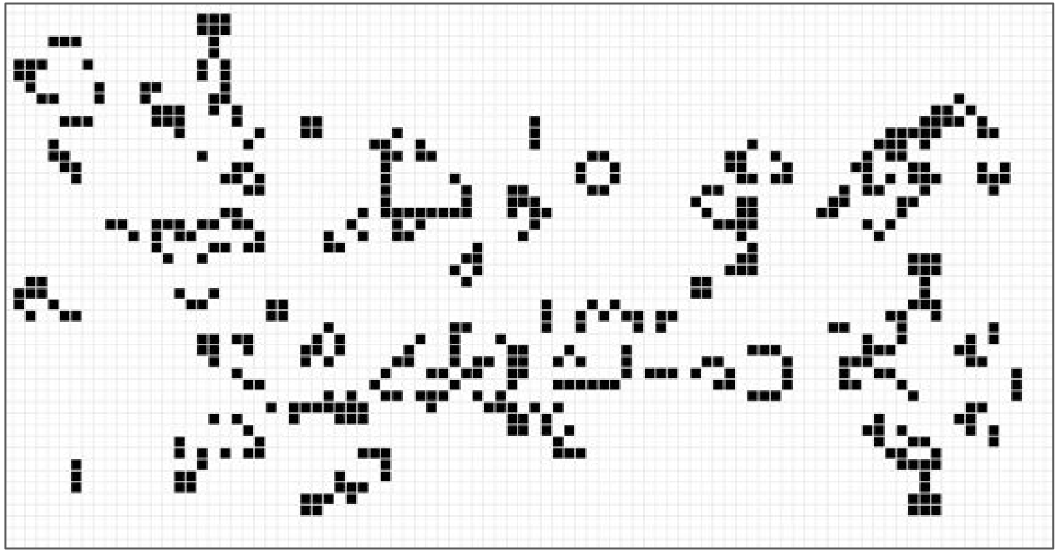
YEAH - Game of Life

Jason Chen

Original slides by: Anton Apostolatos

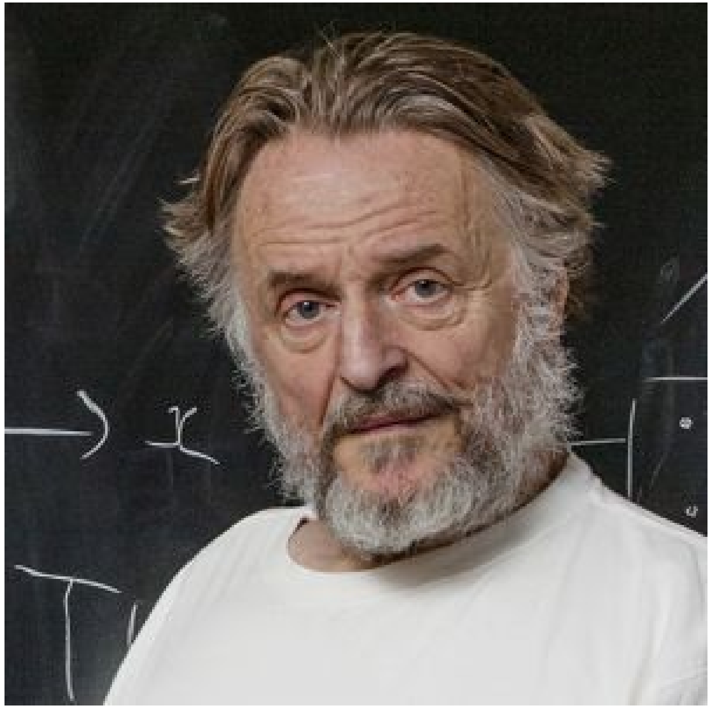


Section leaders are friends, not food

- Once a week to go over material we've gone over in class that week
  - All problems will be found in CodeStepByStep
- Your SL will grade your assignments and will meet with you personally for each assignment for Interactive Grading (IGs)
  - Roughly one week turnaround
- Your SL is your point-person (and a main resource for help)!
- LaIR opens up today!

Von Neumann and Conway's "Game of Life"

