Q → Given an integer array.

A[i] → Money

Find **max sum** without selecting adjacent elements.

i-1 ← i → i+1

		✓		✓	
A →	1	2	5	8	3

Ans = 10

	✓		✓		✓
A →	2	7	9	3	1

Ans = 12

	✓			✓
A →	7	4	1	3

Ans = 10

N = 1 Ans = A[0]

N = 2 Ans = max(A[0], A[1])

N = 3

0 1 2

__ __ __

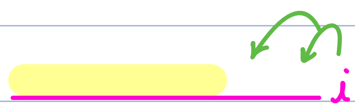
✓ A[2] + ans till A[0] | ✗ ans till A[1]

$$\boxed{sum[N] = \max \begin{cases} A[N] + sum[N-2] & ✓ \\ sum[N-1] & ✗ \end{cases}}$$

sum[0] = A[0]

sum[1] = max(A[0], A[1])

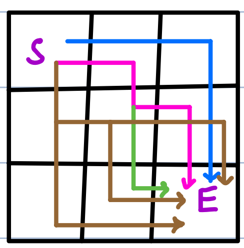
```
for i → 2 to (N-1) {
    sum[i] = max(A[i] + sum[i-2], sum[i-1])
}
return sum[N-1]
```



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

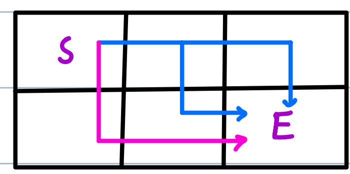
O(1)

Q → Find # ways to travel from (0,0) to (N-1, M-1) s.t in one step we can go → or ↓.

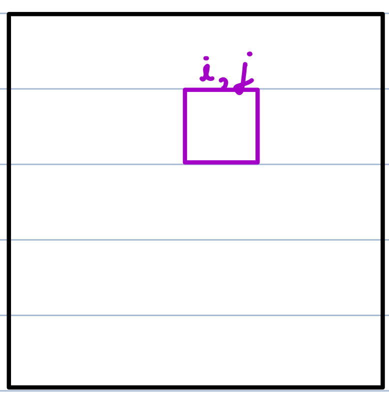


3 * 3

Ans = 6



Ans = 3



start → (i, j) → end

∀ i, j find # ways from start to i, j

⇒ Ans = (i, j) → End ✓

∀ i, j find # ways from i, j to end

⇒ Ans = (i, j) → Start (H.W)

start $\longrightarrow$

$$(i-1, j) \; x$$
$$\downarrow$$
$$(i, j-1) \; y \longrightarrow (i, j) \rightarrow \underline{x + y}$$

ways (i, j)

- ways (i-1, j)
  - ways (i-2, j) ...
  - ways (i-1, j-1) ...
- ways (i, j-1)
  - ways (i-1, j-1) ...
  - ways (i, j-2) ...

ways (0, 0) = 1

ways (0, j) = 1

ways (i, 0) = 1

{ optimal substructure ✓
overlapping subproblems ✓

use DP.

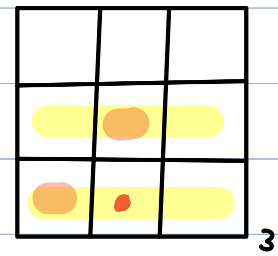
Bottom – Up

```
for i → 0 to (N-1) {
    for j → 0 to (M-1) {
        if (i==0 || j==0)  ways [i][j] = 1
        else  ways [i][j] = ways [i-1][j] + ways [i][j-1]
    }
}  return ways [N-1][M-1]
```

TC = O(N * M)

SC = O(N * M)

$O(2M) \rightarrow \underline{O(M)}$




```
∀j, pre[j] = 1          // row 0
for i → 1 to (N-1) {
    for j → 0 to (M-1) {
        if (j==0)    cur[j] = 1
        else         cur[j] = pre[j] + cur[j-1]
    }
    for j → 0 to (M-1) {
        pre[j] = cur[j]
    }
} return cur[M-1]
```

$TC = \underline{O(N * M)}$

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

Q → Find the count of <u>N</u> digit numbers with digit sum = <u>S</u>.

$\boxed{N, S > 0}$ [0 — 9]

