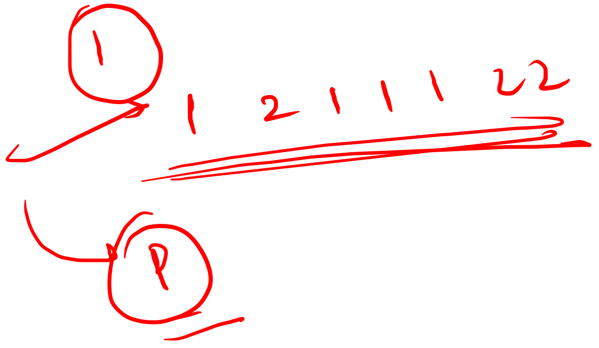
Climbing Stairs

⇒ You are climbing a stair of (n steps)

↳ either you can step 1 } at one
step 2 } time

n = 100

$1 \quad 2 \quad 1 \quad 1 \quad 1 \quad 2 \quad 2$

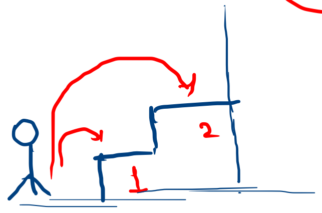


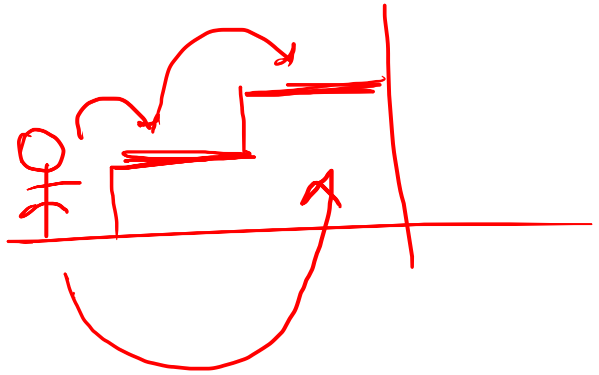
I/p : $\boxed{n = 2}$

↳ 1 way → (1+1)

↳ 2 way → (2)

⇒ (2)

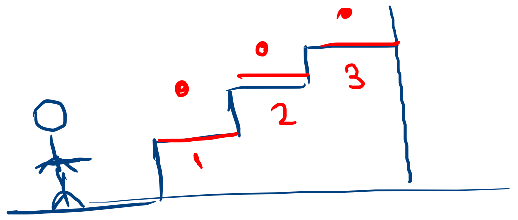


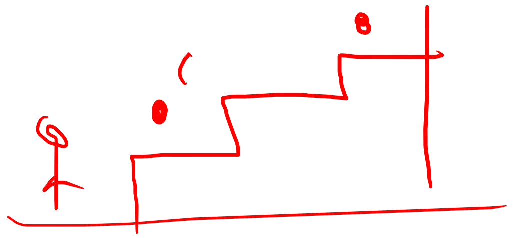


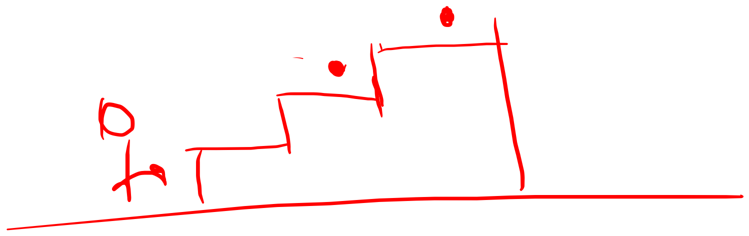
I/p n = 3

O/p :

