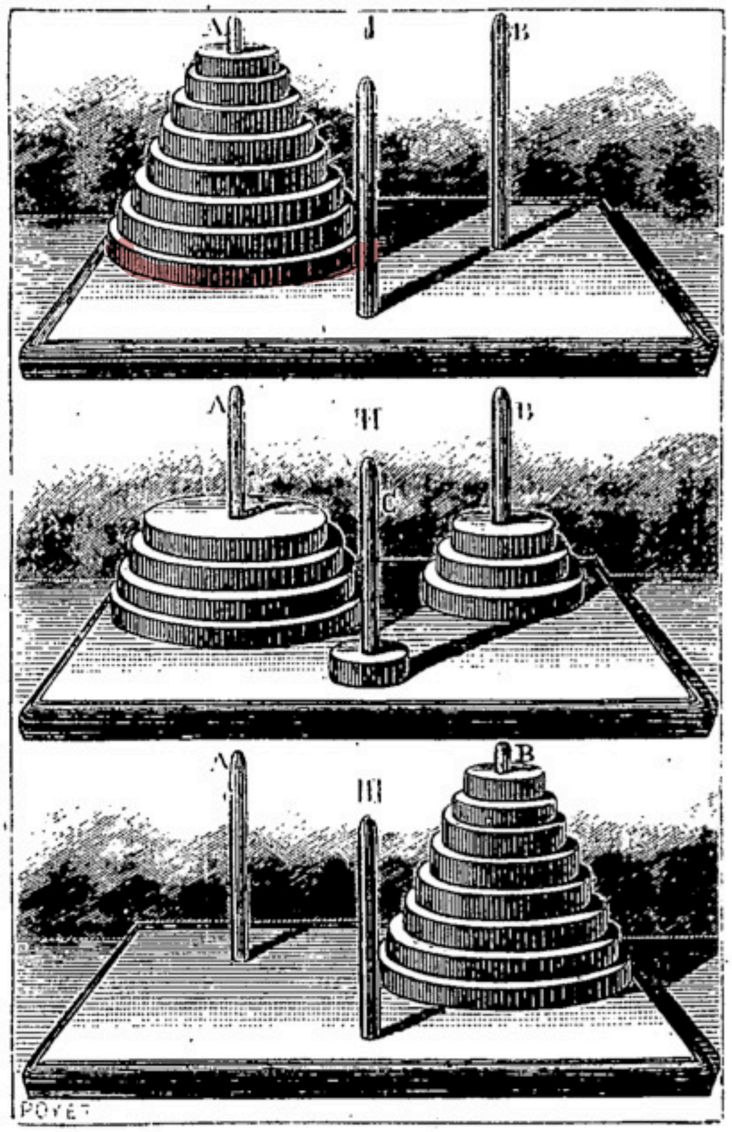
HW5 } out this afternoon due next Tuesday 8pm
GPS 5 }

Algorithms:

Proofs: INDUCTION
Algorithms: RECURSION

Tower of Hanoi Lucas (1887)



Move one disk at a time

Never put a disk on top of a smaller disk

Move n disks from A to B (via C)

if $n>0$:
- Move $n-1$ disks from A to C (via B)
- Move disk n from A to B
- Move $n-1$ disks from C to B (via A)

Recursion!

Smaller instances of exactly the same problem.

