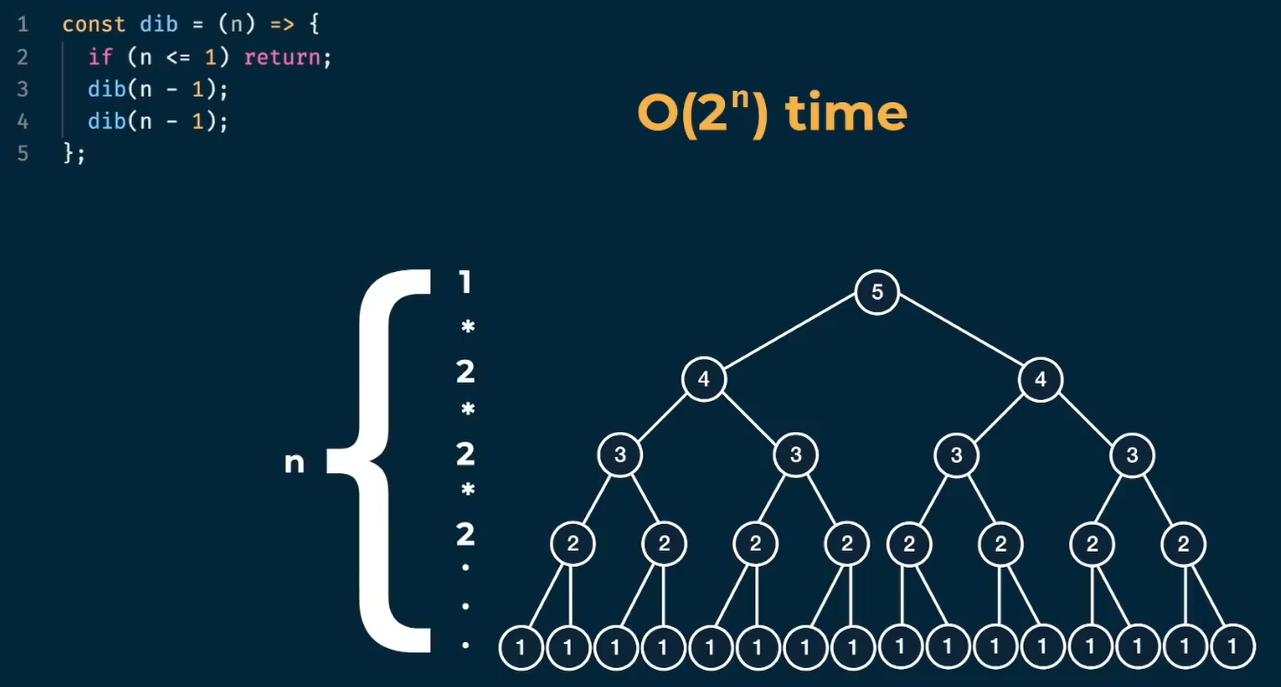
**Dynamic programming**

Large problema that we can decompose into small instances of the same problema and also have an overlaping structure, like the fib tree

**Why fib is O(2^n) time**

Example (This is not fib but the structure is almost the same)



**O(2^n) time | O(n) space**

Imagine this function, the number of nodes in each level(Binary tree) are the last number of nodes times 2:

Level 0: 1

Level 1: 1 \* 2 = 2

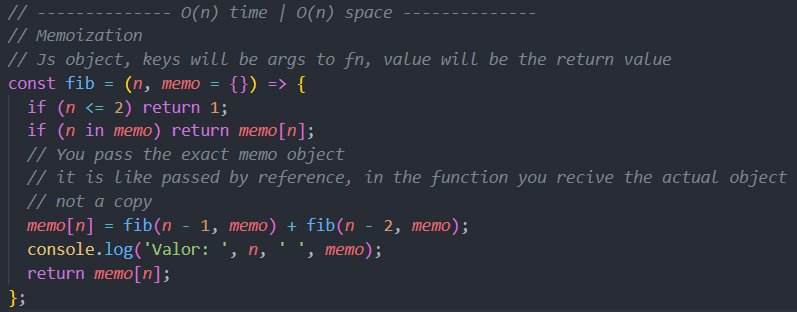
Level 2: 1 \* 2 \* 2 = 4

Level 3: 1 \* 2 \* 2 \* 2 = 8

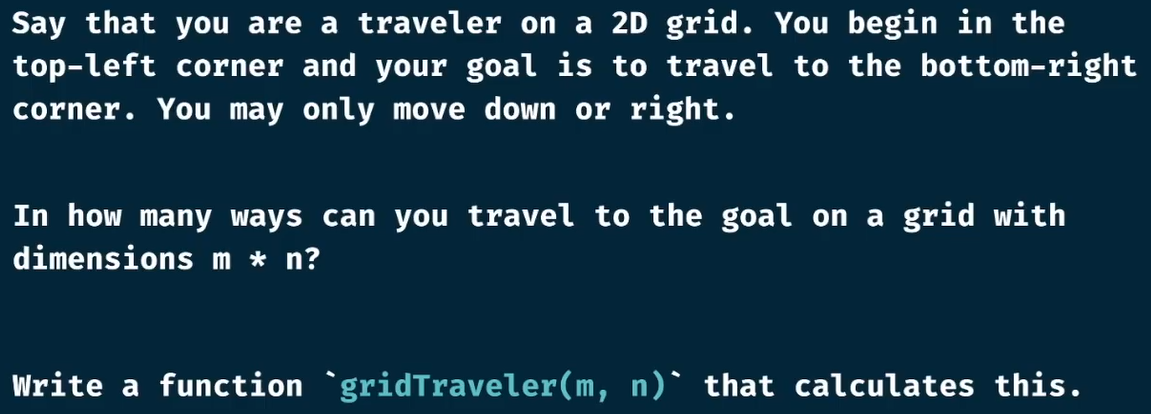
Level 4: 1 \* 2 \* 2 \* 2 \* 2 = 16

