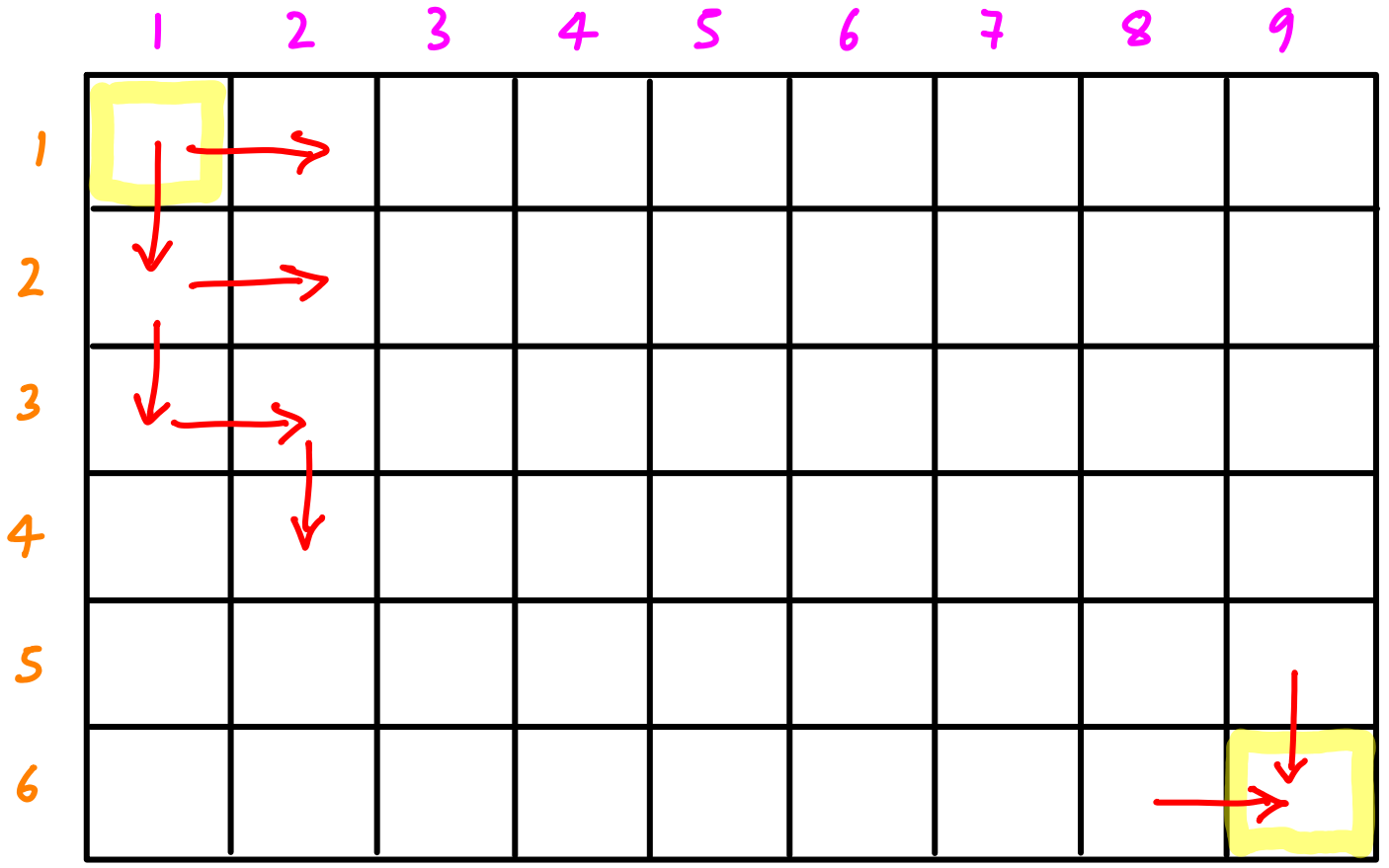
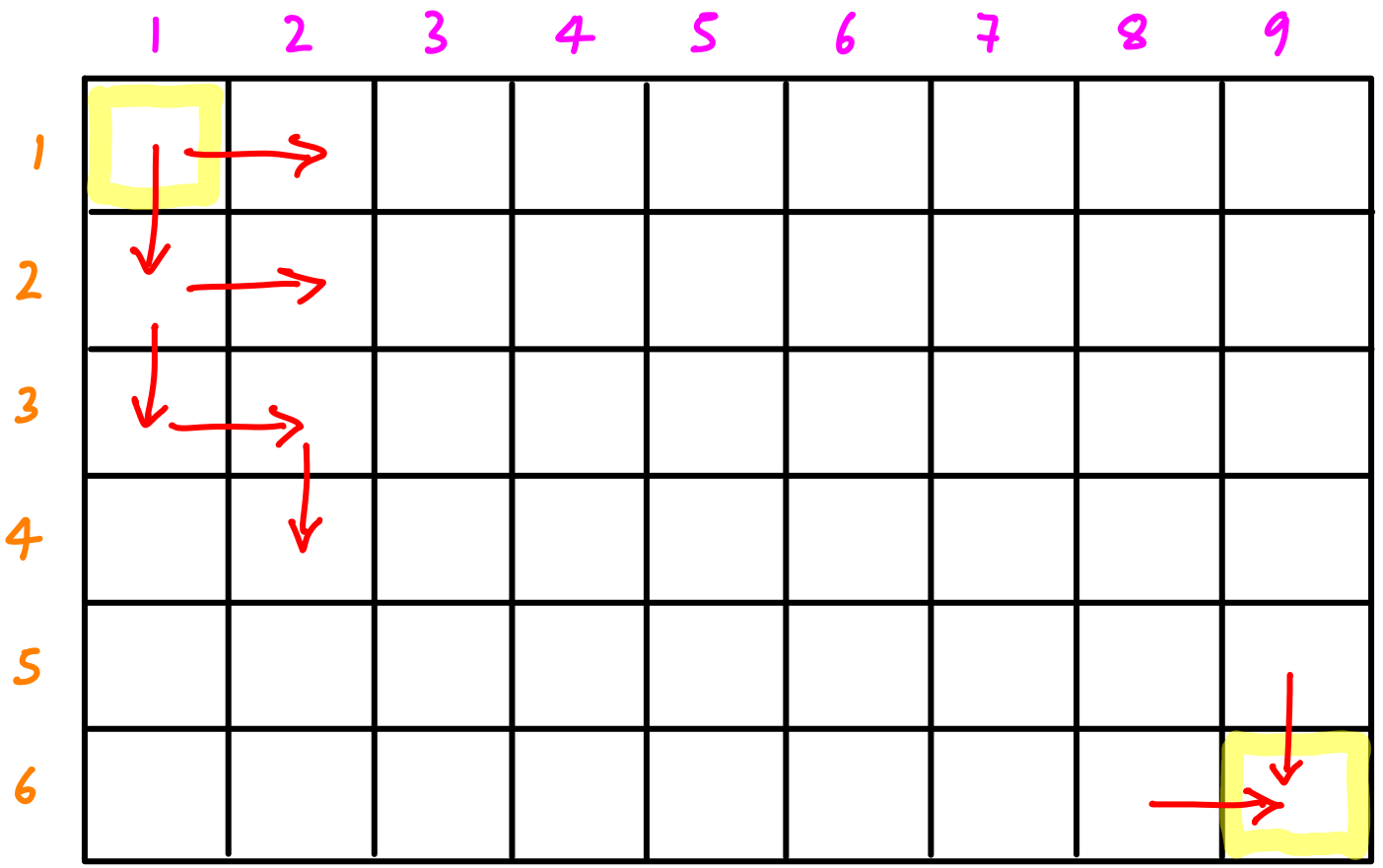
Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?



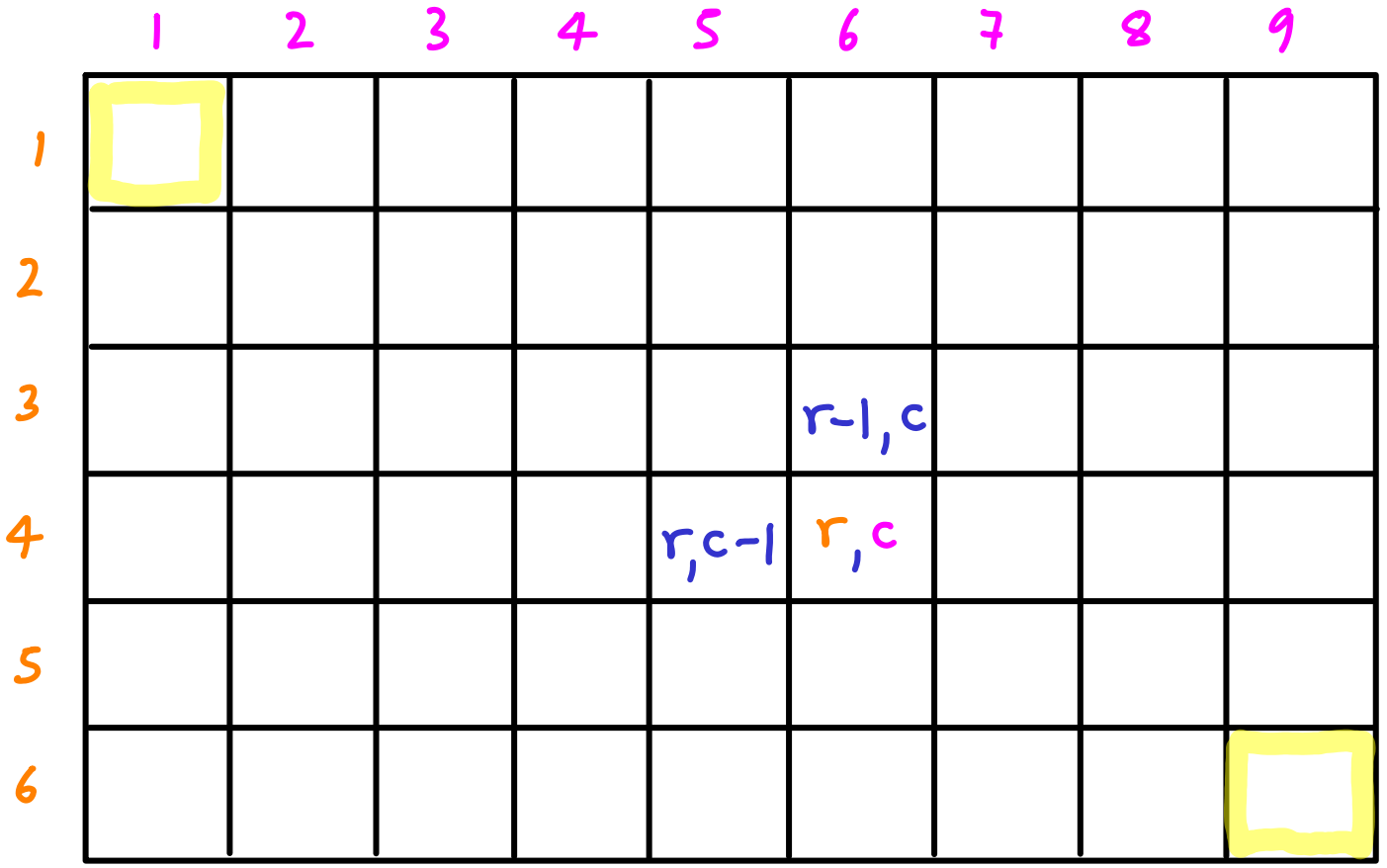
Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?

Recursive form?

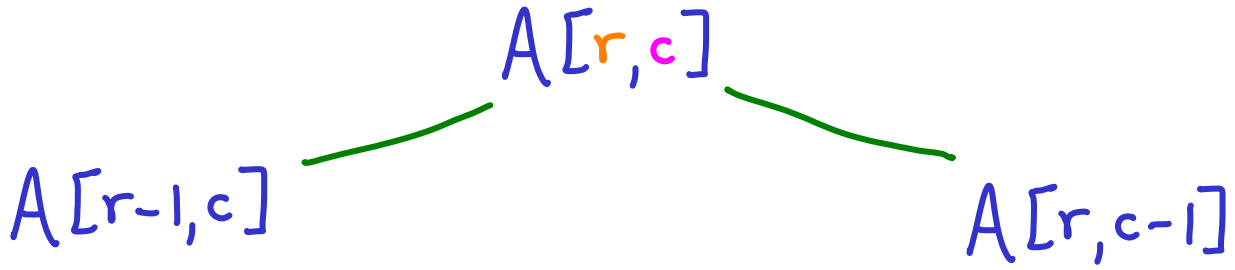


Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?