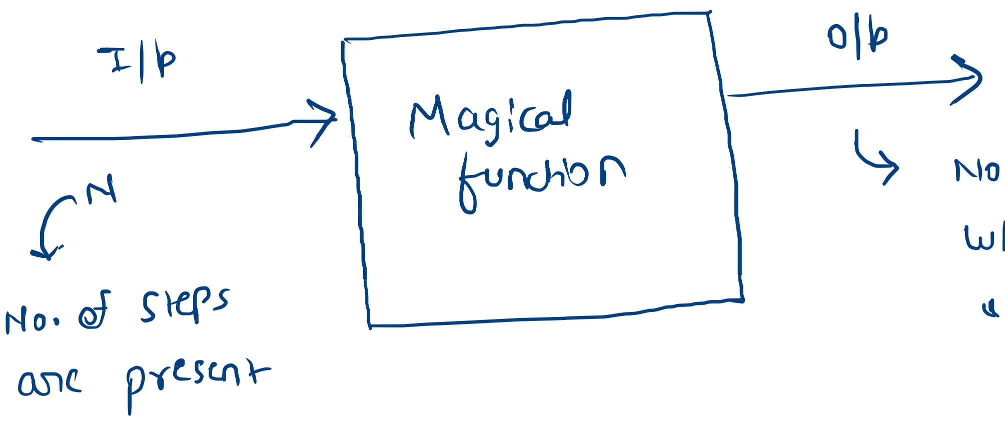
$$\left.\begin{array}{l} 1+1+1 \\ 1+2 \\ 2+1 \end{array}\right\} \textcircled{3}\ \text{way}$$





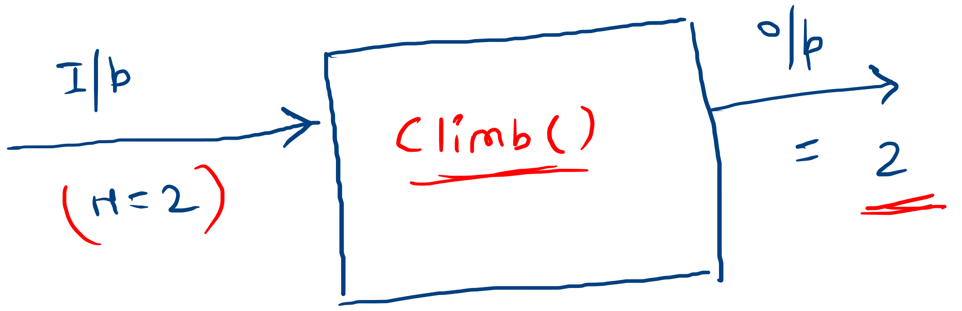


Recursive Solution



No. of steps are present

No. of ways in which we can climb "N" steps in a stair

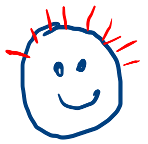


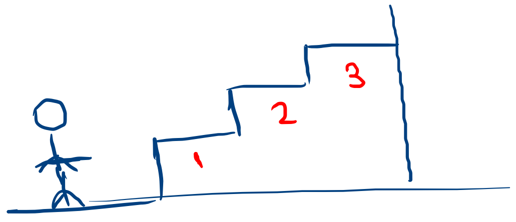
we have total 2 No. of ways to climb stairs having "N" steps

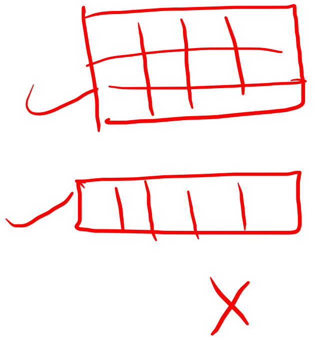
$$\left\{ \begin{array}{l} 1+1 \\ 2 \end{array} \right\}$$

