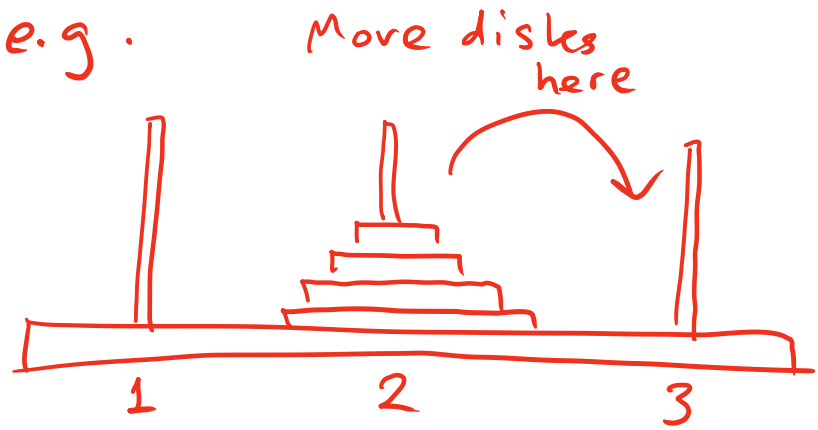
Example 6 -- Towers of Hanoi
(see https://en.wikipedia.org/wiki/Tower_of_Hanoi)

e.g.



from #2
to #3
(other = #1)

```
// Prints instructions for moving disks from one
// pole to another, where the three poles are
// labeled with integers "from", "to", and "other".
// Code from rosettacode.org
public void move(int n, int from, int to, int other) {
  if (n == 1) {
    System.out.println("Move disk from pole " + from +
                       " to pole " + to);
  } else {
    move(n - 1, from, other, to);
    move(1, from, to, other);
    move(n - 1, other, to, from);
  }
}
```

Recursive function!
Let's use recurrence relations.

Let $H(n)$ = # executions to run alg. on n.

Base case @ $n = 1$

```
if (n == 1) {
  System.out.println("Move disk from pole " + from +
                     " to pole " + to);
```

1 execution $\Rightarrow H(1) = 1$

All other $H(n)$:

```
} else {
  move(n - 1, from, other, to);
  move(1, from, to, other);
  move(n - 1, other, to, from);
}
```

$H(n-1)$ executions

$H(1) = 1$

$H(n-1)$

All together:

$$H(n) = H(n-1) + 1 + H(n-1)$$
$$= 1 + 2H(n-1)$$

Expanding (plug in for $H(n)$):

1st $H(n) = 1 + 2H(n-1)$

Plugging in "n-1" into $H(n)$

so $H(n-1) = 1 + 2[1 + 2H([n-1]-1)]$
$= 1 + 2(1 + 2H(n-2)]$
$= 1 + 2 + 4H(n-2)$

2nd $= 1 + 2 + 4H(n-2)$

3rd $= 1 + 2 + 4 + 8H(n-3)$

$\vdots$

The expansions uncovered this pattern!

k^{th} $= 2^0 + 2^1 + 2^2 + \cdots + 2^{k-1} + 2^k H(n-k)$

$$= 2^k - 1 + 2^k H(n-k)$$

Base case is at $H(1)$, so let's solve $n-k=1$

$$\Rightarrow k = n-1$$

Plug it in:

$$H(n) = 2^k - 1 + 2^k H(n-k)$$

$$= 2^{n-1} - 1 + 2^{n-1} H(n-[n-1])$$

$$= 2^{n-1} - 1 + 2^{n-1} H(1)$$

$$= 2^{n-1} - 1 + 2^{n-1}(1)$$

$$= 2(2^{-1}) - 1$$

$$H(n) = 2^n - 1$$

$$\Rightarrow \boxed{O(2^n)}$$