

Dynamic Programming: Change Problem

Pavel Pevzner

Department of Computer Science and Engineering
University of California, San Diego

Data Structures and Algorithms
Algorithmic Toolbox

Outline

- ① Greedy Change
- ② Recursive Change
- ③ Dynamic Programming

Change problem

Find the minimum number of coins needed to make change.



Formally

Change problem

Input: An integer $money$ and positive integers $coin_1, \dots, coin_d$.

Output: The minimum number of coins with denominations $coin_1, \dots, coin_d$ that changes $money$.

Greedy Way

GreedyChange(*money*)

```
Change ← empty collection of coins
while money > 0:
    coin ← largest denomination
        that does not exceed money
    add coin to Change
    money ← money – coin
return Change
```

Changing Money

in the US

$$40 \text{ cents} = 25 + 10 + 5$$

Greedy



Changing Money

in Tanzania

$$40 \text{ cents} = 25 + 10 + 5 = 20 + 20$$

Greedy is not Optimal



Outline

- ① Greedy Change
- ② Recursive Change
- ③ Dynamic Programming

Recursive Change

Given the denominations 6, 5, and 1, what is the minimum number of coins needed to change 9 cents?

$$\text{MinNumCoins}(9) = ?$$

money

1

3

4

5

6

7

8

9

10

MinNumCoins

A horizontal row consisting of ten empty square boxes of equal size, arranged side-by-side. To the right of the tenth box is a single question mark character.

Recursive Change

Given the denominations 6, 5, and 1, what is the minimum number of coins needed to change 9 cents?

$$MinNumCoins(9) = \min \begin{cases} MinNumCoins(9 - 6) + 1 \\ MinNumCoins(9 - 5) + 1 \\ MinNumCoins(9 - 1) + 1 \end{cases}$$

money

1

3

4

5

6

7

8

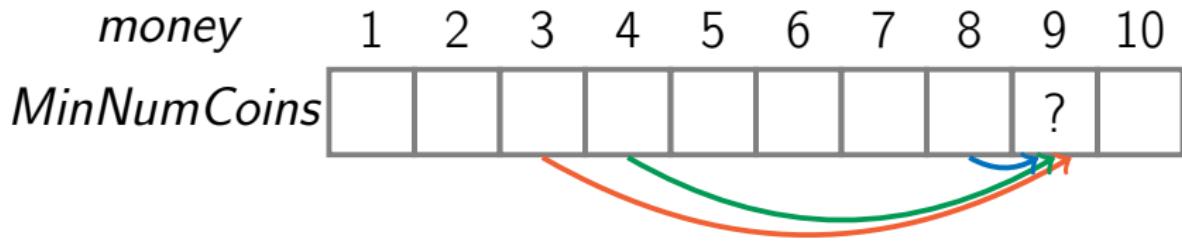
9

10

Recursive Change

Given the denominations 6, 5, and 1, what is the minimum number of coins needed to change 9 cents?

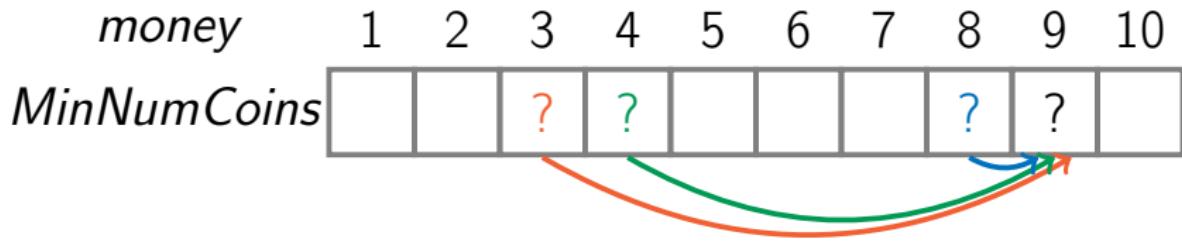
$$\text{MinNumCoins}(9) = \min \begin{cases} \text{MinNumCoins}(3) + 1 \\ \text{MinNumCoins}(4) + 1 \\ \text{MinNumCoins}(8) + 1 \end{cases}$$



