1.13 解答：

$f(n)$	$g(n)$	$f=O(g)$	$f=\Omega(g)$	$f=\Theta(g)$
$2n^3+3n$	$100n^2+2n+100$	False True False		
$50n+\log n$ 10	$n+\log\log n$	True True True		
$50n\log n$ 10	$n\log\log n$	False True False		
$\log n$ log	$^2\ n$	True False False		
$n!$ 5	^{n}	False True False		

1.14 解答：

(a) 因 $\lim\limits_{n\to\infty}\frac{2n+3\log^{100}n}{n}=2$，所以，$2n+3\log^{100}n=\Theta(n)$。

(b) 因 $\lim\limits_{n\to\infty}\frac{7n^3+1000n\log n+3n}{n^3}=7$，所以，$7n^3+1000n\log n+3n=\Theta(n^3)$.

(c) 因 $\lim\limits_{n\to\infty}\frac{3n^{1.5}+n^{1.5}\log n}{n^{1.5}\log n}=1$，所以，$3n^{1.5}+n^{1.5}\log n=\Theta(n^{1.5}\log n)$.

(d) 因 $\lim\limits_{n\to\infty}\frac{2^n+100^n+n!}{n!}=1$，所以，$2^n+100^n+n!=\Theta(n!)$.

1.16 解答：

(a) 算法执行元素比较运算的最小次数是 n－1。当输入 A[1..n] 已经有非减次序时该算法执行元素比较运算次数达到这个最小值。

(b) 算法执行元素比较运算的最大次数是(n－1)+(n－2)+⋯+ 2+1=n(n－1)/2。当输入 A[1..n] 已有递减次序时该算法执行元素比较运算次数达到这个最大值。 (c) 算法执行元素赋值运算的最小次数是 0。当输入 A[1..n] 已经有非减次序时该算法执行元素赋值运算次数达到这个最小值。

(d) 算法执行元素赋值运算的最大次数是 3[(n－1)+(n－2)+⋯+ 2+1]=3n(n－1)/2。当输入 A[1..n] 已有递减次序时该算法执行元素赋值运算次数达到这个最大值。

(e) 使用记号 O 和 ，算法 BUBBLESORT 的运行时间分别表示为 (n2)和 (n)。

(f) 该算法的运行时间不能使用记号 来表示，这是因为算法的运行时间范围为 n 的一次方到二次方。

1.31 (a) 第6步执行了$\lfloor\log n\rfloor(\lfloor\log n\rfloor+1)(2\lfloor\log n\rfloor+1)$.

(b) 用Θ表示更合适。因为$O(\lfloor\log^3 n\rfloor)=\Omega(\lfloor\log^3 n\rfloor)$

(c) 时间复杂性是$\Theta(\lfloor\log^3 n\rfloor)$

2.16 解答：

(a) 一方面，$\sum_{j=1}^{n} j\log j \leqslant \sum_{j=1}^{n} n\log n \leqslant n^2 \log n$，

另一方面，$\sum_{j=1}^{n} j\log j \geqslant \sum_{j=\lceil n/2\rceil}^{n} \lceil n/2\rceil \log\lceil n/2\rceil$

$\geqslant \lfloor n/2\rfloor\lceil n/2\rceil \log\lceil n/2\rceil$

$\geqslant (n-1)n/4 \cdot \log(n/2)$。

因此，$\sum_{j=1}^{n} j\log j = \Theta(n^2\log n)$。

(b) 令 $f(x)=x\log x \quad (x\geqslant 1)$。由于 $f(x)$ 是增函数，故有

$\sum_{j=1}^{n} j\log j \leqslant \int_1^{n+1} x\log x\,dx \leqslant (2(n+1)^2\ln(n+1)-(n+1)^2+1)/(4\ln 2)$；

同时，

$\sum_{j=1}^{n} j\log j = \sum_{j=2}^{n} j\log j \geqslant \int_1^{n} x\log x\,dx \geqslant (2n^2\ln n - n^2+1)/(4\ln 2)$。

因此，$\sum_{j=1}^{n} j\log j = \Theta(n^2\ln n) = \Theta(n^2\log n)$。

2.21 a) $f(n) = 2n^2 - n$

b) $f(n) = \Theta(n^2)$

5.7 参考例 5.1

5.12 为分析插入排序的时间代价，设X_i为桶$B[i]$ $(i = 1,2,\ldots,k)$中元素个数的随机变量。

对桶$B[i]$中的元素进行插入排序，期望时间为$E[O(X_i^2)] = O[E(X_i^2)]$.

