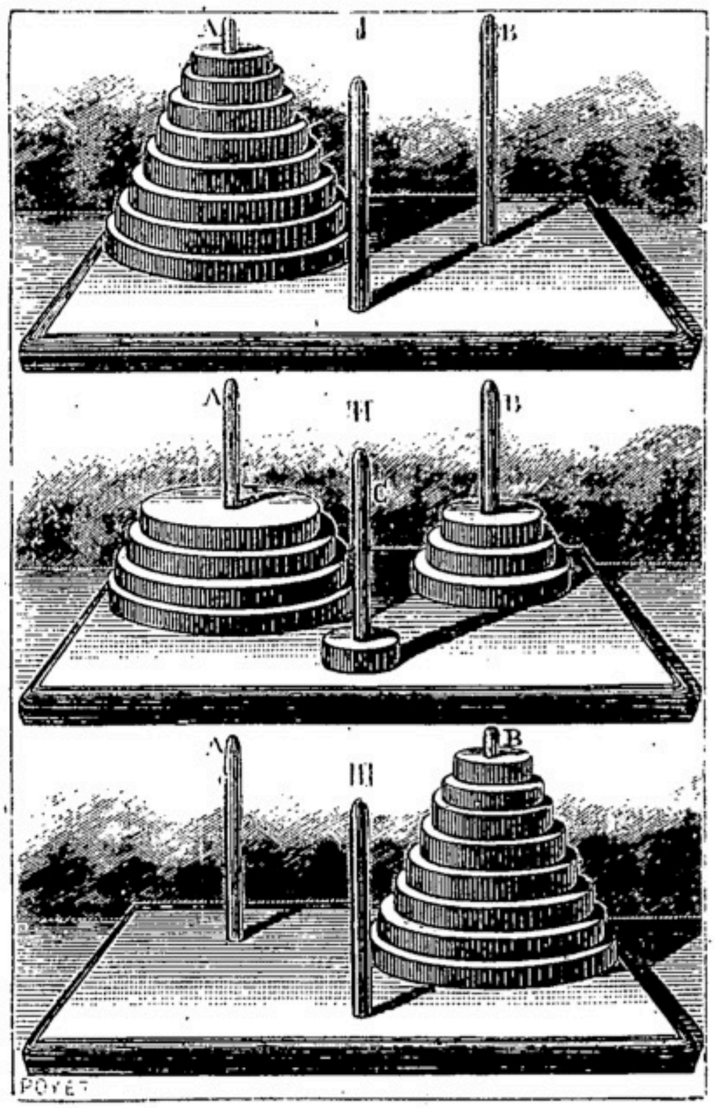
HW4 out later today — due next Tue 8pm

MT1 + solutions also

II. Algorithms

Recursion = Induction

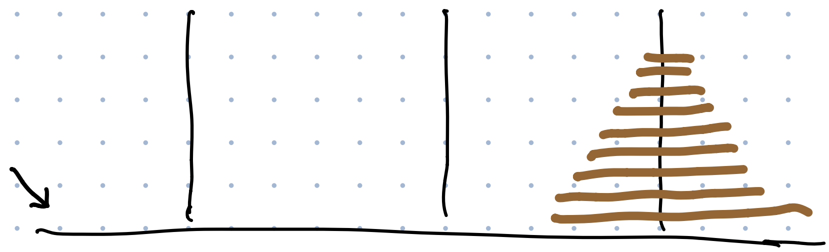
Tower of Hanoi (Lucas 1890s)



Move one disk at a time

Always top disk on any peg

Never larger on top of smaller



Hanoi(n, src, dst, tmp)

if n > 0

Hanoi(n-1, src, tmp, dst) ← MAGIC!

1 → move disk n from src to dst

Hanoi(n-1, tmp, dst, src) ← MAGIC!

Running time (# moves)

$T(n)$ = # moves to get n disks from src to dst

$$T(n) = \begin{cases} 0 & \text{if } n=0 \\ T(n-1)+1+T(n-1) & \text{if } n>0 \end{cases}$$

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

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

Proof: Let n be any non-neg int

Assume $T(k) = 2^k - 1$ for all $k < n$

Two cases:

- $n = 0$ $\quad T(n) = 0 = 2^0 - 1$ ✓
- $n > 0$ $\quad T(n) = 2T(n-1) + 1$

$$= 2(2^{n-1} - 1) + 1 \quad [IH]$$

$$= 2^n - 1$$

Done.

