Dynamic Programming

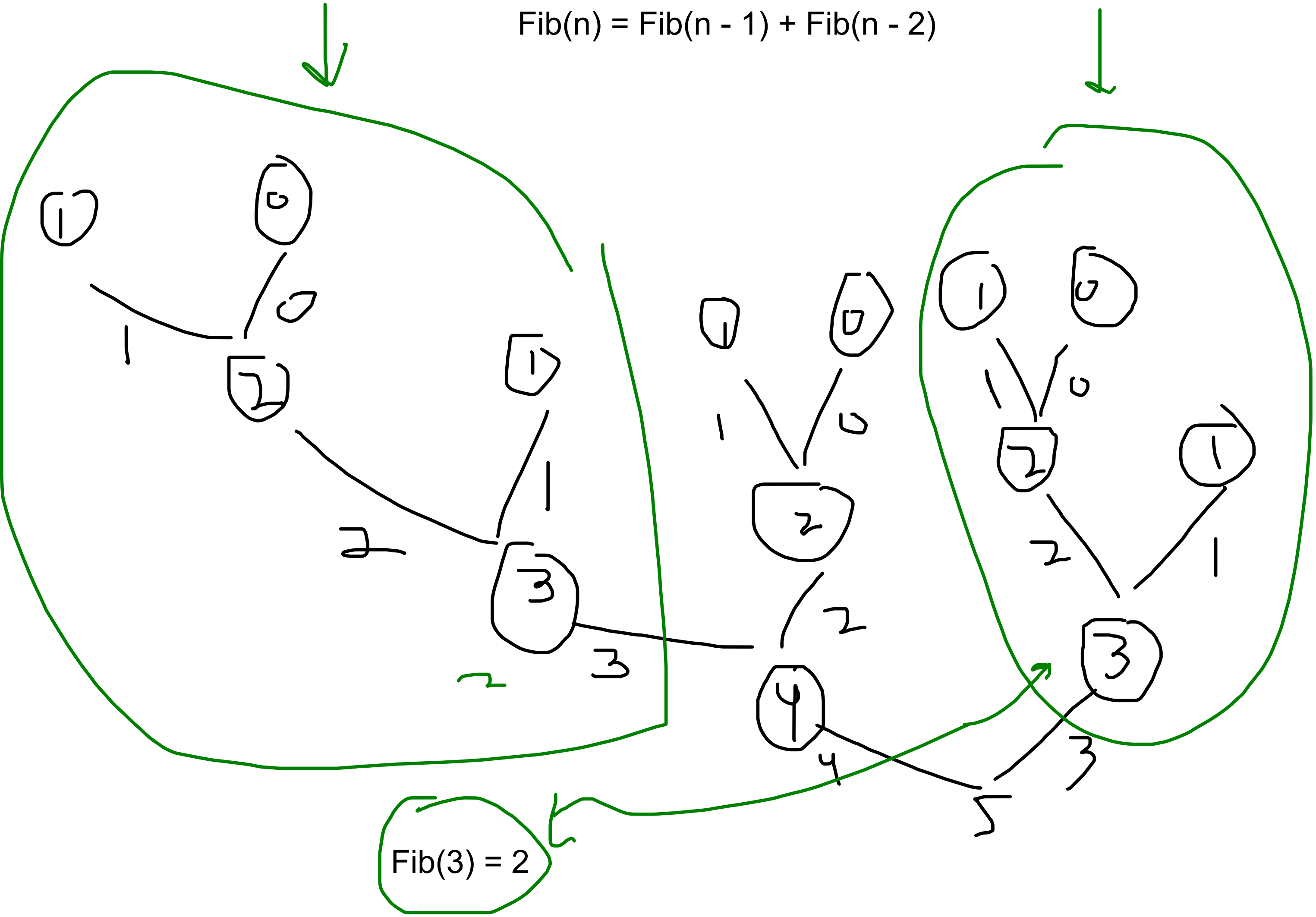
Optimisation over recursion.