Recursive Change

Given the denominations 6, 5, and 1, what is the minimum number of coins needed to change 9 cents?

$$\text{MinNumCoins}(9) = \min \begin{cases} \text{MinNumCoins}(3) + 1 \\ \text{MinNumCoins}(4) + 1 \\ \text{MinNumCoins}(8) + 1 \end{cases}$$



Recurrence for Change Problem

$$\text{MinNumCoins}(\text{money}) = \min \left\{ \begin{array}{l} \text{MinNumCoins}(\text{money} - \text{coin}_1) + 1 \\ \text{MinNumCoins}(\text{money} - \text{coin}_2) + 1 \\ \dots \\ \text{MinNumCoins}(\text{money} - \text{coin}_d) + 1 \end{array} \right.$$

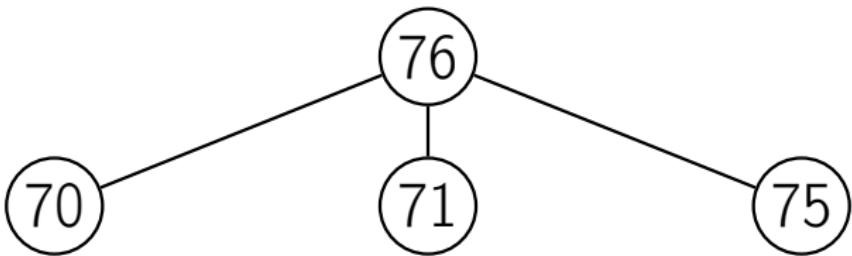
RecursiveChange(*money*, *coins*)

```
if money = 0:  
    return 0  
MinNumCoins ← ∞  
for i from 1 to |coins|:  
    if money ≥ coini:  
        NumCoins ← RecursiveChange(money − coini, coins)  
        if NumCoins + 1 < MinNumCoins:  
            MinNumCoins ← NumCoins + 1  
return MinNumCoins
```

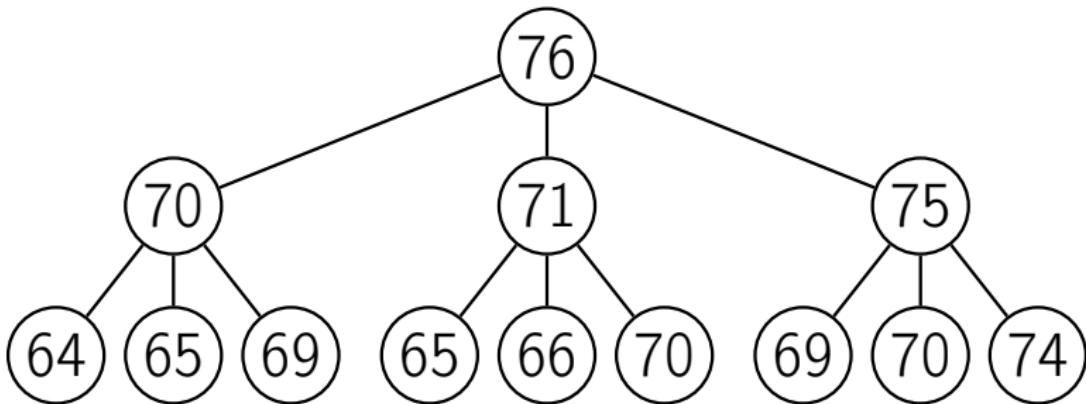
How Fast is RecursiveChange?

