RECURSION Concepts & Qns

Video
4

"मैं, DSA की शपथ लेता हूं कि मैं जो पढ़ाउगा बहोत अच्छे से पढ़ाउगा।"

Facebook
Instagram → codestorywithMIK

(Twitter) → CSwithMIK

codestorywithMIK →

Motivation (भाषण)

- The only way to achieve the impossible is to believe, it is possible.

- who told you, you can't do it

  " करके तो देखो,
  को रोक नही पाएगा तुम्हें। "

#codestorywithMIK

Time & Space Complexity of Recursive Functions

Example-1:- Factorial (n)   Time Complexity

Example-1 :- Factorial (n)

Time Complexity:

```
int Factorial (int n){
    if(n<=1)
        return 1;

    return n * Factorial(n-1);
}
```

1

1

1

$$T(n) = \underbrace{T(n-1)}\; + 3$$

$$= \{T((n-1)-1) + 3\} + 3$$

$$= T(n-2) + 6$$

$$= \{T((n-2)-1) + 3\} + 6$$

$$T(n) = T(n-3) + \cancel{8}\,9$$

$$= T(n-4) + 12$$

$$T(n) = T(n-K) + 3*K$$

$$\vdots$$

$$= T(0) + \cancel{2}\,3 * n$$

$$= 1 + 2*n$$

$$\simeq O(2*n)$$

$$T(n) \simeq O(n)$$

Space Complexity :-

F(n)

F(n-1)

F(n-2)

F(n-3)

⋮

F(1)

Recursion Tree -

depth of tree

$O(n)$

$S.C = O(n)$

Example-2 :- Fibonacci (n)

```
int Fib(n){
    if(n==0)
        return 0;
    if(n==1)
        return 1;
    return Fib(n-1) +
           Fib(n-2);
}
```

1 1 1 1

Time Complexity :-

$$T(n) = T(n-1) + T(n-2) + 4$$

$T(n) = 2T(n-2) + 4$

$T(n-1) \approx T(n-2)$ (approximation) → lower bound.

$T(n) = 2 * \boxed{T(n-2)} + C$

$= 2 * \{\underbrace{2T(n-4)+4}\} + 4$

$= 4T(n-4) + 3*C$

let $C = 4$

$= 8T(n-6) + 7*C$

$= 16T(n-8) + 15*C$

$= \vdots$

$K = 4$

$2^K T(n-2K) + (2^K - 1) * C$

$2^{n/2} T(n - 2 \cdot n/2) + (2^{n/2} - 1) * C$

$(n-2K) = 0$

$n = 2K$

$K = n/2$

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

$\propto 2^{n/2}$

$T.C = O(2^{n/2})$ → (Lower Bound)

$T(n-2) \approx T(n-1)$ (upper bound)

$$T(n) = 2 * T(n-1) + c \qquad (\text{let, } 4 = c)$$

$$= 2\{2\,T(n-2) + c\} + c$$

$$= 2^2\,\underbrace{T(n-2)} + 3c$$

$$= 2^2\{2 * T(n-3) + c\} + 3c$$

$n-k=0$
$n=k$

$$= 2^3 * T(n-3) + 7c$$

$\vdots$

$$= \boxed{2^k * T(n-k) + (2^k-1)\,c}$$

$n-k=0$
$n=k$

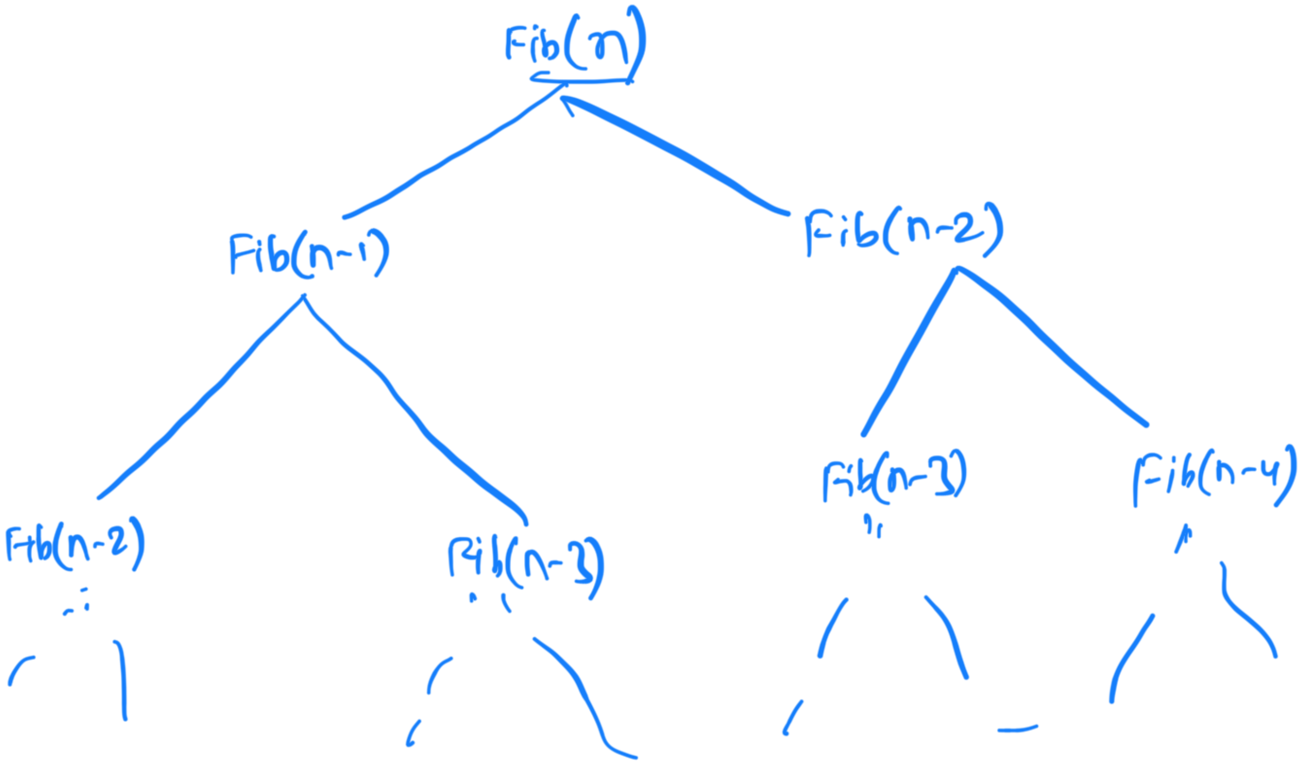
$$= 2^n * \underbrace{T(n-n)} + (2^n-1)\,c$$

