第3次作业

- 1. 书上83页，3.1，3.3和3.5
- 要求写出优化函数的含义，原问题和子问题之间优化函数的关系及递推方程，写出计算过程和备忘录，并追踪最优解

3.1

用动态规划算法求解下面的组合优化问题：

$$\max\ g_1(x_1)+g_2(x_2)+g_3(x_3)$$

$$x_1^2+x_2^2+x_3^2 \leqslant 10$$

x_1, x_2, x_3 为非负整数

其中函数 $g_1(x)$，$g_2(x)$，$g_3(x)$的值给在表 3.9 中.

表 3.9　函数值

x	$g_1(x)$	$g_2(x)$	$g_3(x)$	x	$g_1(x)$	$g_2(x)$	$g_3(x)$
0	2	5	8	2	7	16	17
1	4	10	12	3	11	20	22

3.1 设 $F_k(y)$表示对 $x_1, x_2, \cdots, x_k$ 赋值，且 $x_1^2+x_2^2+\cdots+x_k^2 \leqslant y$ 时所得到的目标函数的最大值，递推关系和初值是

$$F_k(y) = \max_{0 \leqslant x_k \leqslant \lfloor \sqrt{y} \rfloor} \{F_{k-1}(y-x_k^2)+g_k(x_k)\}$$

$$F_1(y) = g_1(\lfloor \sqrt{y} \rfloor)$$

针对给定实例的计算过程如下.

$k=1$：

$F_1(0)=2$ $x_1=0$

$F_1(1)=F_1(2)=F_1(3)=g_1(1)=4$ $x_1=1$

$F_1(4)=F_1(5)=F_1(6)=F_1(7)=F_1(8)=g_1(2)=7$ $x_1=2$

$F_1(9)=F_1(10)=g_1(3)=11$ $x_1=3$

$k=2$：

$F_2(0)=\max\{F_1(0)+g_2(0)\}=7$ $x_2=0$

$F_2(1)=\max\{F_1(1)+g_2(0), F_1(0)+g_2(1)\}=12$ $x_2=1$

$F_2(2)=\max\{F_1(2)+g_2(0), F_1(1)+g_2(1)\}=14$ $x_2=1$

$F_2(3)=\max\{F_1(3)+g_2(0), F_1(2)+g_2(1)\}=14$ $x_2=1$

$F_2(4)=\max\{F_1(4)+g_2(0), F_1(3)+g_2(1), F_1(0)+g_2(2)\}=18$ $x_2=2$

$F_2(5)=\max\{F_1(5)+g_2(0), F_1(4)+g_2(1), F_1(1)+g_2(2)\}=20$ $x_2=2$

$F_2(6)=\max\{F_1(6)+g_2(0), F_1(5)+g_2(1), F_1(2)+g_2(2)\}=20$ $x_2=2$

$F_2(7)=\max\{F_1(7)+g_2(0), F_1(6)+g_2(1), F_1(3)+g_2(2)\}=20$ $x_2=2$

$F_2(8)=\max\{F_1(8)+g_2(0), F_1(7)+g_2(1), F_1(4)+g_2(2)\}=23$ $x_2=2$

$F_2(9)=\max\{F_1(9)+g_2(0), F_1(8)+g_2(1), F_1(5)+g_2(2), F_1(0)+g_2(3)\}=23$ $x_2=2$

$F_2(10)=\max\{F_1(10)+g_2(0), F_1(9)+g_2(1), F_1(6)+g_2(2), F_1(1)+g_2(3)\}=24$ $x_2=3$

$k=3$：

$F_3(0)=\max\{F_2(0)+g_3(0)\}=15$ $x_3=0$

$F_3(1)=\max\{F_2(1)+g_3(0),F_2(0)+g_3(1)\}=20$ $x_3=0$

$F_3(2)=\max\{F_2(2)+g_3(0),F_2(1)+g_3(1)\}=24$ $x_3=1$

$F_3(3)=\max\{F_2(3)+g_3(0),F_2(2)+g_3(1)\}=26$ $x_3=1$

$F_3(4)=\max\{F_2(4)+g_3(0),F_2(3)+g_3(1),F_2(0)+g_3(2)\}=26$ $x_3=1$

$F_3(5)=\max\{F_2(5)+g_3(0),F_2(4)+g_3(1),F_2(1)+g_3(2)\}=30$ $x_3=1$

$F_3(6)=\max\{F_2(6)+g_3(0),F_2(5)+g_3(1),F_2(2)+g_3(2)\}=32$ $x_3=1$

$F_3(7)=\max\{F_2(7)+g_3(0),F_2(6)+g_3(1),F_2(3)+g_3(2)\}=32$ $x_3=1$

$F_3(8)=\max\{F_2(8)+g_3(0),F_2(7)+g_3(1),F_2(4)+g_3(2)\}=35$ $x_3=2$

$F_3(9)=\max\{F_2(9)+g_3(0),F_2(8)+g_3(1),F_2(5)+g_3(2),F_2(0)+g_3(3)\}=37$ $x_3=2$

