight, you will submit your program, and the reflections document, which will discuss how well your design worked, and the results of the testing. You must also provide a makefile. All the files must be submitted in a zip archive.

Grading:

- programming style and documentation (10%)
- build the array/matrix providing a fixed simple oscillator (30%)
- the display allows the user to see the change(s) in the shape(s) (10%)
- allow the user to specify the starting location of the pattern (10%)
- add the glider pattern (5%)
- the glider should simply disappear off an edge (10%)
- add a glider gun or cannon pattern (5%)
- reflections document to include the design description, test plan, test results, and comments about how you resolved problems during the assignment (20%)

IMPORTANT POINT- Keep in mind this is a programming assignment. You are not writing a program to share with others for their enjoyment. Too often students forget this and the attempt to make it entertaining which also makes it more difficult. And has little to do with the grading. You are graded on developing the design, writing the code, and testing your program. You will use text-based output. It may not be pretty, but it will demonstrate your design and implementation works. If you have other programming experience and want to use something else check with your grader before doing it.

HINTS-

1. Use the grading breakdown to plan your program. Use incremental development. Get one part working. Test it. Save a copy and continue working on the next step in your plan. This ensures you will have something to submit!
2. On the Internet you will also find many descriptions and examples of stable patterns. Definitely take advantage of that for your testing!
3. There are only 4 rules that you apply to each cell in your array. There are emergent properties so testing logic errors may be difficult. :-)
4. For cells on the edges you need to handle the case that you are displaying a subset of an infinite grid. Get your program working first, ignoring the edge condition. Once you know your code is working then implement the edge algorithm(s). One option is to have some ghost columns and rows. You can continue calculating the movement, but just not display some of the cells.
5. As one website stated, the *Game of Life* is one of the most programmed games in the world. Be very careful about borrowing any code, or ideas you see in someone else's code. As always, any code submitted must be yours and yours alone.