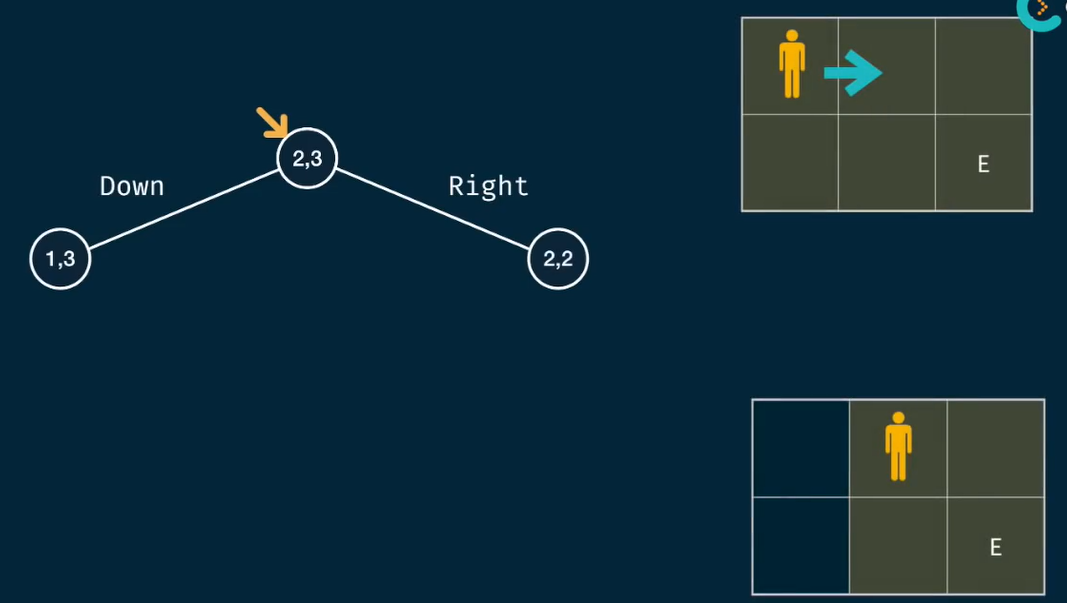
Total: 31 nodes | 2^5 = 32

Solve fib with memoization

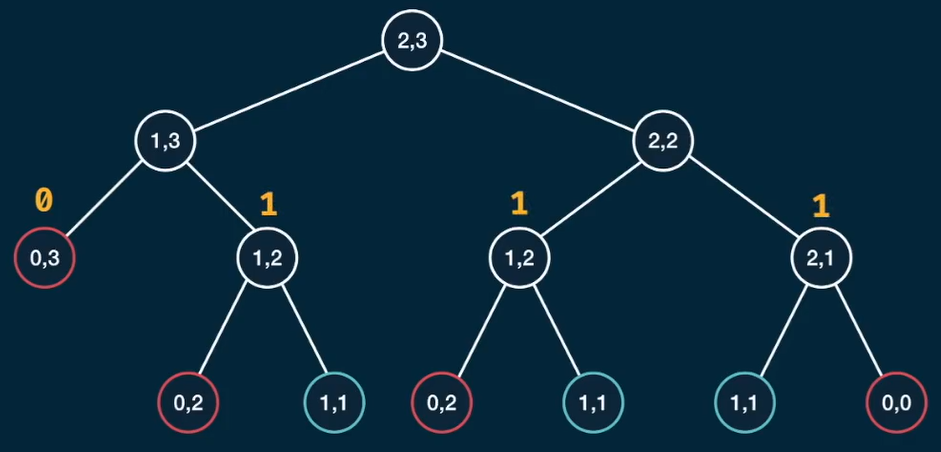


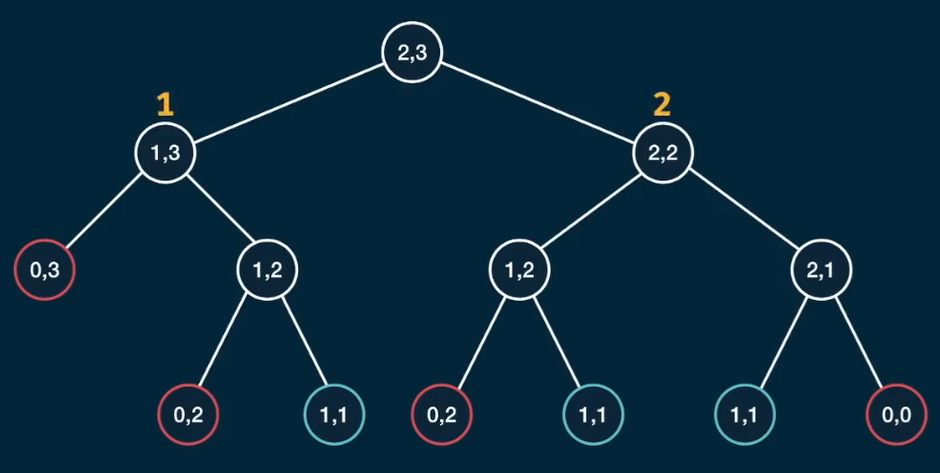
**Grid traveler**

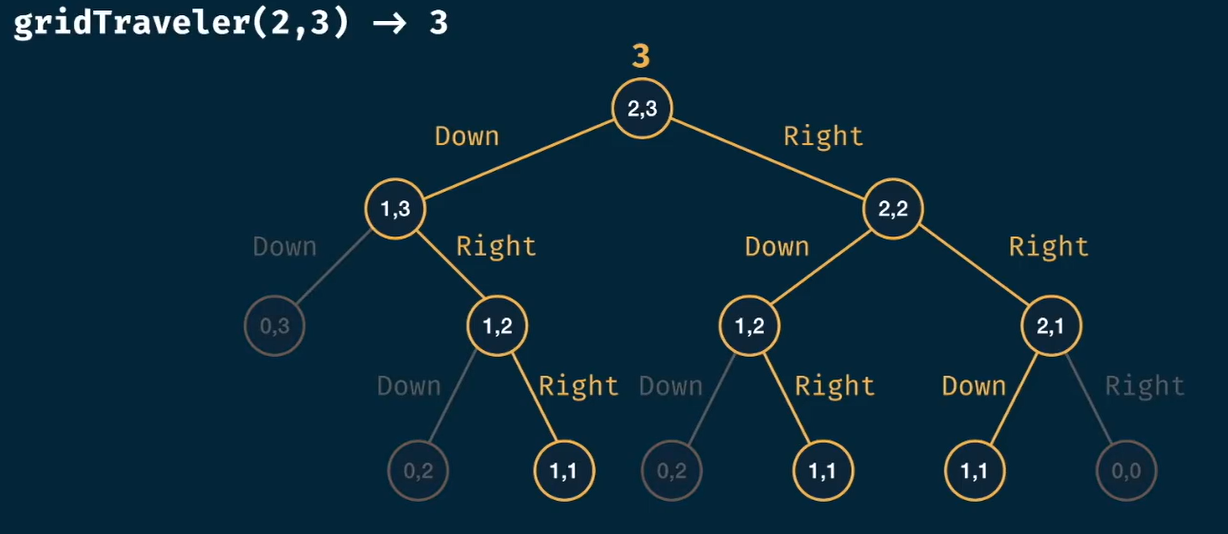