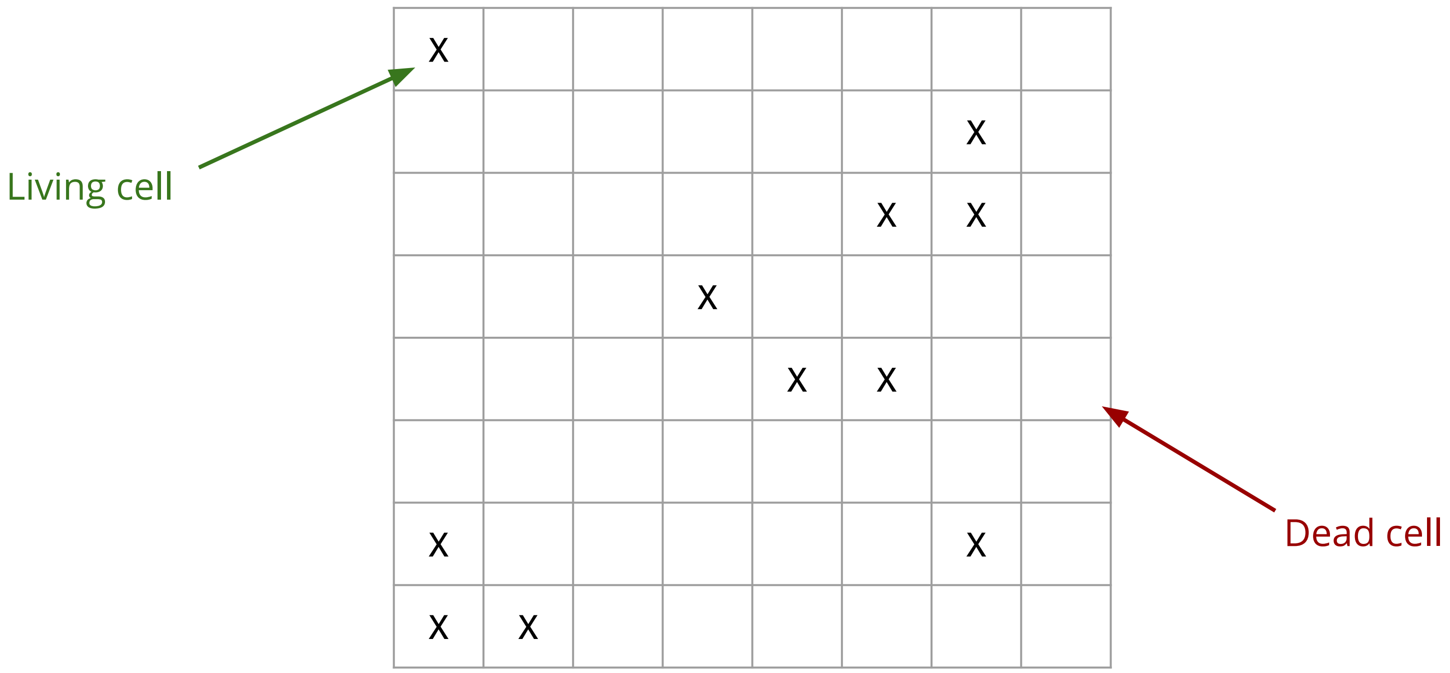
John von Neumann


John Conway

The idea behind cellular automata is that the behavior of a group can be
described by examining the interactions between an individual simple machine, termed an automaton, and the nearby identical automata that directly interact with it. These automata, referred to as cells, affect the cell in focus and define that cell’s neighborhood and change depending on the rules of interaction in the system.

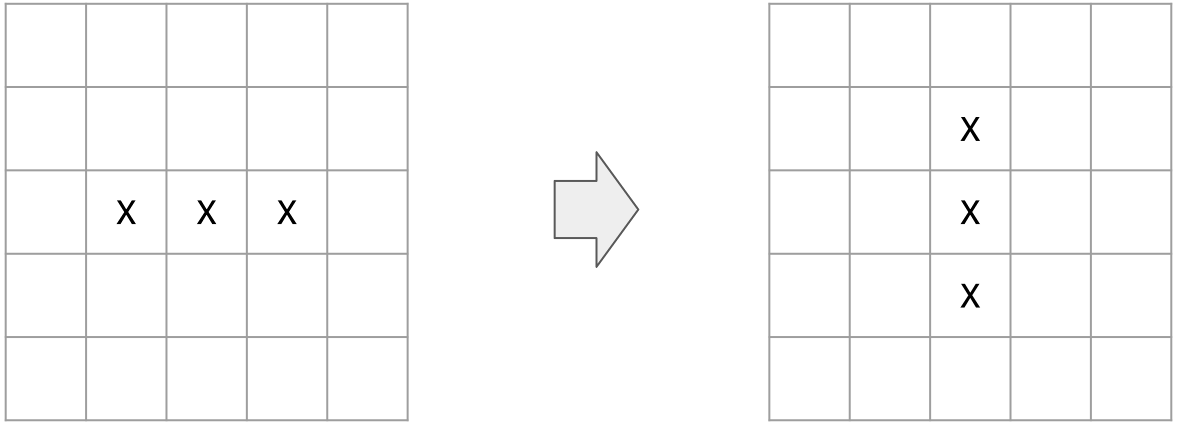
- John von Neumann (1966)

Game of Life - John Horton Conway (1970)



For each cell, from time ***t*** to time ***t + 1***:

0-1 neighbors	→	dead cell
2 neighbors	→	stable
3 neighbors	→	live cell
4-8 neighbors	→	dead cell



Time: ***t*** Time: ***t + 1***

Starter code

```
#include <iostream>
#include <string>
#include "lifegui.h"
using namespace std;

int main() {
  // TODO: Finish the program!

  cout << "Have a nice Life!" << endl;
  return 0;
}
```

The purpose of this assignment is to gain familiarity with basic C++ features such as functions, strings, and I/O streams, using provided libraries, and decomposing a large problem into smaller functions.