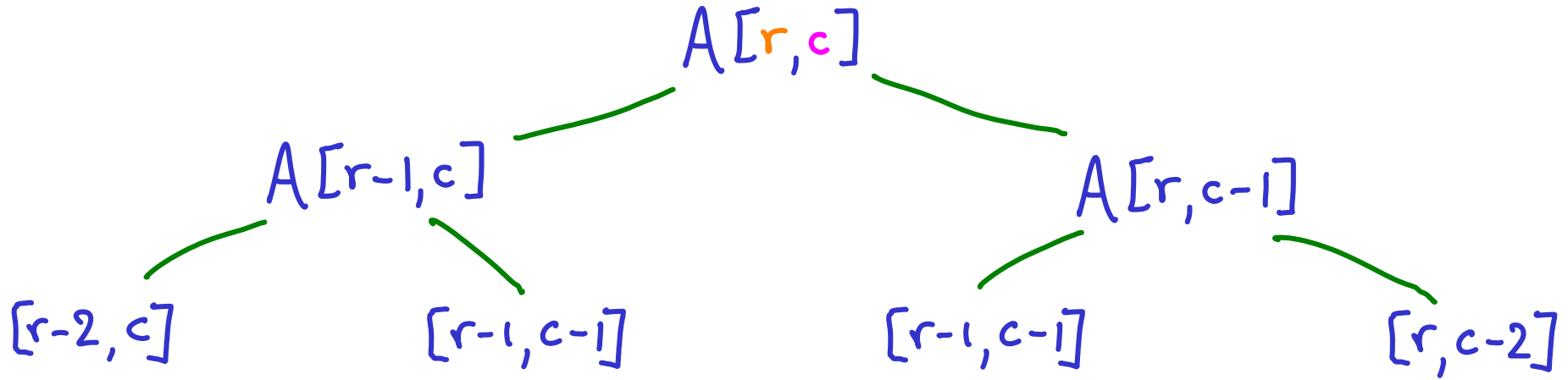
$$A[r,c] = A[r-1,c] + A[r,c-1]$$



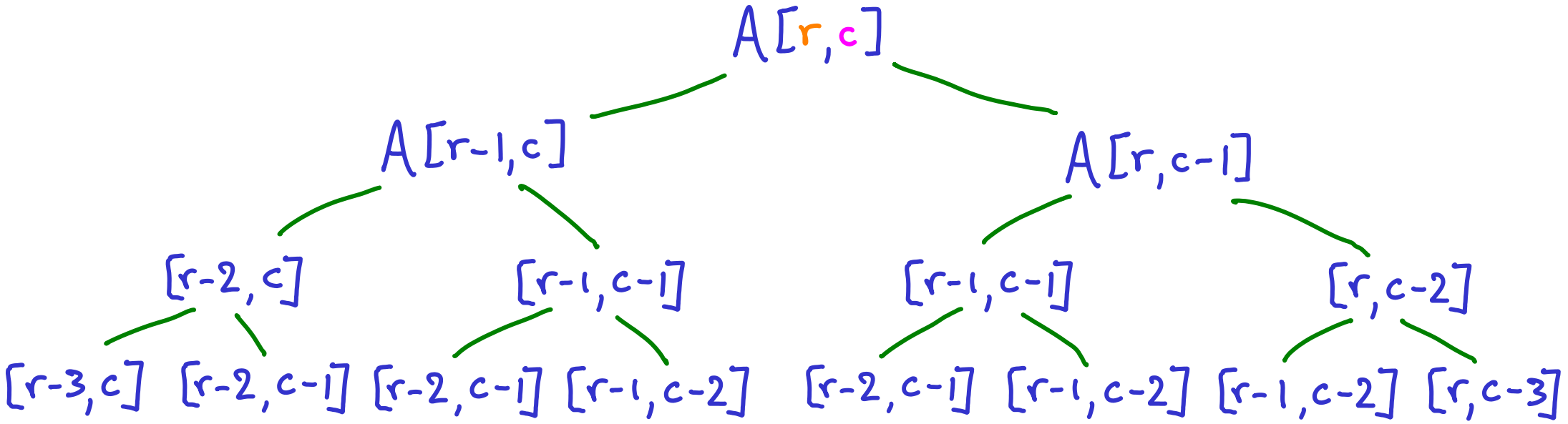
Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?



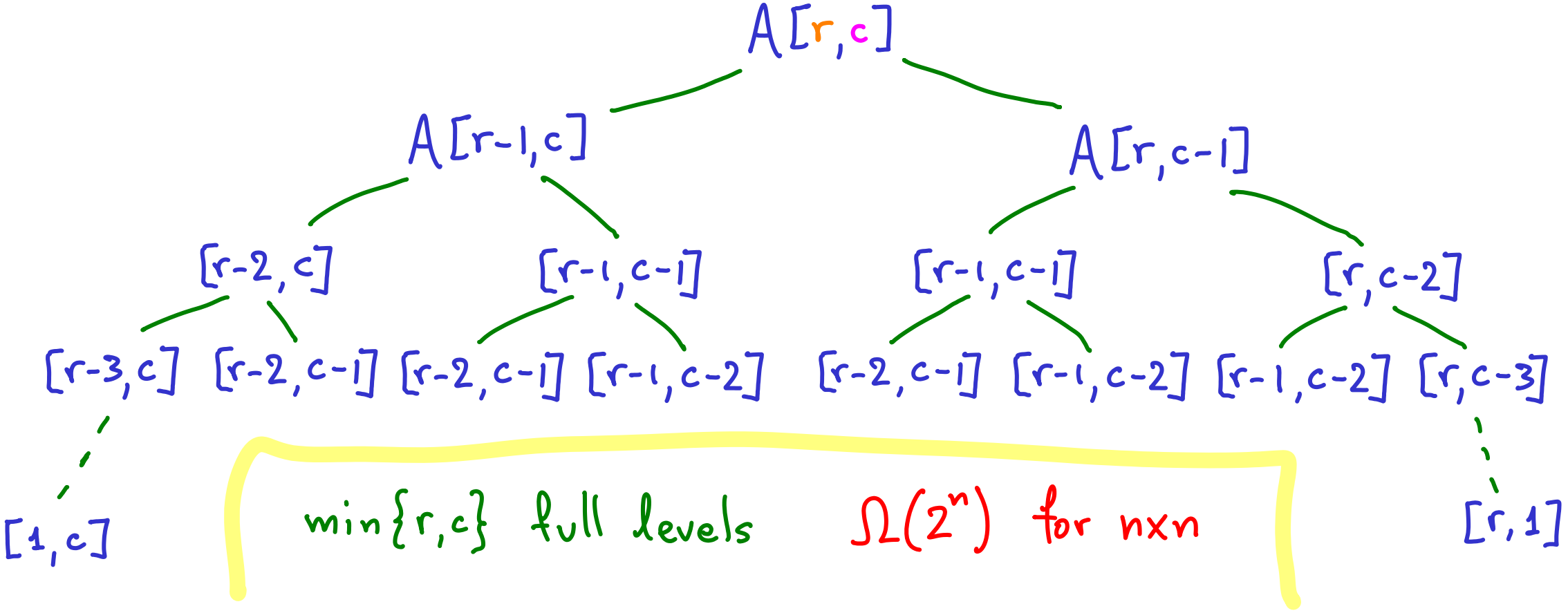
Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?



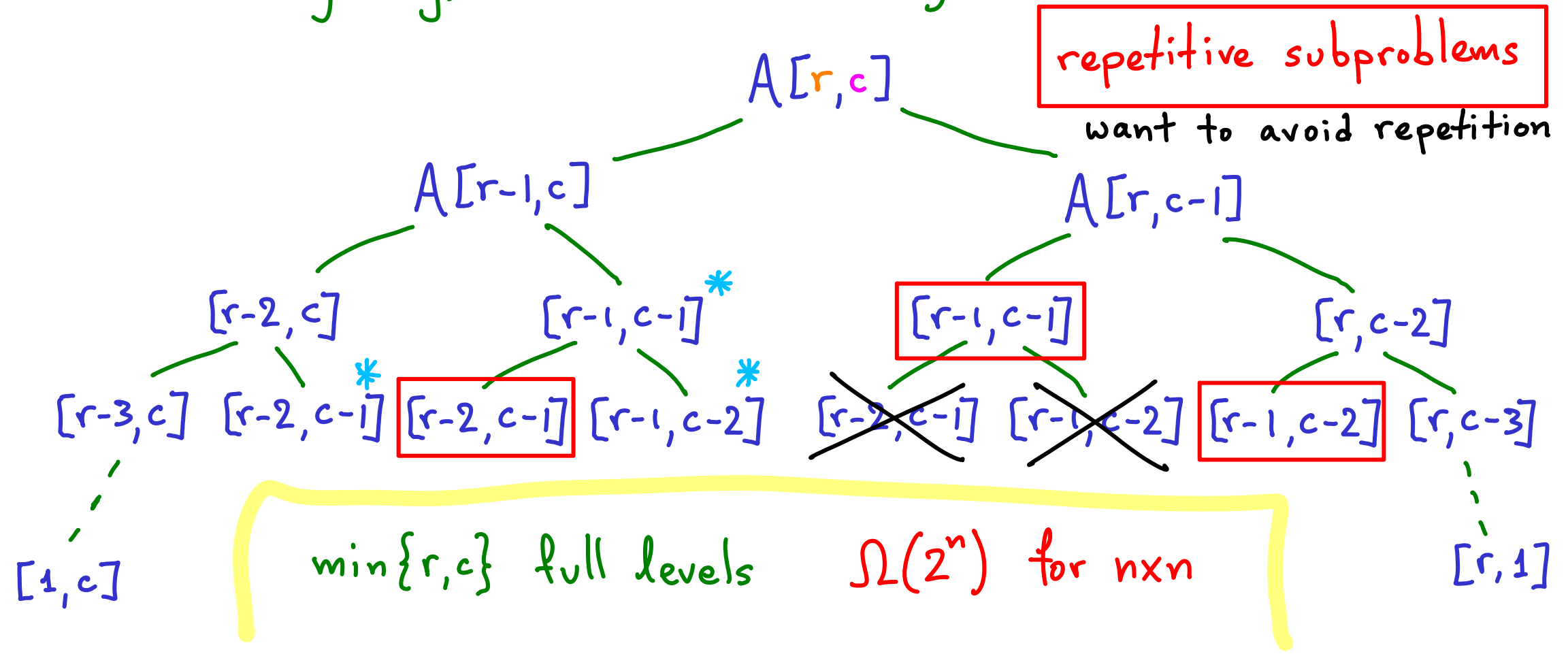
Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?



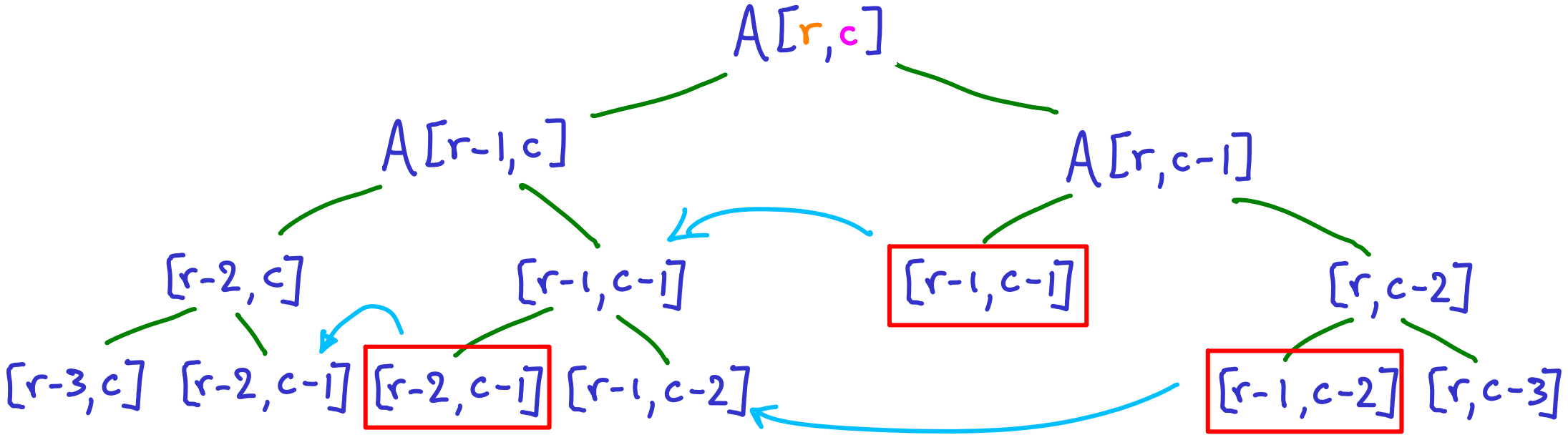
Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?



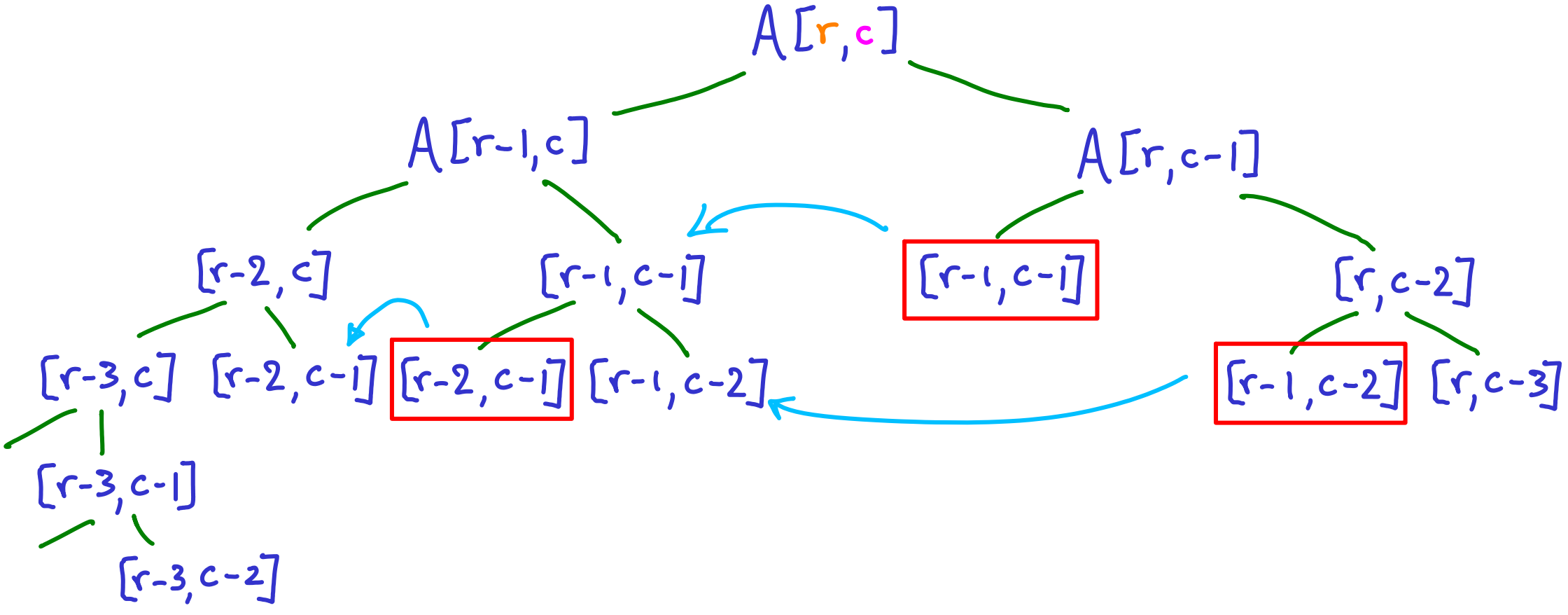
Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?



