Recursion: Towers of Hanoi

Last time we saw recursive solutions for a couple of simple problems, and for the Sierpinski triangle. Today we'll look at another problem that is seemingly very hard, but recursion allows for nice and not-too-hard solution. This is the problem of the *Towers of Hanoi*.

In the Towers of Hanoi problem there are three pegs (posts) and n disks of different sizes. Each disk has a hole in th emiddle so that it can fit on any peg. At the beginning of the game, all n disks are all on the first peg, arranged such that the largest is on the bottom, and the smallest is on the top (so the first peg looks like a tower).

The goal of the game is to end up with all disks on the third peg, in the same order, that is, smallest on top, and increasing order towards the bottom.

But, there are some restrictions to how the disks are moved (which make the problem non-trivial):

(1) The only allowed type of move is to grab *one* disk from the top of one peg and drop it on another peg. That is, you cannot grab several disks at one time.

(2) A larger disk can never lie above a smaller disk, on any post.

To approach this problem, try to solve it by hand for $n = 1, 2, 3, ...$ It quickly becomes hard to keep track of the moves, no? But it gives you the intuition. Think of the moves you are doing, and try to identify smaller sub-problems. That is, can you express the process of moving n disks from one post to another, in terms if moving n-1 disks between two posts.

Try to come up with a recursive formulation of your solution. The trick when thinking recursively is to assume that *you know how to solve the problem on a smaller input*. All you have to do is (1) figure out what is/are the subproblem that come up in solving your problem; (2) figure out how to compose the solution to your oroginal problem from the solution to the subproblem(s); and (3) provide a base-case.

Attached you'll find a skeleton for the Towers of Hanoi that is supposed to ask you for the number of disks, and print the moves. It does no graphics, in order to keep things simple (and focus on recursion). The goal is to fill in the details for

```
public void move (sourcePeg, storagePeg, DestinationPeg)
```

1. CORRECTNESS

Why does this work?

2. ANALYSIS

How long does your move() method take to solve the Towers of Hanoi problem on n disks?