N = 2 13 22 31 40 Ans = <u>4</u>

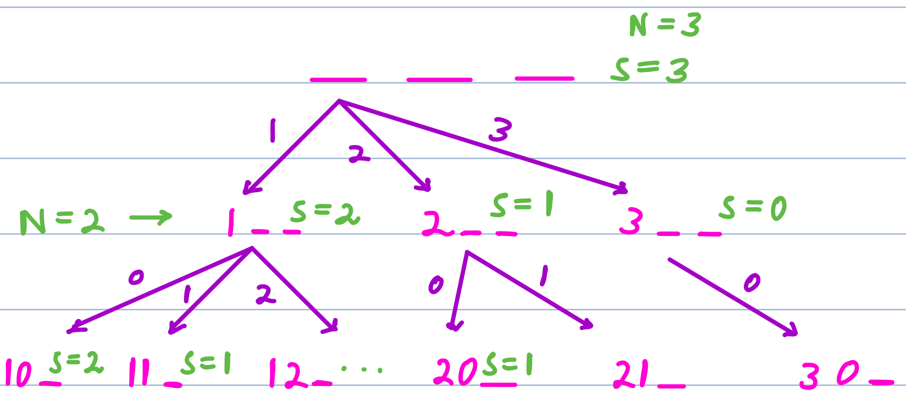
S = 4

N = 3

S = 3

N=2 →

N=1 →



102 111 120 201 210 300 Ans = <u>6</u>

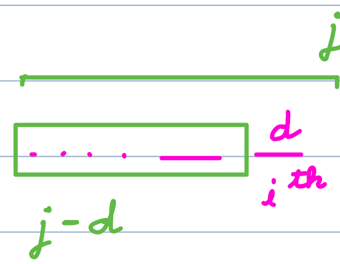
<u>use DP</u> ← { optimal substructure ✓
overlapping subproblems ✓ }

count[i][j] → count of i digit nos. with digit sum j.

count[1][j] → 1 if 1 <= j <= 9
→ 0 else

```
for j → 1 to 9 {
    count[1][j] = 1    ✓
}

for i → 2 to N {
    for j → 0 to S {
        for d → 0 to 9 {
            if (d <= j) {
                count[i][j] += count[i-1][j-d]
                // count[i][j] = (count[i][j] + count[i-1][j-d]) % M
            }
        }
    }
}
return count[N][S]
```



TC = O(N*S)

SC = O(N*S)

optimize SC → H.W

i \ j	0	1	2	3
0	0	0	0	0
1	0	1	1	1
2	0	1	2	3
3	0	1	3	6 → Ans

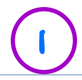
Catalan Numbers

The Catalan numbers form a sequence of natural numbers that have numerous applications in combinatorial mathematics. Each number in the sequence is a solution to a variety of counting problems. The Nth Catalan number, denoted as Cn, can be used to determine:

- The number of correct combinations of N pairs of parentheses.
- The number of distinct binary search trees with N nodes, etc.

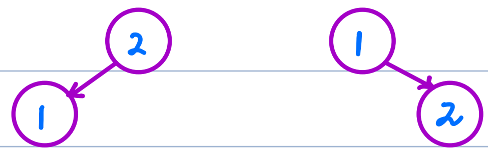
Q → Find the count of unique BST with N distinct nodes.

N = 1



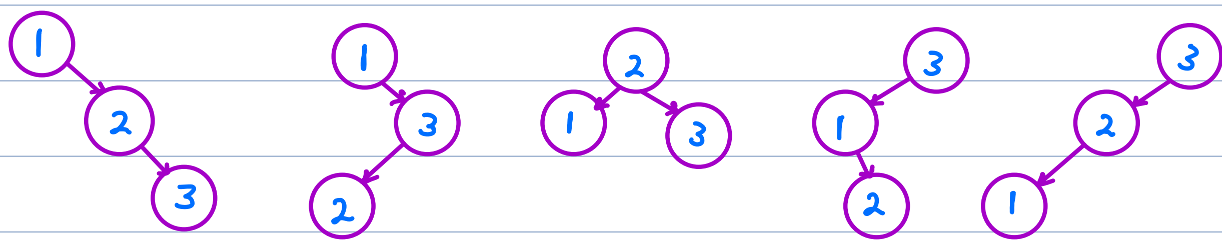
Ans = 1

N = 2



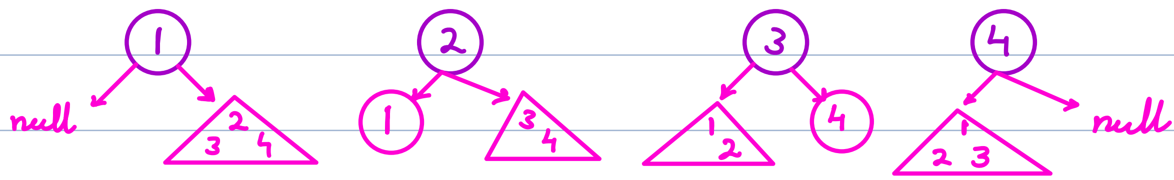
Ans = 2

N = 3



Ans = 5

N = 4



$$\boxed{c(0)}_{=1} * c(3) + c(1) * c(2) + c(2) * c(1) + c(3) * c(0)$$

$$c(4) = 1 * 5 + 1 * 2 + 2 * 1 + 5 * 1 = \underline{14}$$

$$\boxed{c(N) = \sum_{i=0}^{N-1} c(i) * c(N-1-i)} \rightarrow \frac{{}^{2N}C_N}{(N+1)}$$

$TC = \underline{O(N^2)}$ $SC = \underline{O(N)}$

```
c[0] = c[1] = 1
for i → 2 to N {
    for j → 0 to (i-1) {
        c[i] += c[j] * c[i-1-j]
    }
} return c[N]
```