76

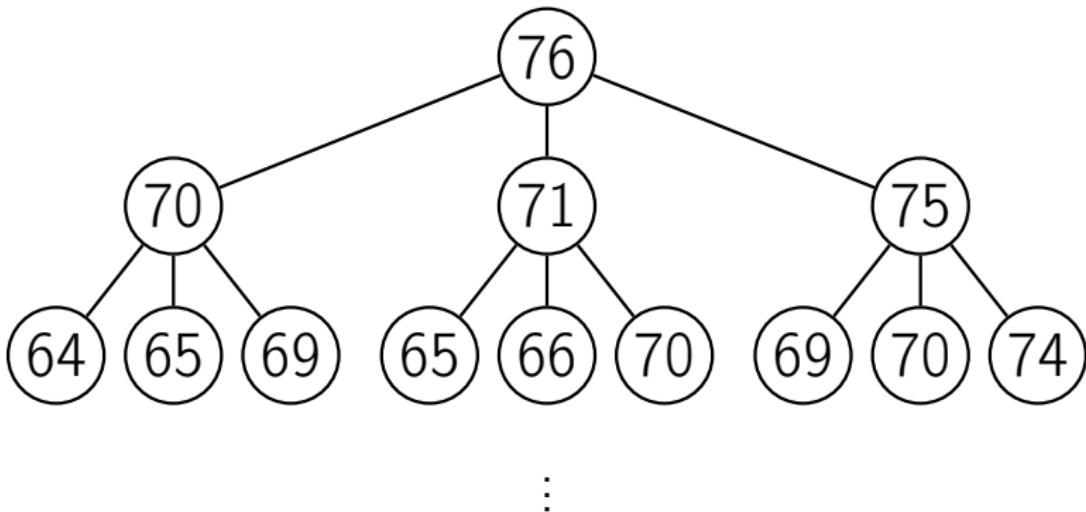
How Fast is RecursiveChange?



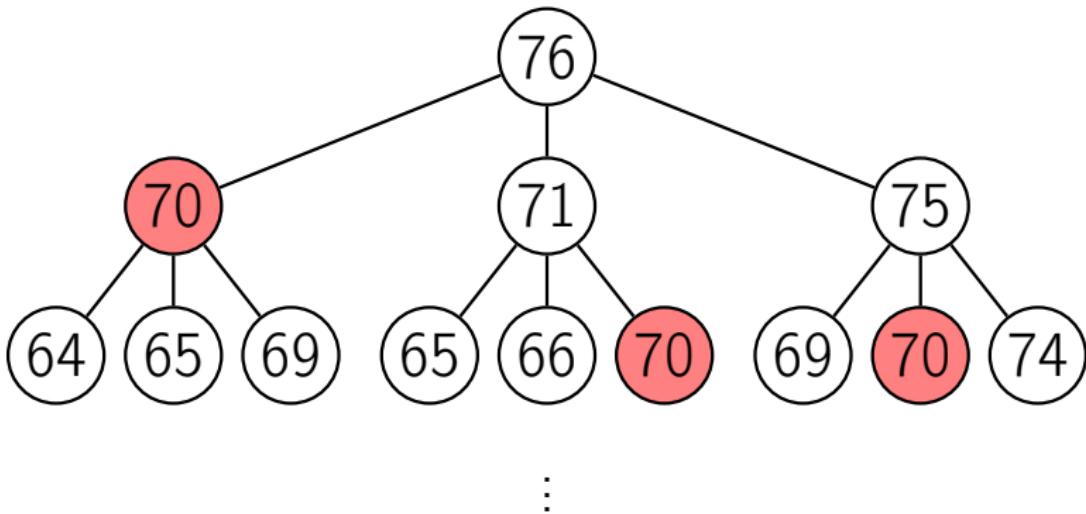
How Fast is RecursiveChange?



