Algorithm HW1

1. 总基本运算次数（加+乘）= 2mnp，时间复杂度：O(mnp)。
2. 两种方法在最坏情况下比较次数相同，均为 2(n-1)。
3.

（1）若 $10n^2+9=O(n)$，则按 O 的定义，存在正常数$c>0$、n_0，使得对所有 $n \geq n_0$：

$$10n^2 \leq cn-9 < cn$$

即：

$$10n \leq c$$

但当 $n \to \infty$ 时，不可能被某个常数 c 上界。产生矛盾。

（2）同理，按 Θ 的定义，若成立，则存在正常数 $c_1, c_2 > 0$和 n_0，使得对所有 $n \geq n_0$：

$$c_1 n^2 \leq n^2 \log n \leq c_2 n^2$$

即：

$$c_1 \leq \log n \leq c_2$$

但不存在有限常数 c_2 使不等式恒成立，产生矛盾。

4.

⇒: 由极限定义：对任意 $\varepsilon > 0$，存在 n_0，当 $n \geq n_0$ 时，

$$\left|\frac{f(n)}{g(n)}\right| < \varepsilon$$

等价于:

$$|f(n)| < \varepsilon |g(n)|$$

根据小 (o) 记号的定义，这正是：$f(n) = o(g(n))$。

$\Leftarrow$: 由小 o 的定义：对任意 $\varepsilon > 0$，存在 n_0，当 $n \geq n_0$ 时，

$$|f(n)| < \varepsilon |g(n)|$$

两边同时除以 $|g(n)|$（n 足够大时 $g(n) \neq 0$）：

$$\left|\frac{f(n)}{g(n)}\right| < \varepsilon$$

这正是：$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$。

5.

第(1)条不成立：$g(n) = O(G(n))$ 只给了 g 的上界，完全可能 g 比 G 小得多，从而把 f/g 放大得很厉害。反例取 $f(n) = 1, F(n) = 1$，取 $g(n) = 1/n, G(n) = 1$。显然 $f(n) = O(F(n))$，也有 $g(n) = O(G(n))$，但 $\frac{f(n)}{g(n)} = n$，而 $\frac{F(n)}{G(n)} = 1$，所以 $\frac{f}{g} \notin O(\frac{F}{G})$。

第(2)条不成立，因为同样只有上界时，f 可以远小于 F，让 $\frac{f}{g}$ 远小于 $\frac{F}{G}$，从而不可能给出 Ω。反例取 $f(n) = 1, F(n) = n$，取 $g(n) = n, G(n) = n$。则 $f(n) = O(n) = O(F(n))$，$g(n) = O(n) = O(G(n))$，但 $\frac{f}{g} = 1/n$，而 $\frac{F}{G} = 1$，显然 $1/n \notin \Omega(1)$。

第(3)条要求同时有 O 和 Ω 才能推出 Θ，所以更不可能成立。

第(4)条不成立：$g(n) = \Omega(G(n))$ 只给了 g 的下界，g 可能比 G 大很多，从而把 $\frac{f}{g}$ 压得很小，无法保证是 $\Omega(\frac{F}{G})$。

第(5)条不成立：只知道 f 不小于 F、g 不小于 G，并不能阻止 f 比 F 大很多、而 g 只是"刚好"大于 G，于是 $\frac{f}{g}$ 可能远大于 $\frac{F}{G}$。

只有第(6)条是正确的：若 $f(n) = \Theta(F(n))$ 且 $g(n) = \Theta(G(n))$，则存在常数 $c_1, c_2, d_1, d_2 > 0$ 和 n_0，使得对所有 $n \geq n_0$，有 $c_1 F(n) \leq f(n) \leq c_2 F(n)$ 且 $d_1 G(n) \leq g(n) \leq d_2 G(n)$。在 $g(n), G(n)$ 最终为正且不为零时两边相除就得到

$$\frac{c_1}{d_2}\frac{F(n)}{G(n)} \leq \frac{f(n)}{g(n)} \leq \frac{c_2}{d_1}\frac{F(n)}{G(n)},$$

这正是 $\frac{f(n)}{g(n)} = \Theta\left(\frac{F(n)}{G(n)}\right)$。

6. $\log n < n^{2/3} < 20n < 4n^2 < 3^n < n!$

7.

(1) 若仍用原算法 $T(n) = 3 \cdot 2^n$，设新机器在 t 秒内能跑到的规模为 n'，则满足 $3 \cdot 2^{n'} = 64 \cdot 3 \cdot 2^n$，两边约掉 3 得 $2^{n'} = 64 \cdot 2^n = 2^6 \cdot 2^n$，所以 $n' = n + 6$。

(2) 若改进后算法 $T(n) = n^2$，设新机器在 t 秒内能处理的规模为 n'，则 $n'^2 = 64 \cdot 3 \cdot 2^n$，因此 $n' = \sqrt{64 \cdot 3 \cdot 2^n} = 8\sqrt{3}\, 2^{n/2}$。

(3) 若进一步改进后 $T(n) = 8$，在新机器的 t 秒内只要能完成至少 8 次基本操作即可，而新机器在 t 秒内能做 $64 \cdot 3 \cdot 2^n = 192 \cdot 2^n$ 次操作，显然对任意 $n \geq 0$ 都有 $192 \cdot 2^n \geq 8$，所以在该模型下输入规模不再受 t 限制：可处理“任意大”的 n。

8. 设解为指数形式 $F(n) = r^n$，代入递推关系得特征方程：

$$r^n = r^{n-1} + r^{n-2} \Longrightarrow r^2 = r + 1$$

解该二次方程：

$$r_{1,2} = \frac{1 \pm \sqrt{5}}{2}$$

因此通解为：

$$F(n) = A \left(\frac{1 + \sqrt{5}}{2} \right)^n + B \left(\frac{1 - \sqrt{5}}{2} \right)^n$$

利用初始条件确定常数 A, B。

当 $n = 0$：$A + B = 1$；当 $n = 1$：$A \frac{1+\sqrt{5}}{2} + B \frac{1-\sqrt{5}}{2} = 1$。

联立解得：

$$A = \frac{1}{\sqrt{5}}, \qquad B = -\frac{1}{\sqrt{5}}$$

$$F(n) = \frac{1}{\sqrt{5}} \left[\left(\frac{1 + \sqrt{5}}{2} \right)^{n+1} \left(\frac{1 - \sqrt{5}}{2} \right)^{n+1} \right]$$