$$= 2^n * 1 + (2^n-1)\,c$$

$$T(n) \propto O(2^n) \rightarrow (\text{upper bound}).$$

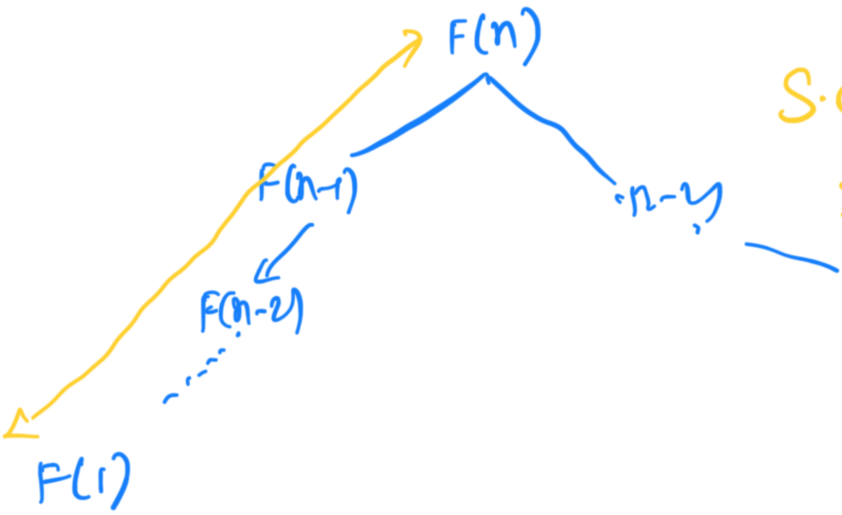
Exponential grow.

Exponential



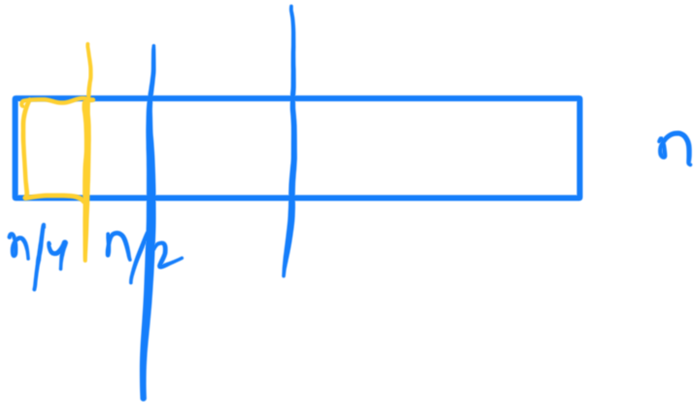
Space Complexity :-

Max depth of recursion tree.



$S.C = O(n)$

Example - 3 "Binary Search"



$$T(n) = T(n/2) + C$$

$$= T(n/4) + 2c$$

$$= T(n/8) + 3c$$

$$\vdots$$

$$\underbrace{T(n/2^k)} + k*c$$

$$n/2^k = 1$$

$$n = 2^k$$

$$\boxed{k = \log_2 n}$$

$$T\left(n/2^{\log_2 n}\right) + \log_2 n * c$$

$$\boxed{T\left(\cancel{n}/\cancel{n}\right)} + \underbrace{\log_2 n + c}$$

$$a^{\log_a c} = c \qquad = \log_2 n$$

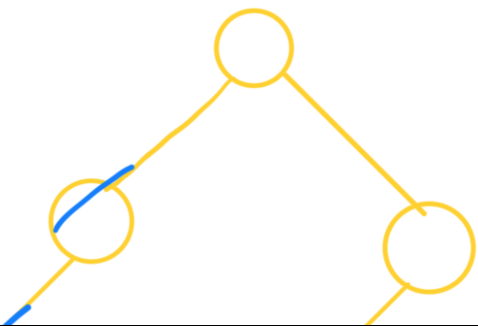
$$T(n) \simeq \underline{\underline{O(\log_2 n)}}$$

depth = $\log_2(n)$

$$S.C = O(\underline{\underline{\log_2(n)}})$$

Example-4 (Binary Tree Traversal)

(Inorder Traversal)



$$\boxed{T.C \simeq O(n)}$$

n nodes

S.C :- O(depth of tree).