第一章习题

1. 试确定下述程序的执行步数，该函数实现一个 m×n 矩阵与一个 n×p 矩阵之间的乘法：

矩阵乘法运算

```
template<class T>
void Mult(T **a, T **b, int m, int n, int p)
{//m×n 矩阵 a 与 n×p 矩阵 b 相成得到 m×p 矩阵 c
   for(int i=0; i<m; i++)
      for(int j=0; j<p; j++){
        T sum=0;
        for(int k=0; k<n; k++)
           Sum+=a[i][k]*b[k][j];
        C[i][j]=sum;
      }
}
```

for(int i=0; i<m; i++)	m+1
for(int j=0; j<p; j++){	m*(p+1)
T sum=0;	m*p
for(int k=0; k<n; k++)	m*p*(n+1)
Sum+=a[i][k]*b[k][j];	m*p*n
C[i][j]=sum;	m*p
	m+1+mp+m+mp+mpn+mp+mpn+mp =1+2m+4mp+2mpn

2． 函数 MinMax 用来查找数组 a[0:n-1]中的最大元素和最小元素，以下给出两个程序。令 n 为实例特征。试问：在各个程序中，a 中元素之间的比较次数在最坏情况下各是多少？

找最大最小元素 （方法一）

```
template<class T>
bool MinMax(T a[], int n, int& Min, int& Max)
{//寻找 a[0:n-1]中的最小元素与最大元素
  //如果数组中的元素数目小于 1，则还回 false
  if(n<1) return false;
  Min=Max=0; //初始化
   for(int i=1; i<n; i++){
      if(a[Min]>a[i]) Min=i;
      if(a[Max]<a[i]) Max=i;
      }
   return true;
}
```

找最大最小元素 （方法二）

```
template<class T>
bool MinMax(T a[], int n, int& Min, int& Max)
{//寻找 a[0:n-1]中的最小元素与最大元素
  //如果数组中的元素数目小于 1，则还回 false
  if(n<1) return false;
  Min=Max=0; //初始化
   for(int i=1; i<n; i++){
      if(a[Min]>a[i]) Min=i;
      else if(a[Max]<a[i]) Max=i;
      }
   return true;
}
```

方法一（没有 else）	2*(n-1)
方法二	2*(n-1) 最大值在末尾的时候

3.证明以下关系式不成立：

1). $10n^2+9=O(n)$;

2). $n^2\log n=\Theta(n^2)$；

3.1 证明：

根据渐近符号 O 定义：

$10n^2+9=O(n)$ 成立当且仅当 $\exists c, n_0$. 对 $\forall n>n_0$. 有

$10n^2+9 \le cn$ ， $c. n_0$ 为正常数

假设 $10n^2+9=O(n)$ 成立. 则 $\exists c. n_0$. 对 $\forall n>n_0$. 有

$10n+\frac{9}{n} \le c$ ， $c. n_0$ 为正常数.

根据极限定义 $\lim\limits_{n\to\infty} 10n+\frac{9}{n}=\infty$. 故不存在 c 使得 $10n^2+9=O(n)$ 成立

3.2 证明：

根据符号 θ 定义

$n^2\log n=\theta(n^2)$ 成立 当且仅当 $\exists$ $c_1. c_2. n_0$. 对 $\forall$ $n>n_0$. 有

$c_1n^2 \le n^2\log n \le c_2n^2$ $c_1. c_2. n_0$ 为正常数

假设 $n^2\log n=\theta(n^2)$ 成立. 则 $\exists$ $c_1. c_2. n_0$. 对 $\forall n>n_0$. 有

$c_1 \le \log n \le c_2$ ， $c_1. c_2. n_0$ 为正常数

根据极限定义 $\lim\limits_{n\to\infty}\log n=\infty$. 故 c_2 不存在.

故 $n^2\log n=\theta(n^2)$ 不成立

4．证明：当且仅当$\lim\limits_{n\to\infty} f(n)/g(n)=0$时，$f(n)=o(g(n))$。

必要性. 即当 $\lim\limits_{n\to\infty}\frac{f(n)}{g(n)}=0 \Rightarrow f(n)=o(g(n))$

由于 $\lim\limits_{n\to\infty}\frac{f(n)}{g(n)}=0$ 即存在正常数 c_1. 如 1. 使得 $\lim\limits_{n\to\infty}\frac{f(n)}{g(n)}\le c_1$

故 $f(n)=O(g(n))$

假设 $g(n)=O(f(n))$ 则 $\exists c_2, n_0>0$ 使得对 $\forall n>n_0$. 都有

$g(n)\le c_2 f(n)$ 即 $\frac{g(n)}{f(n)}\le c_2$.

这与 $\lim\limits_{n\to\infty}\frac{f(n)}{g(n)}=0$ 矛盾. 故 $g(n)\ne O(f(n))$

故 $f(n)=o(g(n))$

充分性：

$$f(n)=o(g(n)) \to \begin{cases} f(n)=O(g(n)) \\ g(n)\ne O(f(n)) \end{cases}$$

由 $g(n)\ne O(f(n))$ 可得 对 $\forall c_3>0$ 有.

$$\lim_{n\to\infty}\frac{g(n)}{f(n)}>c_3.$$

则 $$\lim_{n\to\infty}\frac{f(n)}{g(n)}<\frac{1}{c_3}$$

又 $f(n)=O(g(n))$

故 $$\lim_{n\to\infty}\frac{f(n)}{g(n)}=0.$$

5.下面那些规则是正确的？为什么？

1). $\{f(n)=O(F(n)), g(n)=O(G(n))\}\Rightarrow f(n)/g(n)=O(F(n)/G(n))$；

2). $\{f(n)=O(F(n)), g(n)=O(G(n))\}\Rightarrow f(n)/g(n)=\Omega(F(n)/G(n))$；

3). $\{f(n)=O(F(n)), g(n)=O(G(n))\}\Rightarrow f(n)/g(n)=\Theta(F(n)/G(n))$；

4). $\{f(n)=\Omega(F(n)), g(n)=\Omega(G(n))\}\Rightarrow f(n)/g(n)=\Omega(F(n)/G(n))$；