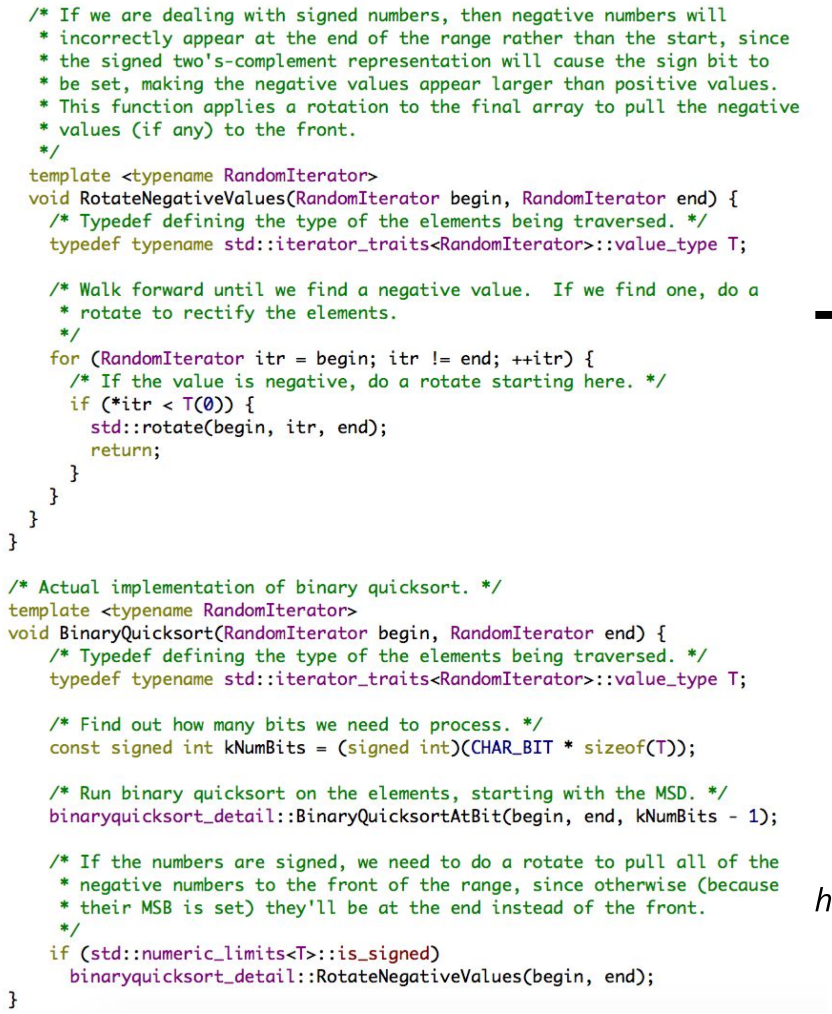
Demo

(also check out Chris’ demo starting from 6:00 into Lecture 2)

```
Welcome to the CS 106B Game of Life,
a simulation of the lifecycle of a bacteria colony.
Cells (X) live and die by the following rules:
- A cell with 1 or fewer neighbors dies.
- Locations with 2 neighbors remain stable.
- Locations with 3 neighbors will create life.
- A cell with 4 or more neighbors dies.

Grid input file name? simple.txt
Should the simulation wrap around the grid (y/n)? n
---------
---------
---XXX---
---------
---------
a)nimate, t)ick, q)uit? t
---------
----X----
----X----
----X----
---------
a)nimate, t)ick, q)uit? t
---------
---------
---XXX---
---------
---------
a)nimate, t)ick, q)uit? t
---------
----X----
----X----
----X----
---------
a)nimate, t)ick, q)uit? q
Have a nice Life!
```

```
a)nimate, t)ick, q)uit? A
How many frames? huh
Illegal integer format. Try again.
How many frames? x
Illegal integer format. Try again.
How many frames? 5
==================== (console cleared) ====================
----------------------
--X-------------------
---XX-----------------
--XX------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
==================== (console cleared) ====================
----------------------
---X------------------
----X-----------------
--XXX-----------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
```

Tips



```
  /* If we are dealing with signed numbers, then negative numbers will
   * incorrectly appear at the end of the range rather than the start, since
   * the signed two's-complement representation will cause the sign bit to
   * be set, making the negative values appear larger than positive values.
   * This function applies a rotation to the final array to pull the negative
   * values (if any) to the front.
   */
  template <typename RandomIterator>
  void RotateNegativeValues(RandomIterator begin, RandomIterator end) {
    /* Typedef defining the type of the elements being traversed. */
    typedef typename std::iterator_traits<RandomIterator>::value_type T;

    /* Walk forward until we find a negative value.  If we find one, do a
     * rotate to rectify the elements.
     */
    for (RandomIterator itr = begin; itr != end; ++itr) {
      /* If the value is negative, do a rotate starting here. */
      if (*itr < T(0)) {
        std::rotate(begin, itr, end);
        return;
      }
    }
  }
}

/* Actual implementation of binary quicksort. */
template <typename RandomIterator>
void BinaryQuicksort(RandomIterator begin, RandomIterator end) {
    /* Typedef defining the type of the elements being traversed. */
    typedef typename std::iterator_traits<RandomIterator>::value_type T;

    /* Find out how many bits we need to process. */
    const signed int kNumBits = (signed int)(CHAR_BIT * sizeof(T));

    /* Run binary quicksort on the elements, starting with the MSD. */
    binaryquicksort_detail::BinaryQuicksortAtBit(begin, end, kNumBits - 1);

    /* If the numbers are signed, we need to do a rotate to pull all of the
     * negative numbers to the front of the range, since otherwise (because
     * their MSB is set) they'll be at the end instead of the front.
     */
    if (std::numeric_limits<T>::is_signed)
      binaryquicksort_detail::RotateNegativeValues(begin, end);
}
```

Tip I: Decompose!