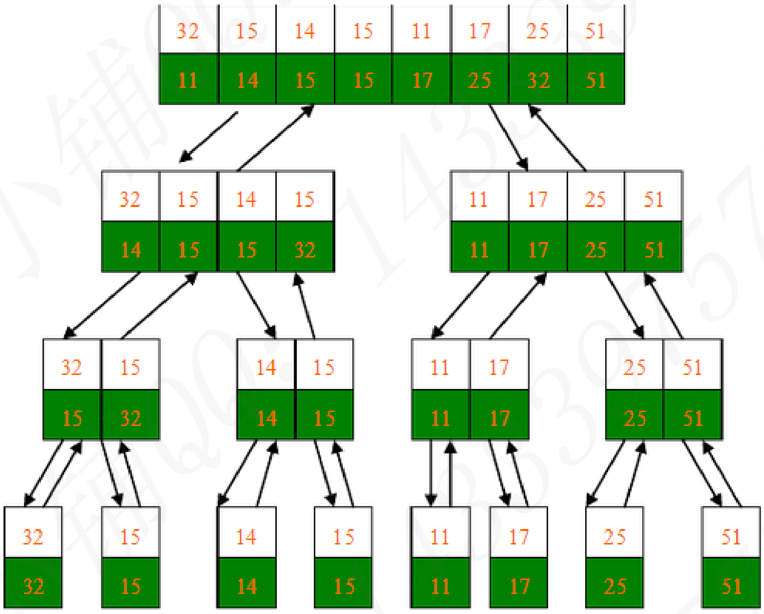
由于每个元素落入桶$B[i]$的概率为$1/k$,且每个元素落入桶中的事件相互独立。故随机变量 X 服从二项分布B(n,p). 故$E(X_i) = np = n/k$, 方差$V(X_i) = np(1-p) = (k-1)n/k^2$.

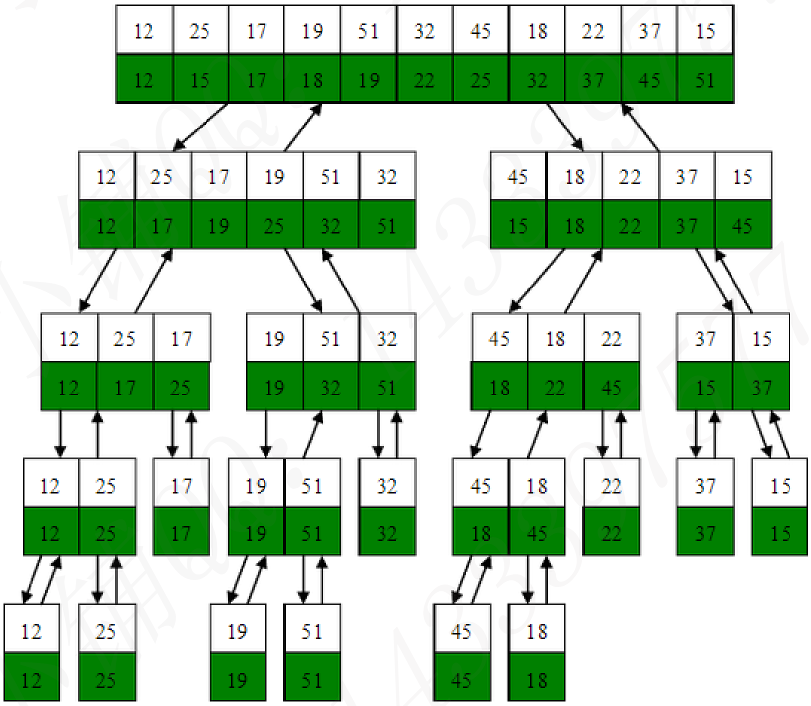
故$E(X_i^2) = V(X_i) + E^2(X_i) = (n^2 + (k-1)n)/k^2$。

总的运行时间为$k \times (n^2 + (k-1)n)/k^2 = (n^2 + (k-1)n)/k$.

