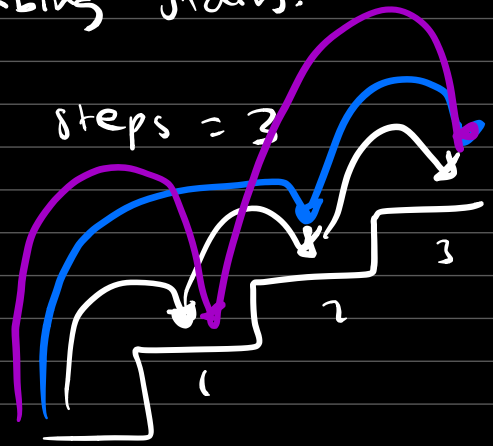
Dynamic Programming

Climbing stairs.

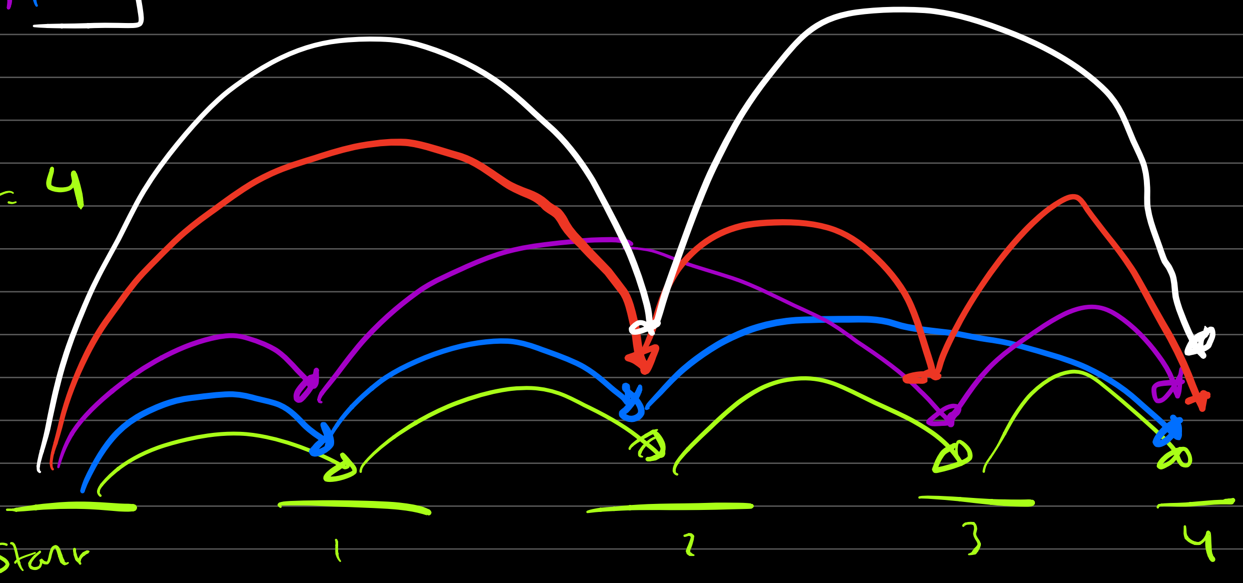
# of steps = 3



ans=3

1, 1, 1
1, 2
2, 1

Steps = 4

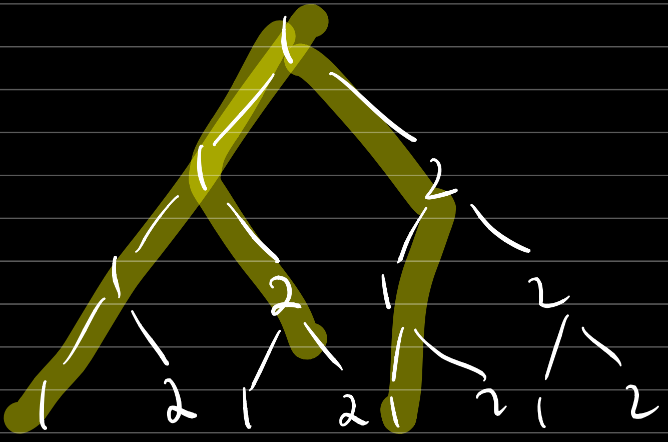


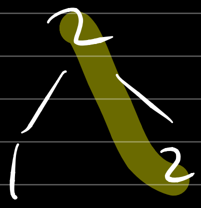
3	4	2
1, 1, 1	1, 1, 1, 1	1, 1
1, 2	1, 1, 2	2
2, 1	1, 2, 1	
	2, 1, 1	
	2, 2	

$O(2^n)$

Permutations of 1, 2 that add up to target.