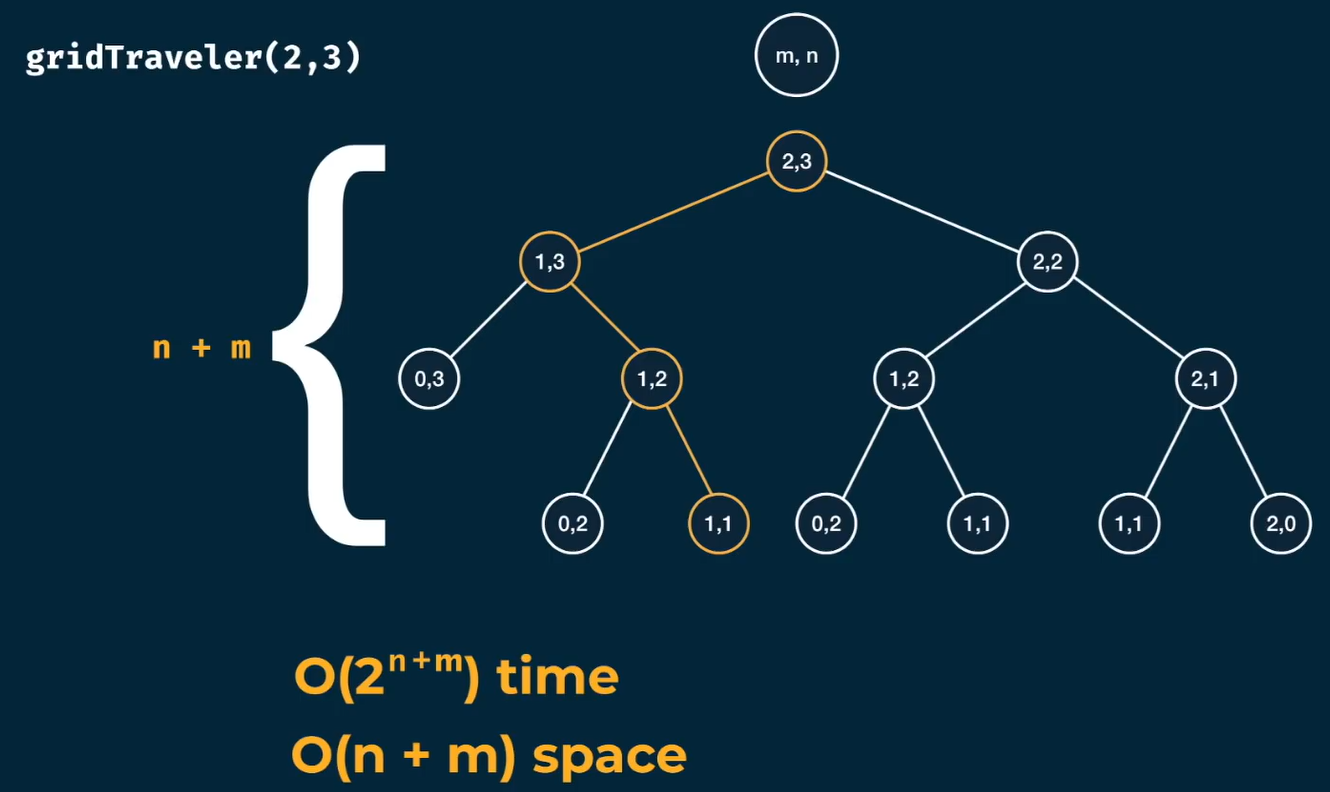
This problem can be solved with the previous approach of reducing the grid área depending on which direction we are taking, and then return to top and sum with the other side