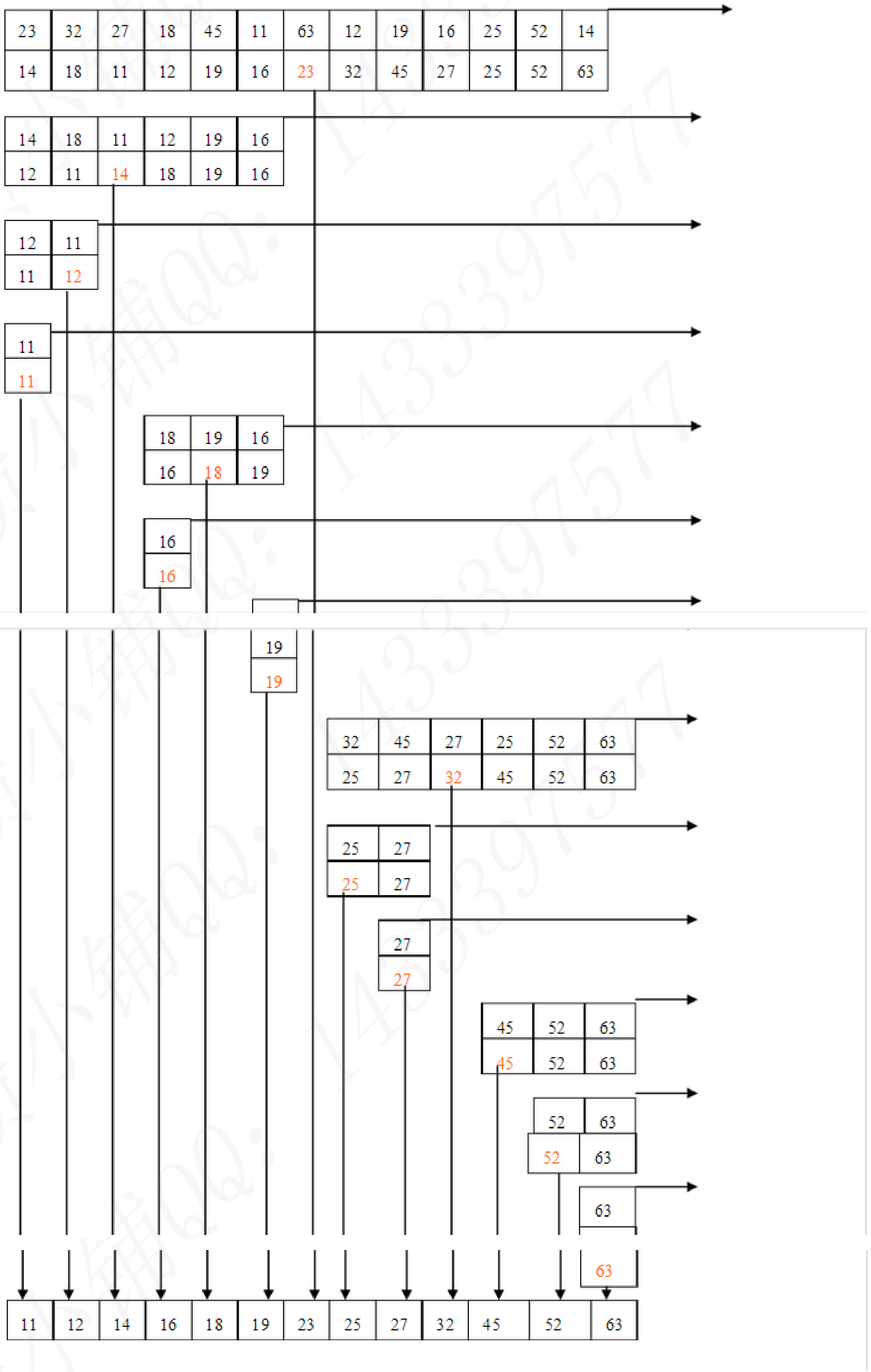
可见，若k十分接近n时，算法的时间复杂度为$\Theta(n)$.

6.10.