Logical Part

I/P: $N = 3$





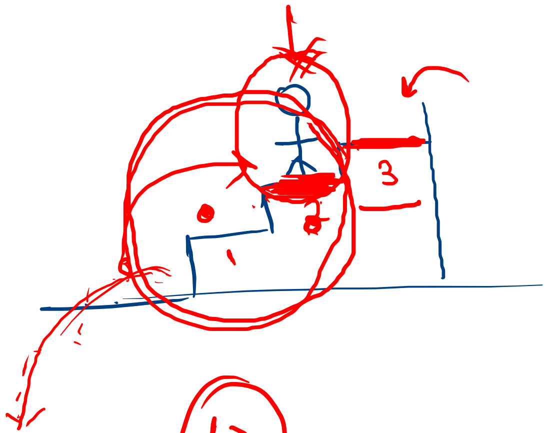


No. of Steps

Ways

1 Step + 1 Step] + 1

2 Step] + 1 Step

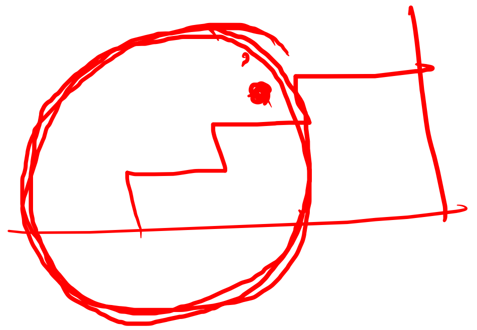


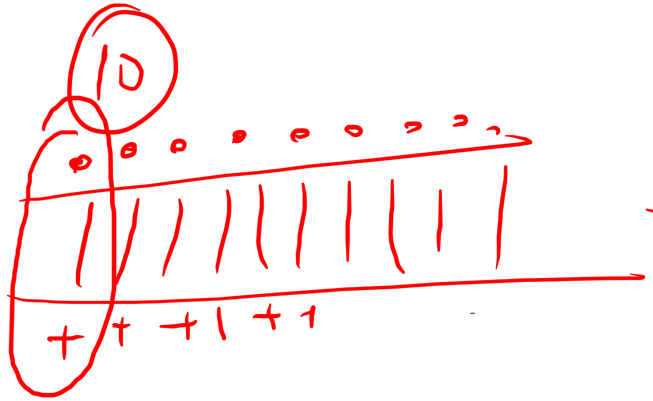
No. of ways to reach to $\boxed{N-1}$ steps

