Week 7: Recursion

Today, we'll go over recursion. You're probably used to iteration (for loops). Another way of coding things up is with recursion (when a function calls itself). By some theorem though, all recursive code can be written as iterative code and vice versa.

Recursion can be difficult at first, so it may be tempting to just only use iteration. But, once you get the hang of it, you will notice that it can be much more elegant than iteration. We'll do some problems with recursion today and maybe they will show how recursion can be nicer than iteration in some cases.

Fibonacci

The Fibonacci sequence is defined as

```
          F(0) = 0
          F(1) = 1
F(n) = F(n-1) + F(n-2)
```

First write an iterate function to calculate the n-th Fibonacci number. Then write a recursive function. If you just applied the straightforward approach, you might notice that it's fast to calculate F(100000) with the iterative approach, but maybe not so easy with the recursive approach. Can you make your recursive approach faster? What's redundant about it?

Combinations

The number of ways to create a k person team from a group of n people is

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!}$$

By definition, we let $\binom{n}{k} = 0$ if k > n so that we avoid negative factorials. What should be your base case(s)?

Using Pascal's Identity, we also know that

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$$

This identity is useful if you want to calculate $\binom{n}{k}$ for all n and k in some big range. Please calculate

$$\sum_{n=0}^{1000} \sum_{k=0}^{1000} \binom{n}{k} \pmod{1234567}$$

If I'm not mistaken, you should get 553338.

Complete Search

Suppose you're in a maze of 1s and 0s and you want to get out:

00001101
111**0**0101
110**0**1111
110**00**001
1101**0**101
1101**0**111
1101**000**1
11111**00**

You start in the top left corner, and the exit the bottom right corner. You can move North, South, East, or West, but you can only move onto a square that has a 0. Can you output a path to get out? It is guaranteed that the top left and bottom right corners are 0.

You probably won't get anywhere here using for loops, so I definitely recommend you try recursion.

Input:
Line 0: n, m – the height and width of the maze, respectively
Lines 1-n: n rows of m characters (1s and 0s) that describe the maze

Output: A solution to the maze, or "impossible" if no solution exists.

Sample Input:
8 8
00001101
11100101
11001111
11000001
11010101
11010111
11010001
11111100

Sample Output:
EEESSSESSSEESE

Sample Input:
2 3
011
110

Sample Output:
impossible