Hard Codewriting 4000

You have a collection of coins, and you know the values of the `coins` and the `quantity` of each type of coin in it. You want to know how many distinct sums you can make from non-empty groupings of these coins.

Example

For `coins = [10, 50, 100]` and `quantity = [1, 2, 1]`, the output should be `possibleSums(coins, quantity) = 9`.

Here are all the possible sums:

- `50 = 50`;
- `10 + 50 = 60`;
- `50 + 100 = 150`;
- `10 + 50 + 100 = 160`;
- `50 + 50 = 100`;
- `10 + 50 + 50 = 110`;
- `50 + 50 + 100 = 200`;
- `10 + 50 + 50 + 100 = 210`;
- `10 = 10`;
- `100 = 100`;
- `10 + 100 = 110`.

As you can see, there are `9` distinct sums that can be created from non-empty groupings of your coins.

Input/Output

- **[execution time limit] 20 seconds (scala)**
- **[input] array.integer coins**

  An array containing the values of the coins in your collection.

  Guaranteed constraints:
  `1 ≤ coins.length ≤ 20`,
  `1 ≤ coins[i] ≤ $10^4$`.

- **[input] array.integer quantity**

  An array containing the quantity of each type of coin in your collection. `quantity[i]` indicates the number of coins that have a value of `coins[i]`.

  Guaranteed constraints:
  `quantity.length = coins.length`,
  `1 ≤ quantity[i] ≤ $10^5$`,
  `(quantity[0] + 1) * (quantity[1] + 1) * ... * (quantity[quantity.length - 1] + 1) <= $10^6$`.

- **[output] integer**

  The number of different possible sums that can be created from non-empty groupings of your coins.

[Scala] Syntax Tips

```
def helloWorld(name: String): String = {
    println("This prints to the console when you Run Tests")
    "Hello, " + name
}
```