CMPS 201 5-26-10

Coin Changing Problem:

Given an unlimited number of coins in denominations $\{d_1, \dots, d_n\}$

(1) what is the least # of coins necessary to pay N monetary units?

(2) which coins are to be disbursed in order to acheive the optimum # coins in (1).

To answer (1), create a 2-dim

$$C[1 \dots n; 0 \dots N]$$

where

$C[i, j]$ = minimum # coins necessary to Pay j units using only coins in $\{d_1, \dots, d_i\}$

Thus we seek $C[n, N]$.

- Observe $C[i, 0] = 0$ for $1 \le i \le n$.
- Also notice, to Pay j units using only coins $\{d_1, \dots, d_i\}$ we have two choices:

(I). use no coins in denom. d_i (even though you can). we can do this using min #

$$C[i-1, j]$$ coins.

or

II. use AT least 1 coin of denom. d_i. After handing over one such coin, have to pay $j-d_i$ units using coins $\{d_1, \ldots, d_i\}$. min # of coins necessary for this is

$$1 + C[i, j-d_i]$$

$C[i,j]$ is whichever alternative yields the fewest coins, i.e.

So

$$C[i,j] = \min\left(C[i-1,j] \,,\, 1 + C[i, j-d_i] \right)$$

Note that if $i=1$ or $j<d_i$ then one of these values falls outside the table $C[\cdot,\cdot]$. Think of values outside Table as being $+\infty$. If Both $i=1$ and $j<d_i$. Then we set $C[i,j] = +\infty$, indicating that its impossible to pay j units using only $\{d_1, \ldots, d_i\}$.

Ex. $N=8$, $n=4$, $\{1,3,5,6\}$

	0	1	2	3	4	5	6	7	8
$d_1 = 1$	0	1	2	3	4	5	6	7	8
$d_2 = 3$	0	1	2	1	2	3	2	3	4
$d_3 = 5$	0	1	2	1	2	1	2	3	2
$d_4 = 6$	0	1	2	1	2	1	1	2	2

$$C[1,1] = \min(C[0,1],\ 1 + C[1,0])$$
$$= \min(\infty,\ 1+0) = 1$$
$$C[1,2] = \min(C[0,2],\ 1 + C[1,1])$$
$$= \min(\infty,\ 1+1) = 2$$

Ex. $N = 8$, $n = 3$, $\{2, 4, 5\}$

	0	1	2	3	4	5	6	7	8
$d_1 = 2$	0	∞	1	∞	2	∞	3	∞	4
$d_2 = 4$	0	∞	1	∞	1	∞	2	∞	2
$d_3 = 5$	0	∞	1	∞	1	1	2	2	2

Input: $d[1 \dots n]$, N

use local array $C[1 \dots n; 0 \dots N]$

CoinChange(d, N)

```
n ← length[d]
for i ← 1 to n
    C[i, 0] ← 0
for i ← 1 to n
    for j ← 1 to N
        if i=1 and j<d[1]
            C[1, j] ← ∞
        else if i=1
            C[1, j] ← 1 + C[1, j-d[1]]
        else if j<d[i]
            C[i, j] ← C[i-1, j]
        else
            C[i, j] ← min(C[i-1, j], 1+C[i, j-d[i]])
return C[n, N]
```

Runtime: $\Theta(n \cdot N)$

since table is of size $n \times (N+1)$

<u>Exercise</u>: modify this algorithm to deal with situation in which some denominations have only a limited # of coins.

<u>Inputs</u>: N = amount to be paid

$d[1 \ldots n]$ denom. set

$\ell[1 \ldots n]$ limits, i.e. only have $\ell[i]$ coins of denom. $d[i]$

once the table $C[1..n; 0...N]$ has been filled, can solve Problem (2) (i.e. which coins exactly to hand over.)

$$C[i,j] = \min\Big(\underset{\text{here?}}{C[i-1,j]},\ \underset{\text{here?}}{1 + C[i, j-d_i]}\Big)$$

where does min occur? or both?

If $C[i,j] = C[i-1,j]$, then no coins of type i are necessary so move up to $C[i-1,j]$ to see what to do.

If $C[i,j] = 1 + C[i, j-d_i]$, then Pay 1 coin of type i (value d_i) then move to left to $C[i, j-d_i]$ to see what to do next.

If $C[i,j] =$ both $C[i-1, j]$ and $1 + C[i, j-d_i]$, then do either.

<u>Exercise</u>

write a recursive algorithm that, Given the filled table

$$C[1 \dots n ; 0 \dots N]$$

Prints a sequence of $C[n,N]$ coin types whose values add to N. If $C[n,N] = \infty$, print appropriate message.