

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 (only one way)

else if $sum > 0$:

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

where, $0 \leq i \leq m - 1$ and $coins[i] \leq sum$.

Dynamic Programming Coin Change – Number of ways to Make Sum Coins[1, 4, 5]. Sum = 12.													
Coins/sum	0	1	2	3	4	5	6	7	8	9	10	11	12
	1												
	1												
	1												

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

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

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