Find the number of subset arrays count to form targetSum.

nums[i] —>
Alll are positive integers

https://www.codingninjas.com/

(Number Of Subsets)

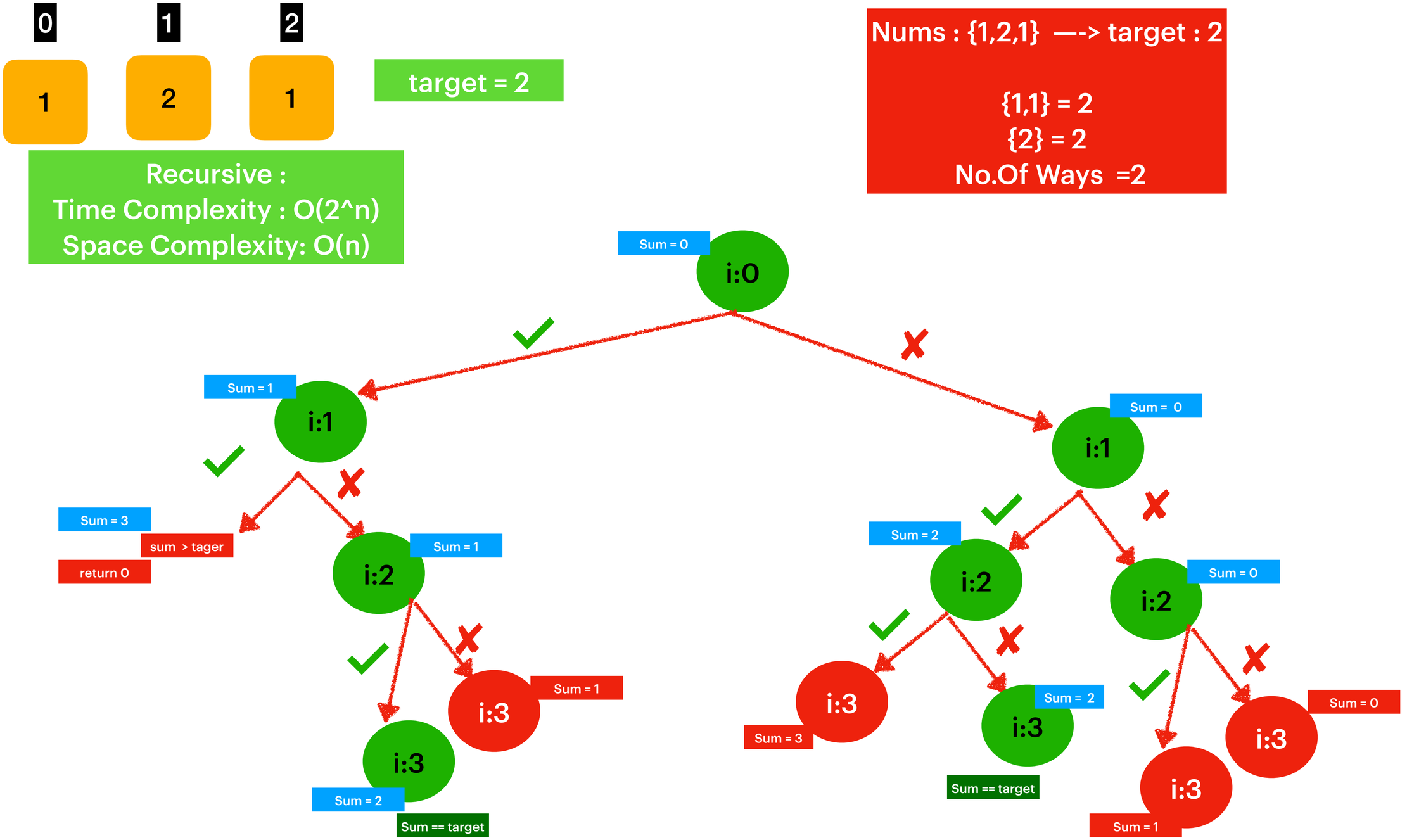
int[] {1,3,7,6,2,9} : targetSum = 9
{1,6,2} = 9
{3,6} = 9
{7,2} = 9
{9} = 9

