The Coin Change Problem

HackerRank

Problem Statement

How many different ways can you make change for an amount, given a list of coins? In this problem, *your code* will need to efficiently compute the answer.

Problem Statement

Write a program that, given two arguments to STDIN

- a list of coins `c1, c2, c3, ..`
- and an amount `N`

Prints out how many different ways you can make change from the coins to STDOUT.

The problem can be formally stated:

Given a value `N`, if we want to make change for `N` cents, and we have infinite supply of each of `C = { C1, C2, .. , Cm}` valued coins, how many ways can we make the change? The order of coins doesn't matter.

Example 1:

For `N = 4` and `C = {1,2,3}` there are four solutions: `{1,1,1,1},{1,1,2},{2,2},{1,3}`

So given the input

```
1, 2, 3
4
```

your program should output:

```
4
```

Example 2:

For `N = 10` and `C = {2, 5, 3, 6}` there are five solutions: `{2,2,2,2,2}, {2,2,3,3}, {2,2,6}, {2,3,5} and {5,5}`

So given the input

```
2, 5, 3, 6
10
```

your program should output:

```
5
```

Constraints

$1 <= C_i <= 50$
$1 <= N <= 250$

Solving the overlapping subproblems using dynamic programming

You can solve this problem recursively, but not all the tests will passs unless you optimise your solution to eliminate the overlapping subproblems using a dynamic programming solution

Or more specifically;

- If you can think of a way to store the checked solutions, then this store can be used to avoid checking the same solution again and again.

Hints

- Think about the degenerate cases:
  - How many ways can you give change for 0 cents?
  - How many ways can you give change for >0 cents, if you have no coins?
- If you are having trouble defining your solutions store, then think about it in terms of the base case `(n = 0)`
- For help on reading from STDIN, see the HackerRank environment help page under the "Sample Problem Statement" section.

Sample Input

```
1, 2, 3
4
```

Sample Output

4