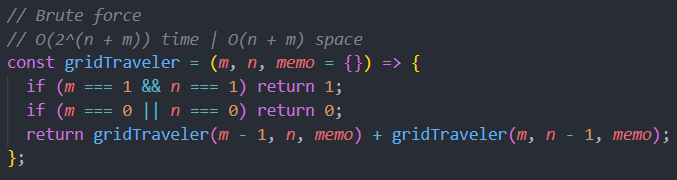






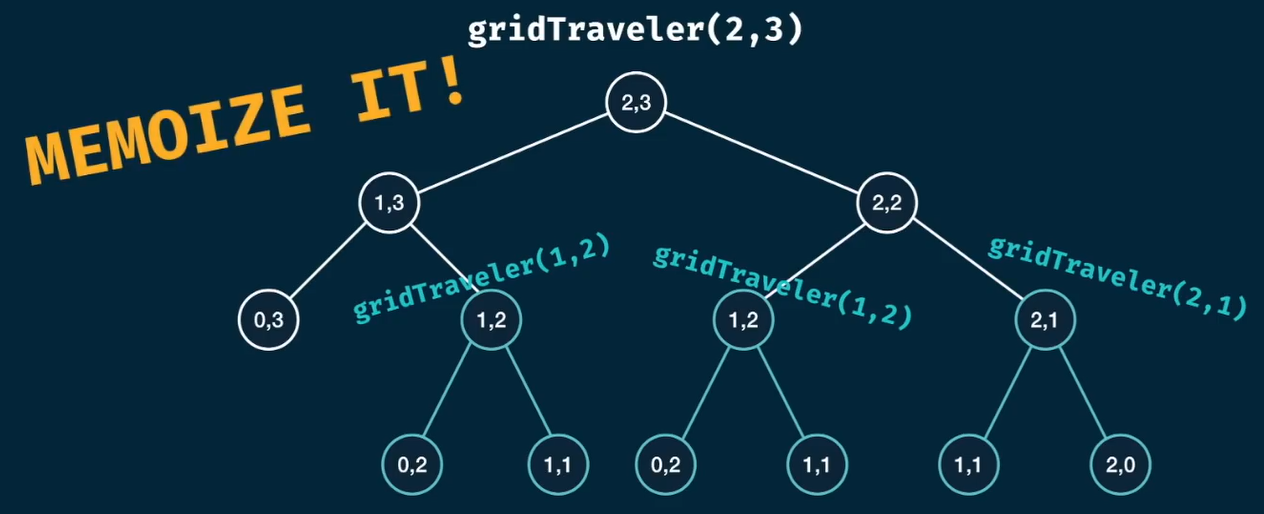
Brute force approach



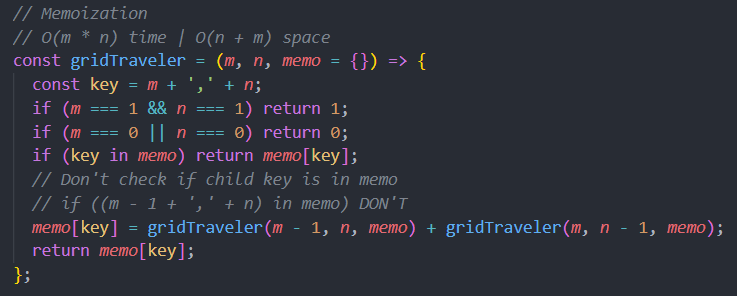


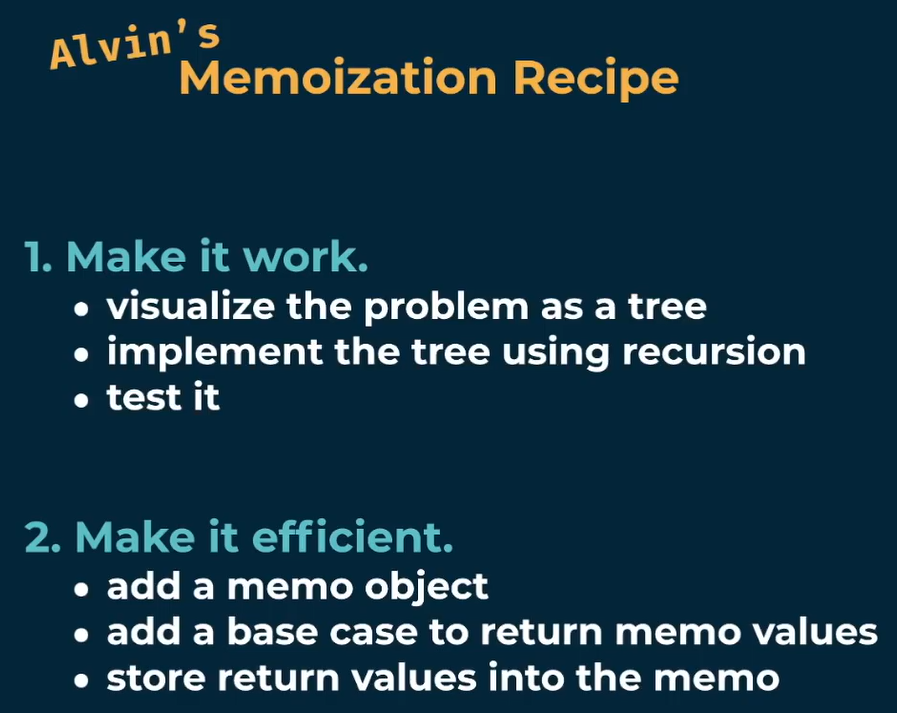


Memoization



Notice that gridTraveler(1, 2) is the same as gridTraveler(2, 1)





**canSum**

