Dr. Walid

Lec 7 Recursion & Applications of Stacks

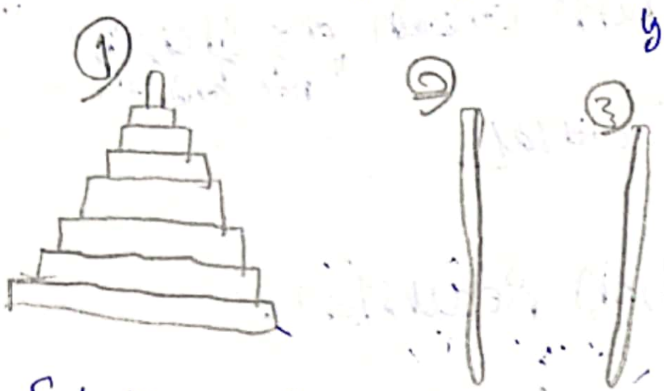
- the great importance of stacks in OS -

Say -- OS called 'A' program and 'A' calle B ... when B finish OS will return to 'A' -- so on "Last In First Out."

APP 1. "tower of Hanoi"

You have to move all disks to tower 3 but:

1- only one disk should be removed at a time
2- No disk should be placed on top of a smaller



Sol. Recursion

MoveDisks(6,1,2,3);
Move disk 7 to 3 from 1
MoveDisks(6,2,3,1); ...

"we need 127 step to move 7"

no# of steps = $2^n - 1$

" to move 100 we need 1×10^{30} " step

10 → 1024   8 → 255
7 → 127   9 → 511

6 → 63   6 + (36 - 1) n^2
5 → 31 $n(2)$
4 → 15 $n + (6-1)$ n^2
3 → 7   2 $n(n-1)$
2 → 3   2 + (2^2 - 1) $3 + 2^{n^2}$ n+9 (13)
1 → 1

Depth o stack: ~~the highest of~~ the length of the longest branch.

!! we can solve it iteratively → yes cuz we need to build stack

When iterative better than recursion:

" if problem does not require stack "
" if the graph contain one branch "
" not condition "

Iterative vs Recursive Factorial
n steps n steps

But iterative faster than Recursion.

Iterative vs Recursive Fibonacci
n steps c^n steps

"Designing Recursive Algo."

- find base cond.
- check termination
- Draw a recursion tree
- Find a stepping rule
- outline your algo.

Tail recursion: IF last statement calling same fn
You can solve it recursive.
without (Removing tail)