$N=?$

$N=2$

$N=3$





Recursion

↓

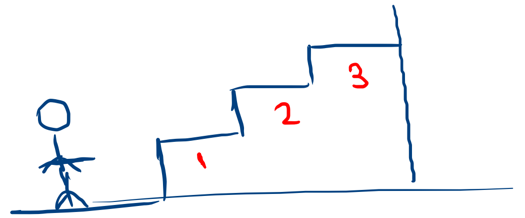
DP

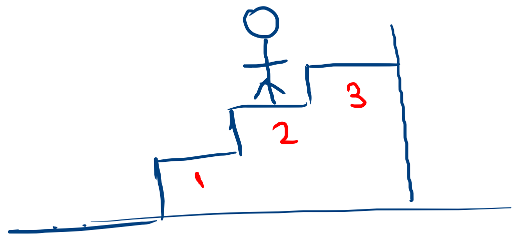
Logical Part ☺

I/p: N = 3

✓ 1 step + 1 step

✓ 2 step



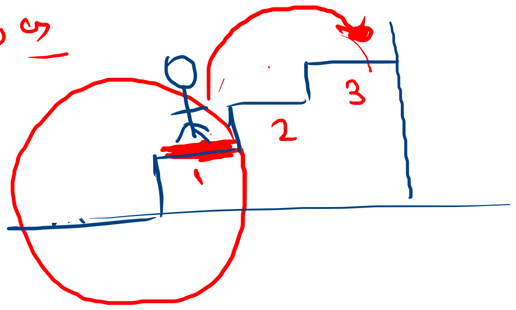


No. of ways to reach to $\boxed{N-1}$ steps = ⓜ

return $\boxed{m+n}$

1 way [$\boxed{1 \text{ step}}$] + 2 step

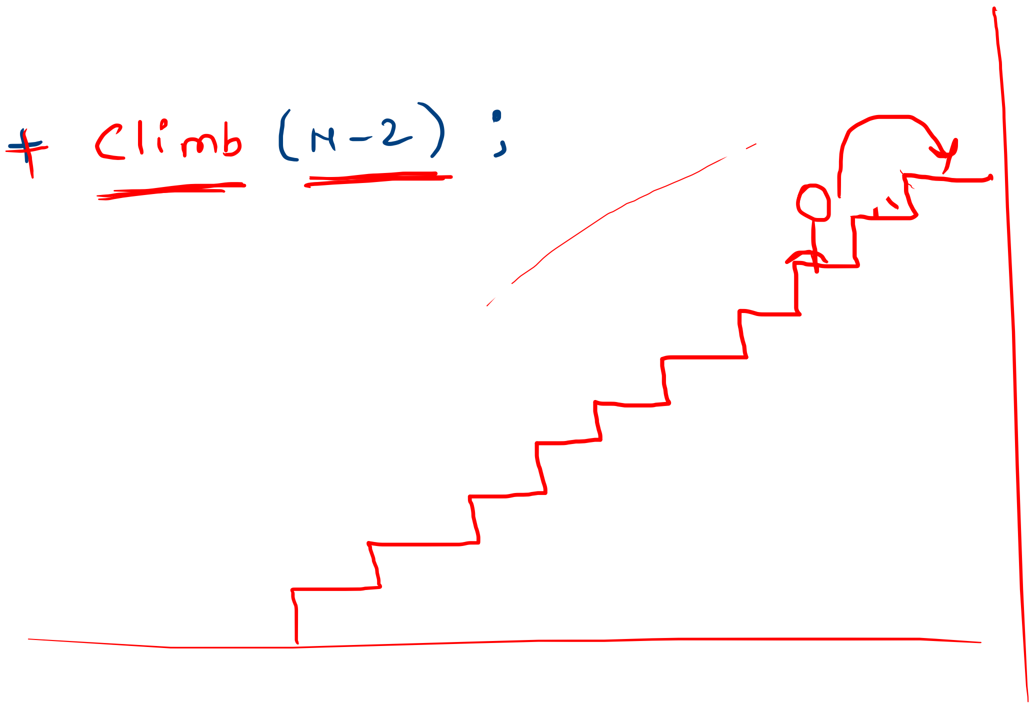
1 way



No. of ways to reach to $\boxed{N-2}$ steps = ⓝ

Recursive
Code

```
int climb (int N)
{
    // Base condition →
    return climb (N-1) + climb (N-2) ;
}
```