"Nothing is more permanent than the temporary"

Styleguide at:
https://web.stanford.edu/class/cs106b/handouts/styleguide.html

Tip II: Outline before you write!

Implementation

```
a)nimate, t)ick, q)uit? a
How many frames? xyz
Illegal integer format. Try again.
How many frames? 5
(five new generations are shown, with screen clear and 50ms pause before ea
```

Input / Output in C++

- You should use either **`getInteger()`** or **`getLine()`** to read input from the keyboard. They work as follows:

Note the uppercase L

```
#include <iostream>
#include "console.h"
#include "simpio.h"

using namespace std;

int main() {
    int a;
    string s;
    a = getInteger("Please enter an integer: ","That wasn't an integer!");
    s = getLine("Please type in a string: ");
    cout << "Your integer: " << a << endl;
    cout << "Your string: " << s << endl;
    return 0;
}
```

File Structure *mycolony.txt: your chance to be creative!*

```
5                    <-- number of rows
9                    <-- number of columns
---------
---------            <-- grid of cells
---XXX---
---------
---------
# simple.txt         <-- optional junk/comments
# This file is a     <-- at bottom (should be ignored)
# basic grid of
# cells.
```

Input / Output in C++

- To read in input from a file, use the getline() function

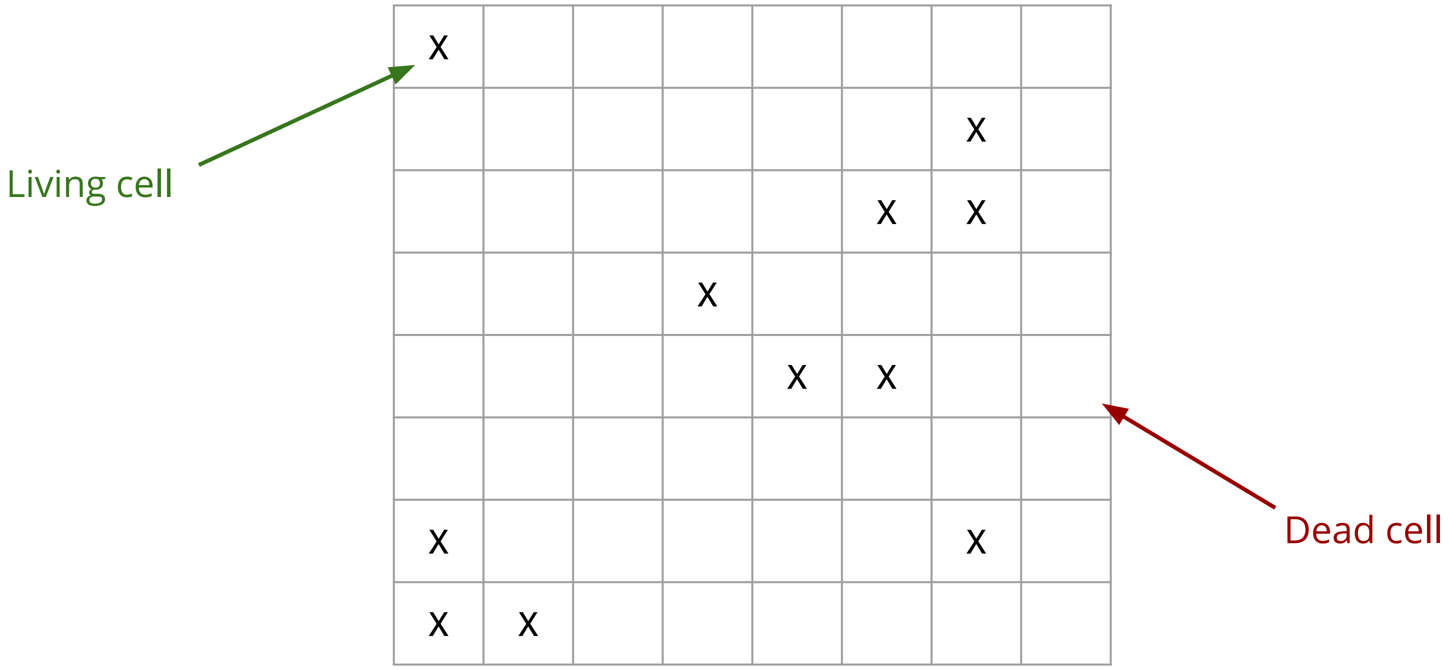
Note the **lowercase** L

```
#include <iostream>
#include <fstream>
#include "console.h"
#include "simpio.h"
#include "filelib.h"

 using namespace std;

 int main() {
     string s;

     ifstream stream;
     openFile(stream, "readme.txt");
     getline(stream,s);
     cout << "The first line of the file is: " << s << endl;
     return 0;
 }
```

Design Decision

How to store the world?