exponential time comp of recursion

0 1 1 2 3 5

Fib

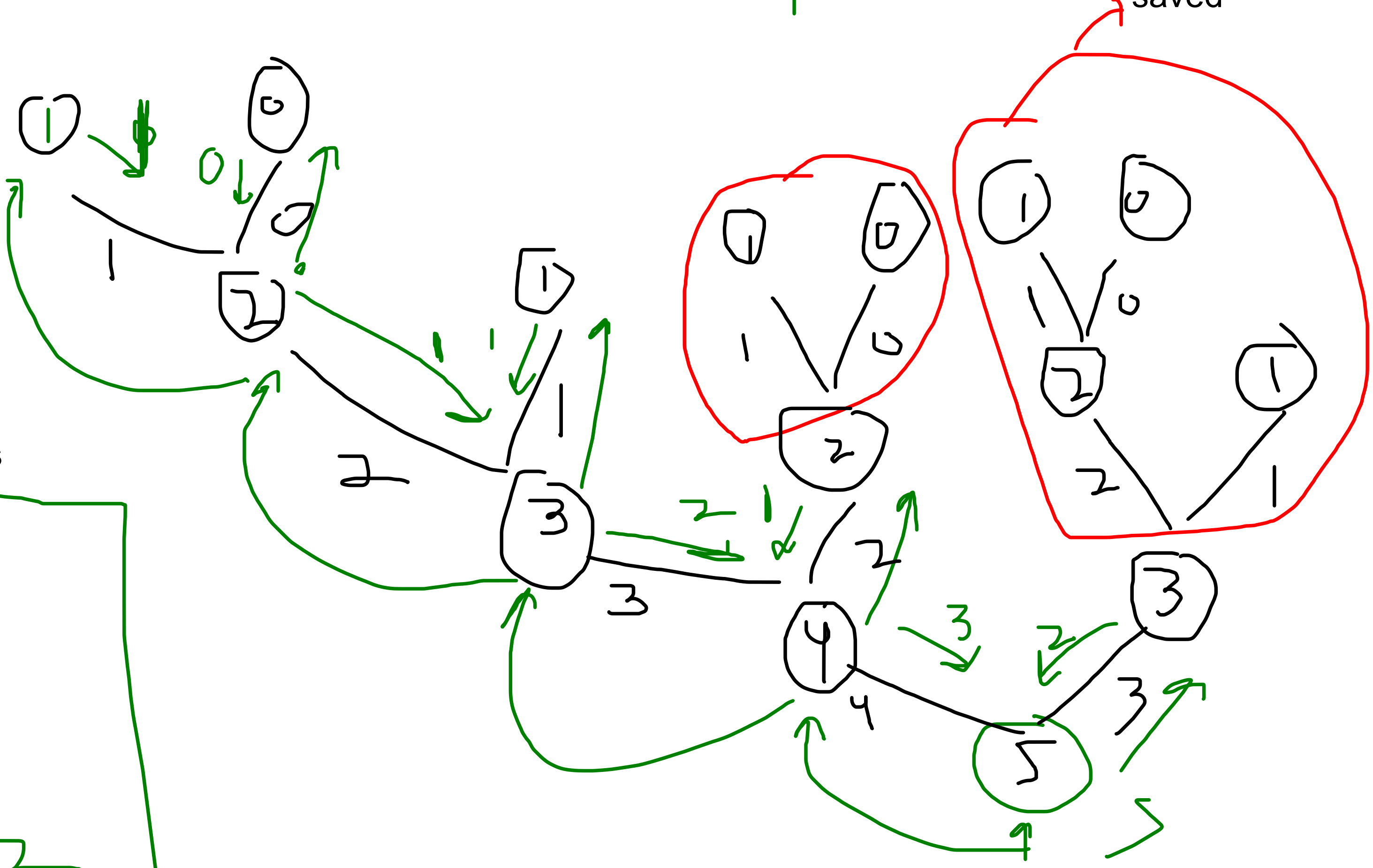
Fib(n) = Fib(n - 1) + Fib(n - 2)



Fib(3) = 2

0 1 1

Fib(n) = Fib(n - 1) + Fib(n - 2)



