Recursion
March 7, Ch. 14

In order to understand recursion, you must understand recursion.
Announcements

Deadline for hw4 pushed back to Sunday March 13 at 6:00pm
Highlights

- recursion

```cpp
int main()
{
    cout << "HI\n!";
    main();
}
```
There are two important parts of recursion:
- A stopping case that ends the recursion
- A reduction case that reduces the problem

What are the base and stopping cases for the Fibonacci numbers?

\[ F_n = F_{n-1} + F_{n-2}, \]
\[ 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, \ldots \]

(sum of the previous two numbers) (see last time: fibonacciRecursion.cpp)
Recursion

What if I wanted to just count down to zero? `countdown(5)` would show:

5
4
3
2
1
0!

(see: countdown.cpp)
Recursion

What if we defined tangent recursively as:

\[ \tan(x) = \frac{x}{1 - \frac{x^2}{3 - \frac{x^3}{5 - \frac{x^4}{7 - \ldots}}}} \]

Assume we take an input for how many times to do this recursion

What is the pattern? What is the stopping case?

How do we move towards the stopping case

(see: tangent.cpp)
Recursion

How would you sum the numbers 1 to n using recursion (not a loop)?

For example, \( \text{sumToN}(5) = 15 \), as \( 1+2+3+4+5 = 15 \).

What is the stopping case?
How do you reduce the problem?

(see: \text{sumToN.cpp})
Recursion is simply when a function calls itself (we did this for the maze in week 5)

This is quite powerful, but also confusing

(see: towerHanoi.cpp)
Recursion

How would you solve a sudoku problem?

Rules:
1. Every row has numbers 1-9
2. Every column has numbers 1-9
3. The nine 3x3 boxes have numbers 1-9

Reduce problem?
Stopping case?

(see: sudokuSolver.cpp)
Recursion

Do not try to solve chess in this manner!

You will segfault
(you will also not finish computing before the sun burns the earth to a crisp)