【示例-1】分析以下算法的时间复杂度。

```
void MATMulti(int n,a[][N],b[][N],int c[][N])
{
   int i,j;
   for (i=0;i<n;i++)                        //①
      for (j=0;j<n;j++)                     //②
      {  c[i][j]=0;                         //③
         for (k=0;k<n;k++)                  //④
           c[i][j]=c[i][j]+a[i][k]*b[k][j];   //⑤
      }
}
```

解法1：从求算法中所有语句的频度来分析算法时间复杂度。

- ✓ 语句①的执行频度为$n+1$（注意$i<n$判断语句需执行$n+1$次）
- ✓ 语句②的执行频度为$n(n+1)$
- ✓ 语句③的执行频度为n^2
- ✓ 语句④的执行频度为$n^2(n+1)$
- ✓ 语句⑤的执行频度为n^3

⇨ $f(n)=2n^3+3n^2+2n+1$

$T(n)=O(n^3)$

解法2：从算法中基本运算的频度来分析算法时间复杂度。

- 基本运算是语句⑤
- 其频度为n^3

⇨ $T(n)=O(n^3)$

结论：从中看到，两种方法的结果相同，而第二种方法更加简洁。

【示例-2】给出以下算法的时间复杂度。

```
void func(int n)
{
    int i=1,k=100;
    while (i<=n)
    {   k++;
        i+=2;
    }
}
```

解：基本运算语句是while循环内的语句。

设while循环语句执行的次数为m，i从1开始递增，最后取值为$1+2m$，有：

$i=1+2m \le n$

即

$T(n)=m \le (n-1)/2=O(n)$

该算法的时间复杂度为$O(n)$。

【示例-3】分析示例-1和示例-2算法的空间复杂度。

解：这两个算法中，局部变量只有固定几个变量，故它们的空间复杂度均为O(1)，即该算法为原时工作算法。

【示例-4】设3个表示算法频度的函数f、g和h分别为：

$f(n)=100n^3+n^2+1000$

$h(n)=n^{1.5}+5000n\log_2 n$

求它们对应的时间复杂度。

解：

$f(n)=100n^3+n^2+1000=O(n^3)$，

当$n\to\infty$时，$\sqrt{n}>\log_2 n$，

所以$h(n)=n^{1.5}+5000n\log_2 n=O(n^{1.5})$。

【示例-5】 2011年全国考研题

设n是描述问题规模的非负整数，下面程序片段的时间复杂度为________。

```
x=2;
while (x<n/2)
      x=2*x;
```

A. $O(\log_2 n)$　　B. $O(n)$

C. $O(n\log_2 n)$　　D. $O(n^2)$

解：

基本算法是语句x=2*x，设其执行时间为$T(n)$，则有：$2^{T(n)} < n/2$，即$T(n)<\log_2 n/2=O(\log_2 n)$。本题答案为A。

【示例-6】本题为2012年全国考研题

求整数n（n≥0）阶乘的算法如下，其时间复杂度是______。

```
int fact(int n)
{
   if (n<=1) return 1;
   else return n*fact(n-1);
}
```

A. $O(\log_2 n)$ **B.** $O(n)$ **C.** $O(n\log_2 n)$ **D.** $O(n^2)$

解：本算法是一个递归算法，设其执行时间为T(n)，则有：

$T(n)=1$ 当n=1时

$T(n)=T(n-1)+1$ 当n>1时

所以，$T(n)=T(n-1)+1= (T(n-2)+1)+1 = T(n-2)+2 =\ldots=T(1)+(n-1) = n = O(n)$。本题答案为B。

END