Game of Life - John Horton Conway (1970)



Stanford C++ Grid class

```
Grid(nRows, nCols, value) // Initializes a new grid of the given size, with
                             every cell set to the given value.

numRows()                 // Returns the number of rows in the grid.

numCols()                 // Returns the number of columns in the grid.

inBounds(row, col)        // Returns true if row/col are inside grid bounds

get(row, col)             // Returns element at row/col position
```

Grid documentation at: https://stanford.edu/~stepp/cppdoc/Grid-class.html

```
[[0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0]
 [0, 0, 0, 0, 0]]
```

```
[[0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0]
 [0, 0, 0, 0, 0]]
```

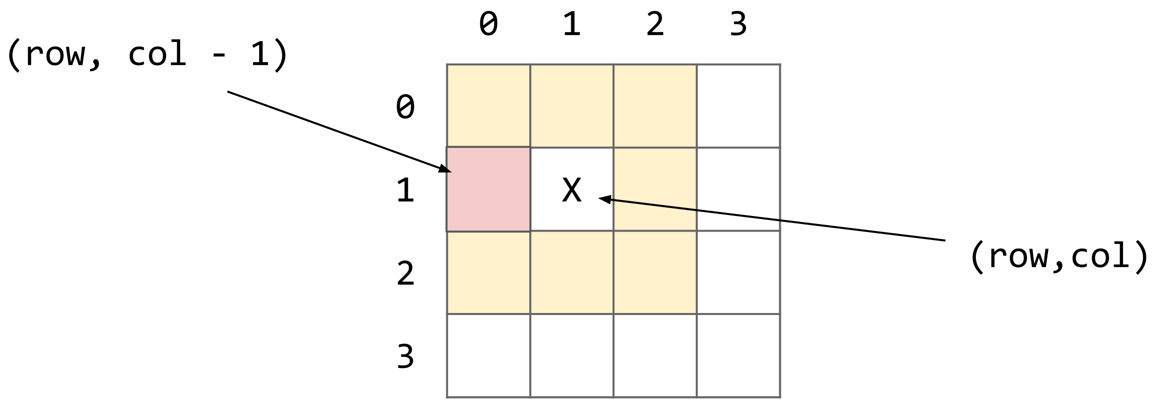
[[0, 0, 0, 0, 0],

[0, 0, 0, 0, 0],

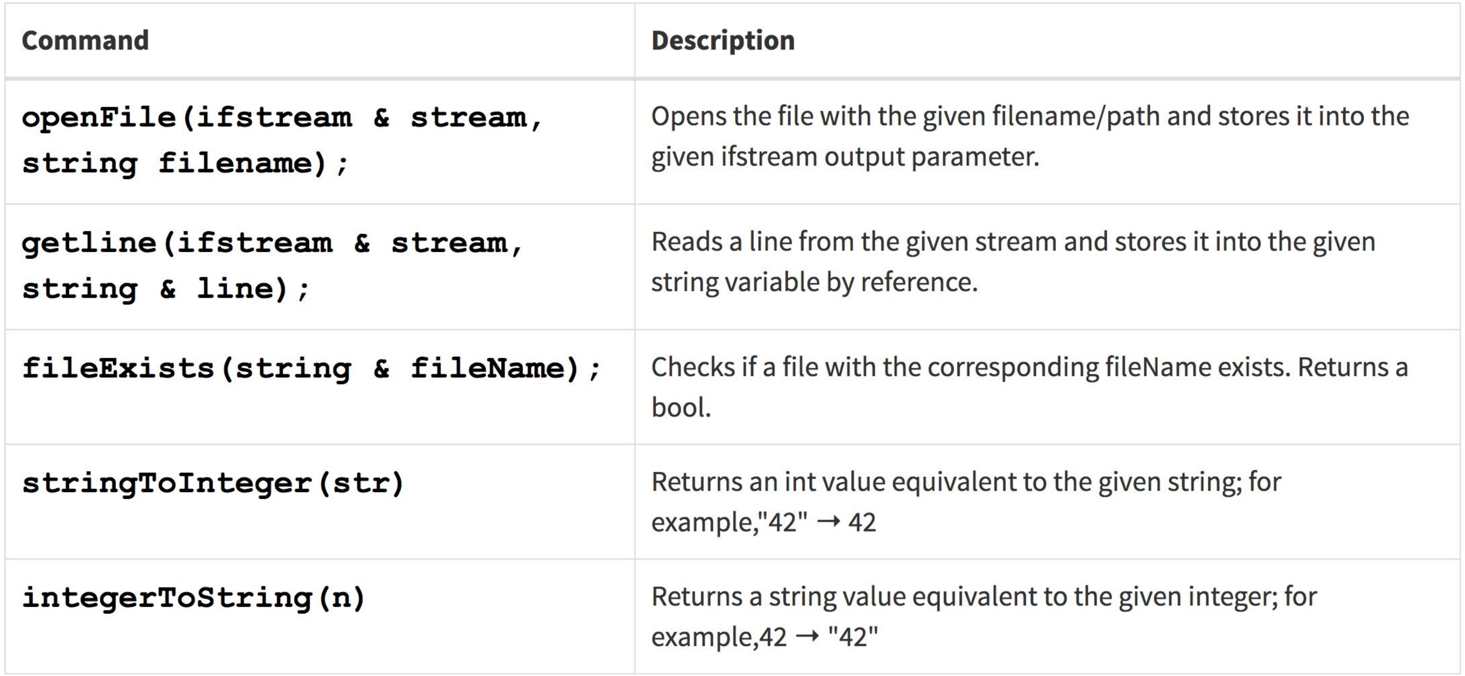
[0, 0, 0, **0**, 0]