Base condition

: Smallest Valid Input → check output

```
int climb (int N)
{

}
```

N → No. of steps in a stair

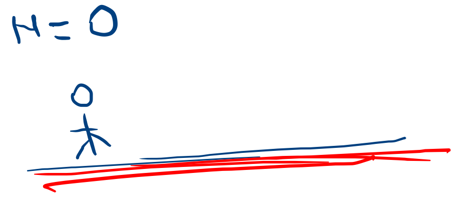
N = 0 , 1 , 2 , 3

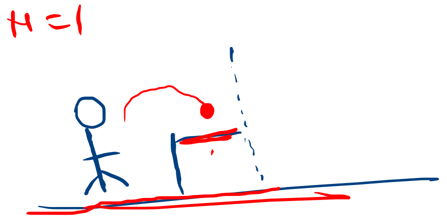
↓ 0, ↓ 1, ↓ 2

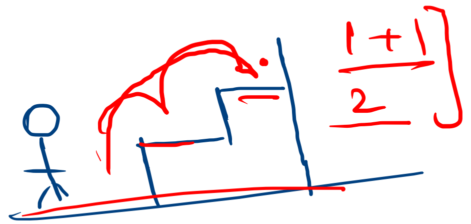
No. of ways

No. of ways

```
if (N<=2) return N;
```







Recursive Code

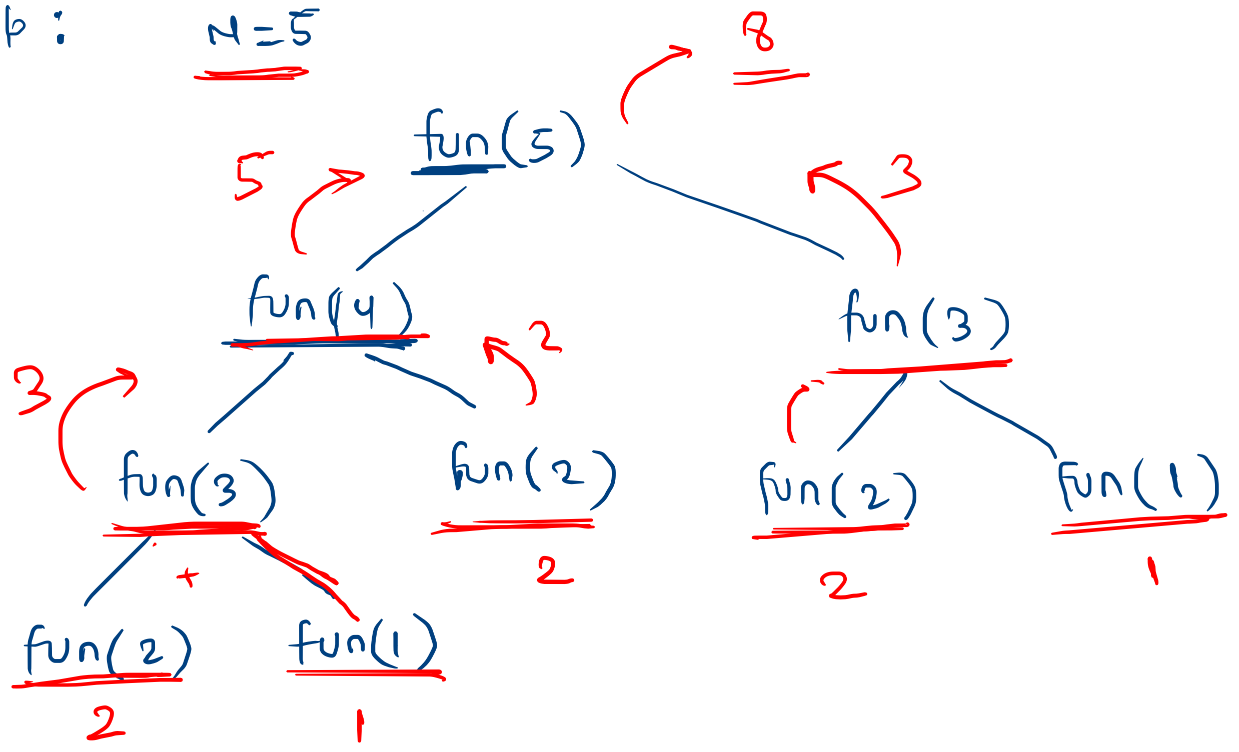
```
int climb (int N)
{
✓ if (N<=2) return N;
✓ return climb (N-1) + climb (N-2) ;

}
```

Let's Code on Leetcode

Recursion Tree

I/p : N=5

fun(n-1) + f(n-2)



