**Dynamic Programming**

In order to make a solution using [Dynamic Programming](http://en.wikipedia.org/wiki/Dynamic_programming) we need to be able to divide the problem into independent sub-problems. There are dimension which this can be done in for this problem, both the number of different coins but also the value we need to split. The problem is also covered[here](http://www.cs.ucf.edu/~reinhard/classes/cop3503/lectures/DynProg03.pdf) but I will try to give an alternative description of the solution.

The question we want to answer repeatedly is. “If I have to give change to *n* pennies, if I give back one coin of *x* how many ways can I give back the rest *n-x* pennies using only coins of value *x* or smaller.” Posing that question means that we can make a smaller problem, which we need to solve. And in the end we get a very small and easy problem to solve “if I want to give change to 1p using 1p, how many ways can this be done?”

Instead of breaking down the problem with the question posed above, we will instead build up an array of solutions so we can look up the answer to the question once we are ready to ask it. And let me show you by hand with a smaller example. Lets figure out how many ways we can give change to 5p using 1p and 2p. This should be doable by hand.

What we will do is build up an array of solutions, at first the array is build up for giving change to 1-5p using only 1p coins.

We need to initialise the array with the question. If we want to give change to 0p using a 1p coin, in how many different ways can we do that. And to that we define the answer to be 1. Now we are ready to answer “If I have to change to 1p using 1p coins, how many ways can I do that?”, so lets give back 1p and look up the value for n-x = 0p, which is 1 add this to the already found 0 solutions for a total of 1 solution.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 |
| 1 | 1 | 0 | 0 | 0 | 0 |

Now we are ready to answer the question, if I want to give change to 2p using 1p coins, and give back 1p, in how many ways can I give back the remaining 1p?. So subtracting 1 from the solution, we can look up the last part of the question, since that was already answered. This goes on for all 5p until we have a solution array that looks like

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 |
| 1 | 1 | 1 | 1 | 1 | 1 |

If you think about it, it is rather obvious that there is only one way to give change to any amount using only 1p coins.

So now we are ready to treat the case when we have 2p coins as well. Since we can’t give back a 2p coin when we need to change 1p, there is no need to pose that question. Lets instead see how many ways we can give back 2p. If we use the 2p coin, we need to further give change to 0p. So that is a solution. Lets add that to the already found number of solutions, so the solution array looks like

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 |
| 1 | 1 | 2 | 1 | 1 | 1 |

For 3p we can also give a 2p coin and then we need to give change to 1p, which in the table and gives us one solution. So the total number of solutions for giving change on 3p, is 1+1.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 |
| 1 | 1 | 2 | 2 | 1 | 1 |

Giving change on 4p. To answer that give 2p change and answer the question how many ways can we give change to n-x = 4-2 = 2p using 1p and 2p coins. The answer to that is in the table and is 2. So there are two ways to give back 2p, plus the already found number of solutions. So the result table now looks like

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 |
| 1 | 1 | 2 | 2 | 3 | 1 |

Lets just make a sanity check on that. 4p we can make change in 4x1p, 2p + 2x1p or 2x2p. Yep, that seems correct. We can do the same with the 5 giving change to 5p so the solution table looks like

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 |
| 1 | 1 | 2 | 2 | 3 | 3 |

You get the idea of how to build the solution now? Lets make the algorithm for it. It is completely analogue to what I just showed you by hand just with more coins and a larger value we need to give change to.