$F_3(10)=\max\{F_2(10)+g_3(0),F_2(9)+g_3(1),F_2(6)+g_3(2),F_2(1)+g_3(3)\}=37$ $x_3=2$

关于中间结果的备忘录如表 3.2 所示.

表 3.2 备忘录

y	$k=1$		$k=2$		$k=3$	
	$F_1(y)$	x_1	$F_2(y)$	x_2	$F_3(y)$	x_3
0	2	0	7	0	15	0
1	4	1	12	1	20	0
2	4	1	14	1	24	1
3	4	1	14	1	26	1
4	7	2	18	2	26	1
5	7	2	20	2	30	1
6	7	2	20	2	32	1
7	7	2	20	2	32	1
8	7	2	23	2	35	2
9	11	3	23	2	37	2
10	11	3	24	3	37	2

从而得到，$F_3(10)=37$，此刻 $x_3=2$，于是

$$x_1^2+x_2^2\leqslant 10-2^2=6$$

再查 $F_2(6)=20$，此刻 $x_2=2$，于是

$$x_1^2\leqslant 6-2^2=2$$

再查 $F_1(2)=4$ 得 $x_1=1$. 问题的解是：在 $x_1=1,x_2=2,x_3=2$ 时得到 $g_1(x_1)+g_2(x_2)+g_3(x_3)$ 的最大值 37.

3.3

有 n 个底面为长方形的货柜需要租用库房存放. 如果每个货柜都必须放在地面上，且所有货柜的底面宽度都等于库房的宽度，那么第 i 个货柜占用库房面积大小只需要用它的底面长度 l_i 来表示，$i=1, 2, \cdots, n$. 设库房总长度是 $D\left(l_i \leqslant D \text{ 且 } \sum_{i=1}^{n} l_i > D\right)$. 设第 i 号货柜的仓储收益是 v_i，若要求库房出租的收益达到最大，问如何选择放入库房的货柜？若 $l_1, l_2, \cdots, l_n, D$ 都是正整数，设计一个算法求解这个问题，给出算法的伪码描述并估计算法最坏情况下的时间复杂度.

3.3 类似于 0-1 背包问题，库房的长度相当于背包的重量限制，每个货柜的收益相当于物品的价值. 于是问题是：

$$\max \sum_{i=1}^{n} v_i x_i$$

$$\sum_{i=1}^{n} l_i x_i \leqslant D, \quad x_i = 0,1$$

令 $C[k,y]$是只允许装前 k 个货柜，库房长度为 y 时的最大收益，那么有

$$C[k,y] = \begin{cases} C[k-1,y] & y < l_k \\ \max\{C[k-1,y], C[k-1,y-l_k]+v_k\} & D \geqslant y \geqslant l_k \end{cases}, \quad k > 1$$

$$C[1,y] = \begin{cases} v_1 & y \geqslant l_1 \\ 0 & y < l_1 \end{cases}$$

算法的伪码是：

```
Store
输入：数组 L[1..n],V[1..n],D      //L 和 V 是货柜长度和价值序列，D 为库房长度
输出：最大的收益 C[n,D]
1. for y←1 to D
2.     C[1,y]←V[1]
3. for k←2 to n
4.     for y←1 to D
5.         C[k,y]←C[k−1,y]
6.         i[k,y]←i[k−1,y]
```

7. if $y \geqslant L[k]$ and $C[k-1, y-L[k]]+V[k]>C[k-1, y]$

8. then $C[k, y] \leftarrow C[k-1, y-L[k]]+V[k]$

9. $i[k, y] \leftarrow k$

算法在第 1 行时间为 $O(D)$，第 3 行和第 4 行的循环进行 $O(nD)$ 次，循环内部是常数时间的操作，于是算法最坏情况下的时间复杂度是 $O(nD)$.