space saved or time saved

fib(n) --> ans

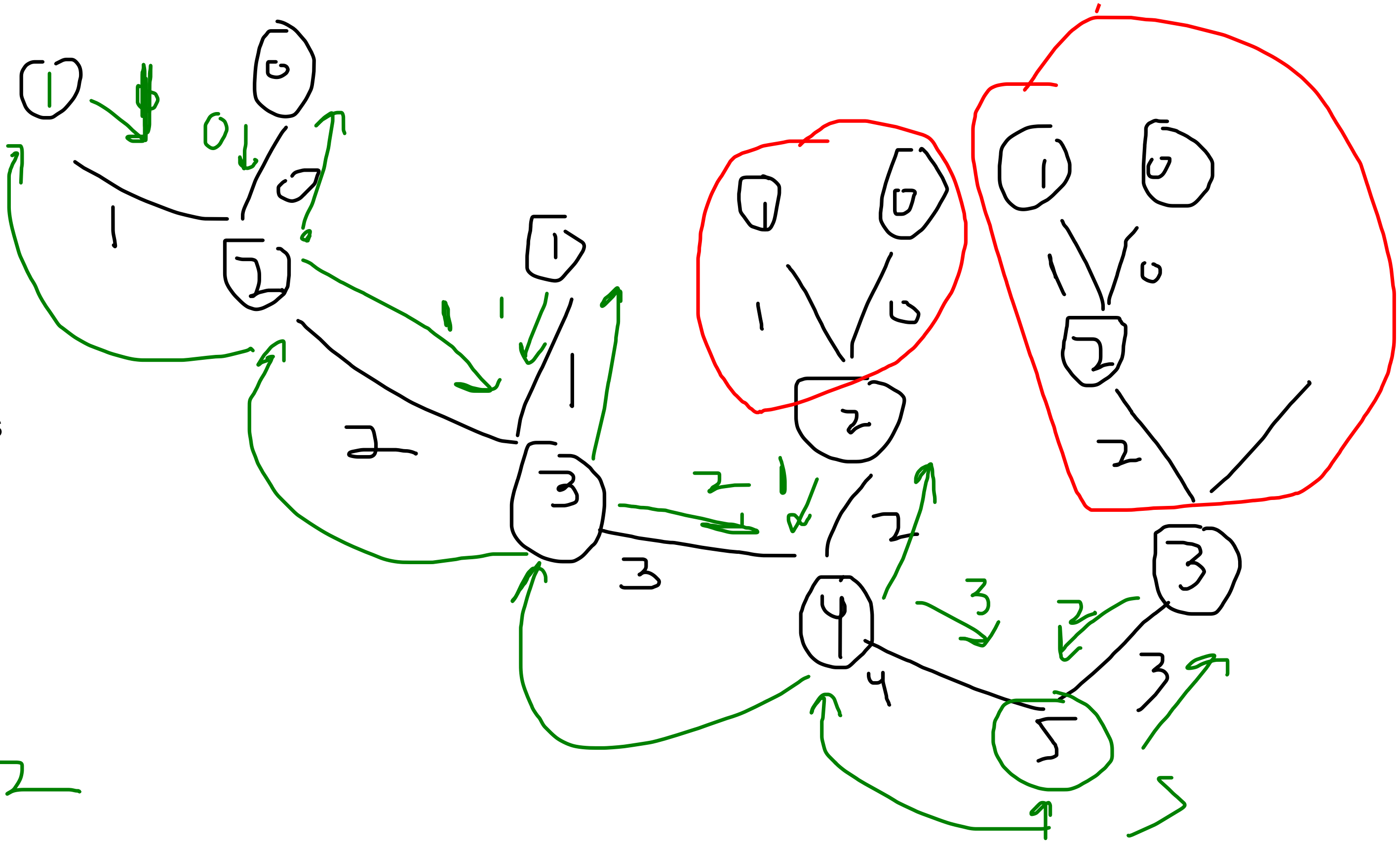
1	1
0	0
2	1
3	2
4	3
5	5

Types of DP

1. Memoization (Top - down approach)
2. Tabulation (Bottom - up approach)

1. Memoization

$$\mathrm{Fib}(n) = \mathrm{Fib}(n - 1) + \mathrm{Fib}(n - 2)$$



dp

fib(n) --> ans

1 →	1
0 →	0
2 →	1
3 →	2
4 →	3
5 →	5

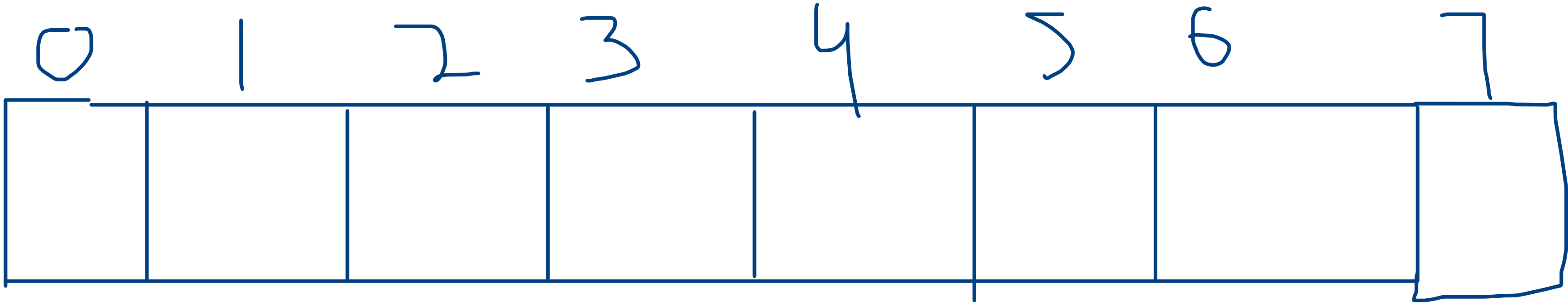
Bigger problem into smaller sub problems

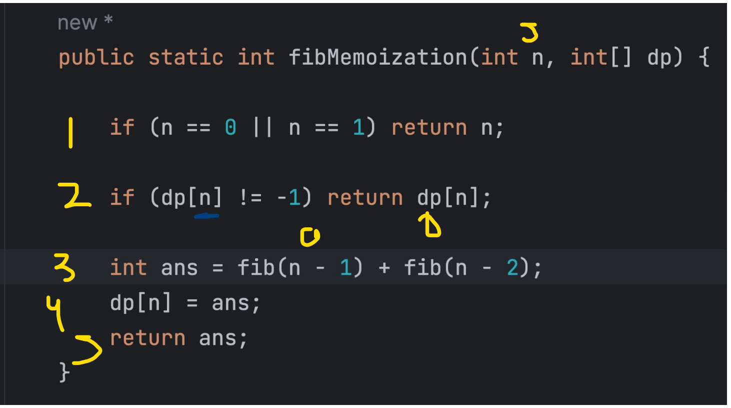
1. properties of memoization

a) It uses same recusive code to optimize with the help of DP array.
b) It is useful for smaller inputs