$T(n) = \#$ moves

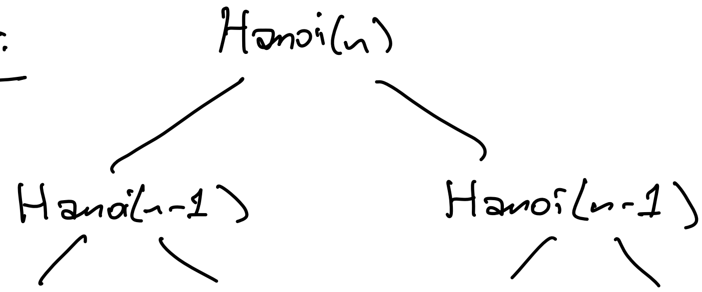
n	0	1	2	3	4	5	6
$T(n)$	0	1	3	7	15	31	63

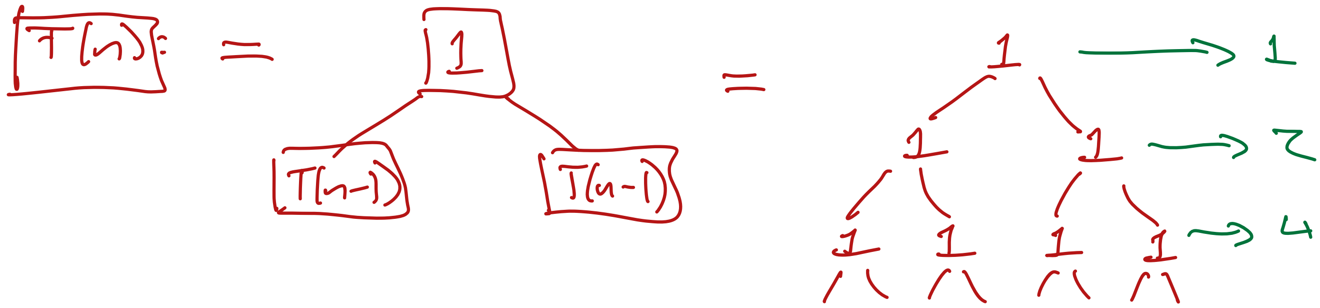
$T(n) = 2T(n-1) + 1$

Guess: $T(n) = 2^n - 1$

Proof: Let n be any non-neg int
Assume $T(m) = 2^m - 1$ for all $m<n$
Two cases:
- $n=0 \Rightarrow T(0) = 0 = 2^0 - 1$ ✓
- $n>0 \Rightarrow T(n) = 2T(n-1) + 1$
$= 2(2^{n-1} - 1) + 1$ [IH]
$= 2^n - 1$ ✓

Recursion Tree:





increasing geom. series

Only largest term matters

n levels $\longrightarrow$ 2^n work at last level

so $\boxed{O(2^n)\text{ moves}}$

Input:	S	O	R	T	I	N	G	E	X	A	M	P	L
Divide:	S	O	R	T	I	N	G	E	X	A	M	P	L
Recurse Left:	I	N	O	R	S	T	G	E	X	A	M	P	L
Recurse Right:	I	N	O	R	S	T	A	E	G	L	M	P	X
Merge:	A	E	G	I	L	M	N	O	P	R	S	T	X

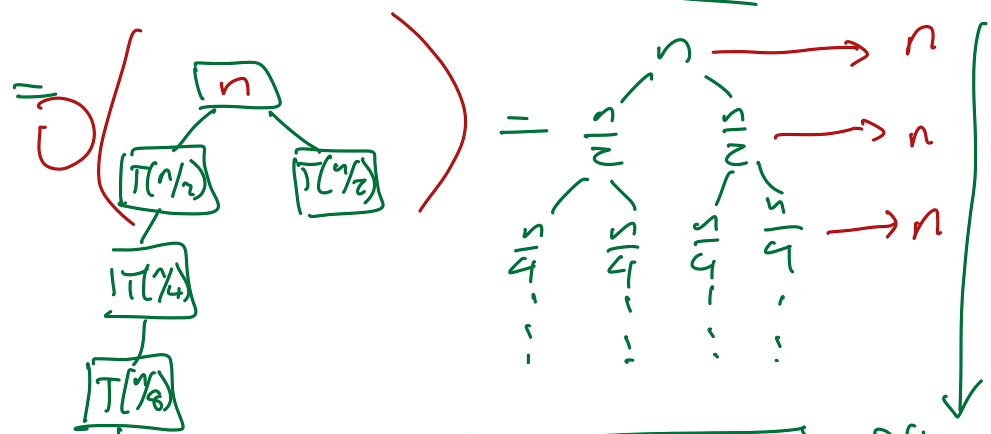
```
T(n) →      MERGESORT(A[1..n]):
              if n > 1
                m ← ⌊n/2⌋
T(n/2) →        MERGESORT(A[1..m])        ⟨⟨Recurse!⟩⟩
T(n/2) →        MERGESORT(A[m+1..n])      ⟨⟨Recurse!⟩⟩
O(n) →          MERGE(A[1..n], m)
```

$$T(n) = 2T(\tfrac{n}{2}) + \underbrace{O(n)}_{\leq c \cdot n \text{ for some } c}$$

$$\cancel{= T(\lceil \tfrac{n}{2} \rceil) + T(\lfloor \tfrac{n}{2} \rfloor) + O(n)}$$

$T(n) = O(1)$ for all $n < 10^{100}$?

