YEAH - Game of Life
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 on Sunday
Von Neumann and Conway’s “Game of Life”

John von Neumann

John Conway
Finite grid world

Living cell

Dead cell
For each cell, from time $t$ to time $t + 1$:

- 0-1 neighbors $\rightarrow$ dead cell
- 2 neighbors $\rightarrow$ stable
- 3 neighbors $\rightarrow$ live cell
- 4-8 neighbors $\rightarrow$ dead cell
Demo!
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;
}
Tips
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
### File Structure

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

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
</table>
| 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, LOLOLOL.
Design Decision

How to store the world?
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
# Useful Functions

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

*Full documentation at:* [https://stanford.edu/~stepp/cppdoc/](https://stanford.edu/~stepp/cppdoc/)
Corners?
Wrapping

- The world wraps around top-bottom and left-right

- Use the \texttt{mod} (\%) operator

\[(a \% b) \text{ returns the remainder of } a / b\]
<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(row, col) → (row, (col - 1) % numCols)
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.
Questions?
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;
}
```
Glider
Pentadecathlon