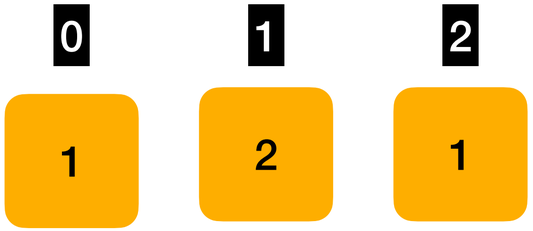
Return : No.Of Ways 4

int[] {1,1} : targetSum = 1
{1} = 1 —> From index :0
{1} = 1 —> From index:1
Return : No.Of Ways 2

int[] {1} : targetSum = 1
{1} = 1
Return : No.Of Ways 1

int[] {1,3,1} : targetSum = 2
{1,1} = 2
Return : No.Of Ways 1


0
1
2
1
2
1
target = 2
Nums : {1,2,1} —-> target : 2
{1,1} = 2
{2} = 2
No.Of Ways =2
Recursive :
Time Complexity : O(2^n)
Space Complexity: O(n)
Sum = 0
i:0
Sum = 1
i:1
Sum = 3
sum > tager
return 0
i:2
Sum = 1
i:3
Sum = 1
i:3
Sum = 2
Sum == target
Sum = 0
i:1
Sum = 2
i:2
i:3
Sum = 3
Sum = 0
i:2
i:3
Sum = 2
Sum == target
i:3
Sum = 0
i:3
Sum = 1