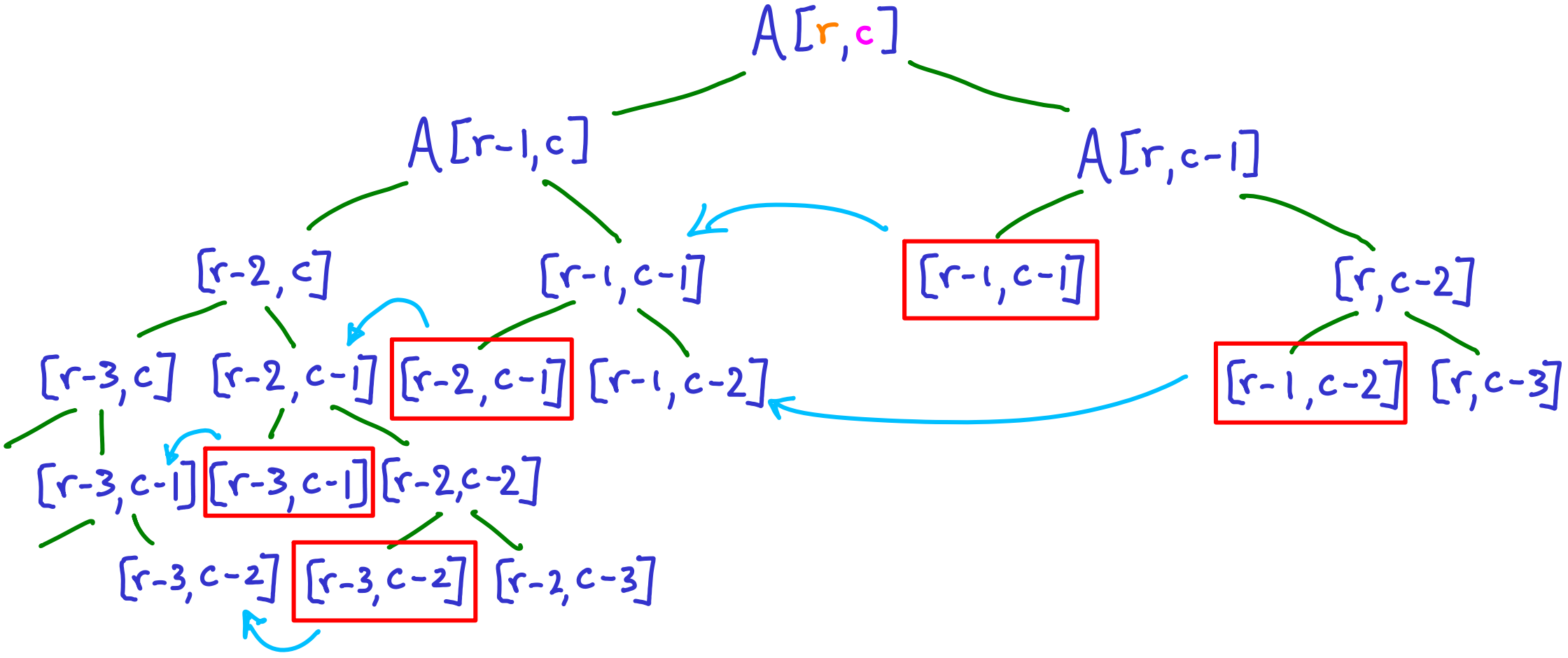
Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?



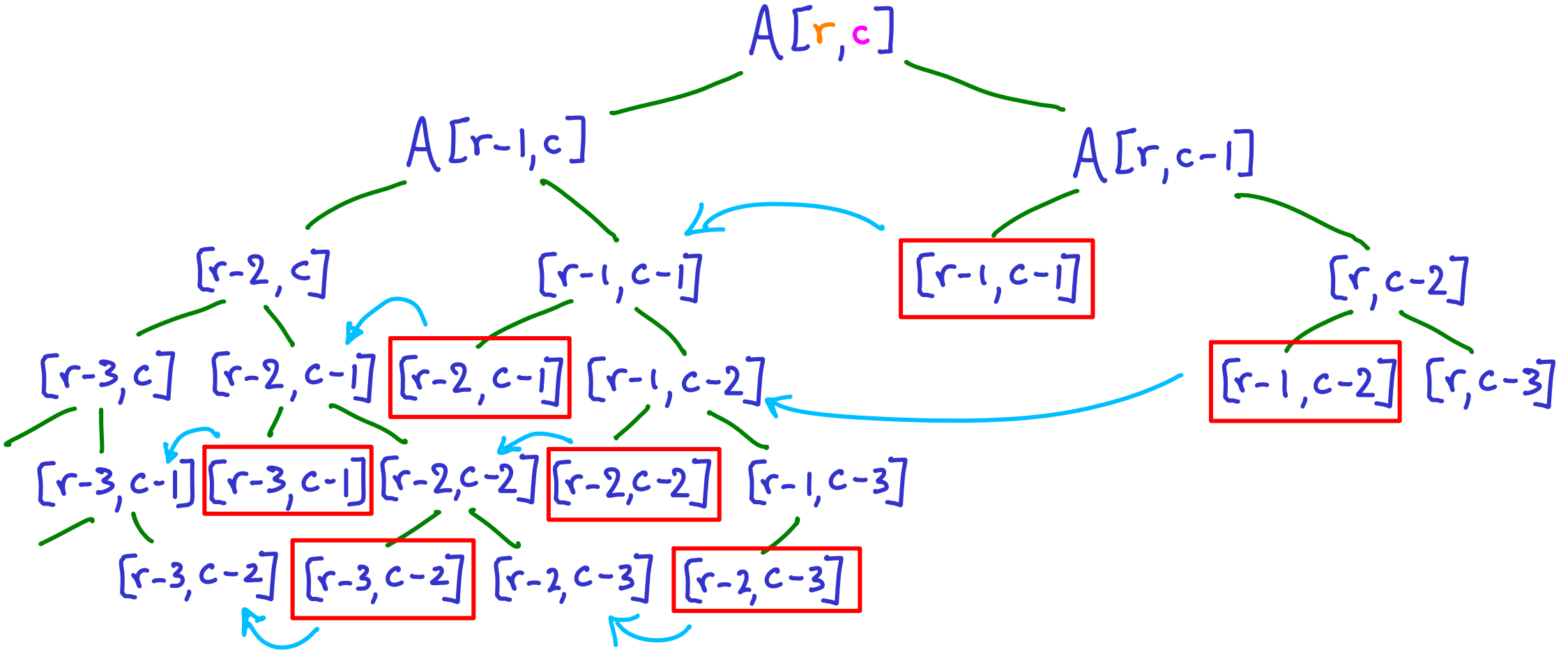
Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?



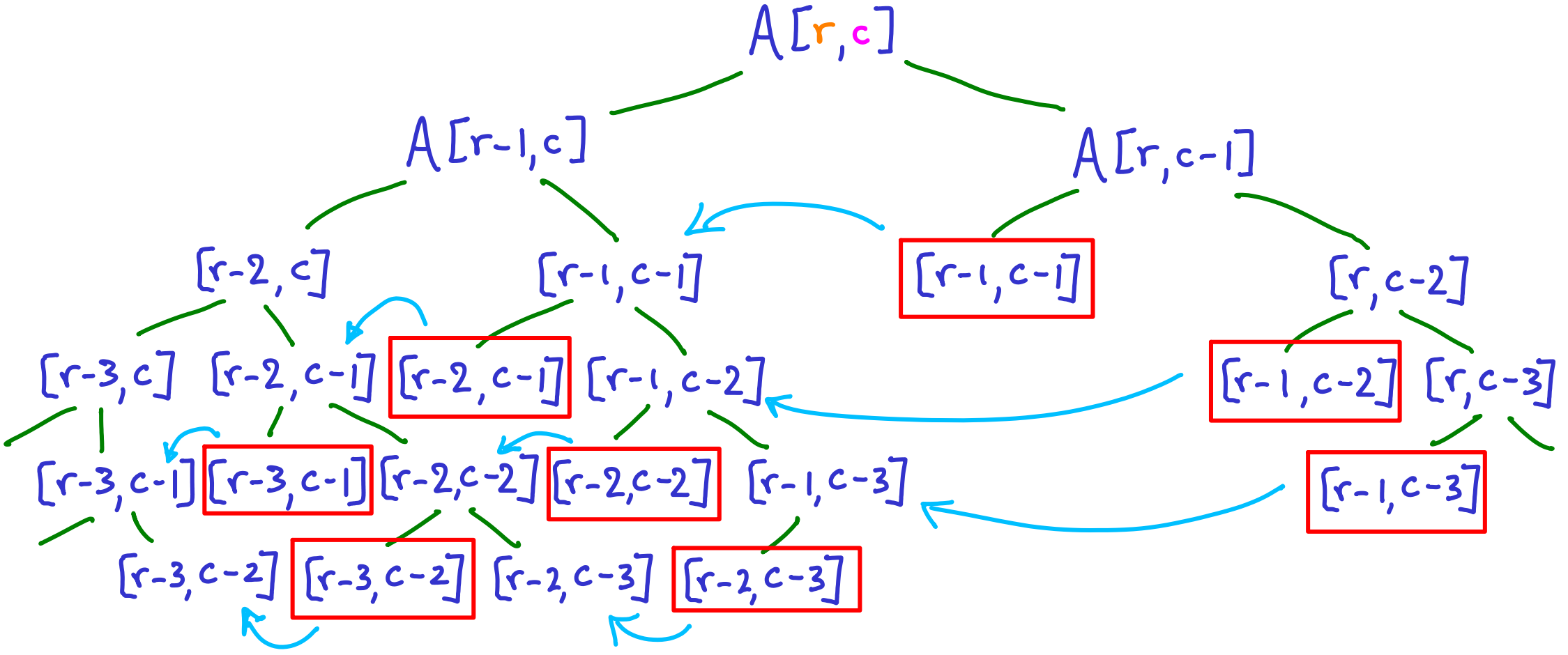
Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?



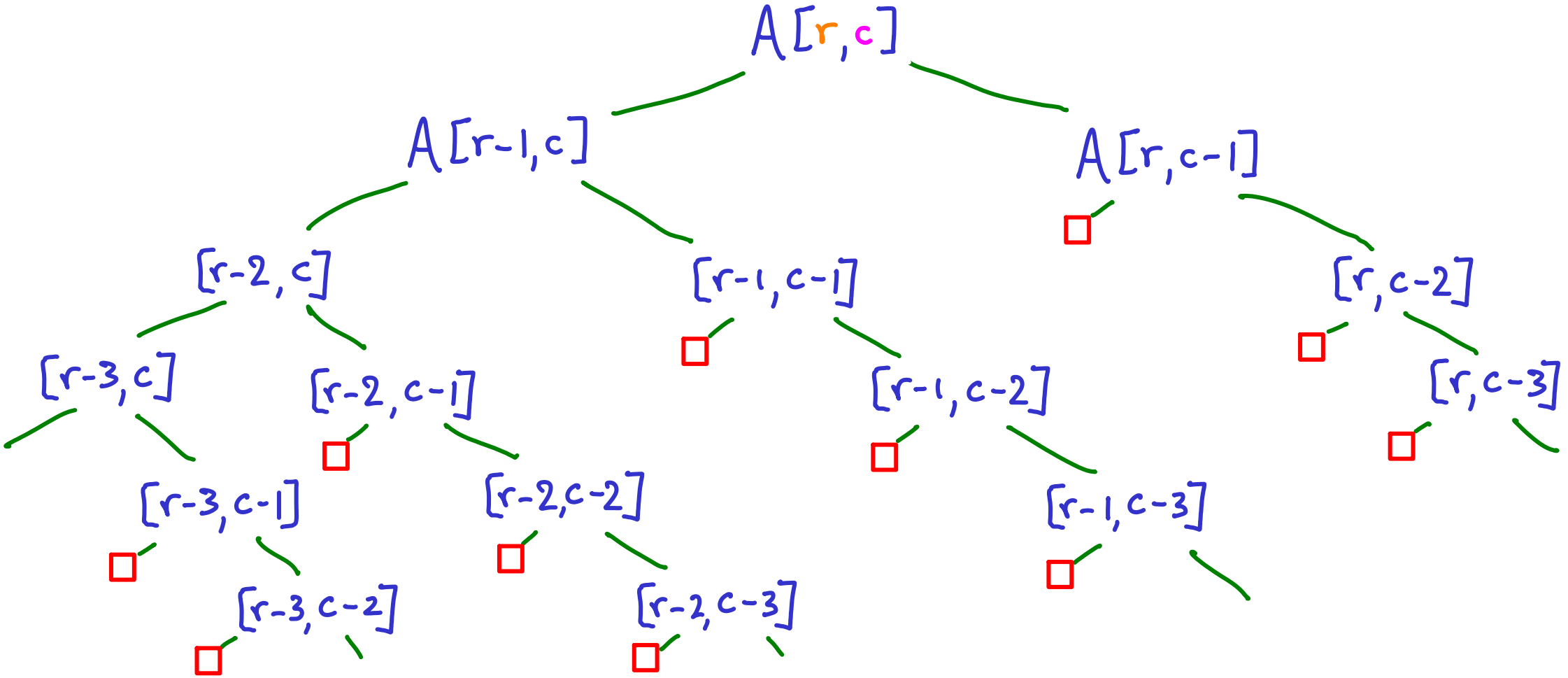
Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?



