Fibonacci Series

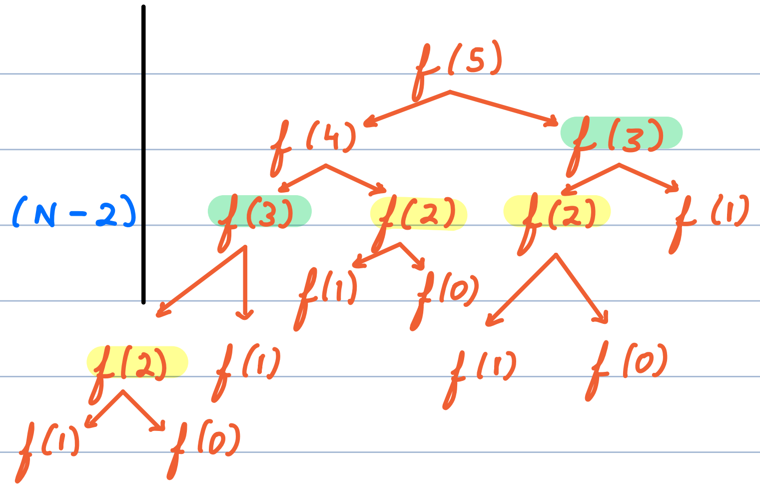
N →	0	1	2	3	4	5	6	7
Ans →	0	1	1	2	3	5	8	13

$$\boxed{fib(i) = fib(i-1) + fib(i-2)}$$

```
int fib(N) {
    if (N <= 1) return N
    return fib(N-1) + fib(N-2)
}
```

TC = $O(2^N)$

SC = $O(N)$



DP identification

✓ 1) Optimal Substructure → Solving the problem by dividing into smaller subproblems.

✓ 2) Overlapping Subproblems → same subproblem repeating multiple times.

DP ← { ⇒ store answer of subproblem & reuse it.

// F[N+1]

∀i, F[i] = -1

```
int fib(N) {
    if (N <= 1) return N
    if (F[N] != -1) return F[N]
    F[N] = fib(N-1) + fib(N-2)
    return F[N]
}
```

TC = O(N)   SC = O(N)

Types

1) Top-Down / Recursive →

Easy to write & understand.

a) Its a recursive solution.

b) Start with actual problem & break it down till we reach base case.

c) Use base case & recursively solve for subproblems & actual problem.

2) Bottom-Up / Iterative →

No recursion space i.e. possibility to optimize SC.

a) Its iterative solution.

b) Start with smallest subproblem & iteratively calculate the answer of bigger problems till we reach the actual answer.

```
F[0] = 0    F[1] = 1
for i → 2 to N {
    F[i] = F[i-1] + F[i-2]
}
return F[N]
```

TC = O(N)   SC = O(N)

0 1 2 3

a b c

a b c

```
a = 0        b = 1
for i → 2 to N {
    c = a + b
    a = b
    b = c
}
return c
```

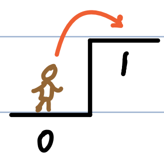
TC = $\underline{O(N)}$ SC = $\underline{O(1)}$

Q → Find #ways to climb N stairs if in 1 step we can move by 1 or 2 stairs.