[0, 0, 0, 0, 0]]

grid[2][3]


(row, col - 1)
0
1
2
3
0
1
2
3
X
(row,col)

Useful Functions



Command	Description
`openFile(ifstream & stream, string filename);`	Opens the file with the given filename/path and stores it into the given ifstream output parameter.
`getline(ifstream & stream, string & line);`	Reads a line from the given stream and stores it into the given string variable by reference.
`fileExists(string & fileName);`	Checks if a file with the corresponding fileName exists. Returns a bool.
`stringToInteger(str)`	Returns an int value equivalent to the given string; for example,"42" → 42
`integerToString(n)`	Returns a string value equivalent to the given integer; for example,42 → "42"

```
promptUserForFile(stream, prompt)
^ another function you could use that wasn't included in the original slides.
```

Full documentation at: https://stanford.edu/~stepp/cppdoc/

Corners?

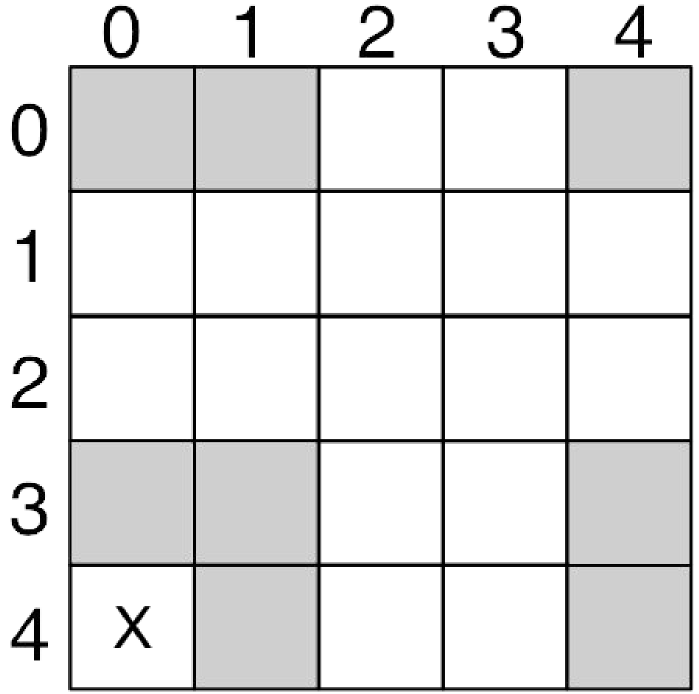
Non-wrapping

- Neighbors outside of the world are ignored

Wrapping

- The world wraps around top-bottom and left-right
- Use the **mod** (%) operator

`(a % b) returns the remainder of a / b`



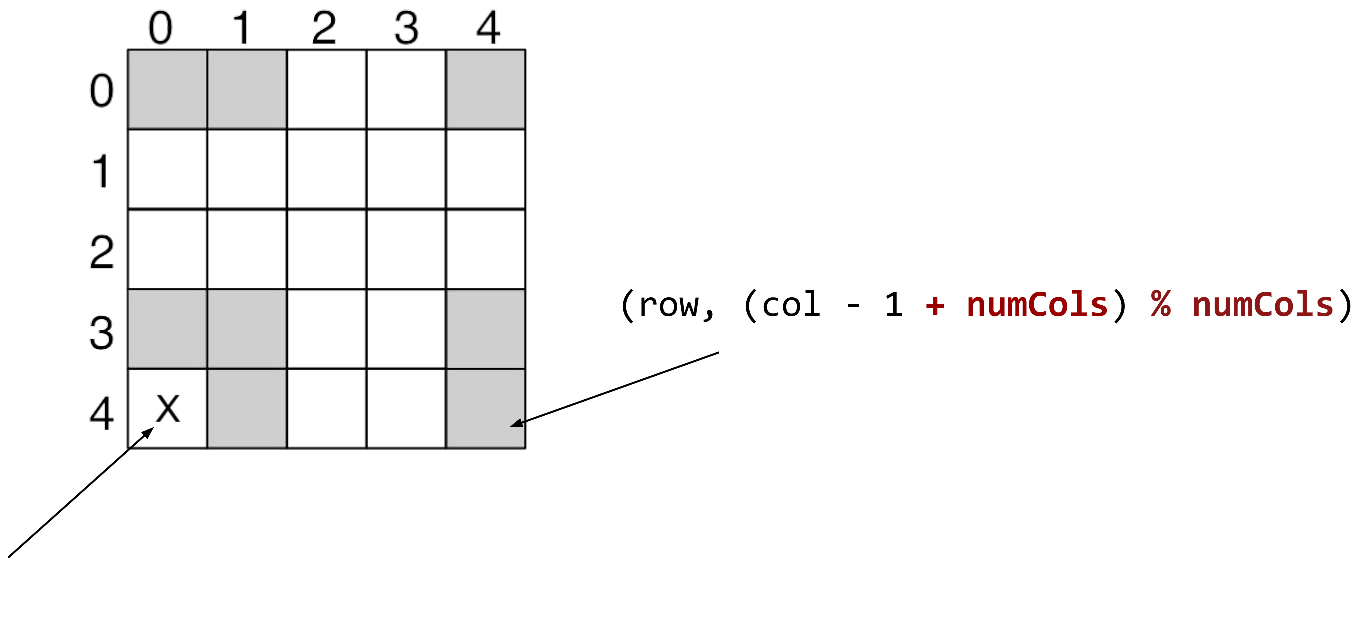
mod %

10 % 10 ?