Fib Question

arr of size == n + 1



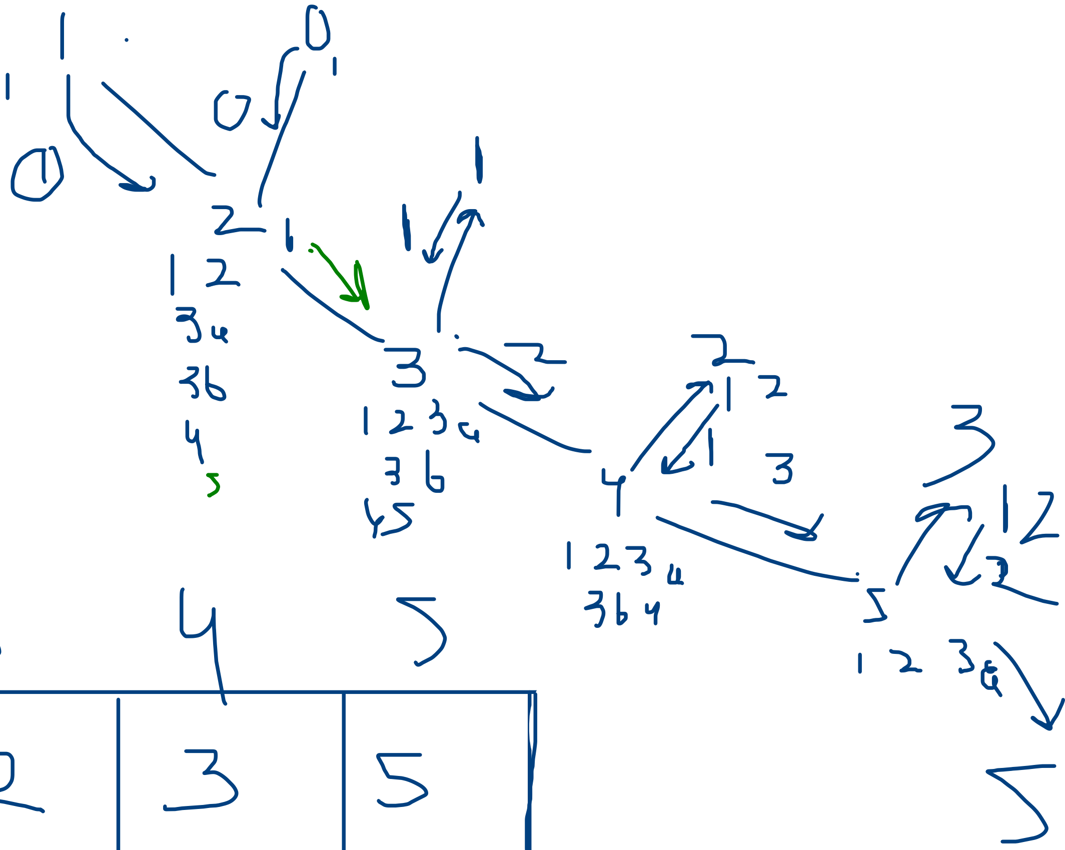


```java
public static int fibMemoization(int n, int[] dp) {

    if (n == 0 || n == 1) return n;

    if (dp[n] != -1) return dp[n];

    int ans = fib(n - 1) + fib(n - 2);
    dp[n] = ans;
    return ans;
}
```



