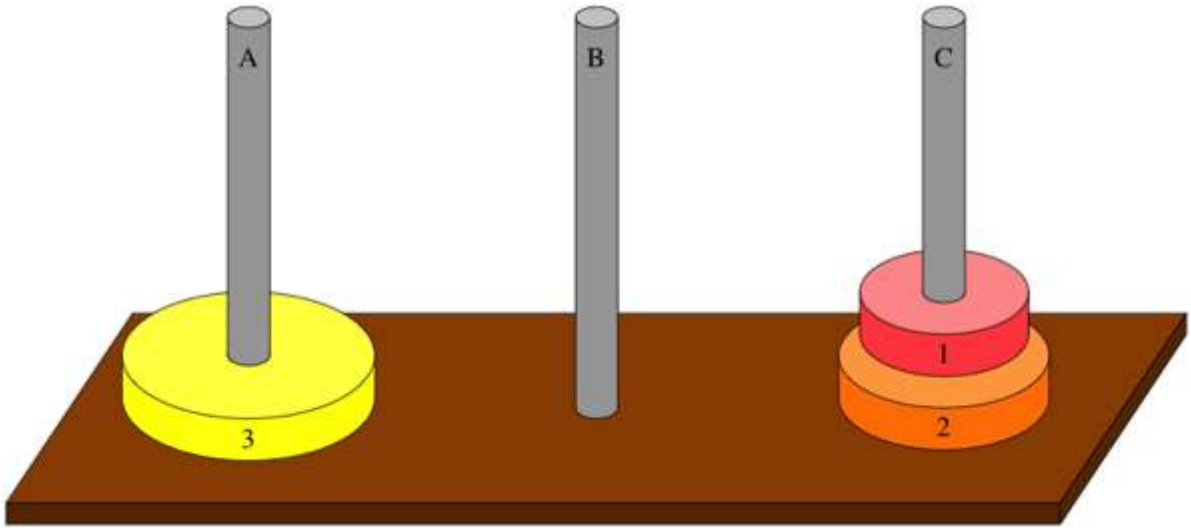
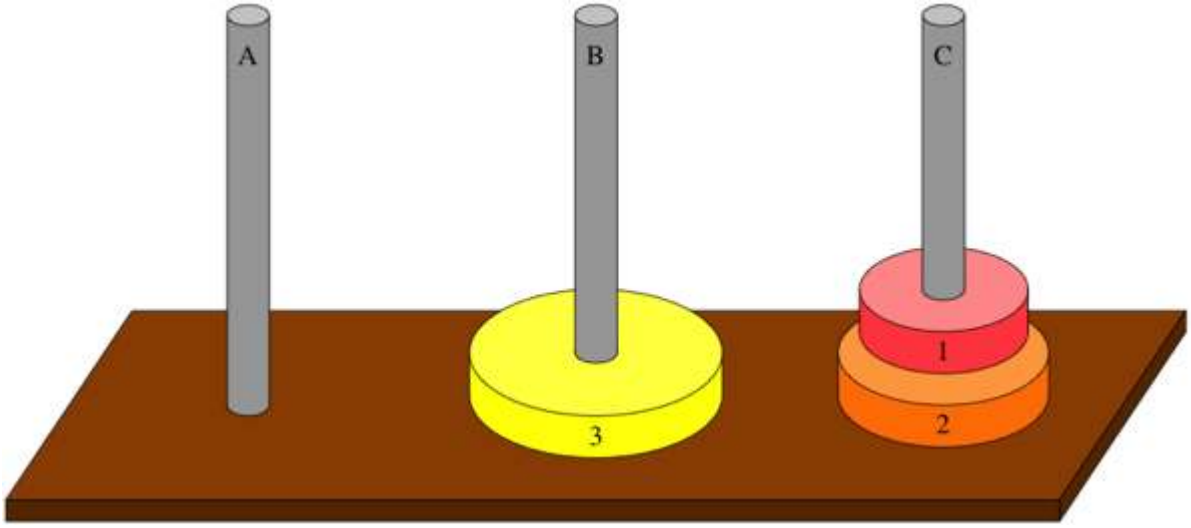
Towers of Hanoi, continued

When you solved the Towers of Hanoi for three disks, you needed to expose the bottom disk, disk 3, so that you could move it from peg A to peg B. In order to expose disk 3, you needed to move disks 1 and 2 from peg A to the spare peg, which is peg C:

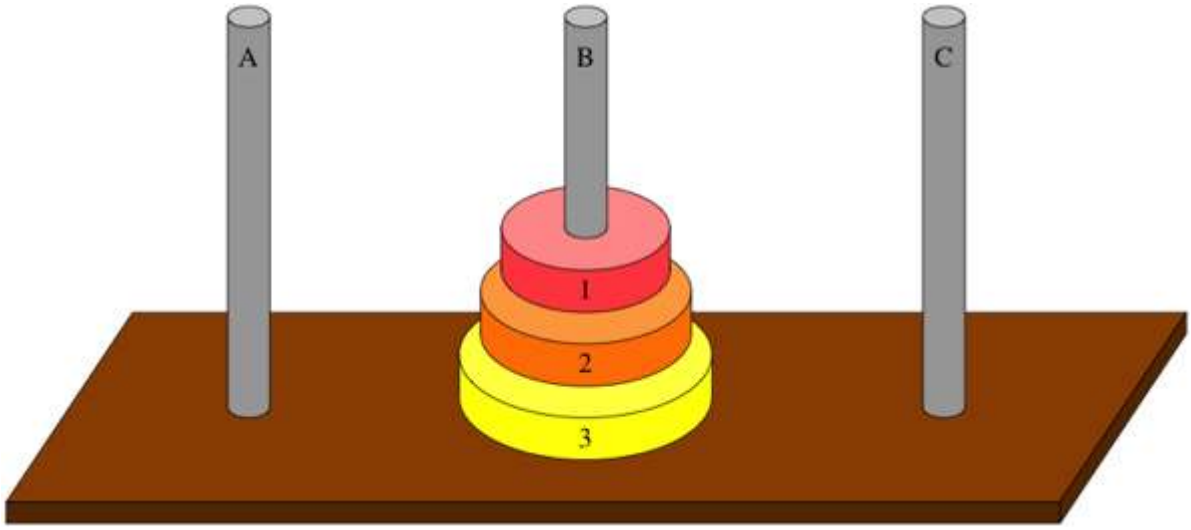


Wait a minute—it looks like two disks moved in one step, violating the first rule. But they did not move in one step. You agreed that you can move disks 1 and 2 from any peg to any peg, using three steps. The situation above represents what you have after three steps. (Move disk 1 from peg A to peg B; move disk 2 from peg A to peg C; move disk 1 from peg B to peg C.)

More to the point, by moving disks 1 and 2 from peg A to peg C, you have recursively solved a subproblem: move disks 1 through **n-1** (remember that **n = 3**) from peg A to peg C. Once you've solved this subproblem, you can move disk 3 from peg A to peg B:



Now, to finish up, you need to recursively solve the subproblem of moving disks 1 through **n-1** from peg C to peg B. Again, you agreed that you can do so in three steps. (Move disk 1 from peg C to peg A; move disk 2 from peg C to peg B; move disk 1 from peg A to peg B.) And you're done:



And—you knew this question is coming—is there anything special about which pegs you moved disks 1 through 3 from and to? You could have moved them from any peg to any peg. For example, from peg B to peg C:

- Recursively solve the subproblem of moving disks 1 and 2 from peg B to the spare peg, peg A. (Move disk 1 from peg B to peg C; move disk 2 from peg B to peg A; move disk 1 from peg C to peg A.)
- Now that disk 3 is exposed on peg B, move to it peg C.

- Recursively solve the subproblem of moving disks 1 and 2 from peg A to peg C. (Move disk 1 from peg A to peg B; move disk 2 from peg A to peg C; move disk 1 from peg B to peg C.)

No matter how you slice it, you can move disks 1 through 3 from any peg to any peg, moving disks seven times. Notice that you move disks three times for each of the two times that you recursively solve the subproblem of moving disks 1 and 2, plus one more move for disk 3.

How about when **n**=**4**? Because you can recursively solve the subproblem of moving disks 1 through 3 from any peg to any peg, you can solve the problem for **n** = **4**:

- Recursively solve the subproblem of moving disks 1 through 3 from peg A to peg C, moving disks seven times.
- Move disk 4 from peg A to peg B.
- Recursively solve the subproblem of moving disks 1 through 3 from peg C to peg B, moving disks seven times.

This solution moves disks 15 times (7 + 1 + 7) and, yes, you can move disks 1 through 4 from any peg to any peg.

At this point, you might have picked up two patterns. First, you can solve the Towers of Hanoi problem recursively. If **n** = **1**, just move disk 1. Otherwise, when **n** ≥ **2**, solve the problem in three steps:

- Recursively solve the subproblem of moving disks 1 through **n**−**1** from whichever peg they start on to the spare peg.
- Move disk **n** from the peg it starts on to the peg it's supposed to end up on.
- Recursively solve the subproblem of moving disks 1 through **n**−**1** from the spare peg to the peg they're supposed to end up on.

Second, solving a problem for **n** disks requires $2^n - 1$ moves. We've seen that it's true for **n** = **1** ($2^1 - 1 = 1$, and we needed one move), **n** = **2** ($2^2 - 1 = 3$, and three moves), **n** = **3** ($2^3 - 1 = 7$, and seven moves), and **n** = **4** ($2^4 - 1 = 15$, and 15 moves). If you can solve a problem for **n-1** disks in $2^{n-1} - 1$ moves, then you can solve a problem for **n** disks in $2^n - 1$ moves: you need $2^{n-1} - 1$ moves to recursively solve the first subproblem of moving disks 1 through **n-1**, one move to move disk **n**, and another $2^{n-1} - 1$ moves to recursively solve the second subproblem of moving disks 1 through **n-1**. If you add up the moves, you get $2^n - 1$.

Back to the monks. They're using **n** = **64** disks, and so they will need to move a disk $2^{64} - 1$ times. These monks are nimble and strong. They can move one disk every second, night and day. How long is $2^{64} - 1$ seconds? Using the rough estimate of 365.25 days per year (we're not accounting for skipping the leap year once every 400 years), that comes to 584,542,046,090.6263 years. That's 584+ billion years. The sun has only about another five to seven billion years left before it goes all supernova on us. So, yes, the world will end, but no matter how tenacious the monks may be, it will happen long before they can get all 64 disks onto peg B.

Wondering how else we can use this algorithm, besides predicting the end of the world? Wikipedia lists several interesting applications.