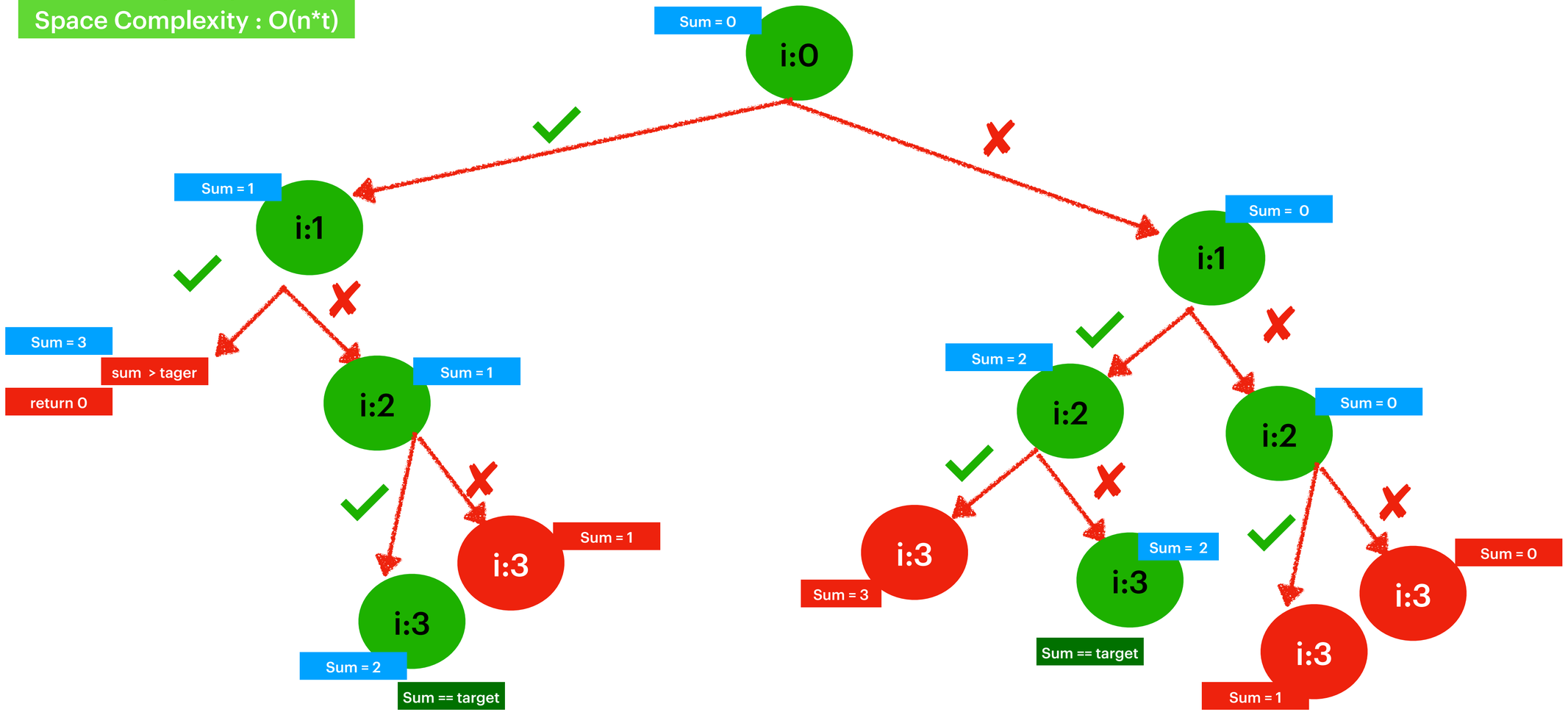
target = 2

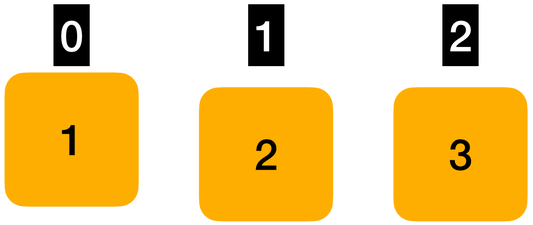
```
int[][] dp =
new   int[nums.length] [target+1];
```

Nums : {1,2,1} —-> target : 2

{1,1} = 2
{2} = 2
No.Of Ways =2

Time Complexity : O(n*t)
Space Complexity : O(n*t)





int[][] dp = new [nums.length] [target+1]

