Dynamic Programming Coin Change – Number of ways to Make Sum Coins[1, 2, 3]. Sum = 11.												
Coins/sum	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
1	**1**	1	1	1	1	1	1	1	1	1	1	1
2	**1**											
	1											

Coins/sum	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
1	**1**	1	1	1	1	1	1	1	1	1	1	1
2	**1**	1	2	2	3	3	4	4	5	5	6	6
	1											

Coins/sum	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
1	**1**	1	1	1	1	1	1	1	1	1	1	1
2	**1**	1	2	2	3	3	4	4	5	5	6	6
3	**1**	1	2	3	4	5	7	8	10	12	14	16

*The **distinct ways** coins can be dispersed for a value sum can be computed using the recursive formula.*

if *sum == 0:*

$1 \quad (\text{only one way})$

else if *sum > 0:*

$$diffWays(i\ ,\ sum\ ) = diffWays(i,\ sum\ - coin[i]) + diffWays(i - 1,\ sum)$$

$$\text{where, } 0 \le i \le m - 1 \text{ and } coins[i] \le sum.$$

Dynamic Programming

Climbing Stairs – Number of ways to climb

steps[1, 2, 3]. Sum = 11.

Coins/sum	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
1	**1**	1	1	1	1	1	1	1	1	1	1	1
2	**1**											
	1											

Coins/sum	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
1	**1**	1	1	1	1	1	1	1	1	1	1	1
2	**1**	1	2	3	4	5	6	7	8	9	10	11
	1											

Coins/sum	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
1	**1**	1	1	1	1	1	1	1	1	1	1	1
2	**1**	1	2	3	5	8	13	21	34	55	89	144
3	**1**	1	2	4	7	13	24	44	81	149	274	504

*The **distinct ways** to climb n stairs is given by*

diffWays(n) = diffWays(n-1) + diffWays(n-2) + diffWays(n-3) for n > 2 and

diffWays(0) = 1, diffWays(1) = 1 and diffWays(2) = 2.

Since recurrence has "no i" (or "row"), all we need is one 1D array.

The difference between coin and stairs is that 3 = 1 + 2 and 3 = 2 + 1 two different ways in the stairs case. However the are treated as one way in the coins situation.