Brute force



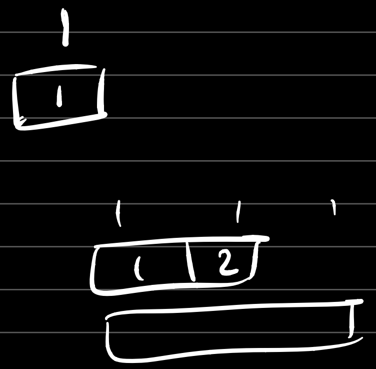


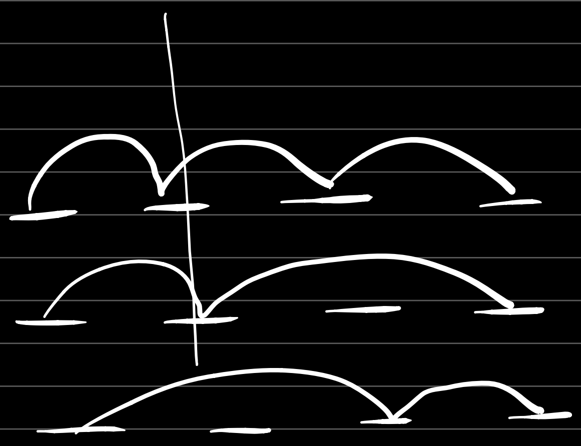
f(1)=1

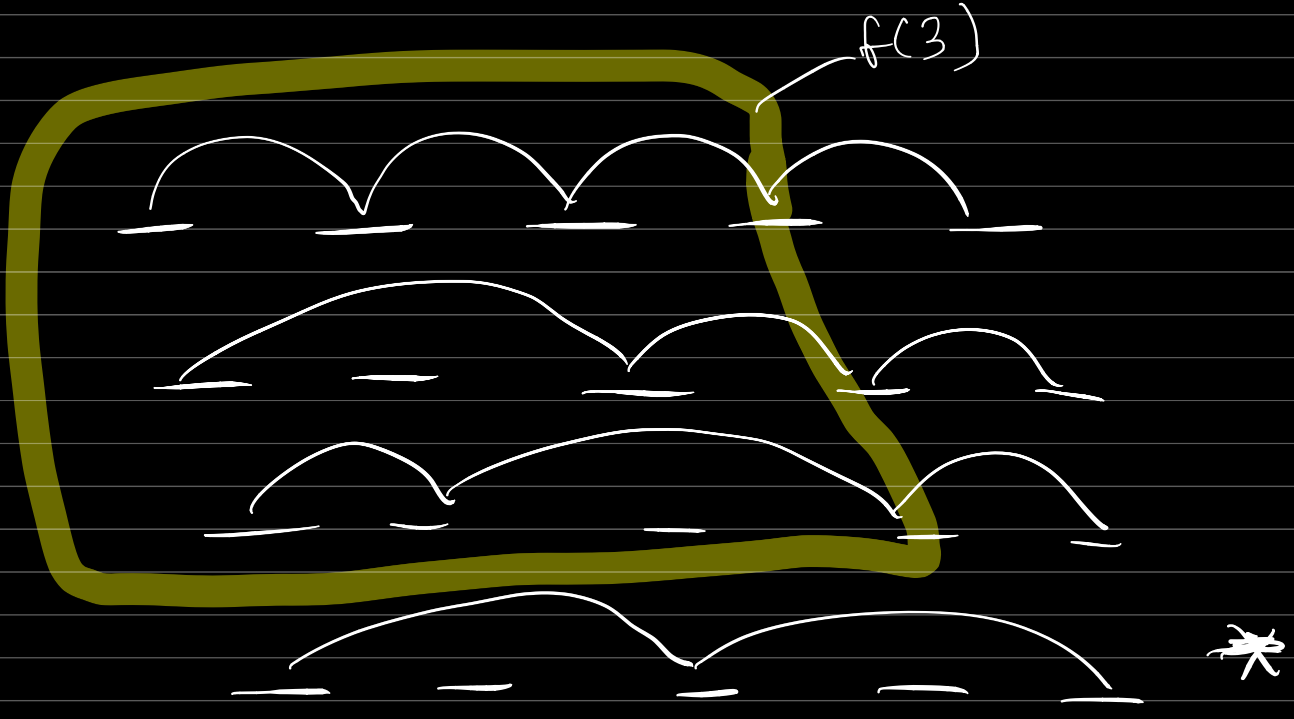
f(2)=2

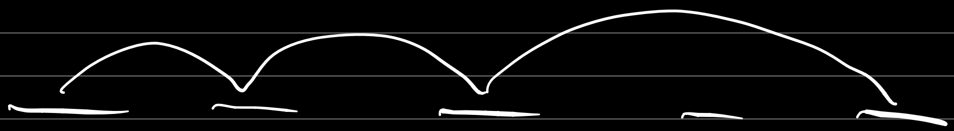
f(3)=3

f(4)=5









f(5)

1 1 1 1 1 - 5

1 1 1 2
1 1 2 1
1 2 1 1
2 1 1 1 } 4

2 2 1
2 1 2
1 2 2 } 3

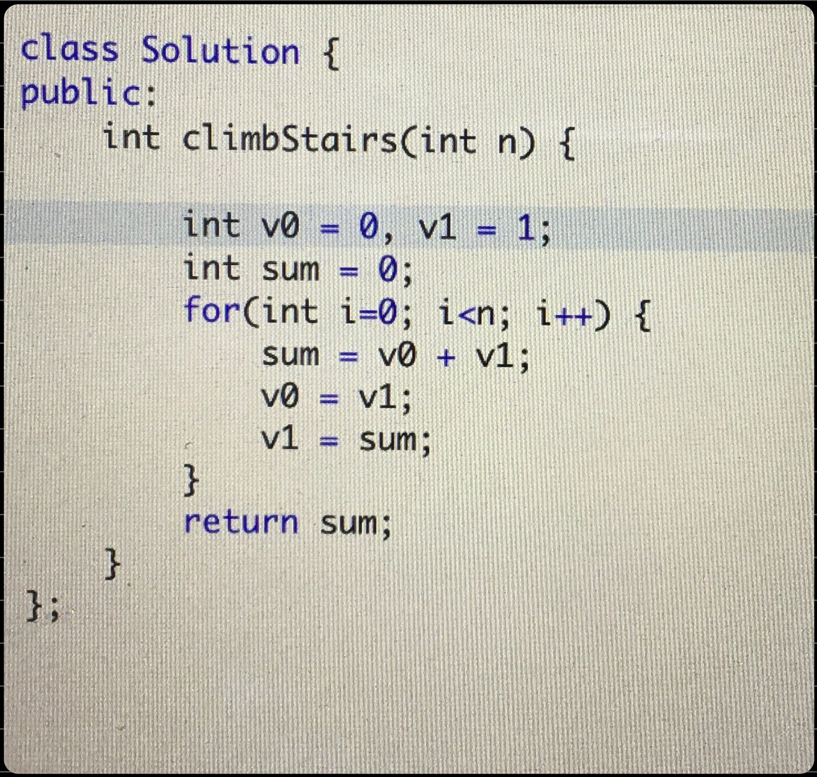
1 1 1 1 } 4

1 1 2
1 2 1
2 1 1 } 3

2 2 } 2

1
2
3
5
8

→ fibonacci-sequence



```
class Solution {
public:
    int climbStairs(int n) {

        int v0 = 0, v1 = 1;
        int sum = 0;
        for(int i=0; i<n; i++) {
            sum = v0 + v1;
            v0 = v1;
            v1 = sum;
        }
        return sum;
    }
};
```