0	1	2	3	4	5
-1	-1	1	2	3	5

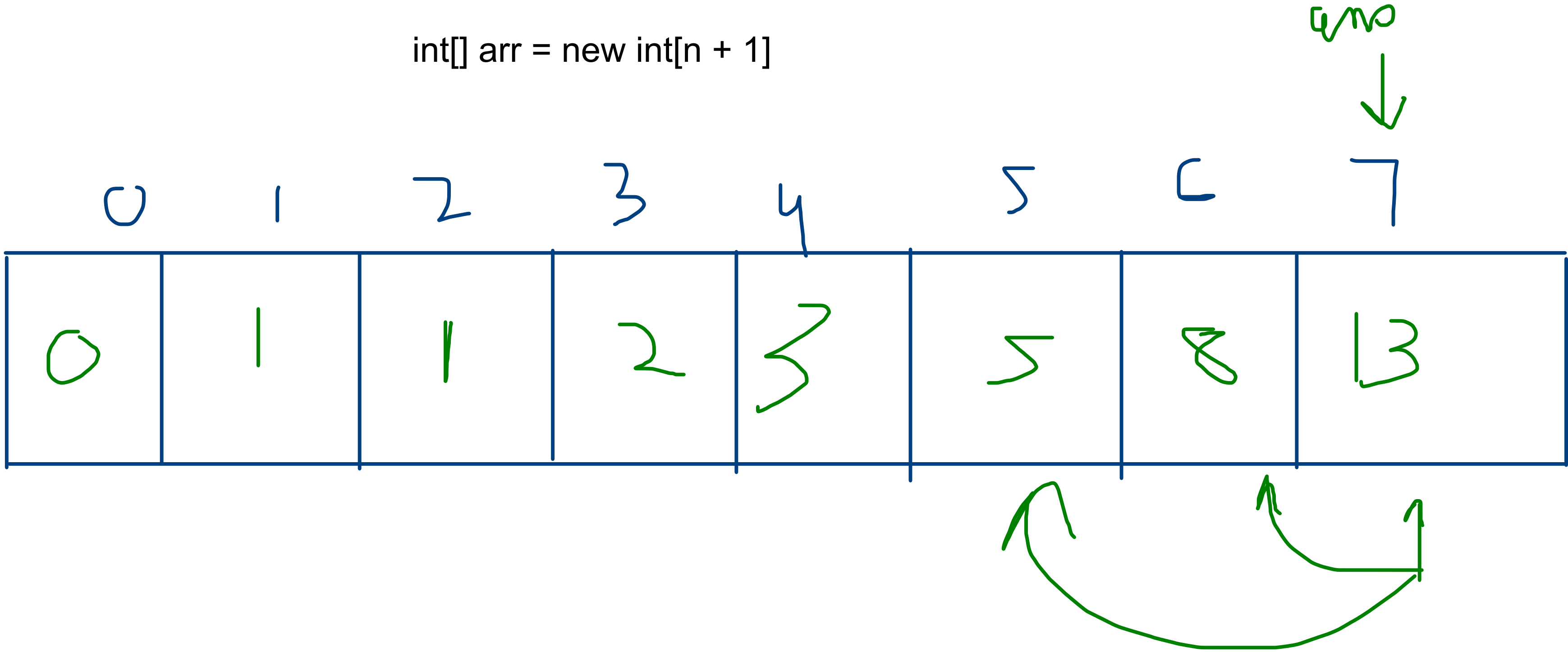
Tabulation --> Bottom-up

1. It do not require recusive code, ie its code is iterative

Fib(n) = Fib(n - 1) + Fib(n - 2)

$n = 7$

int[] arr = new int[n + 1]



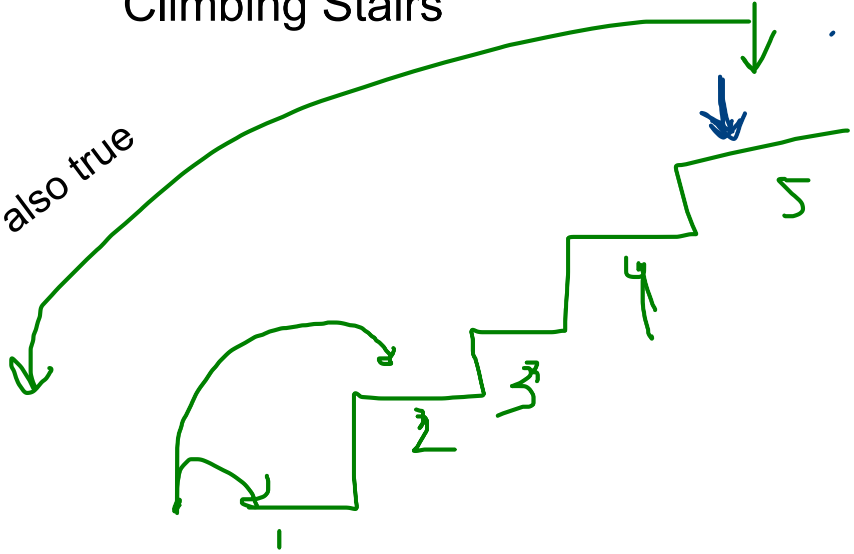
0	1	2	3	4	5	6	7
0	1	1	2	3	5	8	13

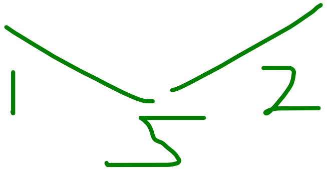
Memoization	Tabultaion
applied over small input	applied over large input
Complex to understand --> recusive code	Simple to understand --> iterative code