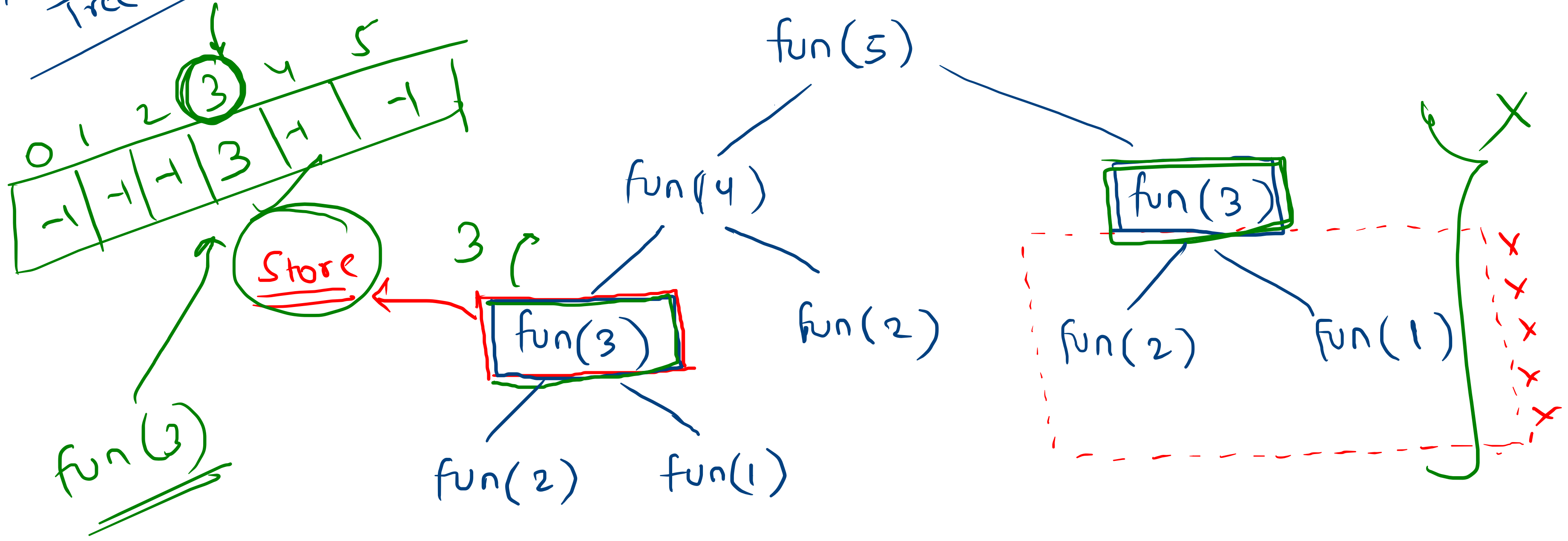
Climb()

$\Downarrow$

fun()

Recursion Tree

I/p : N=5



Climb()

$\Downarrow$

fun()

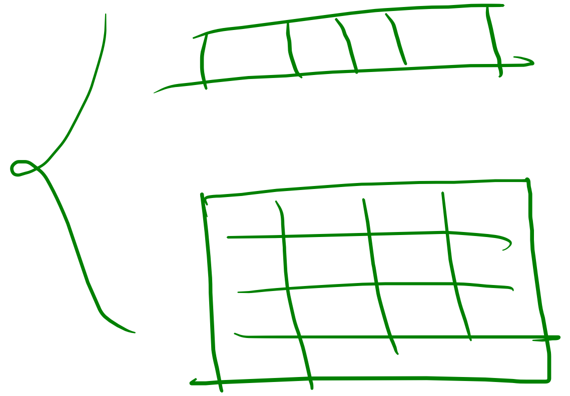
N=5

$\downarrow$

$\boxed{N = 100}$

Now, we have to convert

$$\begin{tikzcd}
\boxed{\text{Recursive Code}} \arrow[rr] \arrow[dr, bend right] & & \boxed{\text{DP Code}} \\
& \text{Memoization} \arrow[ur, bend right] &
\end{tikzcd}$$

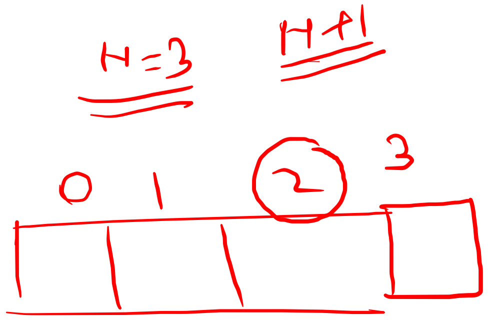


Recursive code

Imp

```
int climb (int N)
{
    if (N<=2) return N;
    return climb (N-1) + climb (N-2) ;

}
```

fun(3) → X



DP code

C++

```
vector<int> dp (N+1, -1);

int climb (int N )
{
    if( dP[N] ! = -1 ) return dp[N];

    if( N<=2)
    {   return dp[N] = n;
    }
    return dp[N] =
        climb(N-1) +
            climb (N-2);
}
```