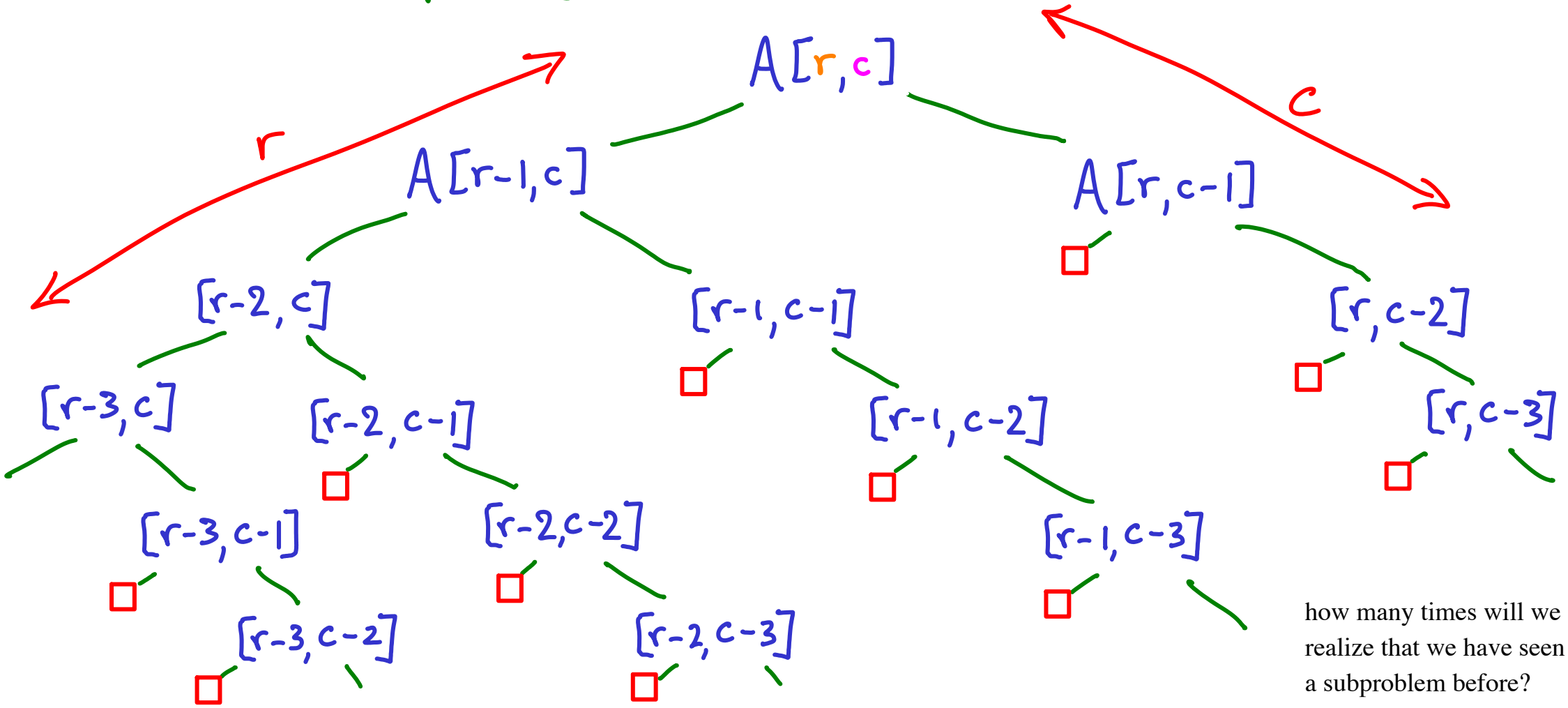
Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?



How many times will we recurse in a unique way?



How many times will we recurse in a unique way? → r·c distinct subproblems

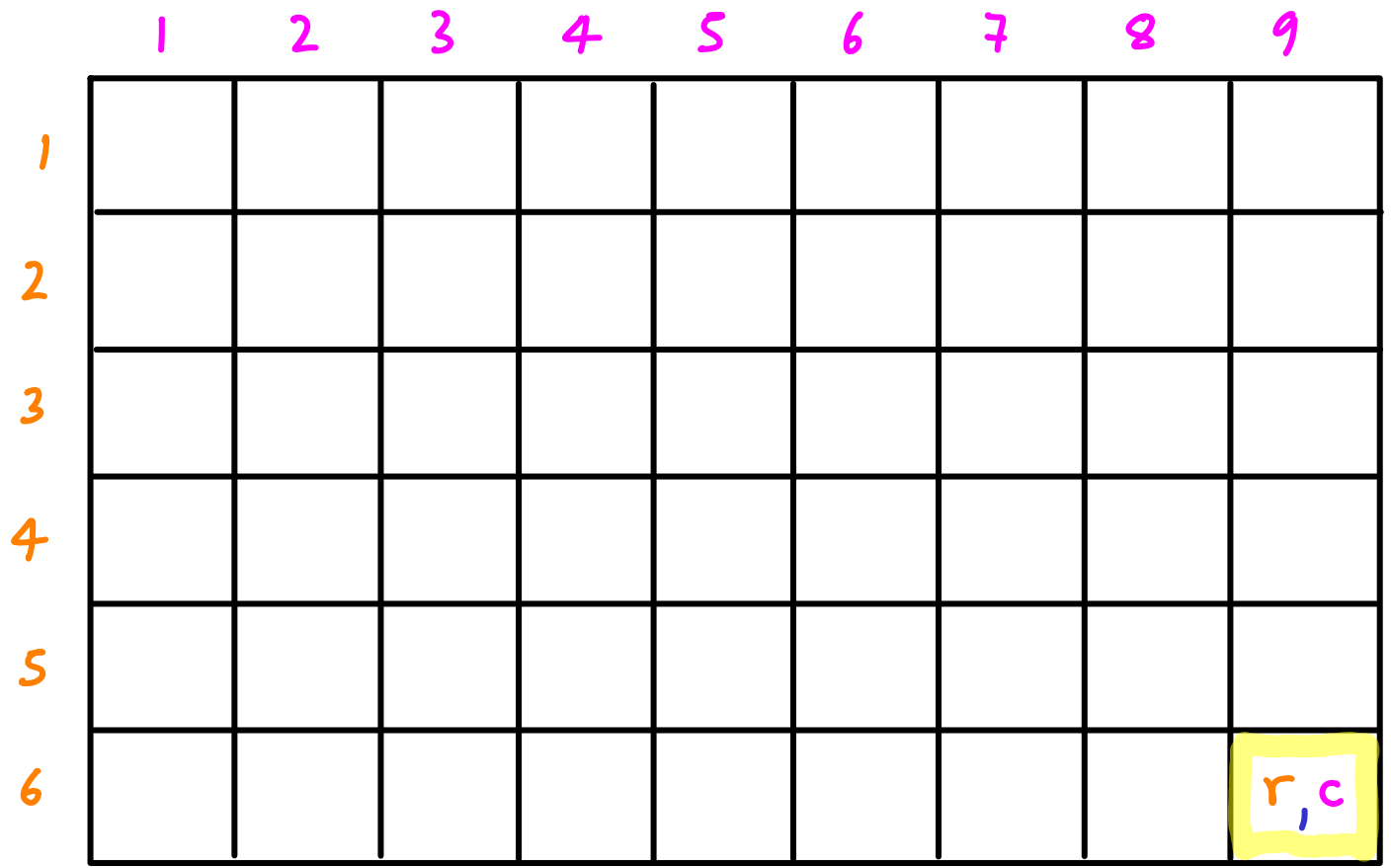


how many times will we realize that we have seen a subproblem before?

MEMOIZATION (making memos)

For this problem, mxn table

$$A[r,c] = A[r-1,c] + A[r,c-1]$$



Recursion:

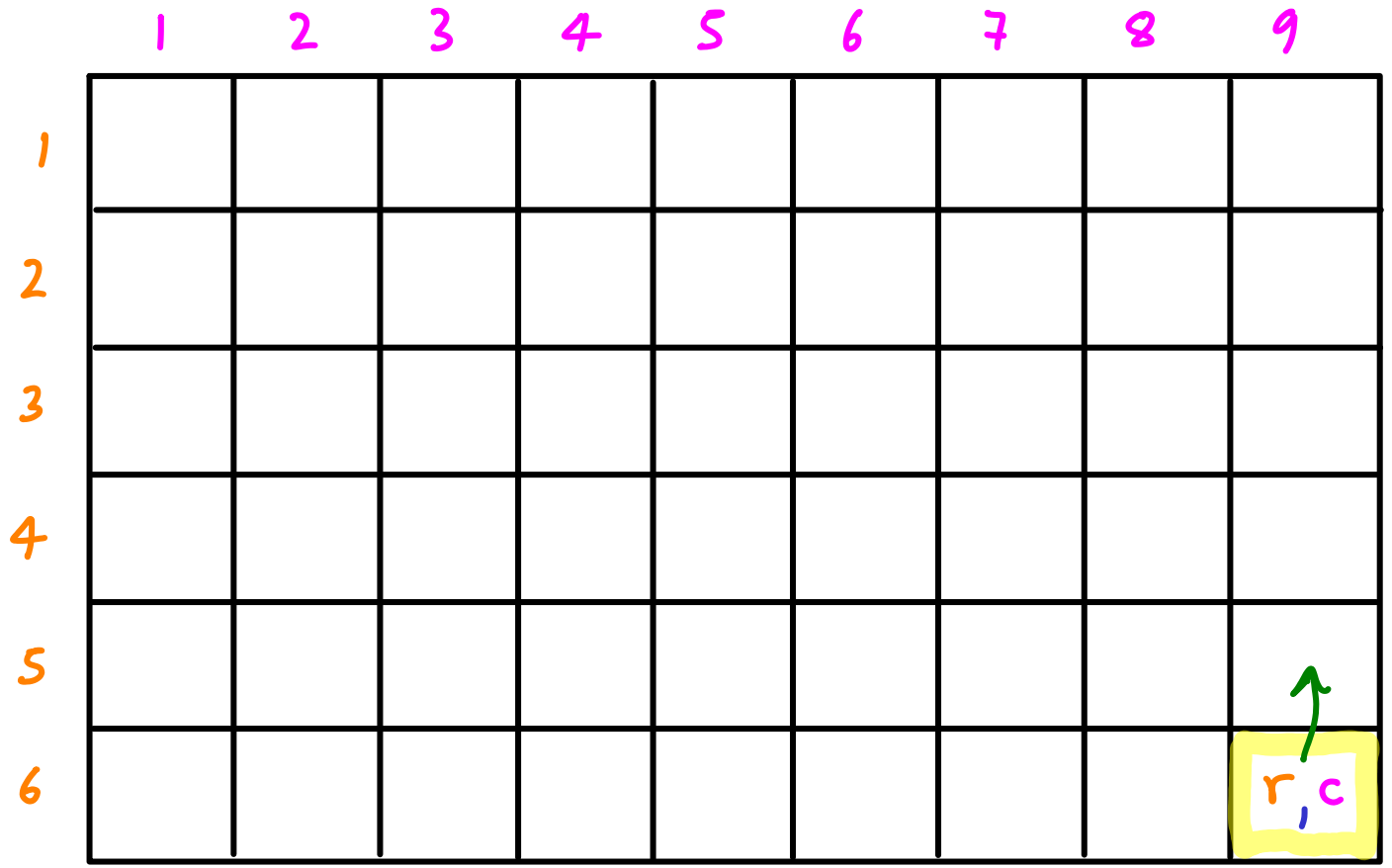
first find $A[r-1,c]$ ↑

then find $A[r,c-1]$ ←

MEMOIZATION (making memos)

For this problem, mxn table

$$A[r,c] = A[r-1,c] + A[r,c-1]$$



Recursion:

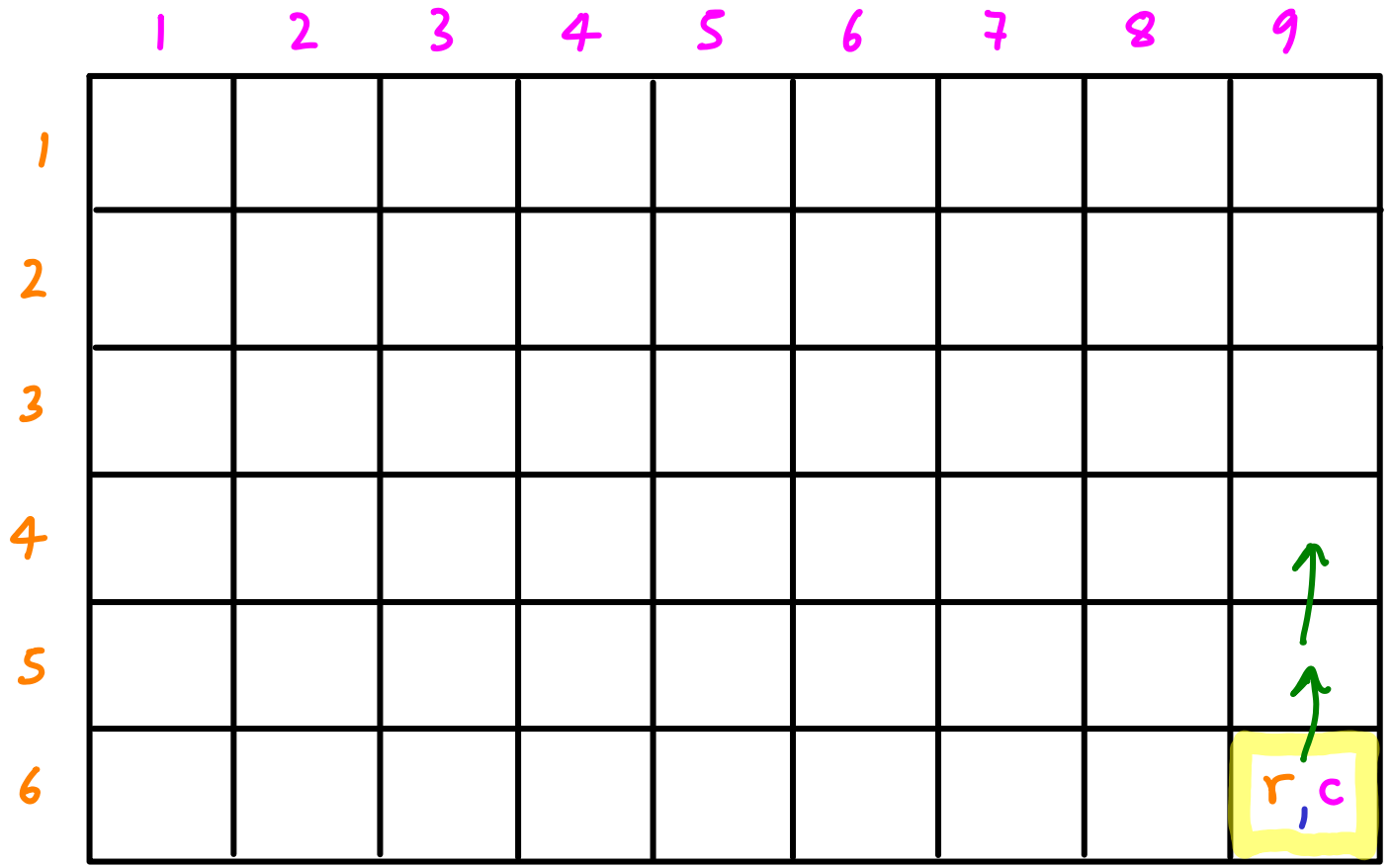
first find $A[r-1,c]$ ↑

then find $A[r,c-1]$ ←

MEMOIZATION (making memos)