3.5

设有 n 种不同面值的硬币，第 i 种硬币的币值是 v_k（其中 $v_1=1$），重量是 w_i，$i=1,2,\cdots,n$ 且现在购买总价值为 y 的某些商品，需要用这些硬币付款，如果每种钱币使用的个数不限，问如何选择付款的方法使得付出钱币的总重量最轻？设计一个求解该问题的算法，给出算法的伪码描述并分析算法的时间复杂度. 假设问题的输入实例是：

$$v_1=1,\quad v_2=4,\quad v_3=6,\quad v_4=8$$

$$w_1=1,\quad w_2=2,\quad w_3=4,\quad w_4=6$$

$$y=12$$

给出算法在该实例上计算的备忘录表和标记函数表，并说明付钱的方法.

3.5 设 x_i 表示第 i 种硬币使用的个数，$i=1,2,\cdots,n$. 该问题的描述为

$$\min \sum_{i=1}^{n} w_i x_i$$

$$\sum_{i=1}^{n} v_i x_i = y \quad x_i \text{ 为非负整数}$$

令 $F_k(x)$表示只允许使用前 k 种钱，总付款为 x 时所使用零钱的最轻重量，则

$$F_k(x) = \min\{F_{k-1}(x), F_k(x-v_k)+w_k\}, \quad k>1,\ 0<x\leqslant y$$

$$F_1(x) = w_1\left\lfloor \frac{x}{v_1} \right\rfloor = w_1 x$$

$$F_k(0) = 0$$

$$F_k(x) = +\infty, \quad x<0$$

设立标记函数 $t_k(y)$记录 $F_k(y)$取得最小值时最大币值的标号是否为 k.

$$t_k(x) = \begin{cases} k & F_k(x-v_k)+w_k \leqslant F_{k-1}(x) \\ t_{k-1}(x) & \text{否则} \end{cases}, \quad k>1,\ 0<x\leqslant y$$

$$t_1(x) = 1, \quad 0<x\leqslant y$$

$$t_k(0) = 0, \quad k=1,2,\cdots,n$$

算法的伪码是：

```
Coin
输入：w[1..n],v[1..n],y      //w,v 分别为硬币的重量和币值数组，y 是付款数
输出：F[i,j],t[i,j] i=1,2,…,n,j=1,2,…,y
1. for j←1 to y do
2.     F[1,j]←j * w[1]
3.     t[1,j]←1
4. for i←2 to n do
5.     for   j←1 to y do
6.         F[i,j]←F[i−1,j]
7.         t[i,j]←t[i−1,j]
8.         if F[i,j−v[i]]+w[i]≤F[i−1,j]
9.         then F[i,j]←F[i,j−v[i]]+w[i]
10.                t[i,j]←i
11. return 二维数组 F,t
```

算法的时间复杂度主要取决于第 4 行和第 5 行的 for 循环，内部工作量是常数时间，算法的时间是 $O(ny)$. 通过数组 t 追踪解的过程比较简单，时间不超过 $O(ny)$. 于是算法最坏情况下的时间复杂度是 $O(ny)$.

针对给定实例，算法的备忘录如表 3.5 和表 3.6 所示.

表 3.5 $F_k(x)$

k \ x	1	2	3	4	5	6	7	8	9	10	11	12
1	1	2	3	4	5	6	7	8	9	10	11	12
2	1	2	3	2	3	4	5	4	5	6	7	6
3	1	2	3	2	3	4	5	4	5	6	7	6
4	1	2	3	2	3	4	5	4	5	6	7	6

表 3.6 $t_k(x)$

k \ x	1	2	3	4	5	6	7	8	9	10	11	12
1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	2	2	2	2	2	2	2	2	2
3	1	1	1	2	2	3	3	2	2	3	3	2
4	1	1	1	2	2	3	3	2	2	3	3	2

问题实例的解是：最轻重量是 6，由 $t_4(12)=2$ 知道 $x_4=0, x_3=0, x_2\geqslant 1$，再由

$$t_2(12-4)=t_2(8)=2 \Rightarrow x_2 \geqslant 2$$

$$t_2(8-4)=t_2(4)=2 \Rightarrow x_2 \geqslant 3$$

$$t_2(4-4)=t_2(0)=0 \Rightarrow x_2=3, \quad x_1=0$$

从而得到 $x_1=x_3=x_4=0, x_2=3$，只用了 3 枚币值为 4 的硬币.