Dynamic Programming

- Optimization problems
  - (minima & maxima)
- Overlapping subproblems

$T(n)$

1	2	3	4	5	fib. series
1,	1,	2,	3,	5,	8, - - - - - -

fib(n):

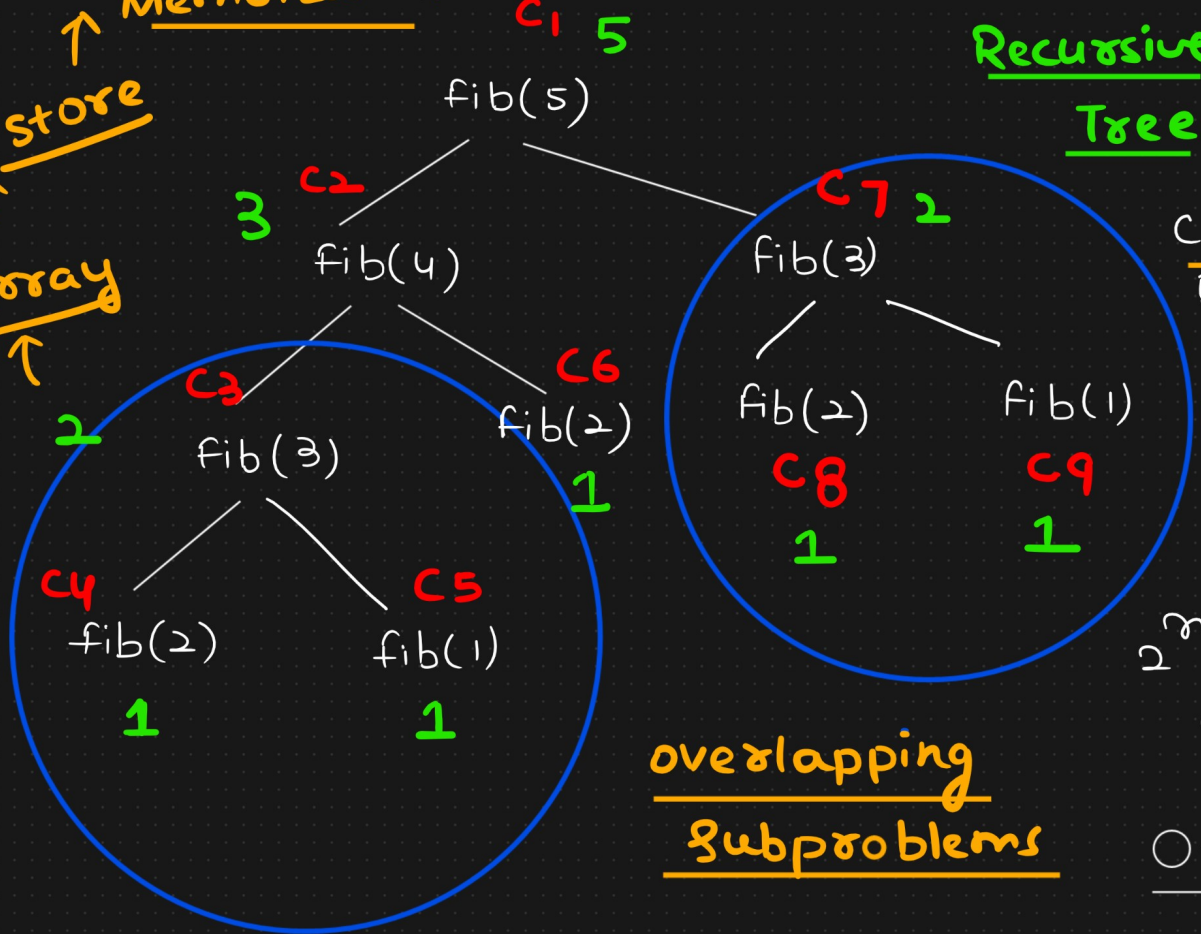
```
c { n==1 or n==2
      result=1
  Otherwise ---> n>2
T(n-1)+T(n-2) { result = fib(n-1) + fib(n-2)
                return result
```