Time is $O(2^n)$

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

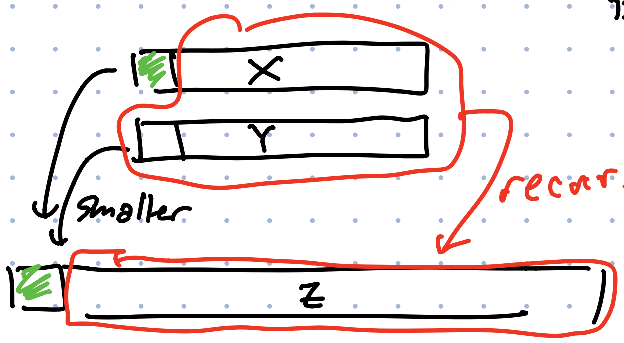
base case

```
MergeSort(A[1..n]):
  if n > 10000000
    m ← ⌊n/2⌋
    MergeSort(A[1..m])        ⟨⟨Recurse!⟩⟩
    MergeSort(A[m+1..n])      ⟨⟨Recurse!⟩⟩
    Merge(A[1..n], m)
```

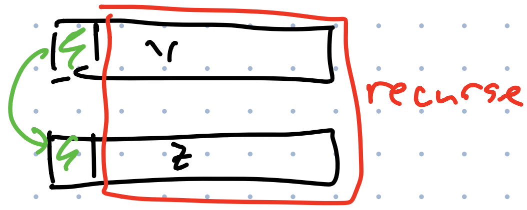
else BubbleSort

```
Merge(A[1..n], m):
  i ← 1; j ← m+1
  for k ← 1 to n
    if j > n
      B[k] ← A[i]; i ← i+1
    else if i > m
      B[k] ← A[j]; j ← j+1
    else if A[i] < A[j]
      B[k] ← A[i]; i ← i+1
    else
      B[k] ← A[j]; j ← j+1
  for k ← 1 to n
    A[k] ← B[k]
```

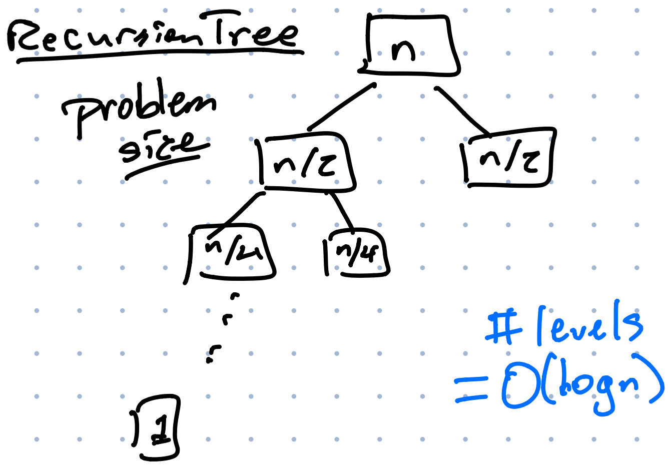
$\boxed{O(n)}$

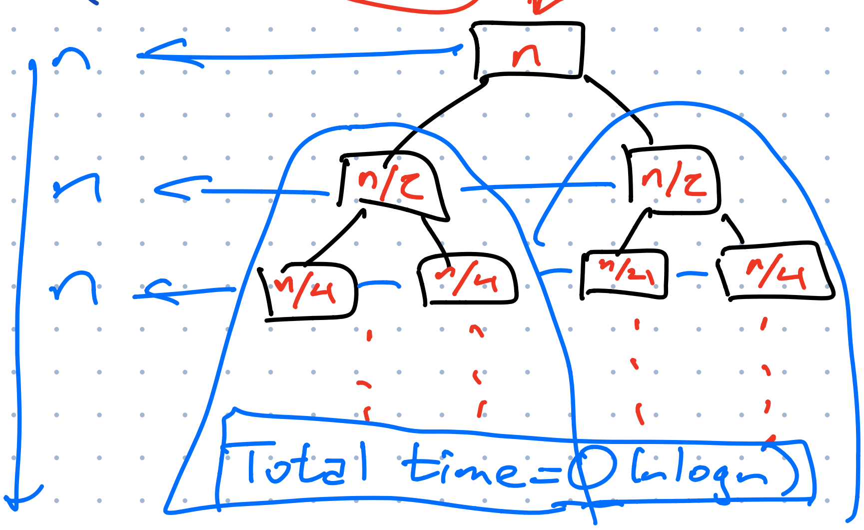


if x is empty



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





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	$\boxed{\text{P}}$	L
Partition:	A	G	O	E	I	N	L	M	$\boxed{\text{P}}$	T	X	S	R
Recurse Left:	A	E	G	I	L	M	N	O	$\boxed{\text{P}}$	T	X	S	R
Recurse Right:	A	E	G	I	L	M	N	O	$\boxed{\text{P}}$	R	S	T	X

```
QuickSort(A[1..n]):
  if (n > 1)
      Choose a pivot element A[p]
      r ← Partition(A, p)
      QuickSort(A[1..r − 1])      ⟨⟨Recurse!⟩⟩
      QuickSort(A[r + 1..n])      ⟨⟨Recurse!⟩⟩
```

unspecified

Practice : whatever

Good practice : random

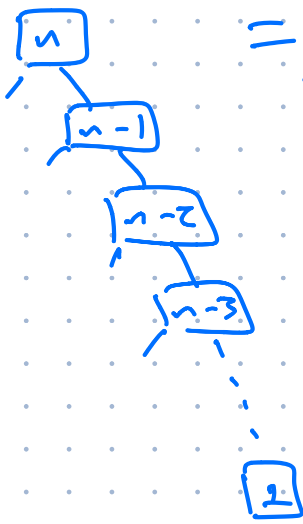
Theoretical practice : median

```
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
```

$\boxed{O(n)}$

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

$$= O(n) + T(0) + T(n-1)$$



$$= \underline{O(n^2)}$$