target = 3

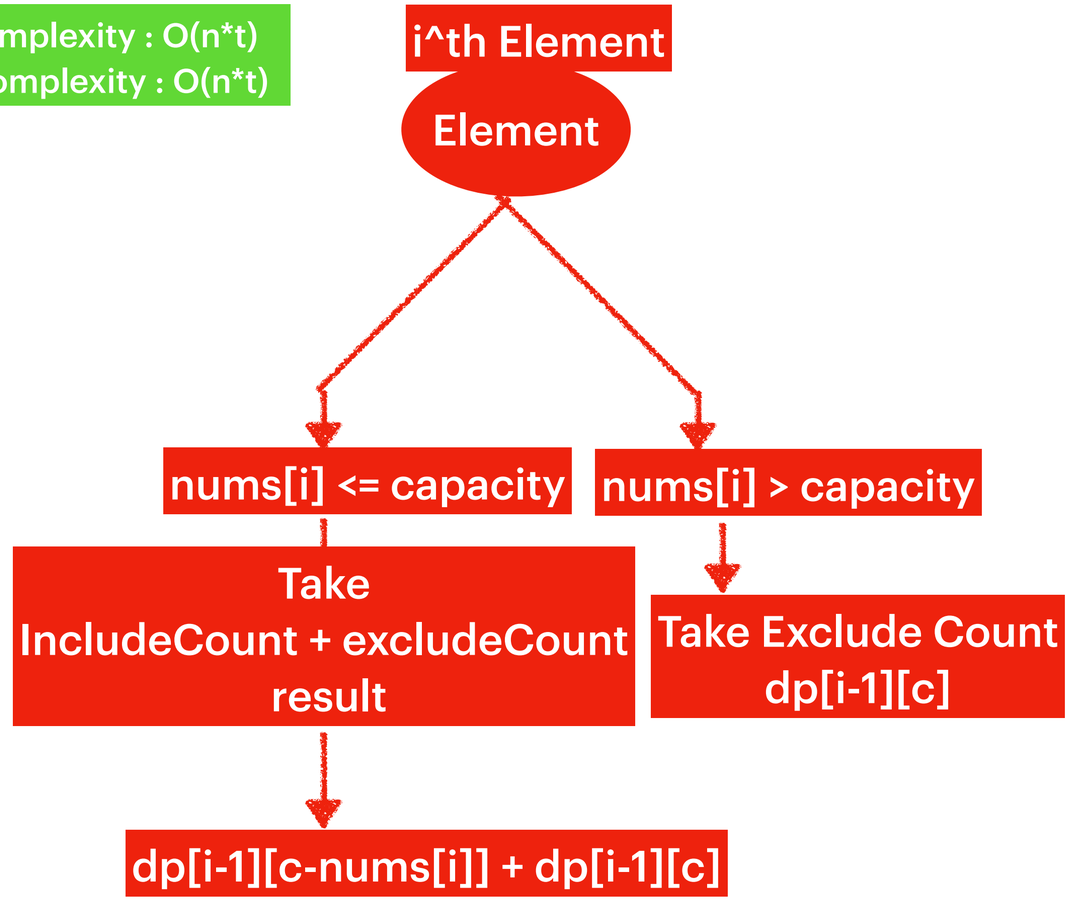
An Empty SubSet represents 0 so for 0 always counter will be 1.

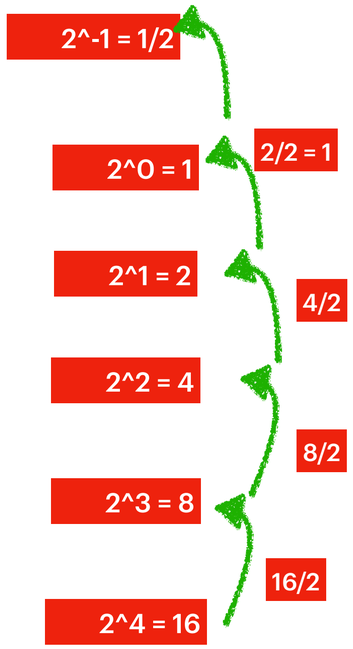
Possible Subsets.
{1,2} {3}
Output: 2

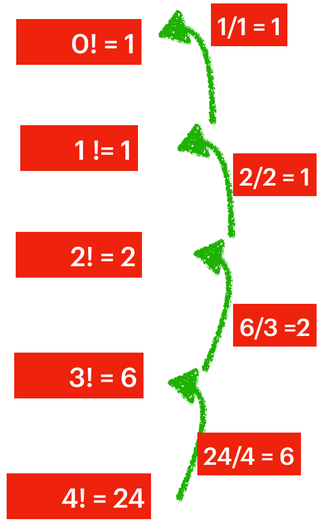
Time Complexity : O(n*t)
Space Complexity : O(n*t)

Column Target

	t:0	t:1	t:2	t:3
1 i:0 -> {1}	1	1	0	0
1 2 i:1 -> {1,2}	1	1	1	1
1 2 3 i:2 -> {1,2,3}	1	1	1	2




2^-1 = 1/2
2/2 = 1
2^0 = 1
2^1 = 2
4/2
2^2 = 4
8/2
2^3 = 8
16/2
2^4 = 16


0! = 1
1/1 = 1
1 != 1
2/2 = 1
2! = 2
6/3 =2
3! = 6
24/4 = 6
4! = 24

0 {} 1

1 {1} 1

2