5). $\{f(n)=\Omega(F(n)), g(n)=\Omega(G(n))\}\Rightarrow f(n)/g(n)=O(F(n)/G(n))$。

6). $\{f(n)=\Theta(F(n)), g(n)=\Theta(G(n))\}\Rightarrow f(n)/g(n)=\Theta(F(n)/G(n))$

5.1 错误 反例：f(n)=n^2 F(n)=n^2 g(n)=n G(n)=n^2 则 f(n)/g(n)=n 但是 F(n)/G(n)=1

5.2 错误 反例：f(n)=n F(n)=n^3 g(n)=n G(n)=n 则 f(n)/g(n)=1 但是 F(n)/G(n)=n^2
5.3 错误 因为上界错误（理由见 5.1）
5.4 错误 反例：f(n)=n^2 F(n)=n^2 g(n)=n^2 G(n)=n 则 f(n)/g(n)=1 但是 F(n)/G(n)=n
5.5 错误 反例：f(n)=n^2 F(n)=n g(n)=n G(n)=n 则 f(n)/g(n)=n 但是 F(n)/G(n)=1
5.6 正确
证明：

$f(n) = \Theta(F(n))$. $g(n) = \Theta(G(n))$

$\Rightarrow$ ∃正常数 c_1, c_2, n_0. 对 $\forall n > n_0$ 有 $c_1 F(n) \le f(n) \le c_2 F(n)$ ①

∃正常数 c_3, c_4, n_1. 对 $\forall n > n_1$ 有 $c_3 G(n) \le g(n) \le c_4 G(n)$ ②

①、② 相除、得到

$$\frac{c_1}{c_3}\frac{F(n)}{G(n)} \le \frac{f(n)}{g(n)} \le \frac{c_2}{c_4}\frac{F(n)}{G(n)}$$

即 ∃正常数 $c_5 = \frac{c_1}{c_3}$. $c_6 = \frac{c_2}{c_4}$. $n_2 = \max\{n_0, n_1\}$

对 $\forall n > n_2$. 成立.

故 $\frac{f(n)}{g(n)} = \Theta \frac{F(n)}{G(n)}$ 成立

6. 按照渐近阶从低到高的顺序排列以下表达式:

$$4n^2, \log n, 3^n, 20n, n^{2/3}, n!$$

4n^2	O(n^2)
logn	O(logn)
3^n	O(3^n)
20n	O(n)
n^(2/3)	O(n^(2/3))
n!	O(n!)

顺序从小到大为：logn < n^(2/3) < 20n < 4n^2 < 3^n < n!

7. 1) 假设某算法在输入规模是 n 时为 $T(n) = 3 * 2^n$. 在某台计算机上实现并完成该算

法的时间是t秒.现有另一台计算机,其运行速度为第一台的64倍，那么,在这台计算机上用同一算法在t秒内能解决规模为多大的问题?

2) 若上述算法改进后的新算法的时间复杂度为$T(n)=n^2$，则在新机器上用t秒时间能解决输入规模为多大的问题？

3）若进一步改进算法，最新的算法的时间复杂度为 $T(n)=8$，其余条件不变，在新机器上运行，在t秒内能够解决输入规模为多大的问题？

7.1 新机器的运行速度是第一台的 64 倍，那么在 t 秒时间内，能够解决 64*3*2^n
由于算法未改变，故：
设输入规模为 x，则 3*2^x = 64*3*2^n
求得 x=6+n

7.2 新算法时间复杂度为 T(n) = n^2 又因为在新机器上，t 秒能够解决 64*3*2^n
因原算法输入规模为 n
设新算法输入规模为 x
则 x^2 = 64*3*2^n
求得 x=根号下[3*2*(6+n)]

7.3 新算法的执行时间与输入规模无关，故 t 妙内能够解决任意输入规模的问题

8. Fibonacci 数有递推关系：

$$F(n)=\begin{cases}1, & n=0\\ 1, & n=1\\ F(n-1)+F(n-2), & n>1\end{cases}$$

试求出$F(n)$的表达式。

$$F(n)=F(n-1)+F(n-2)$$

$$F_n-\alpha F(n-1)=\beta(F(n-1)-\alpha F(n-2))+y$$

$$F_n=(\alpha+\beta)F_{n-1}-\alpha\beta F_{n-2}+y \Rightarrow \begin{cases}\alpha+\beta=1\\ -\alpha\beta=1\\ y=0\end{cases}$$

$$\Rightarrow \begin{cases}\alpha=\frac{1+\sqrt{5}}{2}\\ \beta=\frac{1-\sqrt{5}}{2}\\ y=0\end{cases}$$

则 $F_n=C_1\alpha^n+C_2\beta^n$

$$C_1\alpha+C_2\beta=C_1\alpha^2+C_2\beta^2=1$$

求得 $C_1=\frac{1}{\sqrt{5}}$ $C_2=-\frac{1}{\sqrt{5}}$

$$\therefore F_n=\begin{cases}\frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^n-\left(\frac{1-\sqrt{5}}{2}\right)^n\right]. & n=1.2.3\cdots\\ 1 & n=0\end{cases}$$