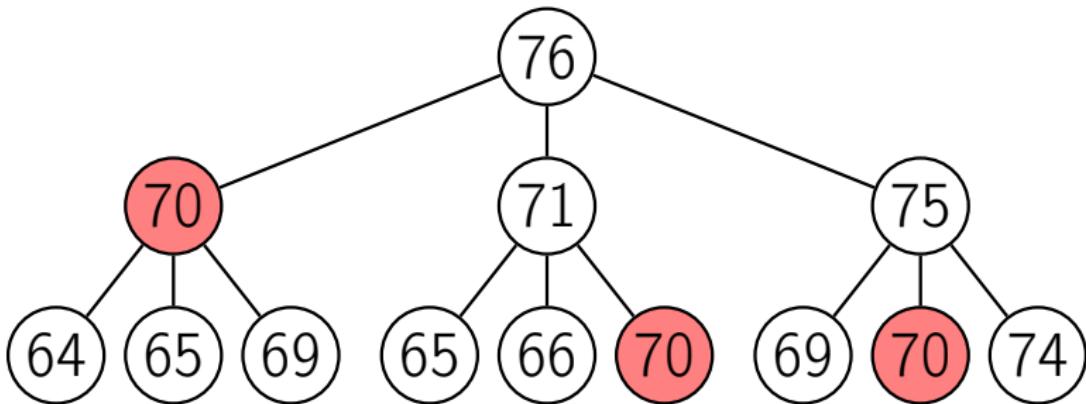
How Fast is RecursiveChange?



How Fast is RecursiveChange?

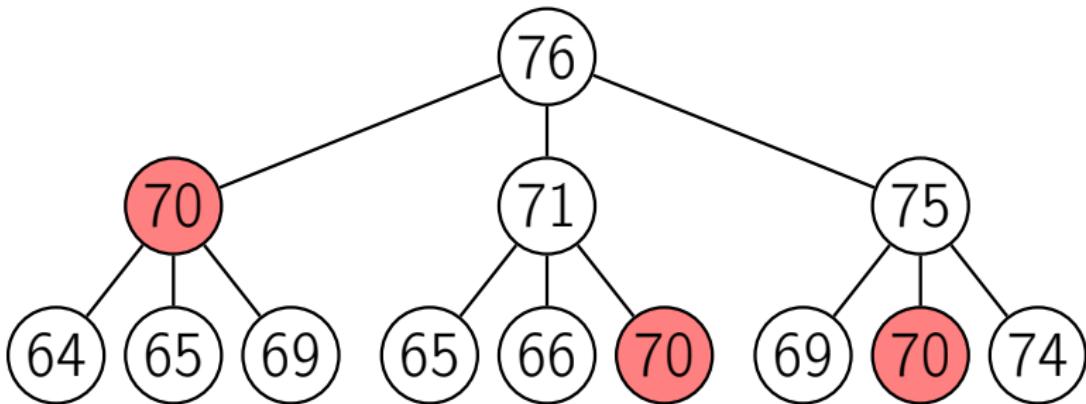


How Fast is RecursiveChange?



the optimal coin combination for 70 cents is computed at least **three** times!

How Fast is RecursiveChange?



the optimal coin combination for 70 cents is computed at least **three** times!

the optimal coin combination for 30 cents is computed **trillions** of times!

Hint

Wouldn't it be nice to know all the answers for changing *money* – *coin*; by the time we need to compute an optimal way of changing *money*?



Hint

Wouldn't it be nice to know all the answers for changing $money - coin_i$ by the time we need to compute an optimal way of changing $money$?

Instead of the time-consuming calls to

`RecursiveChange($money - coin_i$, coins)`

we would simply look up these values!



Outline

- ① Greedy Change
- ② Recursive Change
- ③ Dynamic Programming

Dynamic Programming

What is the minimum number of coins needed to change 0 cents for denominations 6, 5, and 1?

money

0

1

2

3

4

5

6

7

8

9

MinNumCoins

0

Dynamic Programming

What is the minimum number of coins needed to change 1 cent for denominations 6, 5, and 1?

Dynamic Programming

What is the minimum number of coins needed to change 2 cents for denominations 6, 5, and 1?