4 % 3 ?

7 % 1?

12 % 5?


0
1
2
3
4
0
1
2
3
4
X
(row, (col - 1 + numCols) % numCols)
(row,col)

Steps

1. **Setup.** Get the project running and print intro welcome message
2. **File input.** Write code to prompt for a filename, and open and print that file's lines to the console. Once this works, try reading the individual grid cells and turning them into a Grid object.
3. **Grid display.** Write code to print the current state of the grid, without modifying that state.

Steps II

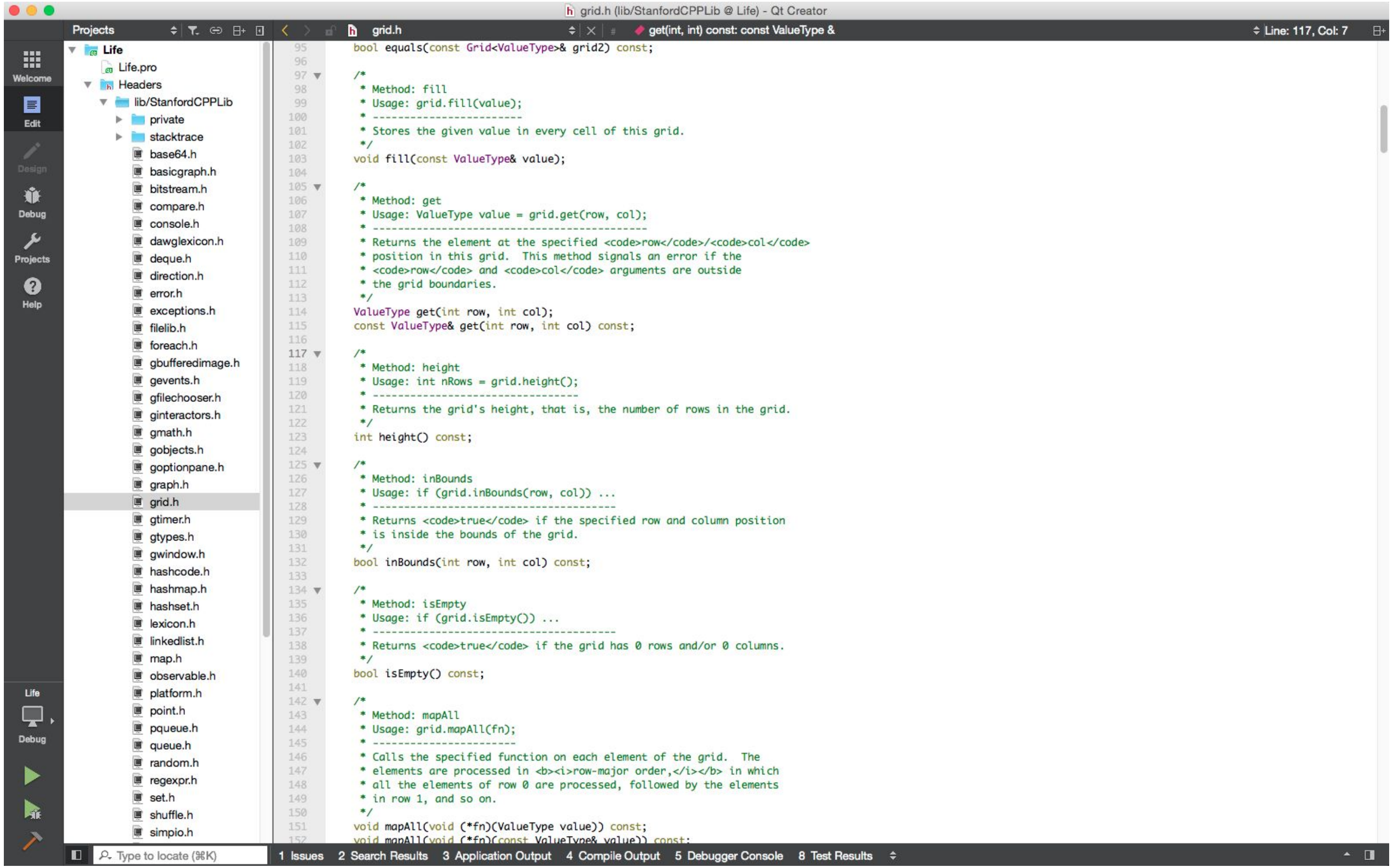
4. **Updating to next generation.** Write code to advance the grid from one generation to the next.
5. **Overall menu and animation.** Implement the program's main menu and the animation feature.
6. **Extensions.** If you do one step a day starting from today, you’ll still have three days to do extensions! :) A list of suggestions is offered at the end of the handout. I personally suggest adding the graphical component. It’s super pretty!

