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)

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
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;
}
```
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)animate, t)ick, q)uit? t
---------
-----X----
-------X---
-------X---
a)animate, t)ick, q)uit? t
---------
--XXX--
---------
a)animate, t)ick, q)uit? t
---------
-----X----
-------X---
------X----
a)animate, t)ick, q)uit? q
Have a nice Life!
animate, tick, quit? 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
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
You should use either `getInteger()` or `getLine()` to read input from the keyboard. They work as follows:

```cpp
#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!

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---XXX---

# simple.txt  
# This file is a basic grid of cells.

--- --

# optional junk/comments

--- --

# at bottom (should be ignored)
To read in input from a file, use the getline() function

```cpp
#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;
}
```

Note the lowercase L
Design Decision

How to store the world?
Game of Life - John Horton Conway (1970)

Living cell

Dead cell
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](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]
]
$$\text{grid}[2][3]$$

$$=[[0, 0, 0, 0, 0],
[0, 0, 0, 0, 0],
[0, 0, 0, 0, 0],
[0, 0, 0, 0, 0]]$$
### 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>

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

**Full documentation at:** [https://stanford.edu/~stepp/cppdoc/](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
  $(a \mod b)$ returns the remainder of $a / b$
mod %

10 % 10?
4 % 3?
7 % 1?
12 % 5?
$$\text{(row, (col - 1 + numCols) \mod numCols)}$$
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.

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.
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;
}
```
Glider
Pentadecathlon
Pulsar