<i>money</i>	0	1	2	3	4	5	6	7	8	9
<i>MinNumCoins</i>	0	1	2							



Dynamic Programming

What is the minimum number of coins needed to change 3 cents for denominations 6, 5, and 1?

<i>money</i>	0	1	2	3	4	5	6	7	8	9
<i>MinNumCoins</i>	0	1	2	3						

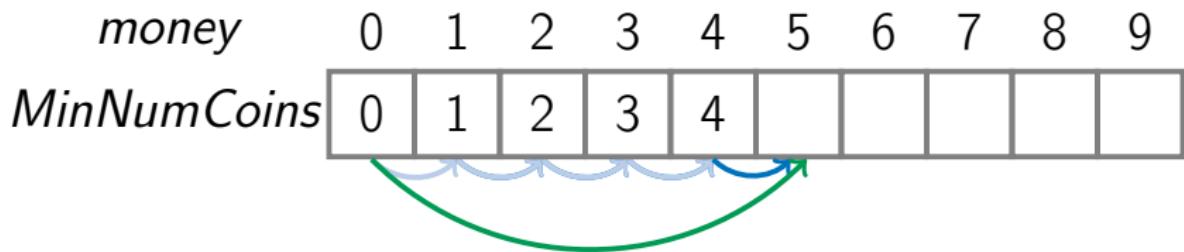
Dynamic Programming

What is the minimum number of coins needed to change 4 cents for denominations 6, 5, and 1?

<i>money</i>	0	1	2	3	4	5	6	7	8	9
<i>MinNumCoins</i>	0	1	2	3	4					

Dynamic Programming

What is the minimum number of coins needed to change 5 cents for denominations 6, 5, and 1?



$$\min \begin{cases} \text{MinNumCoins}(0) + 1 \\ \text{MinNumCoins}(4) + 1 \end{cases}$$

Dynamic Programming

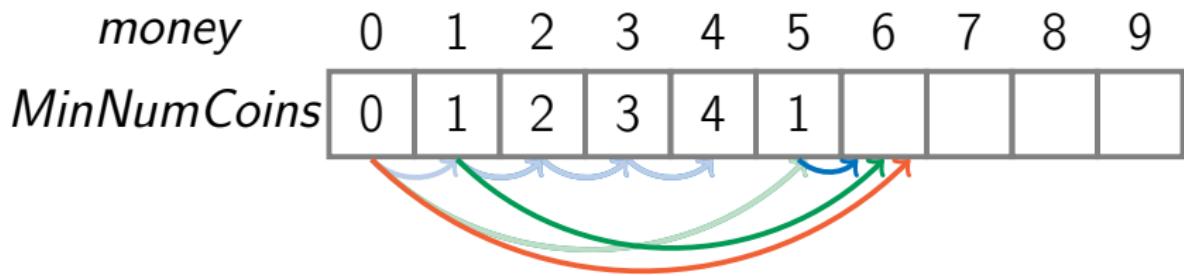
What is the minimum number of coins needed to change 5 cents for denominations 6, 5, and 1?

<i>money</i>	0	1	2	3	4	5	6	7	8	9
<i>MinNumCoins</i>	0	1	2	3	4	1				



Dynamic Programming

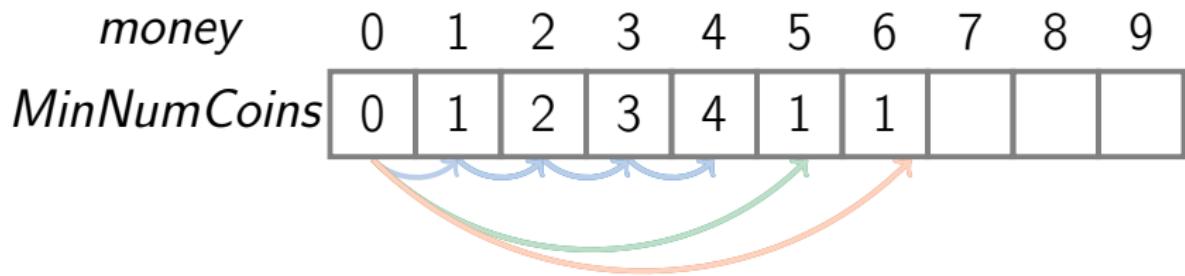
What is the minimum number of coins needed to change 6 cents for denominations 6, 5, and 1?