For this problem, mxn table

$$A[r,c] = A[r-1,c] + A[r,c-1]$$



Recursion:

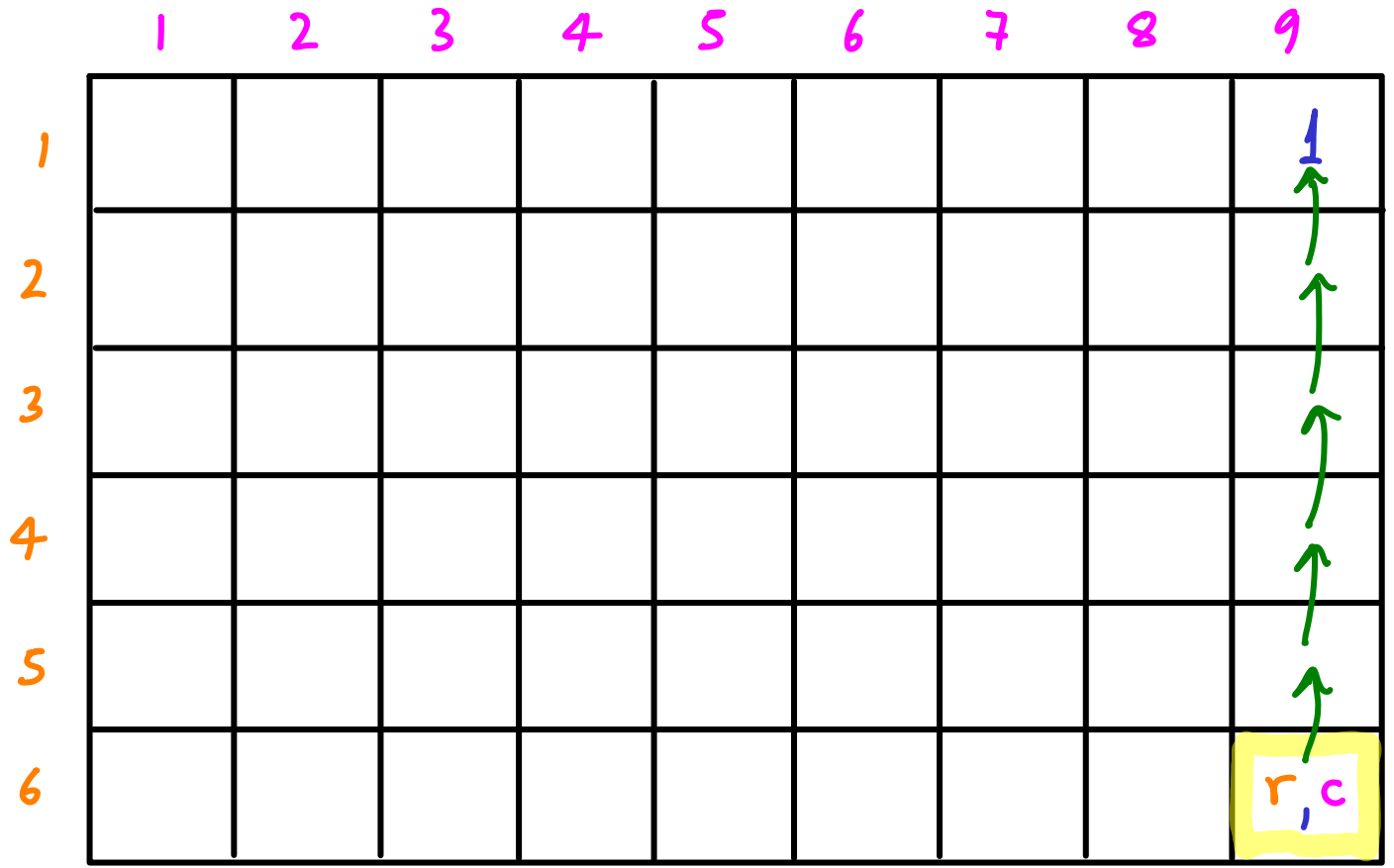
first find $A[r-1,c]$ ↑

then find $A[r,c-1]$ ←

MEMOIZATION (making memos)

For this problem, mxn table

$$A[r,c] = A[r-1,c] + A[r,c-1]$$



Recursion:

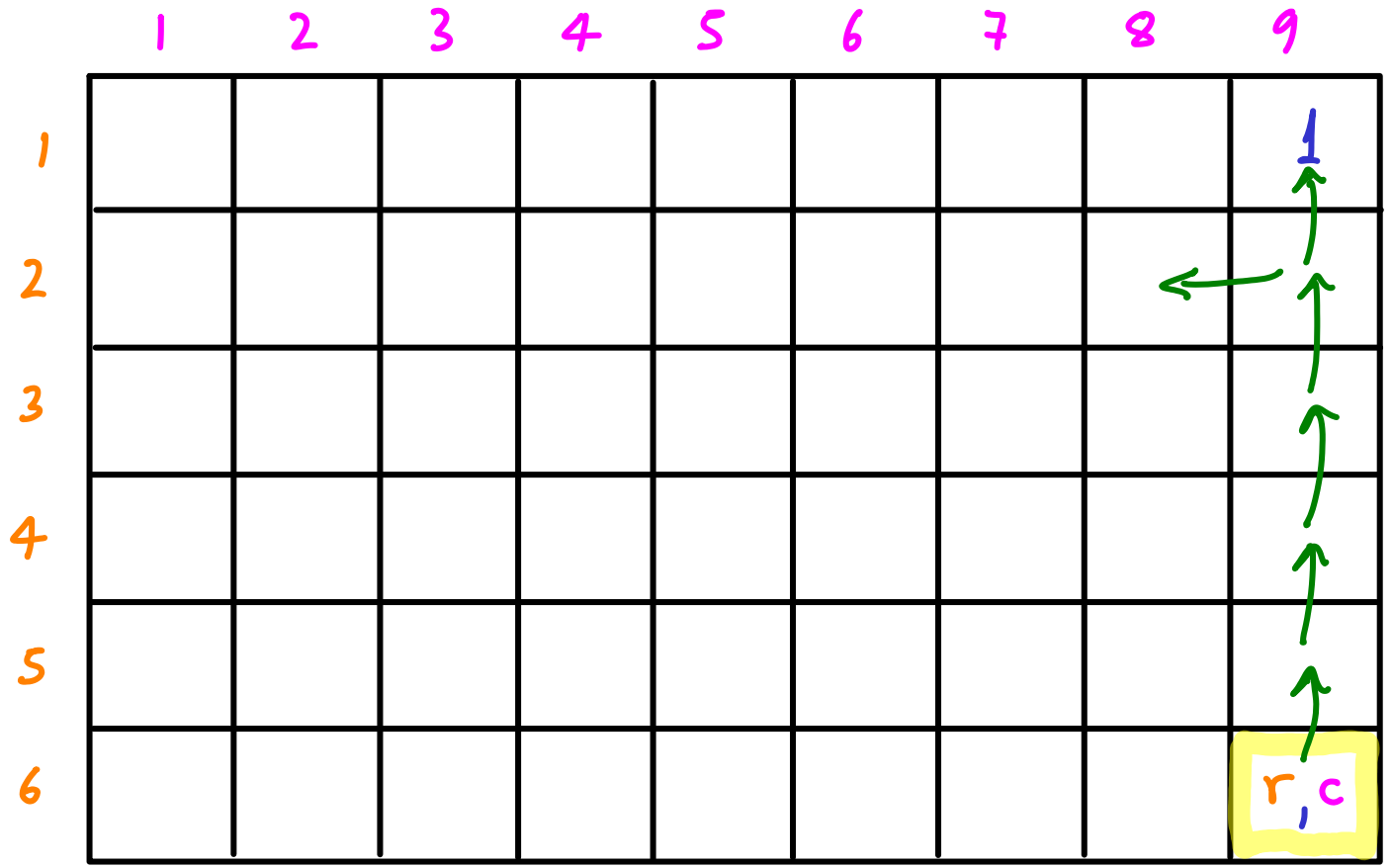
first find $A[r-1,c]$ ↑

then find $A[r,c-1]$ ←

MEMOIZATION (making memos)

For this problem, mxn table

$$A[r,c] = A[r-1,c] + A[r,c-1]$$



Recursion:

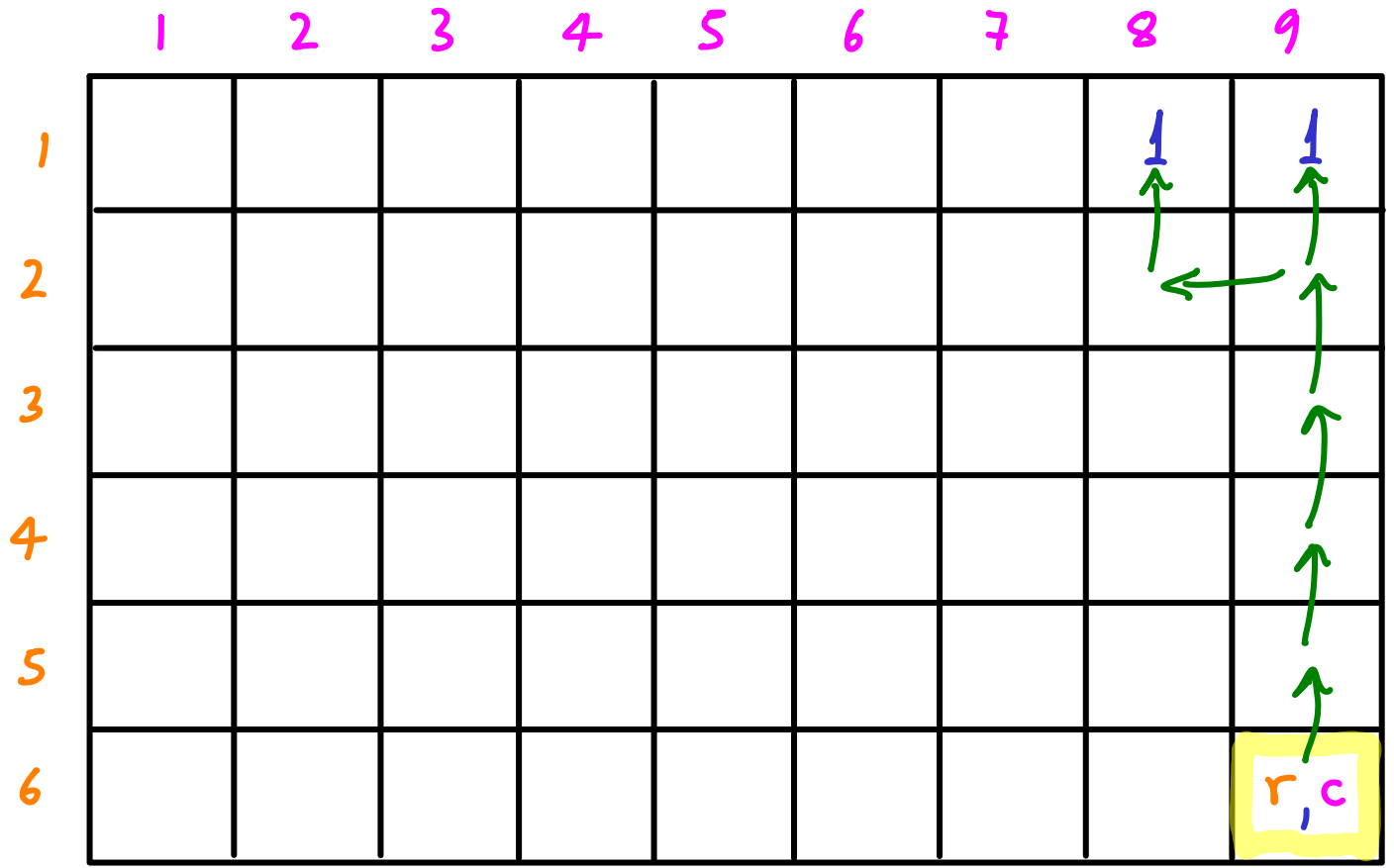
first find $A[r-1,c]$ ↑

then find $A[r,c-1]$ ←

MEMOIZATION (making memos)

For this problem, mxn table

$$A[r,c] = A[r-1,c] + A[r,c-1]$$



Recursion:

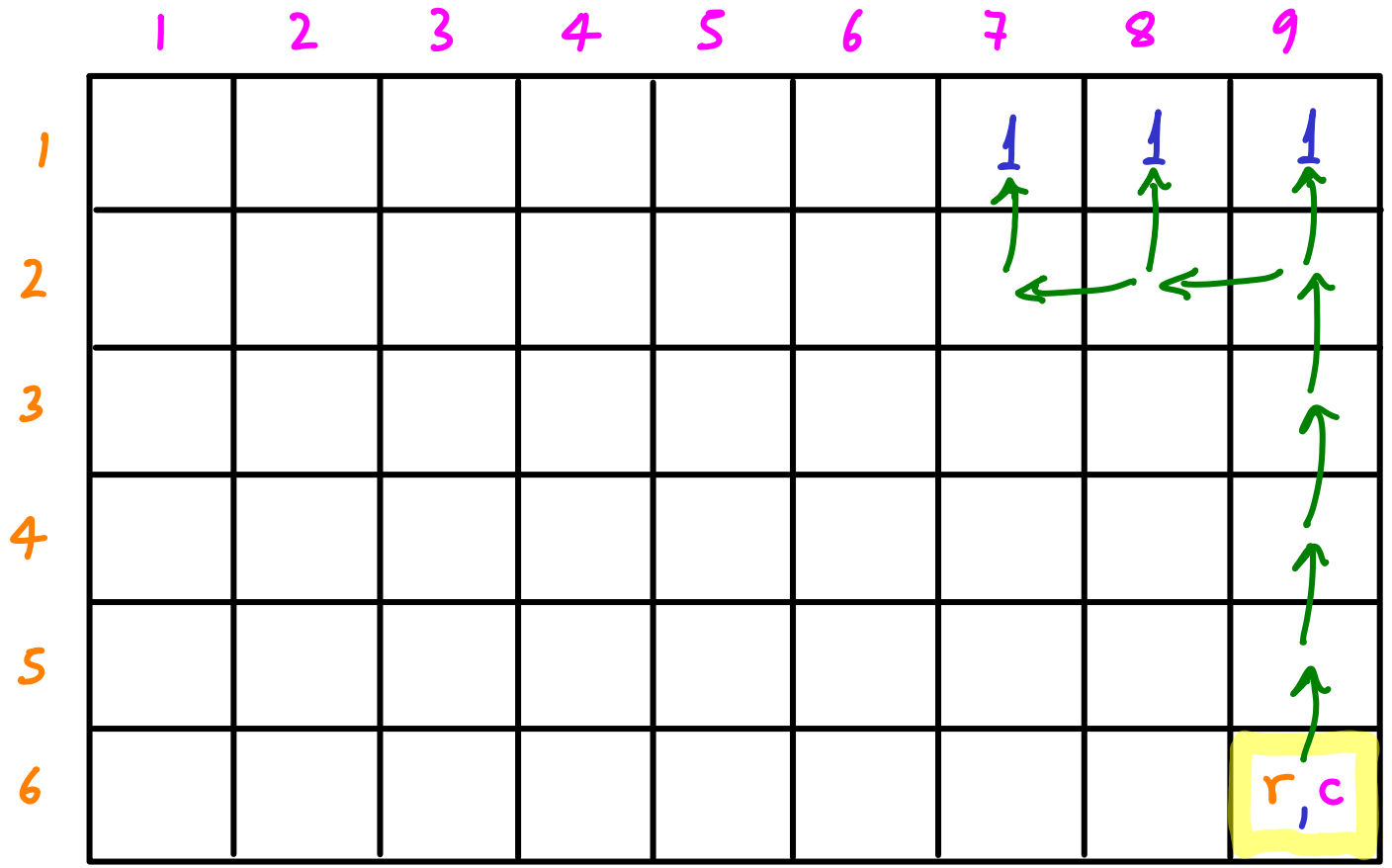
first find $A[r-1,c]$ ↑

then find $A[r,c-1]$ ←

MEMOIZATION (making memos)

For this problem, mxn table

$$A[r,c] = A[r-1,c] + A[r,c-1]$$



Recursion:

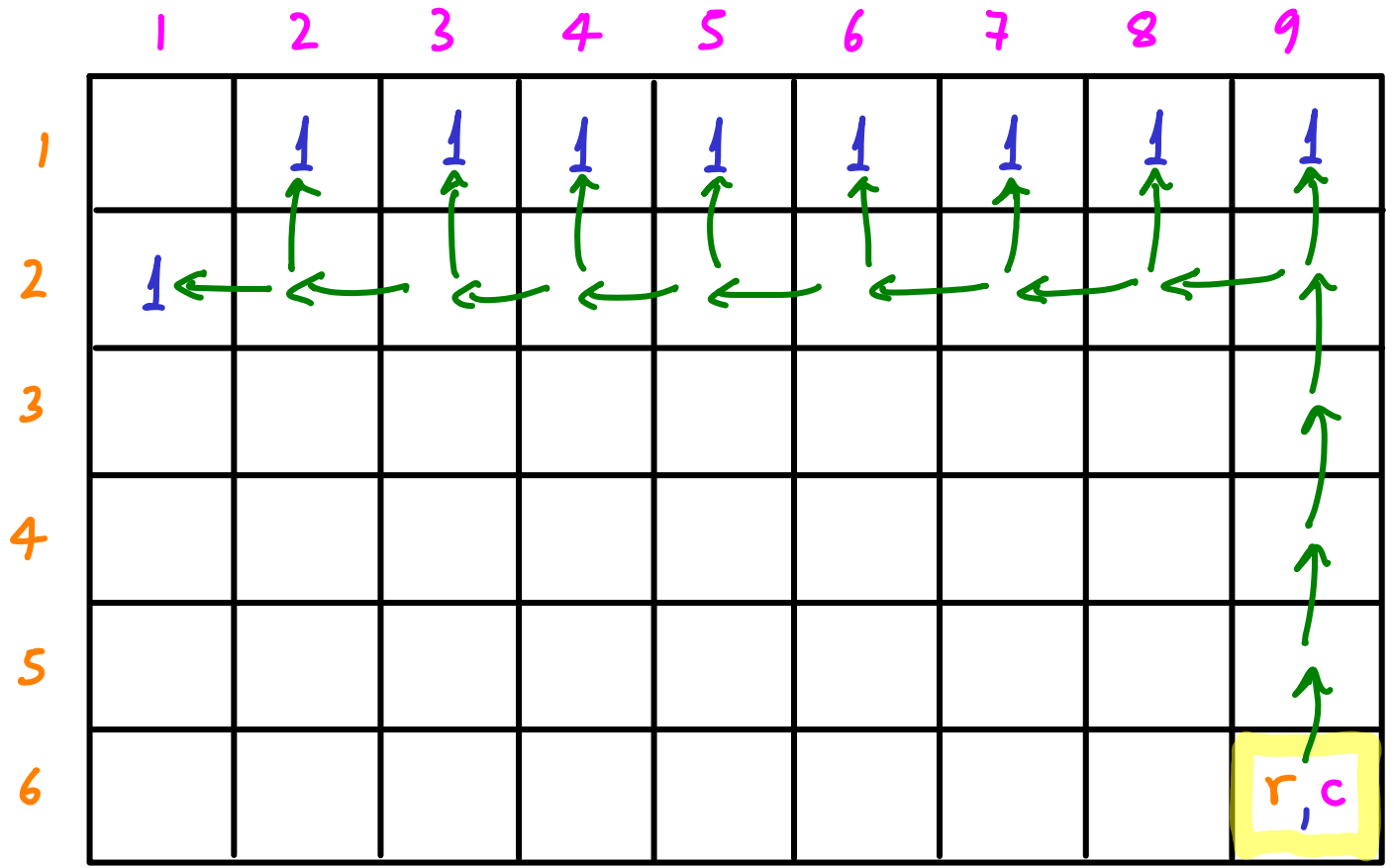
first find $A[r-1,c]$ ↑

then find $A[r,c-1]$ ←

MEMOIZATION (making memos)

For this problem, mxn table

$$A[r,c] = A[r-1,c] + A[r,c-1]$$



Recursion:

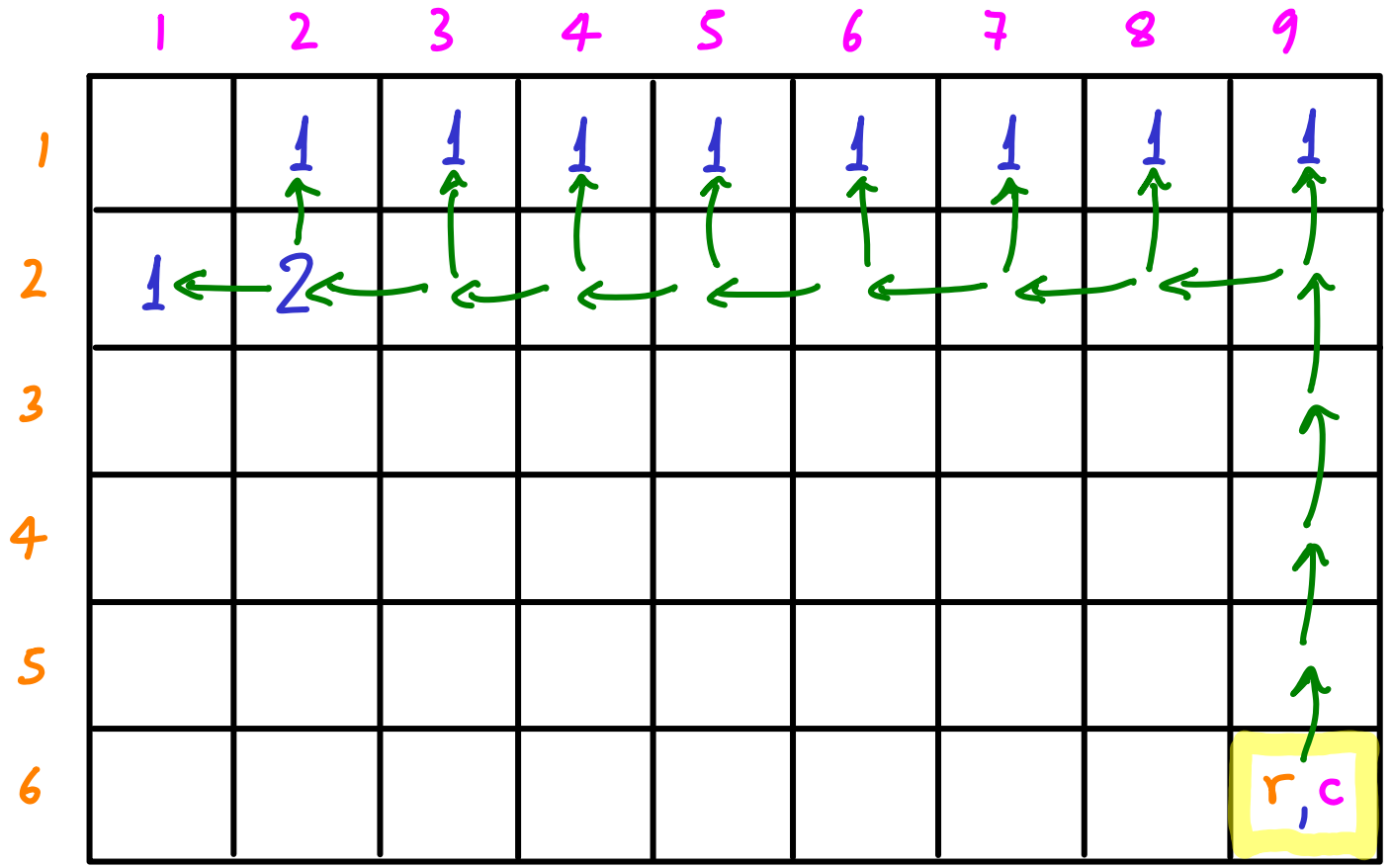
first find $A[r-1,c]$ ↑

then find $A[r,c-1]$ ←

MEMOIZATION (making memos)

For this problem, mxn table

$$A[r,c] = A[r-1,c] + A[r,c-1]$$



Recursion:

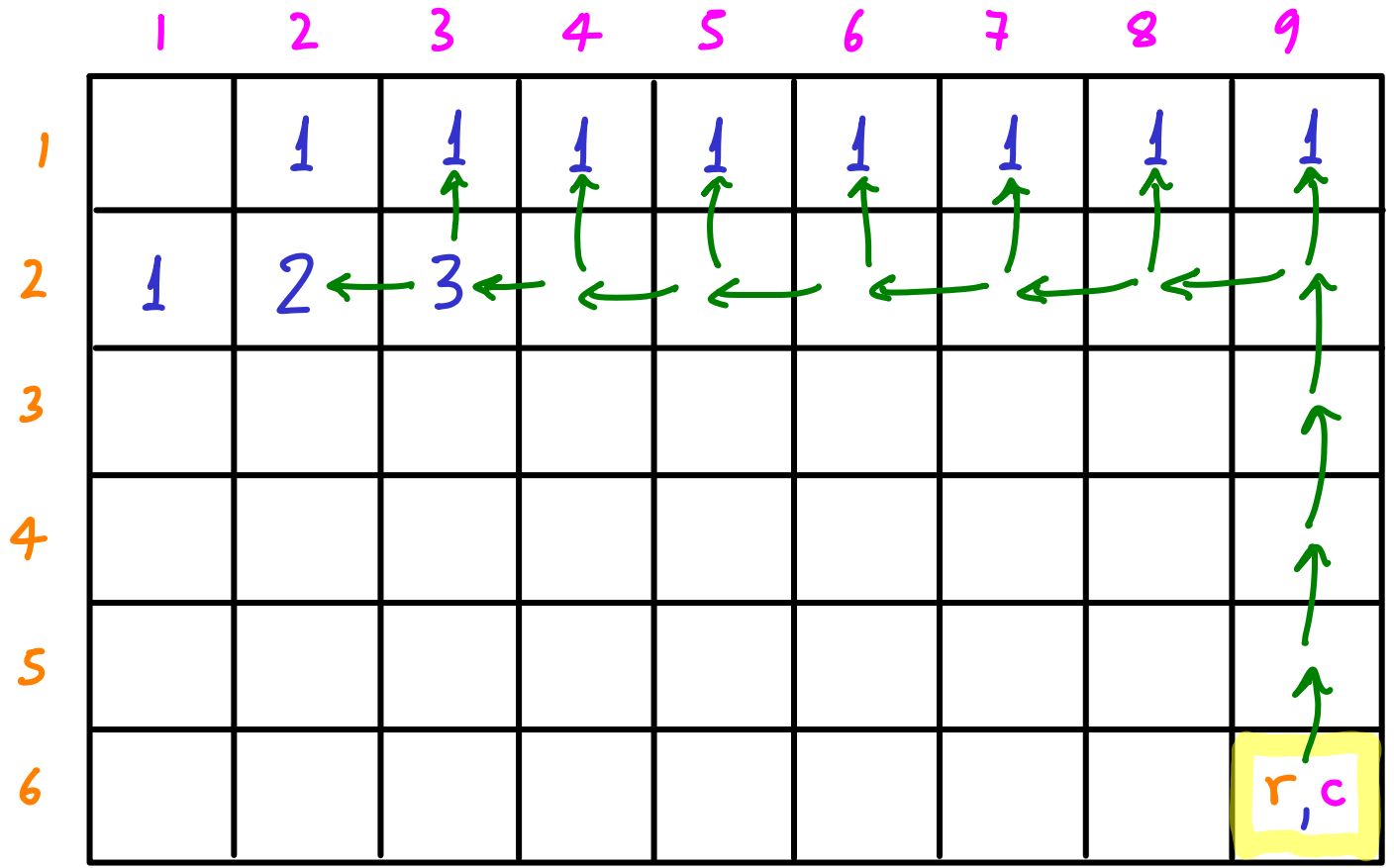
first find $A[r-1,c]$ ↑

then find $A[r,c-1]$ ←

MEMOIZATION (making memos)

For this problem, mxn table

$$A[r,c] = A[r-1,c] + A[r,c-1]$$



Recursion:

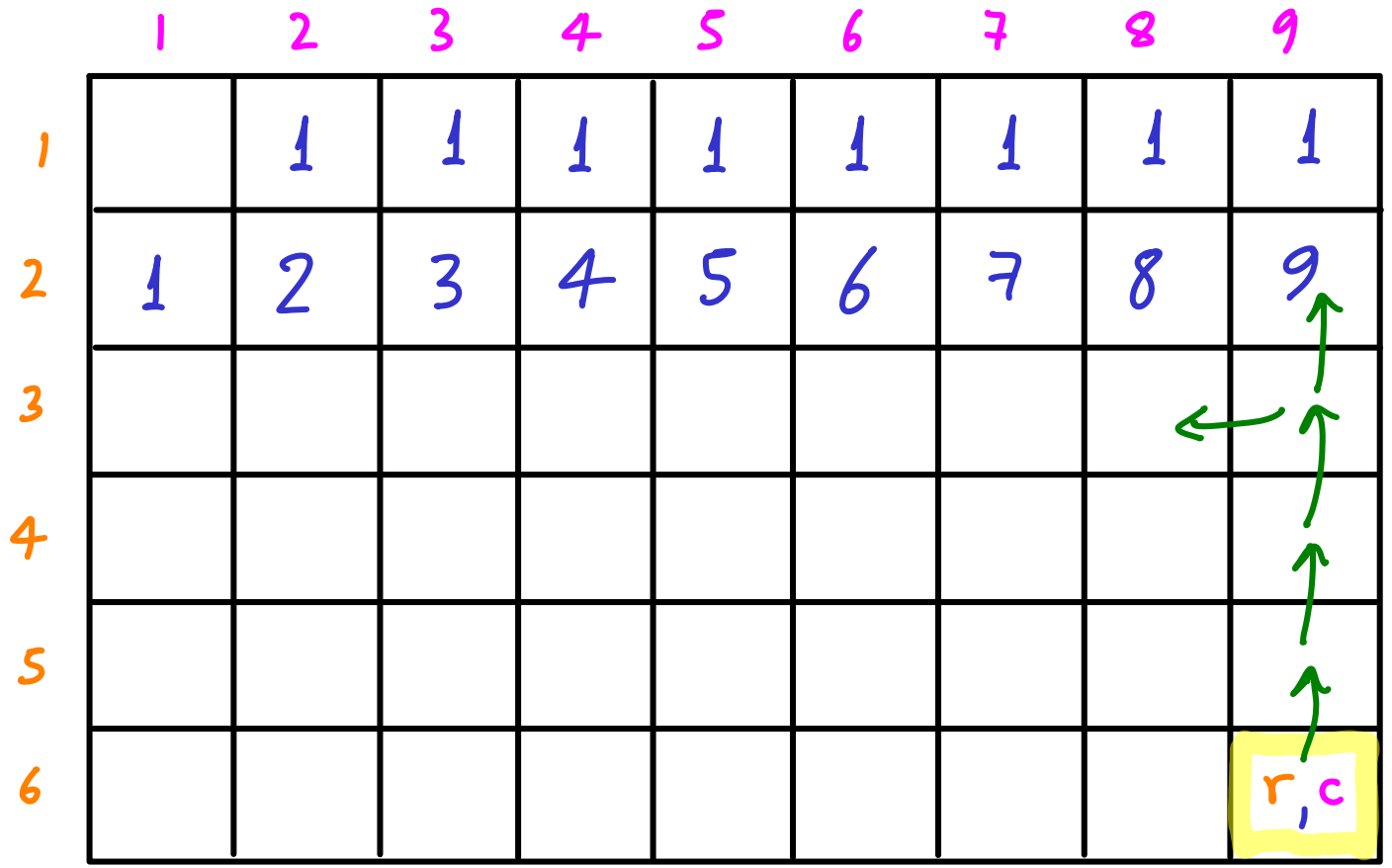
first find $A[r-1,c]$ ↑

then find $A[r,c-1]$ ←

MEMOIZATION (making memos)

For this problem, mxn table

$$A[r,c] = A[r-1,c] + A[r,c-1]$$



Recursion:

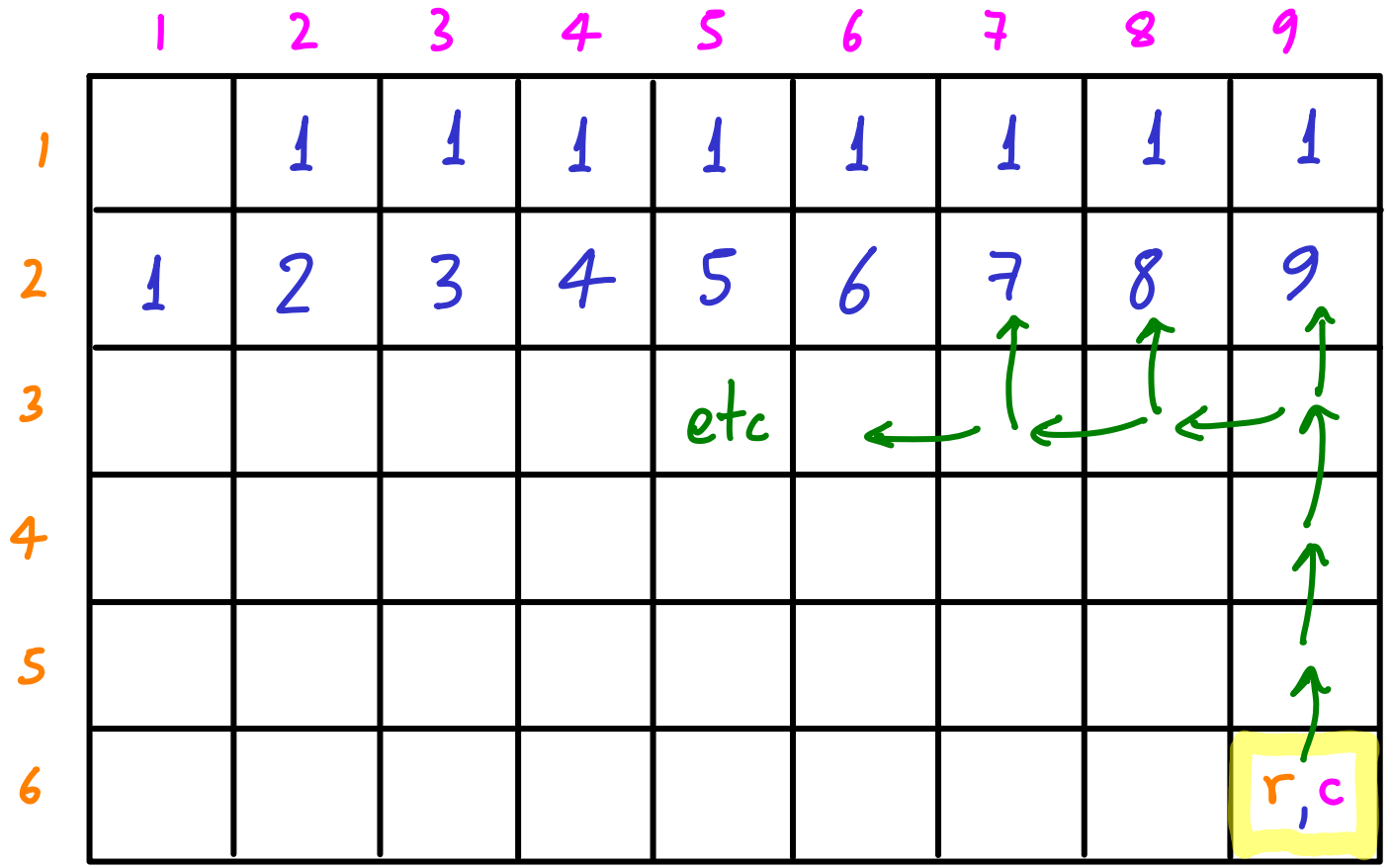
first find $A[r-1,c]$ ↑

then find $A[r,c-1]$ ←

MEMOIZATION (making memos)

For this problem, mxn table

$$A[r,c] = A[r-1,c] + A[r,c-1]$$



Recursion:

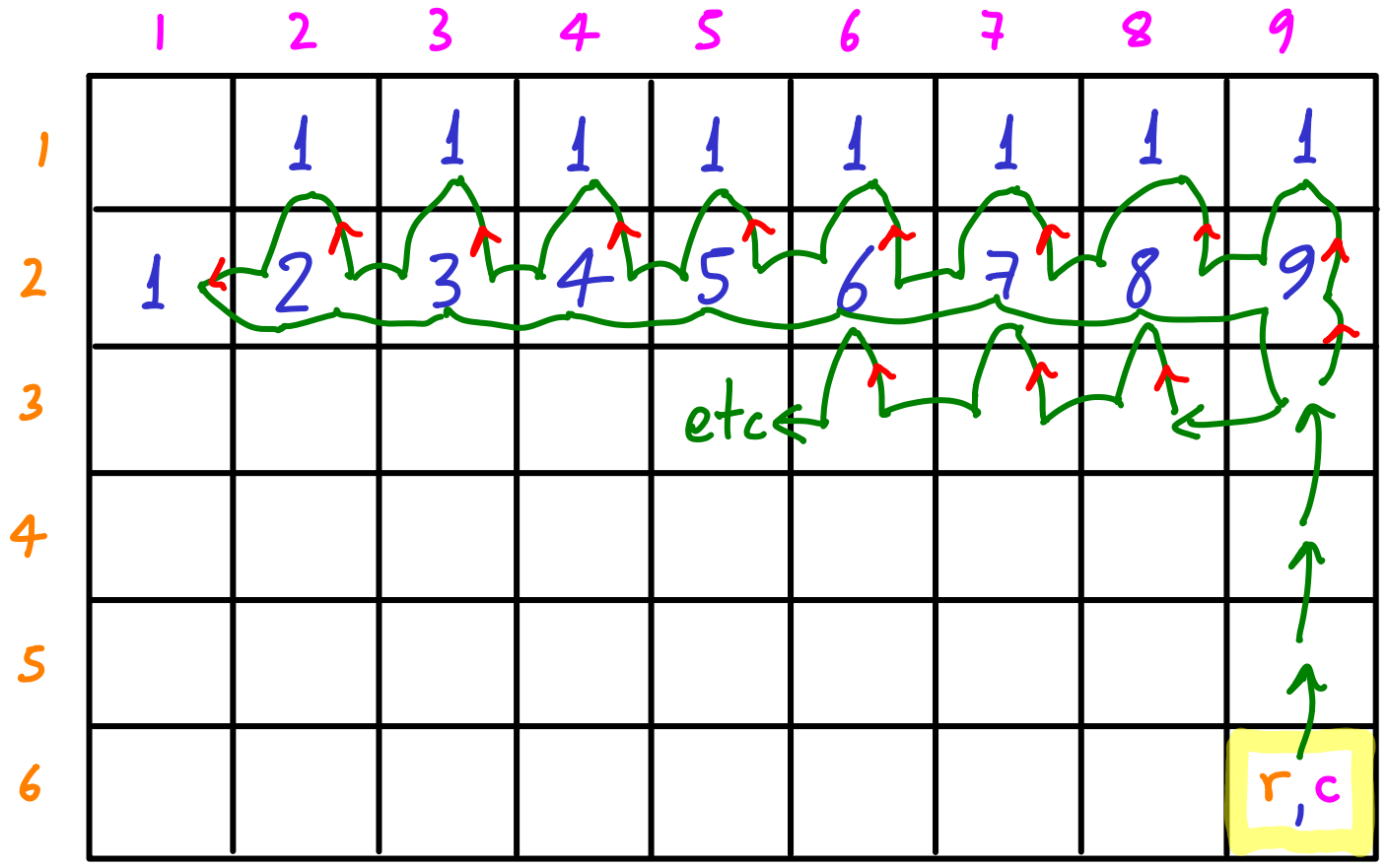
first find $A[r-1,c]$ ↑

then find $A[r,c-1]$ ←

MEMOIZATION (making memos)

For this problem, m x n table

$$A[r,c] = A[r-1,c] + A[r,c-1]$$



Recursion:

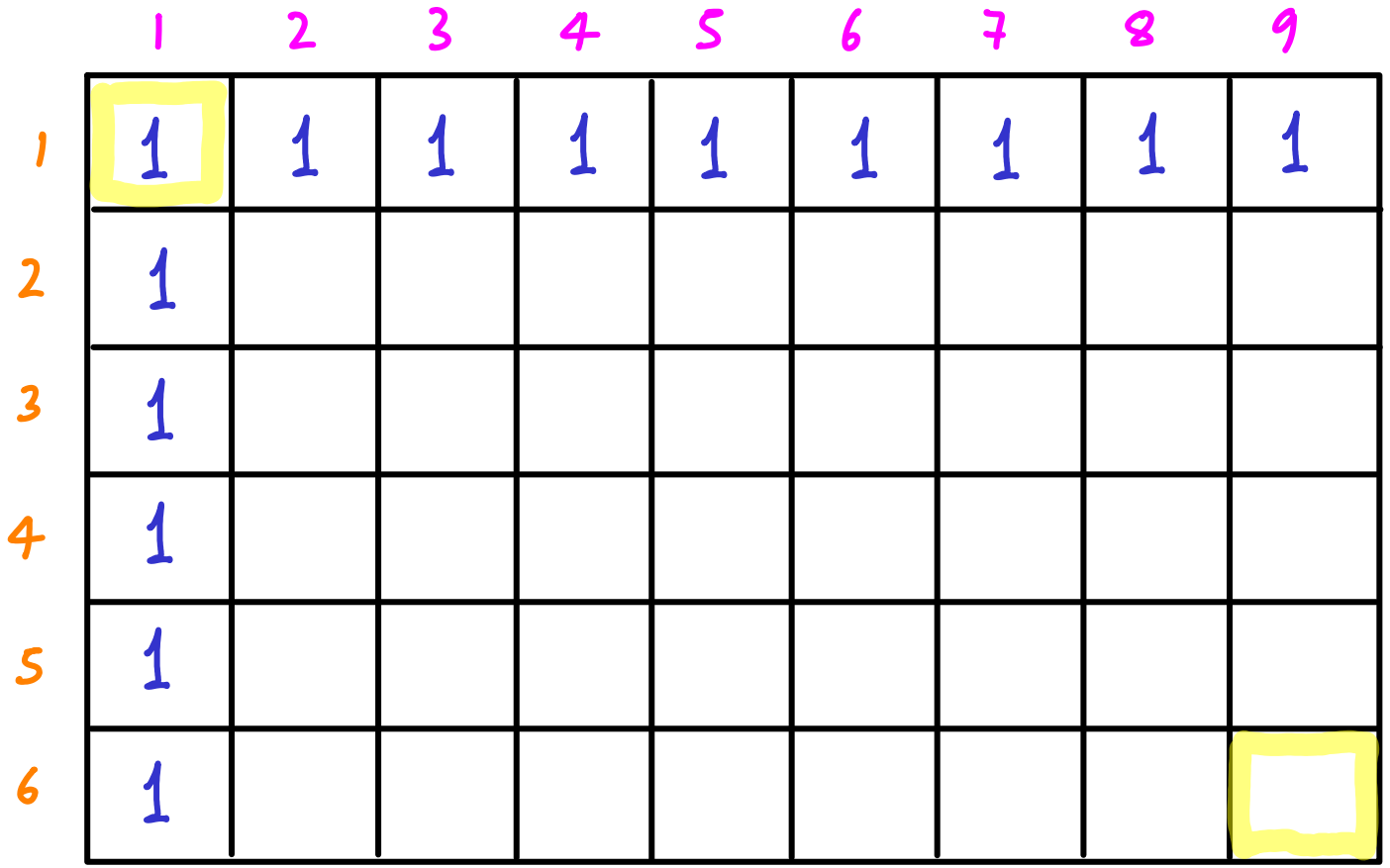
first find $A[r-1,c]$ ↑

then find $A[r,c-1]$ ←

$\Theta(n \cdot m)$ time & space

Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?