Detect DP

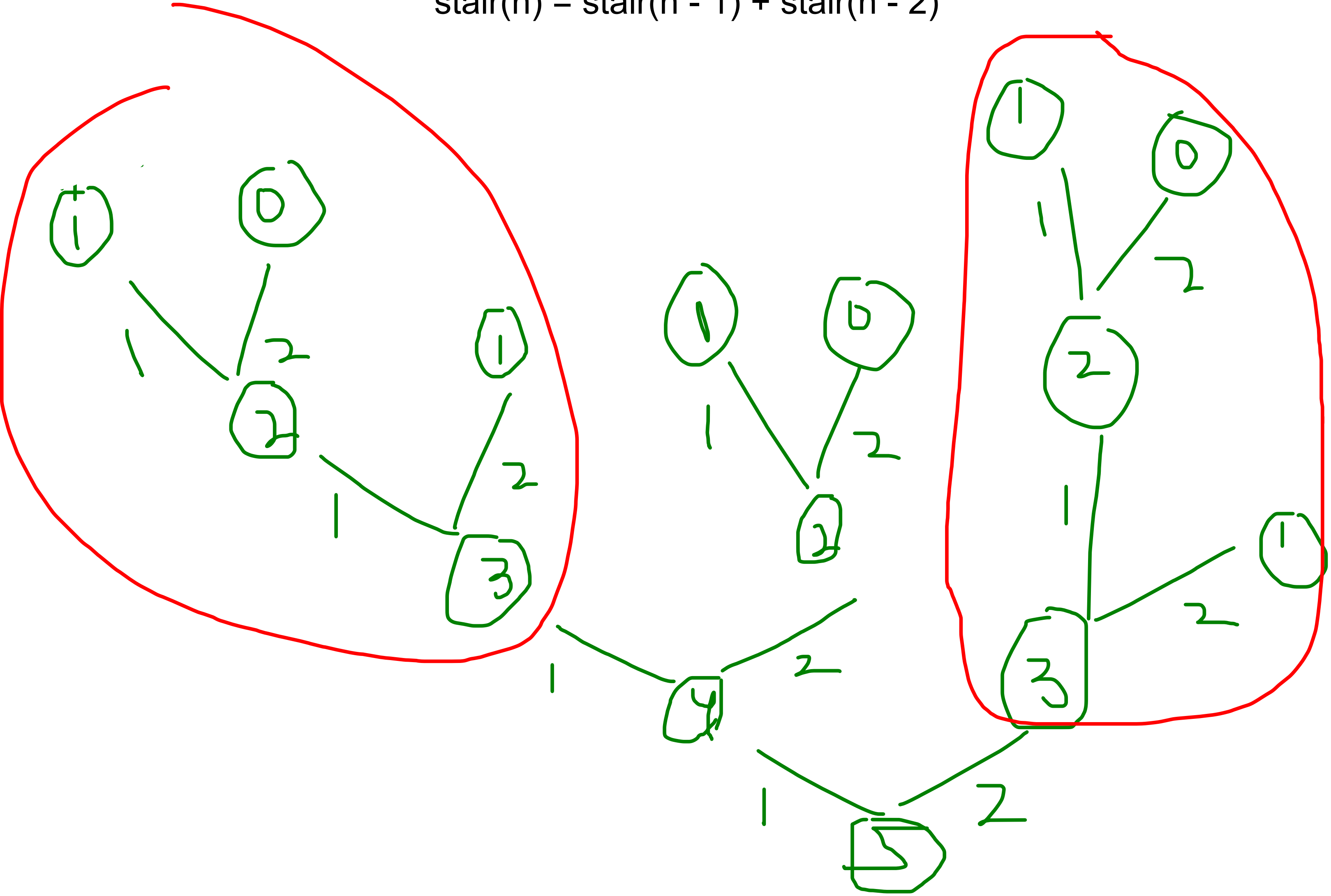
Repeated Subproblems --> Detect DP

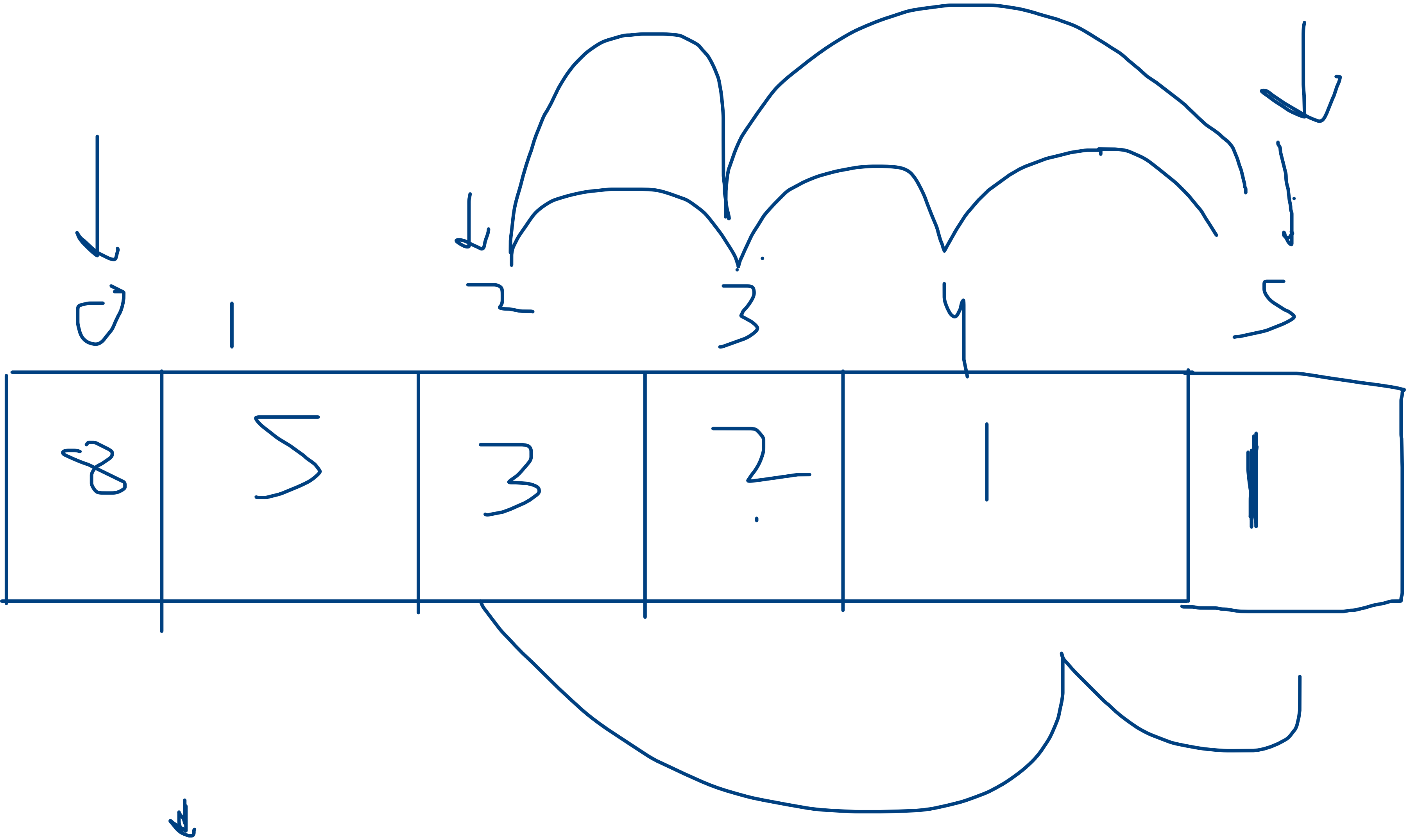
Climbing Stairs


also true



stair(n) = stair(n - 1) + stair(n - 2)

