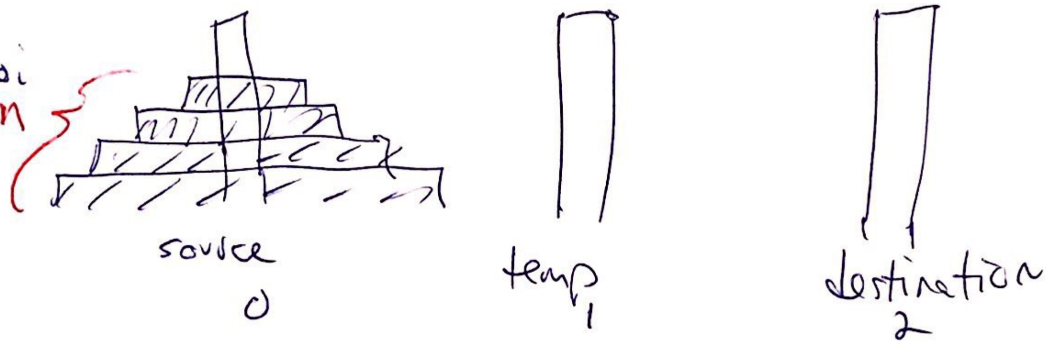
Towers of Hanoi




```
TOH(int n, int src, int temp, int dest) {
    // select moves to move n disks from src to dest

    if n=1
        print "move a disk from src to dest"

    else {
        TOH(n-1, src, dest, temp)  // move n-1 from src to temp
        print "move a disk from src to dest"
        TOH(n-1, temp, src, dest)  // move n-1 from temp to dest
    }
```

Design technique: Assume recursive calls work, justified by principle of mathematical induction

Challenge find a closed form for the # of disk moves to solve tower of size n, $T(n)$.

$$T(n) = \begin{cases} 1 & \text{if } n=1 \\ \underbrace{2T(n-1)}_{\text{two recursive calls}}+\underbrace{1}_{\text{one more}} & \text{if } n>1 \end{cases}$$

Recurrence equation
defines $T(n)$ in terms of $T(k)$ for $k<n$

$T(1)=1$

$T(2)=2(T(1))+1=3$

$T(3)=2T(2)+1=7$

guess
$T(n)=2^n-1$

Check guess by induction
prove $\forall n^{\geq 1}$ ~~T~~ $T(n)=2^n-1$

base case $n=1$ $T(n)=2^1-1=1$ ✓

⋮

Left as exercise

Expand — Guess — Check.

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

$$= 2\left[2T(n-2)+1\right] + 1 = 2^2 T(n-2) + 3$$

$$= 2\left\{2\left[2T(n-3)+1\right]+1\right\}+1 = 2^3 T(n-3) + 7$$

guess $\longrightarrow$ $= 2^k T(n-k) + (2^k - 1)$

then set $k = n-1$

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

$$= 2^n - 1$$

still need induction to check.

Fibonacci numbers $T(n) = T(n-1) + T(n-2)$

$$T(0) = T(1) = 1$$

— has a closed form using $\sqrt{5}$

Linear recurrence equations with constant coefficients.

$T(n)$ is defined as a linear combination of smaller calls $T(n-k$

$$= \boxed{\cancel{\sum_{k=1}^{n-1} a_k T(k)}} + f(n)$$ plus some function of n.

for $a_i \in \mathbb{R}$

$$\Downarrow$$

$$\sum_{i=1}^{k} a_i T(n-i) + f(n)$$

For Fibonacci (homogeneous)
$a_1 = 1 \quad a_2 = 1 \quad k = 2 \quad f(n) = 0$

For TOH (non homogeneous)
$a_1 = 2 \quad \cancel{a_2} \quad k = 1 \quad f(n) = 1$

k = degree of the recurrence
= # of base cases needed.

$f(n) = 0$ iff recurrence is called <u>homogeneous</u>. $T(n) = 2T(n-1)+1$

Example recurrence $T(n) = 3T(n-1) - 2T(n-2)$

$T(0) = 1 \quad T(1) = 2$

<u>Step 1</u> Guess $T(n) = r^n$

plug into recursive eqn to get constraint on r:

$$T(n) = 3T(n-1) - 2T(n-2)$$

$$r^n = 3r^{n-1} - 2r^{n-2}$$

characteristic equation

$$r^2 = 3r - 2 \Rightarrow \boxed{r^2 - 3r + 2 = 0} \quad (r-2)(r-1) = 0$$

so $T(n) = c_1 2^n$ matches the recursive eqn.

$T(n) = c_2 1^n$ also matches.

so generally $T(n) = \underline{c_1 2^n + c_2 (1)^n}$ matches the recursive equation.

- this is a whole family of sequences, one for each choice of c_1 & c_2

but only one will match the base cases.

$$\left.\begin{aligned} T(0) &= c_1 2^0 + c_2 1^0 = 1 \\ T(1) &= c_1 2^1 + c_2 1^1 = 2 \end{aligned}\right\}$$ find c_1 & c_2 plug back in