Assignment 11: Intro to Dynamic Programming Due **in class** *Thursday, 12/6*

December 3, 2018
CS: DS&A
Proof School

Rod Cutting with Instructions

Modify `rod_mem_top` to return some representation of how to cut the rod into pieces, in addition to the maximum price. You can do this by simply changing the `values` memoization object, but I suggest *planning* before you write a line of code!

Longest Increasing Subsequence

For your second problem, you will use the dynamic programming paradigm to write a function `LIS` that takes a list of numbers, and returns the length of the longest strictly increasing subsequence. For example, `LIS[2,1,3,7,3,9,4,11,11]` returns `5`.

(a) The first step in dynamic programming is setting up the recursive relationship. Sometimes there's a bit of a trick, and there is here, so read carefully!

Let `input` be an input list. You might think to find a recursive relation using

A_i = the length of the longest increasing subsequence formed from the first i elements of the input

Instead, I want you to consider the quantities

B_i = the length of the longest increasing subsequence formed from the first i elements of the input *that ends with the ith element*

For example, given `input = [2,1,3,7,3,9,4,11,11]`, we have $A_5 = 3$ (via the subsequence `1,3,7`, for example), but $B_5 = 2$ (because the subsequence has to end on the 5th element (which isn't love, but 3)).

Your first step is to come up with a recursive relationship between B_i and all the B_j for $j < i$. (You wouldn't have been able to do this with the A_i.) Look at examples if you're stuck.

(b) Now implement `LIS(input)`. You may use either top-down or bottom-up dynamic programming. Your overall runtime should be $O(n^2)$.