32 15 14 15 11 17 25 51
11 14 15 15 17 25 32 51
32 15 14 15
14 15 15 32
11 17 25 51
11 17 25 51
32 15
15 32
14 15
14 15
11 17
11 17
25 51
25 51
32
32
15
15
14
14
15
15
11
11
17
17
25
25
51
51


12 25 17 19 51 32 45 18 22 37 15
12 15 17 18 19 22 25 32 37 45 51
12 25 17 19 51 32
12 17 19 25 32 51
45 18 22 37 15
15 18 22 37 45
12 25 17
12 17 25
19 51 32
19 32 51
45 18 22
18 22 45
37 15
15 37
12 25
12 25
17
17
19 51
19 51
32
32
45 18
18 45
22
22
37
37
15
15
12
12
25
25
19
19
51
51
45
45
18
18

6.31


23 32 27 18 45 11 63 12 19 16 25 52 14
14 18 11 12 19 16 23 32 45 27 25 52 63
14 18 11 12 19 16
12 11 14 18 19 16
12 11
11 12
11
11
18 19 16
16 18 19
16
16
19
19
32 45 27 25 52 63
25 27 32 45 52 63
25 27
25 27
27
27
45 52 63
45 52 63
52 63
52 63
63
63
11 12 14 16 18 19 23 25 27 32 45 52 63

7.5

字符串 A=”xzyzzyx”和 B=”zxyyzxz”的最长公共子序列长度为 4，共有 6 个最长公共子序列，分别是：①zyyx ②zyzz ③zyzx ④xyyx ⑤xyzz ⑥xyzx

7.9

C[1,5]=C[1,1]+C[2,5]+r[1]*r[2]*r[6]=307

C[1,5]=C[1,2]+C[3,5]+r[1]*r[3]*r[6]=252

C[1,5]=C[1,3]+C[4,5]+r[1]*r[4]*r[6]=372

C[1,5]=C[1,4]+C[5,5]+r[1]*r[5]*r[6]=260

所以最优括号表达式为（M1M2）((M3M4)M5)

7.21

	0	1	2	3	4	5	6	7	8	9	10	11
0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	3	3	3	3	3	3	3	3	3	3
2	0	0	3	4	4	7	7	7	7	7	7	7
3	0	0	3	4	4	7	7	8	9	9	12	12
4	0	0	3	4	4	7	7	8	10	11	12	14

7.30 讲解.感谢周爱爱同学发现原有方案的问题。

(a)设$c[j], j = 0 \cdots y$,表示用用该硬币系统找钱j的最少硬币个数。

对于任何$0 \le j \le y$及$0 \le i \le n$,若$j - v_i > 0$, 则$c[j - v[i]]$所表示的找钱$j - v[i]$的最优序列，再加1枚面值为 $v[i]$的硬币是一种找钱j的方法，且所用的硬币个数为$c[j - v[i]] + 1$。由此可得以下递规式：

$$c[j] = \min_{0 \le i \le n} \{c[j - v_i] + 1\}$$

其初始条件为：

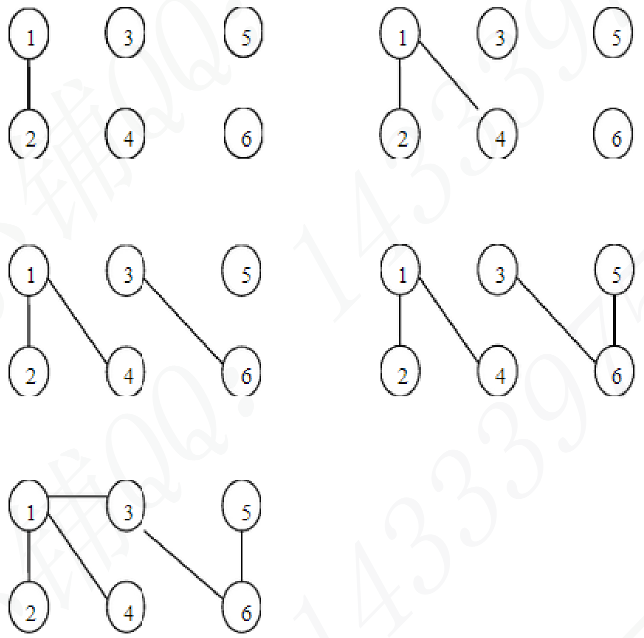
$$c[0] = 0$$

算法自己写.

(b) 算法的时间复杂性为$\Theta(ny)$,空间复杂性为$O(y)$。

(c) y相当于背包容量，n种硬币相当于n种体积为$v[i]$、价值为1的物品。 但该题要求背包恰好装满，且总价值最小。

8.23



8.24（共有 4 棵最小生成树，此处仅举一例）