$$\min \begin{cases} \textcolor{red}{\textit{MinNumCoins}(0) + 1} \\ \textcolor{green}{\textit{MinNumCoins}(1) + 1} \\ \textcolor{blue}{\textit{MinNumCoins}(5) + 1} \end{cases}$$

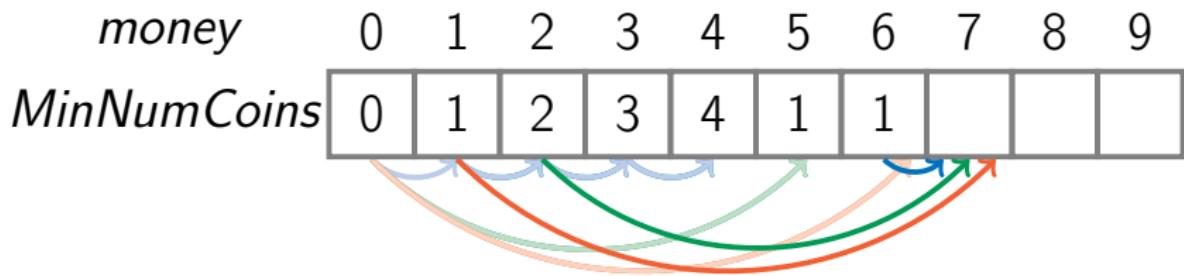
Dynamic Programming

What is the minimum number of coins needed to change 6 cents for denominations 6, 5, and 1?



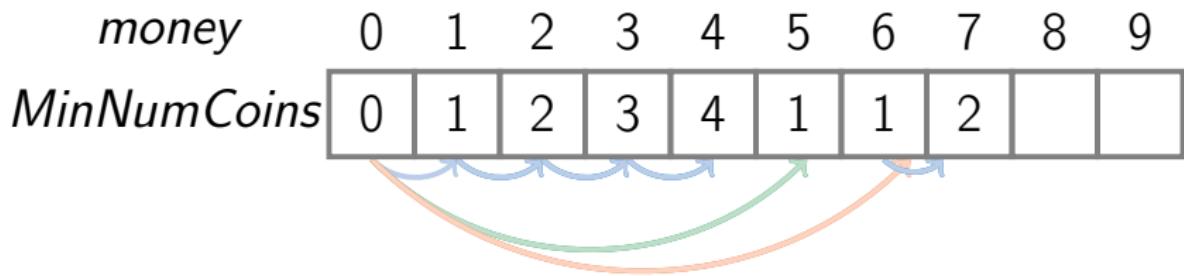
Dynamic Programming

What is the minimum number of coins needed to change 7 cents for denominations 6, 5, and 1?



Dynamic Programming

What is the minimum number of coins needed to change 7 cents for denominations 6, 5, and 1?



Dynamic Programming

What is the minimum number of coins needed to change 8 cents for denominations 6, 5, and 1?