DYNAMIC PROGRAMMING (bottom-up : base cases first)

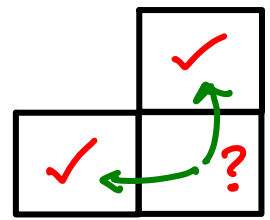


Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?

DYNAMIC PROGRAMMING (bottom-up : base cases first)

	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	1	1	1	1
2	1								
3	1								
4	1								
5	1								
6	1								

$A[r,c] = A[r-1,c] + A[r,c-1]$



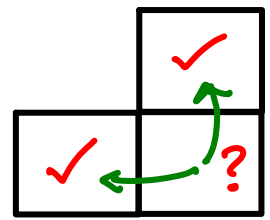
fill any cell as long as what it depends on is full

Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?

DYNAMIC PROGRAMMING (bottom-up : base cases first)

	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	1	1	1	1
2	1	2							
3	1								
4	1								
5	1								
6	1								

$$A[r,c] = A[r-1,c] + A[r,c-1]$$



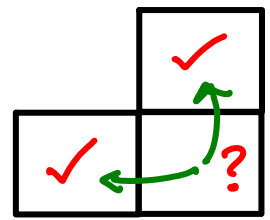
fill any cell as long as what it depends on is full

Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?

DYNAMIC PROGRAMMING (bottom-up : base cases first)

	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	1	1	1	1
2	1	2	3						
3	1								
4	1								
5	1								
6	1								

$A[r,c] = A[r-1,c] + A[r,c-1]$



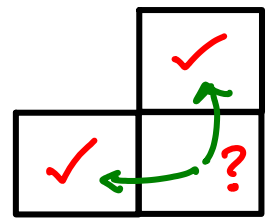
fill any cell as long as what it depends on is full

Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?

DYNAMIC PROGRAMMING (bottom-up : base cases first)

	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	1	1	1	1
2	1	2	3	4					
3	1								
4	1								
5	1								
6	1								

$A[r,c] = A[r-1,c] + A[r,c-1]$



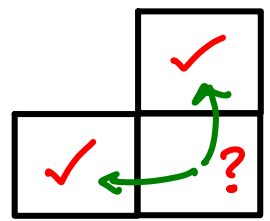
fill any cell as long as what it depends on is full

Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?

DYNAMIC PROGRAMMING (bottom-up : base cases first)

	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	1	1	1	1
2	1	2	3	4					
3	1	3							
4	1								
5	1								
6	1								

$A[r,c] = A[r-1,c] + A[r,c-1]$



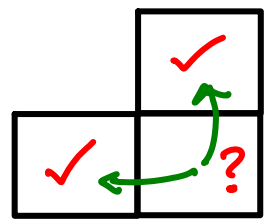
fill any cell as long as what it depends on is full

Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?

DYNAMIC PROGRAMMING (bottom-up : base cases first)

	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	1	1	1	1
2	1	2	3	4	5	6	7	8	
3	1	3	6						
4	1	4	10						
5	1	5							
6	1	6							

$A[r,c] = A[r-1,c] + A[r,c-1]$



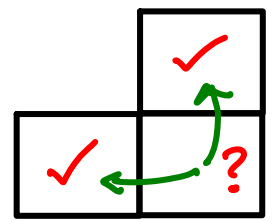
fill any cell as long as what it depends on is full

Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right?

DYNAMIC PROGRAMMING (bottom-up : base cases first)

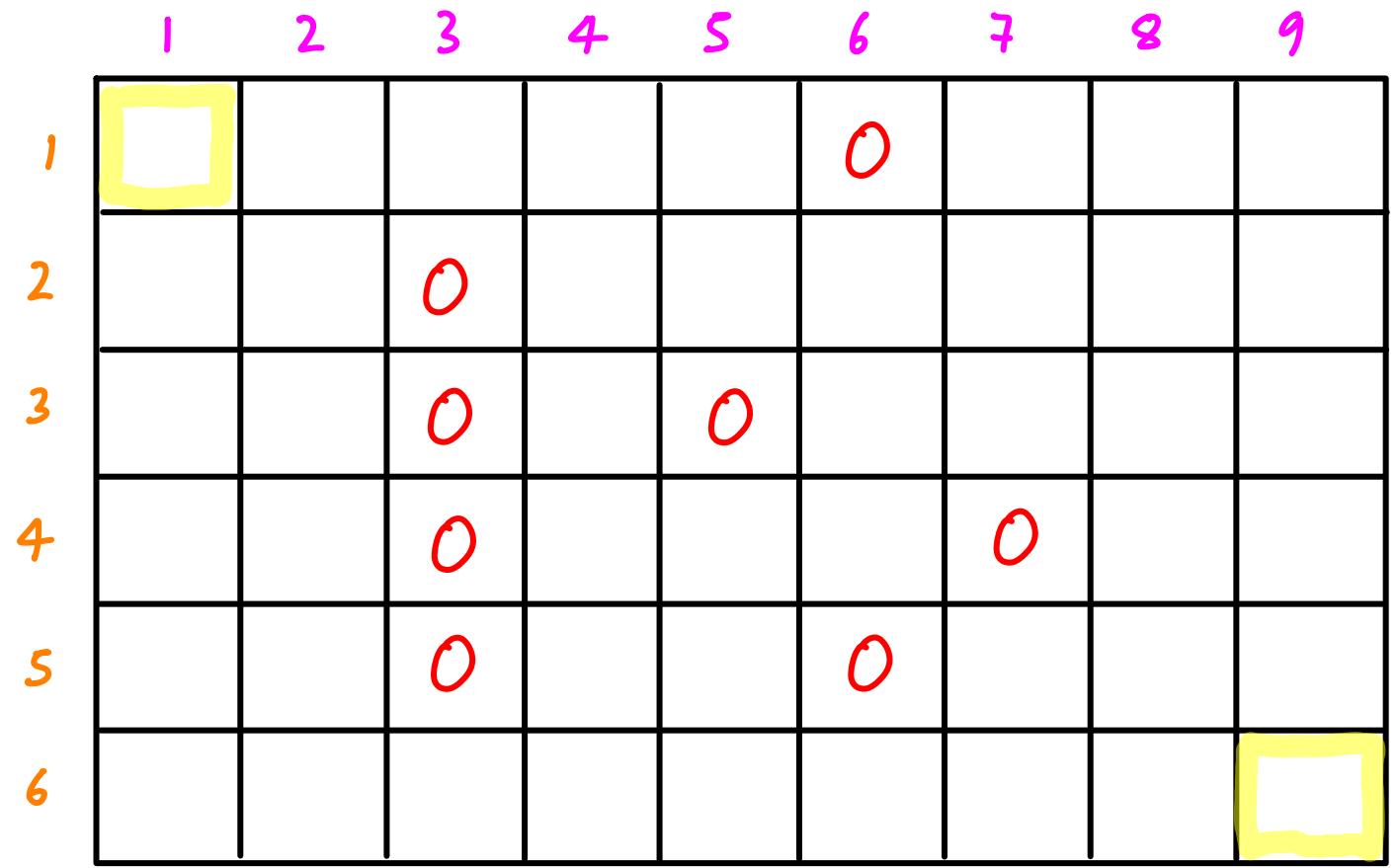
	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	1	1	1	1
2	1	2	3	4	5	6	7	8	9
3	1	3	6	10	15	21	28	36	45
4	1	4	10	20	35	56	84	120	165
5	1	5	15	35	70	126	210	330	495
6	1	6	21	56	126	252	462	792	1287

$A[r,c] = A[r-1,c] + A[r,c-1]$

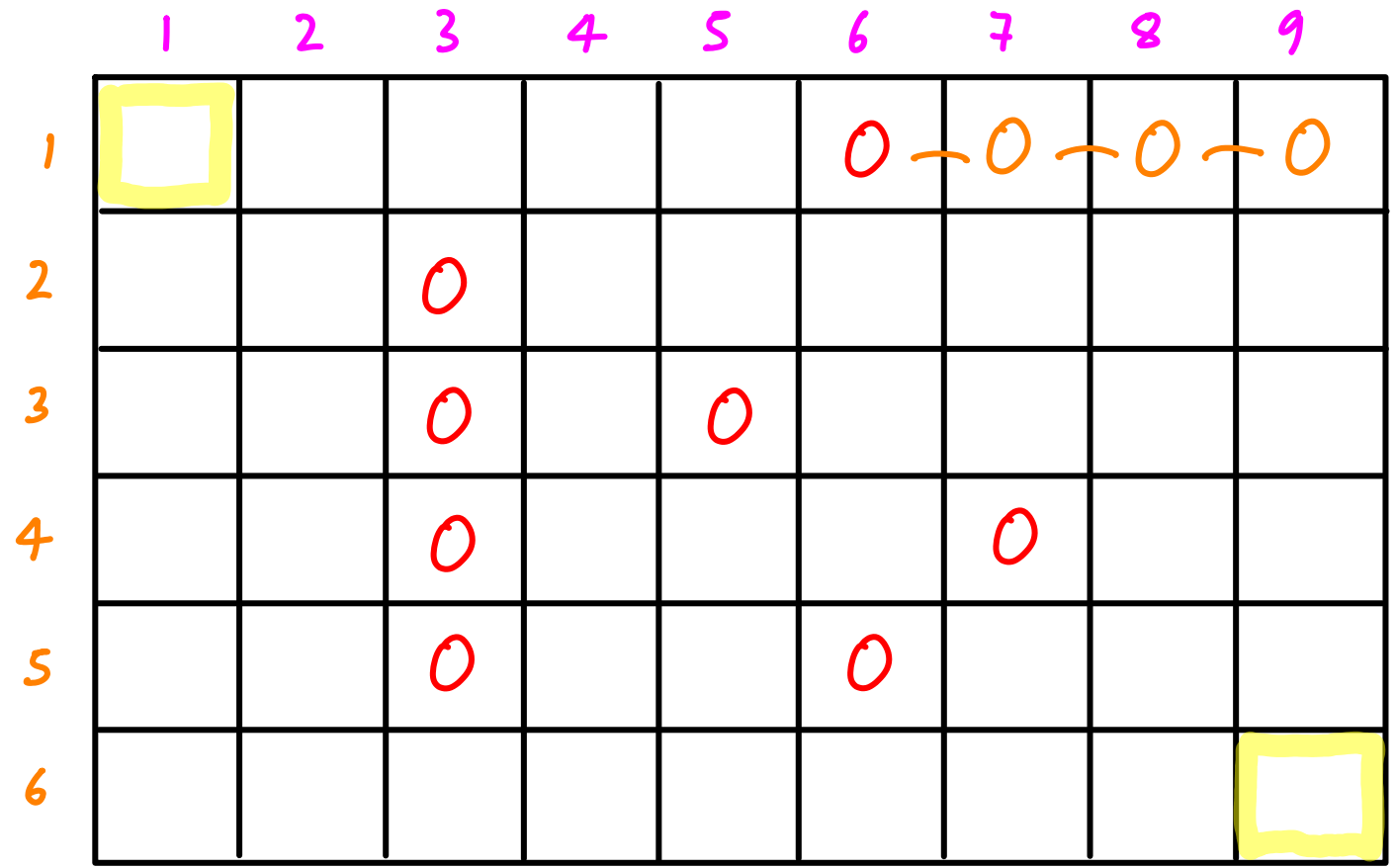


fill any cell as long as what it depends on is full

Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right? ... with obstacles



Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right? ... with obstacles



Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right? ... with obstacles

	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	0	0	0	0
2	1		0						
3	1		0		0				
4	1		0				0		
5	1		0			0			
6	1								

Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right? ... with obstacles

	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	0	0	0	0
2	1	2	0	1	2	2	2	2	2
3	1	3	0		0				
4	1	4	0				0		
5	1	5	0			0			
6	1	6							

Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right? ... with obstacles

	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	0	0	0	0
2	1	2	0	1	2	2	2	2	2
3	1	3	0	1	0				
4	1	4	0	1			0		
5	1	5	0	1		0			
6	1	6	6	7					

Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right? ... with obstacles

	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	0	0	0	0
2	1	2	0	1	2	2	2	2	2
3	1	3	0	1	0	2	4	6	8
4	1	4	0	1	1	3	0		
5	1	5	0	1	2	0	0		
6	1	6	6	7	9				

Starting at top-left of nxm grid, moving only down or right, how many ways to reach bottom-right? ... with obstacles

	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	0	0	0	0
2	1	2	0	1	2	2	2	2	2
3	1	3	0	1	0	2	4	6	8
4	1	4	0	1	1	3	0	6	14
5	1	5	0	1	2	0	0	6	20
6	1	6	6	7	9	9	9	15	35