GAME OF LIFE NOTES:

RULES:
A) Each cell has 8 neighbors
B) If a cell has 0 or 1 neighbors, it dies
C) If a cell has > 3 neighbors, it dies
D) An empty cell with 3 neighbors births a new cell to replace the empty cell
-These changes occur at each generation change

Suggested format:
world array with 80x22
function generation: scans the world array and modifies cells, marking for births and deaths as rules are applied (Examines each cell, killing, letting live, or birthing, as needed by marking with */x or ' ')
*/x/etc for living cells
' ' blank for dead cells
function display: accepts the world array and outputs it to the screen - include a delay between this and the next instance of generation

NEEDS:
Ability to abort
Ability to start with more than one arrangement in the grid
Ability to run a different number of times

World Array and Generation function:
Build an array with [80][22] and fill the initial values with x's or ' 's to begin generating the program. Using these initial positions, apply the rules to the array and begin altering what will be the initial instance of the array. (Potentially randomize the initial ones?)
Because we are going to be looping through the program for each generation, we will need a temporary array to store the altered cells. This will then be set as the initial array for the next generation to run off of, which will be stored into a temp, and on and on for as many generations as the user runs.
To apply the rules to the cells, we need two loops (nested b.c its a 2d array) to go through each cell and then to test to see if there is in fact an * or x in it. We can then use a counter for the number of x's in surrounding cells, and if there are 0 or 1 or > 3, we kill the cell. Similarly, if our cell is blank and it has 3 around it, we birth a new cell.
To test the cells we can have a series of conditions, where if [i - + or i itself][j], or [i][j- + or j itself] etc is true (from a bool that tests for x or * in the cell), then we can calculate what the neighbors are and use this to apply to our counter.
Display:
This will be a simply void function where we will loop through the arrays and output their values to the screen to be called in main.

Main: Call each of our functions, and have the user push enter after each generation to run the program to output the next generation. Offer to quit the program during runtime, and offer to play again once their current game has ended.