- Recursion

$n = 4$ → output = 3

Memoization

- store
- array



Recursive Tree

Complete Binary Tree

$2^n - 1$

2^n function calls

$O(2^n)$

overlapping subproblems

Note: $k = 2^n - 1$ (CBT)

1) Recursion (store the results)
↳ overlapping subproblems
(Memoization)

Merge Sort
↳ No Overlapping Subproblem
↳ Divide & conquer

↳ Top Down Approach

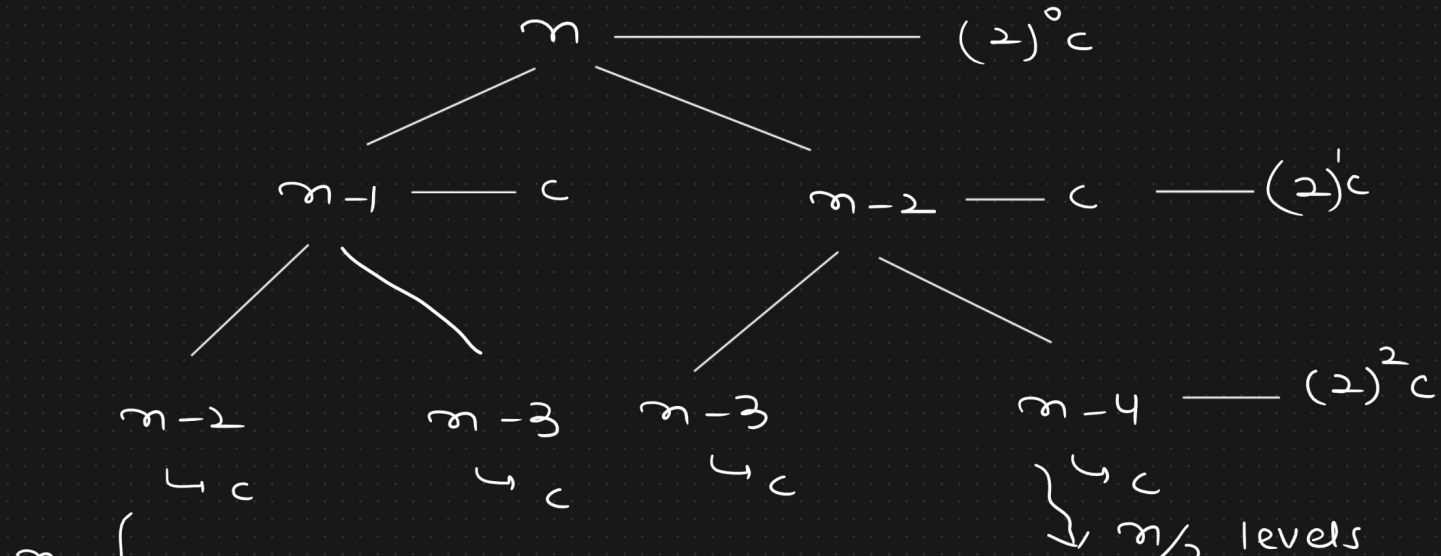
2) Tabulation (Bottom up Approach)
↳ Not Recursion

1	2	3	4	5	6
1	1	2	3	5	8

Recurrence Relation :- (Recursion)

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

Recursive Tree Approach



Left side

$n - k = 1$

$k = n - 1$

Right side

$n - 2k = 1$

$\frac{n-1}{2} = k$

$c(2^0 + 2^1 + 2^2 + \dots + 2^k)$ $k = n$ (max)

$c(2^0 + 2^1 + 2^2 + \dots + 2^n)$

⇓

GP series

$r = 2$ $r > 1$

$$S = \frac{a(r^n - 1)}{r - 1} = \frac{2^n - 1}{1}$$

$\Rightarrow O(2^n)$

↳ Exponential Time complexity

$O(n)$

fib(5)
↓
fib(4)
↓
fib(3)
↓
fib(2)
↓
fib(1)

(Recursion)
Exponential Time complexity
⇓
Linear Time complexity
(Dynamic Programming)