Goal: simulate a "universe" using a 2D array of booleans where each cell is either dead or alive, and simulate a game based on the rules given in the documentation, showing each step of the simulation and the end result. Command line arguments specify the input and output files, whether the bounds of the universe exist or it loops, whether to display the simulation, and how many steps to use in the simulation.

Pseudo code:

Life.c (main):

- use getopt to parse command line options, setting toroidal, ncurses, and generations. then use fscanf to read the first line of the infile and set rows and columns accordingly. create 2 universes and populate using the rest of the infile. setup ncurses, and iterating over the number of generations, take a census of each cell and set each cell based on the given rules, then swap the universes. Then close the ncurses screen and print the final universe using uv_print

Universe.c:

- uv_create:
  - mallocate universe, iterate over rows given as parameter to mallocate each column of bools
- uv_delete:
  - iterate over rows value located within universe to free each column, then free the array of rows as well as voiding the universe pointer
- uv_rows:
  - accessor function for universe's rows
- uv_cols:
  - accessor function for universe's columns
- uv_live_cell:
  - sets value at given coordinates to alive if within bounds
- uv_dead_cell:
  - sets value at given coordinates to dead if within bounds
- uv_get_cell:
  - returns value at given coordinates if within bounds
- uv_populate:
  - iterate over values located in parameter infile and parse them into the universe's bool grid
- uv_census:
  - at a given coordinate, iterates over every possible neighbor, including the opposite edge of the grid if the universe is set to toroidal, and returns the number of living neighbors.
- uv_print:

    - print universe by iterating over each value, printing living cells as o and dead cells as .