The diagram shows the state transition graph for the knapsack problem. The horizontal axis represents the amount of money, labeled *money*, from 0 to 9. The vertical axis represents the number of coins used, labeled *MinNumCoins*, from 0 to 9. A grid of nodes connects these points. Colored arrows indicate transitions between states:

- Blue arrows:** From (0,0) to (1,0), (2,0), (3,0), (4,0), (5,0), (6,0), (7,0), (8,0).
- Green arrows:** From (0,0) to (0,1), (1,1), (2,1), (3,1), (4,1).
- Red arrows:** From (0,0) to (0,2), (1,2), (2,2), (3,2), (4,2).
- Orange arrows:** From (0,0) to (0,3), (1,3), (2,3), (3,3), (4,3).
- Yellow arrows:** From (0,0) to (0,4), (1,4), (2,4), (3,4).
- Grey arrows:** From (0,0) to (0,5), (1,5), (2,5), (3,5).
- Purple arrows:** From (0,0) to (0,6), (1,6), (2,6), (3,6).
- Cyan arrows:** From (0,0) to (0,7), (1,7), (2,7), (3,7).
- Magenta arrows:** From (0,0) to (0,8), (1,8), (2,8), (3,8).

The final solution path is highlighted with red arrows, starting from (0,0) and ending at (9,2). The grid also contains some grey nodes, likely representing invalid or already visited states.

Dynamic Programming

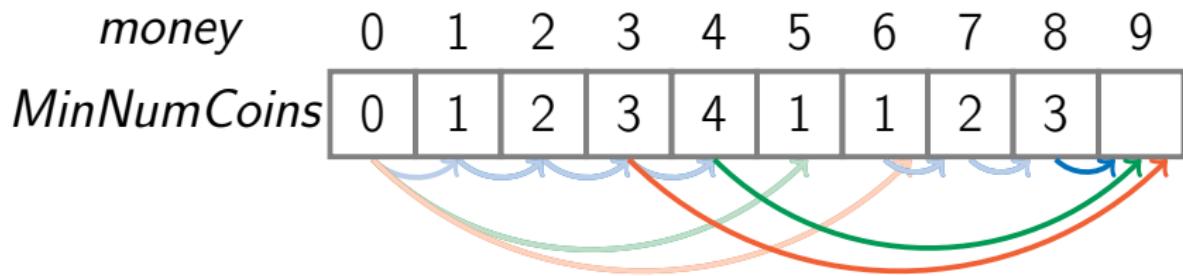
What is the minimum number of coins needed to change 8 cents for denominations 6, 5, and 1?

<i>money</i>	0	1	2	3	4	5	6	7	8	9
<i>MinNumCoins</i>	0	1	2	3	4	1	1	2	3	



Dynamic Programming

What is the minimum number of coins needed to change 9 cents for denominations 6, 5, and 1?



Dynamic Programming

What is the minimum number of coins needed to change 9 cents for denominations 6, 5, and 1?

<i>money</i>	0	1	2	3	4	5	6	7	8	9
<i>MinNumCoins</i>	0	1	2	3	4	1	1	2	3	4

The diagram illustrates the results of a dynamic programming algorithm for changing 9 cents. The array *MinNumCoins* contains the minimum number of coins required for each amount from 0 to 9. The value at index 5 (1) is highlighted with a green arrow, indicating the solution to the problem of changing 5 cents. Blue arrows point to the values at indices 6, 7, 8, and 9, which are also highlighted with green arrows, showing the path of dependencies from the bottom row to the top row.

DPChange(*money*, *coins*)

```
MinNumCoins(0) ← 0
for m from 1 to money:
    MinNumCoins(m) ← ∞
    for i from 1 to |coins|:
        if m ≥ coini:
            NumCoins ← MinNumCoins(m − coini) + 1
            if NumCoins < MinNumCoins(m):
                MinNumCoins(m) ← NumCoins
return MinNumCoins(money)
```

“Programming” in “Dynamic Programming” Has Nothing to Do with Programming!

Richard Bellman developed this idea in 1950s working on an Air Force project.

At that time, his approach seemed completely impractical.

He wanted to hide that he is really doing math from the Secretary of Defense.



Richard
Bellman

...What name could I choose? I was interested in planning but planning, is not a good word for various reasons. I decided therefore to use the word, “programming” and I wanted to get across the idea that this was dynamic. **It was something not even a Congressman could object to.** So I used it as an umbrella for my activities.