

Dynamic Programming

forward dp

we know $dp_i \rightarrow$ calculate dp_{i+x}

backward dp

we want to calculate $dp_i \rightarrow$ calculate it from dp_{i-x}

avoiding double counting:

Consider a money system consisting of n coins. Each coin has a positive integer value. Your task is to calculate the number of distinct *ordered* ways you can produce a money sum x using the available coins.

For example, if the coins are $\{2, 3, 5\}$ and the desired sum is 9, there are 3 ways:

- $2 + 2 + 5$
- $3 + 3 + 3$
- $2 + 2 + 2 + 3$

$dp_{i,j} \rightarrow$ number of ways to form sum i using first j coins.

$\{c_1, c_2, c_3, \dots\}$

```
for (i: 1 → x)
  for (j: 1 → n)
     $dp_{i,j} = dp_{i-c_j,j} + dp_{i,j-1}$ 
```

$dp_{i,j} \rightarrow dp_{i+c_j,j}$
 $dp_{i,j} \rightarrow dp_{i,j+1}$

$dp_{i,j} \leftarrow dp_{i-c_j,j}$
 $dp_{i,j} \leftarrow dp_{i,j-1}$