Ad. Robotics

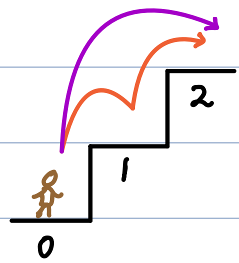
N = 1

Ans = $\underline{1}$



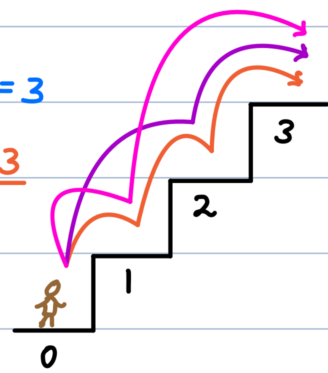
N = 2

Ans = $\underline{2}$



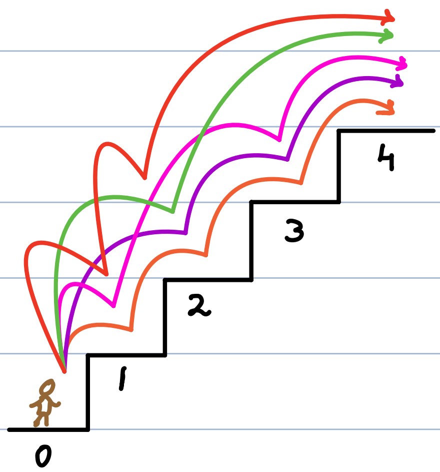
N = 3

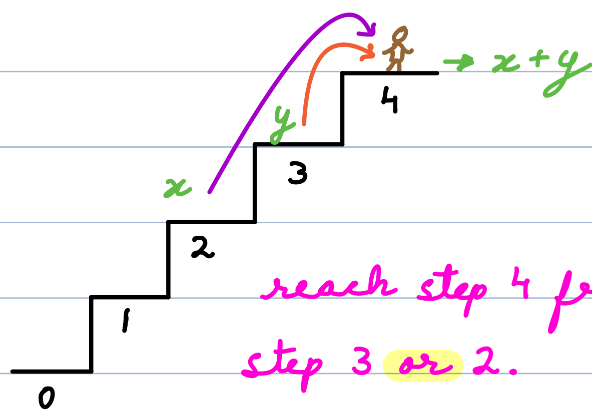
Ans = $\underline{3}$



N = 4

Ans = $\underline{5}$





reach step 4 from step 3 or 2.

ways (N) = ways (N-1) + ways (N-2) ← fibonacci seq.

ways (0) = 1 ways (1) = 1

(# ways to do a task = 0) ⇒ impossible task

Q → Find minimum count of perfect squares required to get sum = N.

$N = 5$ → $1^2 + 1^2 + 1^2 + 1^2 + 1^2$
→ $2^2 + 1^2$ ✓ Ans = 2

$N = 10$ → $1^2 + 1^2 + \ldots + 1^2$ (10 times)
→ $2^2 + 1^2 + 1^2 \ldots 1^2$ (6 times)
→ $2^2 + 2^2 + 1^2 + 1^2$
→ $3^2 + 1^2$ ✓ Ans = 2

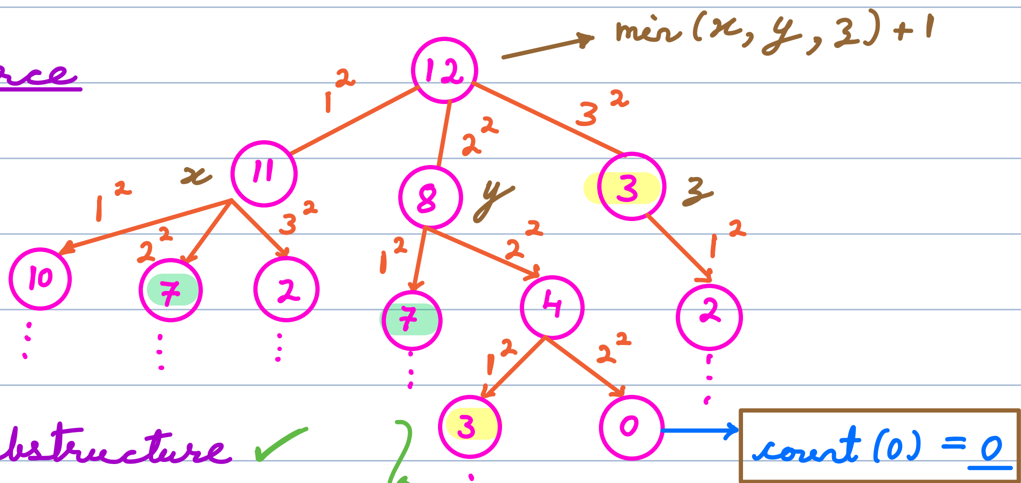
Greedy → select large perfect squares. ✗

$N = 50$ $\quad 50 - 7^2 = 1 - 1^2 = 0$ $\quad$ Ans = 2

$N = 12$ $\quad 12 - 3^2 = 3 - 1^2 = 2 - 1^2 = 1 - 1^2 = 0$

$2^2 + 2^2 + 2^2 = 12$ $\quad$ Ans = 3

Bruteforce



optimal substructure ✓
overlapping subproblems ✓

use DP

$$\boxed{\begin{array}{c} count(N) = \min\ count(N - x^2) + 1 \\ \forall x,\ s.t\ x^2 <= N \end{array}}$$

```
cnt[0] = 0
for i -> 1 to N {
    cnt[i] = i  ✓
    for (x = 1 ; x*x <= i ; x++) {
        cnt[i] = min(cnt[i], cnt[i - x*x] + 1)
    }
} return cnt[N]
```

8   8 $8 - 2^2$

TC = $O(N * \sqrt{N})$

SC = $\underline{O(N)}$

0	1	2	3	4	5	6	7	8
0	1	2	3	~~4~~ 1	~~5~~ 2	~~6~~ 3	~~7~~ 4	~~8~~ ~~5~~ 2

Ans = $\underline{2}$