$\boxed{T(n)} =$



$\log_2 n =$
~~starting~~ with n
times we can divide by 2 until we reach ≤ 1

$\Rightarrow \boxed{T(n) = O(n \log n)}$

$O(\log n)$ levels

	MERGE($A[1..n], m$):	
	$i \leftarrow 1$; $j \leftarrow m+1$	
	for $k \leftarrow 1$ to n	$O(n)$ iteration
R empty	if $j > n$	$O(1)$ per iteration
	$B[k] \leftarrow A[i]$; $i \leftarrow i+1$	
L empty	else if $i > m$	
	$B[k] \leftarrow A[j]$; $j \leftarrow j+1$	
$L[1] < R[1]$	else if $A[i] < A[j]$	
	$B[k] \leftarrow A[i]$; $i \leftarrow i+1$	
$L[1] > R[1]$	else	
	$B[k] \leftarrow A[j]$; $j \leftarrow j+1$	
	for $k \leftarrow 1$ to n	
	$A[k] \leftarrow B[k]$	

if L and R are not both empty:

Move min(L[1], R[1]) to output

Recursivey merge rest of L and R

Input:	S	O	R	T	I	N	G	E	X	A	M	P	L
Choose a pivot:	S	O	R	T	I	N	G	E	X	A	M	P	L
Partition:	A	G	O	E	I	N	L	M	P	T	X	S	R
Recurse Left:	A	E	G	I	L	M	N	O	P	T	X	S	R
Recurse Right:	A	E	G	I	L	M	N	O	P	R	S	T	X

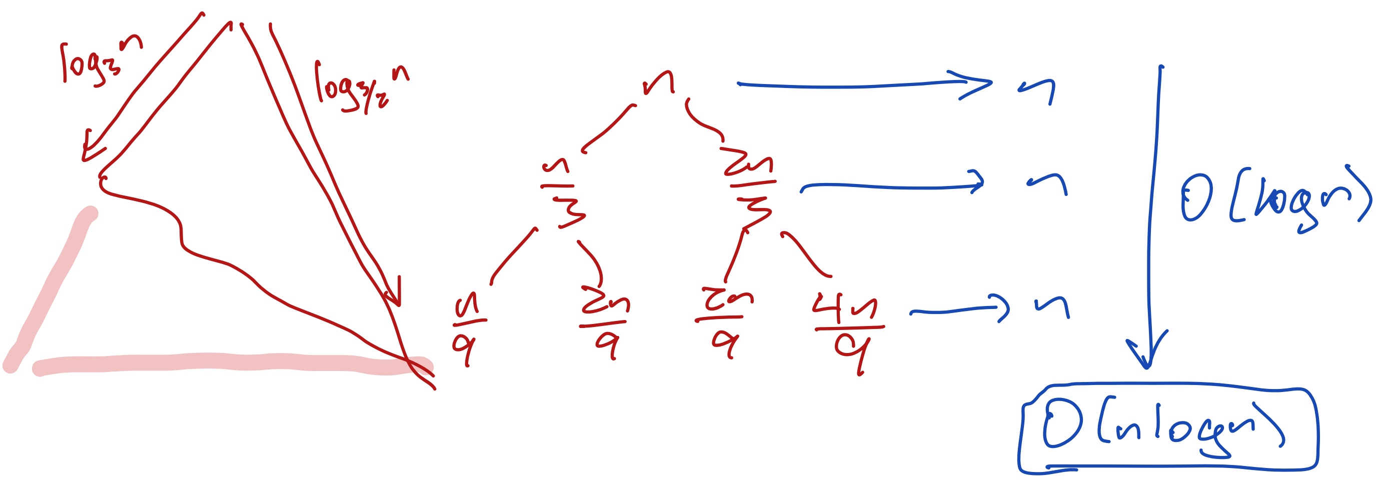
<u>QuickSort($A[1..n]$):</u>
  if ($n > 1$)
    Choose a pivot element $A[p]$
    $r \leftarrow$ Partition(A, p)
    QuickSort($A[1..r-1]$) ⟨⟨*Recurse!*⟩⟩
    QuickSort($A[r+1..n]$) ⟨⟨*Recurse!*⟩⟩

$$T(n) = \max_r \left(T(r-1) + T(n-r) \right) + O(n)$$

$$\leq T(0) + T(n-1) + O(n) = O(n^2)$$

If we could guarantee $\frac{n}{3} \leq r \leq \frac{2n}{3}$

$$T(n) \leq T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) + O(n)$$



```
PARTITION(A[1..n], p):
    swap A[p] ↔ A[n]
    ℓ ← 0                    ⟨⟨#items < pivot⟩⟩
    for i ← 1 to n − 1
        if A[i] < A[n]
            ℓ ← ℓ + 1
            swap A[ℓ] ↔ A[i]
    swap A[n] ↔ A[ℓ + 1]
    return ℓ + 1
```