Documentation

http://stanford.edu/~stepp/cppdoc/

if you start to think “there must be an easier way to do this nitty gritty string processing…”

there most probably is.



```cpp
    bool equals(const Grid<ValueType>& grid2) const;

    /*
     * Method: fill
     * Usage: grid.fill(value);
     * ------------------------
     * Stores the given value in every cell of this grid.
     */
    void fill(const ValueType& value);

    /*
     * Method: get
     * Usage: ValueType value = grid.get(row, col);
     * --------------------------------------------
     * Returns the element at the specified <code>row</code>/<code>col</code>
     * position in this grid.  This method signals an error if the
     * <code>row</code> and <code>col</code> arguments are outside
     * the grid boundaries.
     */
    ValueType get(int row, int col);
    const ValueType& get(int row, int col) const;

    /*
     * Method: height
     * Usage: int nRows = grid.height();
     * ---------------------------------
     * Returns the grid's height, that is, the number of rows in the grid.
     */
    int height() const;

    /*
     * Method: inBounds
     * Usage: if (grid.inBounds(row, col)) ...
     * ---------------------------------------
     * Returns <code>true</code> if the specified row and column position
     * is inside the bounds of the grid.
     */
    bool inBounds(int row, int col) const;

    /*
     * Method: isEmpty
     * Usage: if (grid.isEmpty()) ...
     * ---------------------------------------
     * Returns <code>true</code> if the grid has 0 rows and/or 0 columns.
     */
    bool isEmpty() const;

    /*
     * Method: mapAll
     * Usage: grid.mapAll(fn);
     * -----------------------
     * Calls the specified function on each element of the grid.  The
     * elements are processed in <b><i>row-major order,</i></b> in which
     * all the elements of row 0 are processed, followed by the elements
     * in row 1, and so on.
     */
    void mapAll(void (*fn)(ValueType value)) const;
    void mapAll(void (*fn)(const ValueType& value)) const;
```

filelib.h

grid.h

simpio.h

strlib.h

Questions?

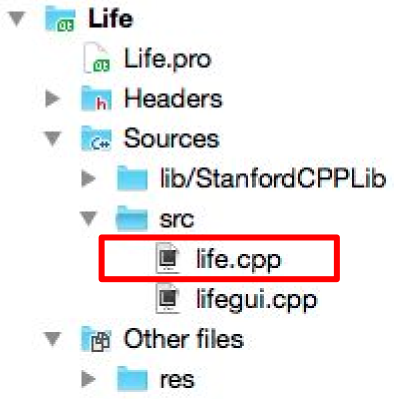
piazza
your section leader
your classmates
Chris
Jason

piazza
your section leader
your classmates
Chris
Jason

the assignment handout!! :)

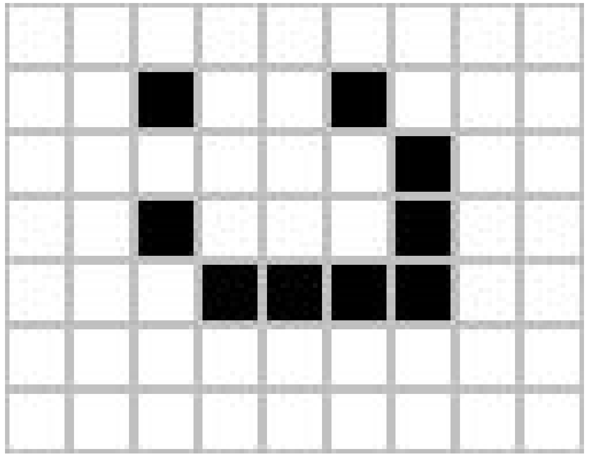
Starter code

```cpp
#include <iostream>
#include <string>
#include "lifegui.h"
using namespace std;

int main() {
  // TODO: Finish the program!

  cout << "Have a nice Life!" << endl;
  return 0;
}
```

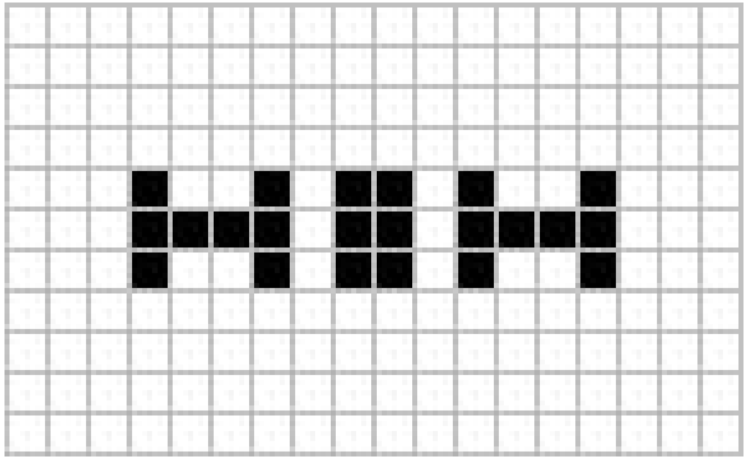


```
▼ Life
    Life.pro
  ▶ Headers
  ▼ Sources
    ▶ lib/StanfordCPPLib
    ▼ src
        life.cpp
        lifegui.cpp
  ▼ Other files
    ▶ res
```

Glider



Pentadecathlon



Pulsar