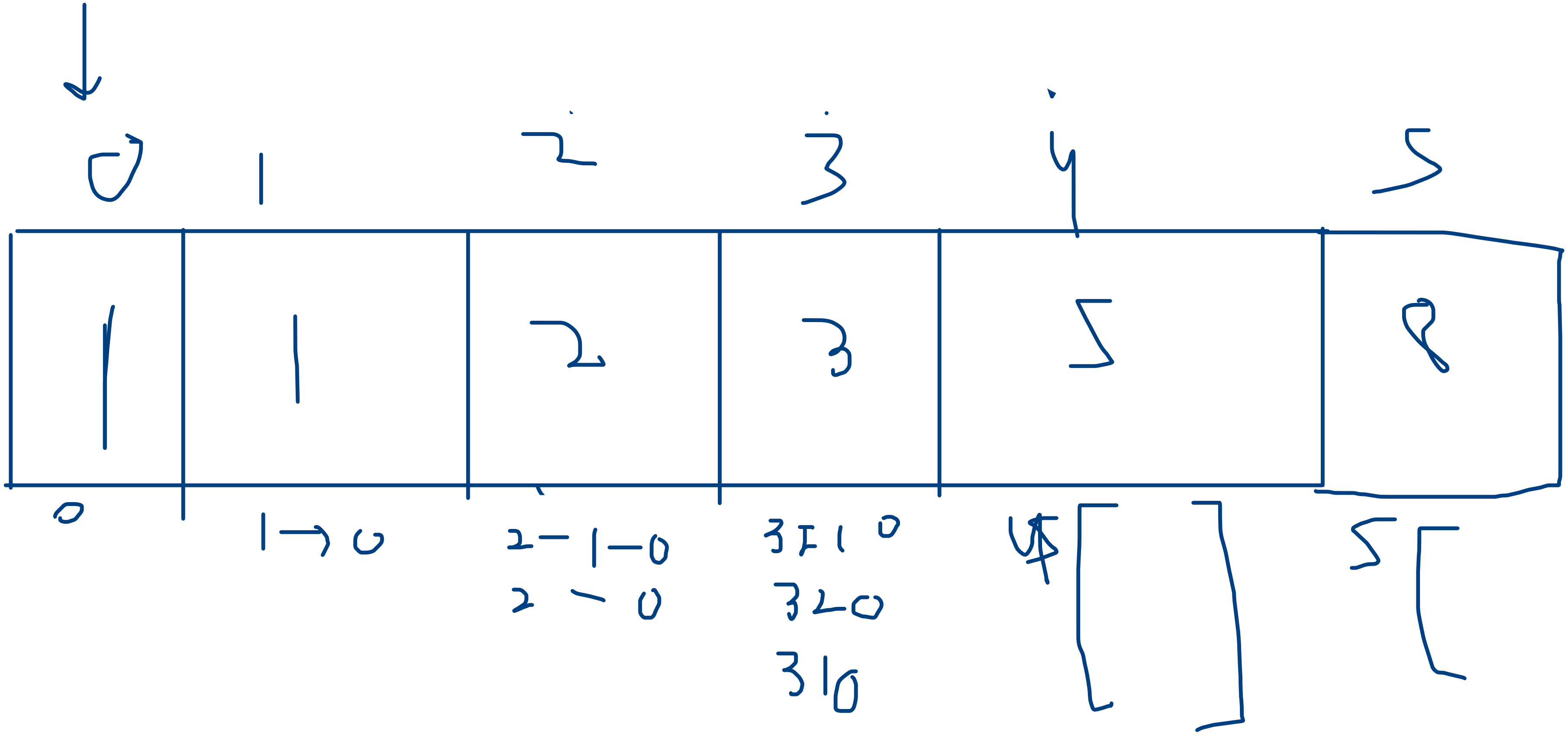




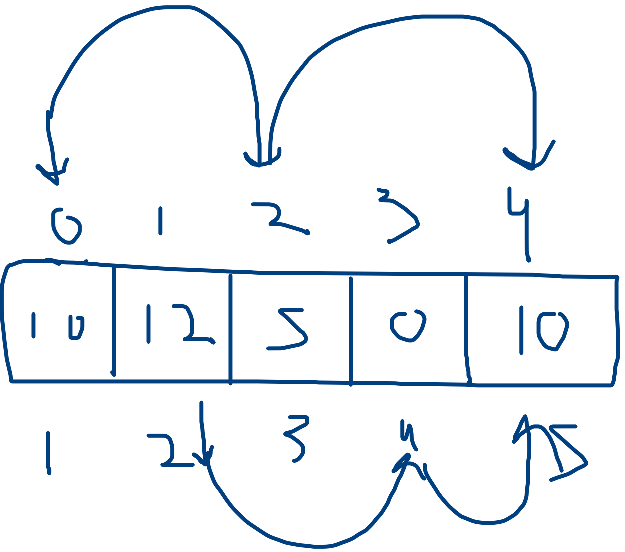
0	1	2	3	4	5
8	5	3	2	1	1

num of ways == 1 way

path --> .


0
1
2
3
4
5
1
1
2
3
5
8
0
1 → 0
2 — 1 — 0
2 — 0
3 2 1 0
3 2 0
3 1 0
4 [ ]
5 [

Min Cost Climbing Stairs



10 15

5

12

0 12

0

General rules

0	1	2	3	4
10	12	5	0	10

Step 1: Type of DP (1D, 2D, 3D, 4D....)

1D → n+1

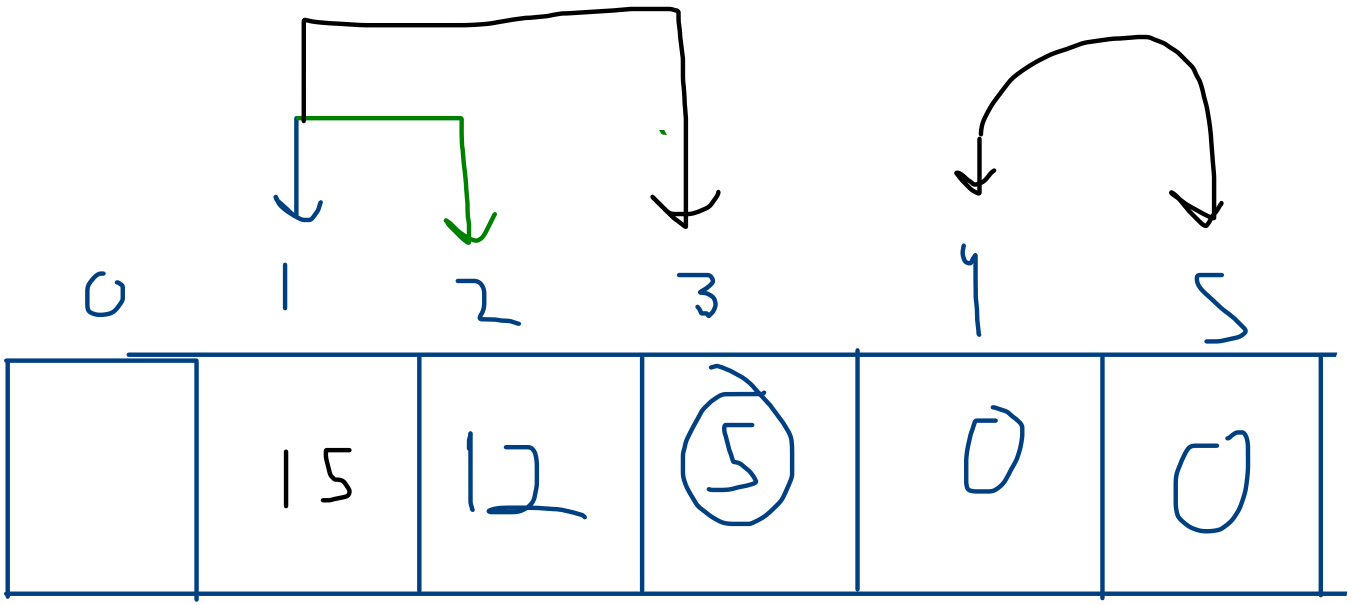
Step2: identify smaller and bigger problem

Step3: Define meaning of dp[i] --> most imp --> min cost need to pay to reach to 5th step (smaller problem

Step 4: solve from smaller to bigger

0	1	2	3	4	5

0	1	2	3	4
10	12	5	0	10



$$\begin{array}{r} 10 \\ 12 \\ \hline 22 \end{array} \quad \begin{array}{r} 10 \\ 5 \\ \hline 15 \end{array} \quad \begin{array}{r} 12 \\ 5 \\ \hline 17 \end{array} \quad \begin{array}{r} 12 \\ 0 \\ \hline 12 \end{array}$$