CSC311: The Design and Analysis of Algorithms

Computer Science Department	Second Semester 1443
King Saud University	Assignment 3

Question 1. Apply the bottom-up dynamic programming algorithm to the following instance of the 0/1-knapsack problem, with capacity=16 and four items with weights: 3, 5, 7, 8 and values: 4, 6, 7, 9.

item	weight	value
1	3	4
2	5	6
3	7	7
4	8	9

$$F(i,j) = max\{F(i-1,j), v_i + F(i-1,j-w_i)\} \quad \textbf{if } j - w_i \geq 0$$
$$= F(i-1,j) \qquad \textbf{if } j - w_i \leq 0$$
$$F(0,j) = 0 \ \textbf{ and } \ F(i,0) = 0$$

Question 2. Design a dynamic programming algorithm for the **change-making problem**: given an amount n and unlimited quantities of coins of each of the denominations $d_1, d_2, ..., d_m$, find the smallest number of coins that add up to n or indicate that the problem does not have a solution.

Question 3. Given a number N, you've to find the number of different ways to write it as the sum of 1, 3 and 4.
For example, if $N = 5$, the answer would be 6.

1) . 1 + 1 + 1 + 1 + 1
2) . 1 + 4
3) . 4 + 1
4) . 1 + 1 + 3
5) . 1 + 3 + 1
6) . 3 + 1 + 1

Find dynamic-programming solution for this problem.

Question 4. Apply Warshalls algorithm to find the transitive closure of the digraph defined by the following adjacency matrix:

$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Q1:

	K \ w_K	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$w_K=$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$w_K=3$, $v_K=4$	1	0	0	0	4	4	4	4	4	4	4	4	4	4	4	4	4	4
$w_K=5$, $v_K=6$	2	0	0	0	4	4	6	6	6	10	10	10	10	10	10	10	10	10
$w_K=7$, $v_K=7$	3	0	0	0	4	4	6	6	7	10	10	11	11	13	13	13	17	17
$w_K=8$, $v_K=9$	4	0	0	0	4	4	6	6	7	10	10	11	13	13	15	15	17	19

Solution = $\{J_4, J_2, J_1\}$

max value = 19

$v[K-1, w]$ if $v_K > w$

if $w_K \leq w$

$v[K,w] = \max(v[K-1,w], v[K-1, w-w_K]+v_K$

Q2:

$d_1 = 3$, $d_2 = 5$, $d_3 = 7$, $n = 11$

$i \backslash j$	0	1	2	3	4	5	6	7	8	9	10	11
$d_1 = 3$	0	∞	∞	1	∞	∞	2	∞	∞	3	∞	∞
$d_2 = 5$	0	∞	∞	1	∞	1	2	∞	+2	3	2	3
$d = 7$	0	∞	∞	1	∞	1	2	1	2	3	2	3

```
Algorthim CoinChainging (d1 ..., dn, T)
// d1 = 1

for t <- 0 to T do
    N [1, t] <- t
    A [1, t] <- t
for i <- 2 to n
    for t <- 0 to T
        N [i, t] <- N [i-1, t]
        A [i, t] <- 0
        for j <- 1 to ⌊(t / di⌋
            if  j + N [i-1, t - jdi] < N [i, t]
                    N [i, t] <- j + N [i-1, t - jdi]
                    A [i, t] <- j
return (N [n, T])
```

The A [i, t] values makes it easy to determine number of coins of each denomination in the optimal solution N [i, T]

Q 3 :

	No of ways	
F[1]	1	{ 1 }
F[2]	1	{ 1+1 }
F[3]	2	{ 1+1+1 , 3 }
F[4]	4	{ 4, 3+1, 1+3, 1+1+1+1 }
F[5]	6	{ 4+1, 3+1+1, 1+4, 1+3+1, 1+1+3, 1+1+1+1+1 }

the ways of to get to 5   1+4 $\overset{1\ \ 1}{2}+3$ $\overset{1+1+1+1}{4}+1$

F [5−4] + F [5−3] + F [5−1]

F [1] + F [2] + F [4]

1 + 1 + 4 = 6

Algorthim :

F [0 … 2] = 1

F [3] = 2

for i = 4 to n

   F[i] = F [i−4] + F [i−3] + F [i−1]

return F [n]

Q 4 :

$$R_0 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad R_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad R_2 = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$R_3 = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad R_4 = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} = T$$