1.2.1

> 一个算法所需时间由下述递归方程表示，试求出该算法的时间复杂度的级别(或阶).
>
> $$T(n)=\begin{cases}1,n=1\\2T(n/2)+n,n>1\end{cases}$$
>
> 式中， n是问题的规模，为简单起见，设 n是 2的整数次幂。

$T(n)=2T(\frac{n}{2})+nC$

$=2[2T(\frac{n}{2^2})+\frac{nC}{2}]+nC$

$=2^2T(\frac{n}{2^2})+2nC$

$=2^2[2T(\frac{n}{2^3})+\frac{nC}{2^2}]+2nC$

$=2^3T(\frac{n}{2^3})+3nC$

$=\cdots$

$=2^kT(\frac{n}{2^k})+knC$

令 $\frac{n}{2^k}=1$,即 $n=2^k,k=\log_2 n$

代入原式得: $T(n)=2^k+\log_2 n\times nC=n+n\log_2 nC=O(n\log_2 n)$

1.2.2

> 分析以下各程序段，求出算法的时间复杂度.

```
i=1;k=0;
while(i<n-1){
    k=k+10*i;
    i++;
}
```

```
y=0;
while((y+1)*(y+1)<=n)
        y=y+1;
```

```
for(int i=1;i<=n;i++)
        for(int j=1;j<=i;j++)
            for(int k=1;k<=j;k++)
                    x++;
```

```
for(int i=0;i<n;i++)
    for(int j=0;j<m;j++)
            a[i][j]=0;
```

时间复杂度分别为 $O(n)$, $O(\sqrt{n})$, $O(n^3)$, $O(nm)$.

③

$$\sum_{i=1}^{n}\sum_{j=1}^{i}\boxed{\sum_{k=1}^{j}1} = \sum_{i=1}^{n}\boxed{\sum_{j=1}^{i} j}$$

for $(k=1;\ k\le j;\ k{+}{+})$
循环一次.

$$= \boxed{\sum_{i=1}^{n}\frac{i(i+1)}{2}}$$

for (~~int~~ $j=1;\ j\le i;\ j{+}{+}$)
循环 j 次
$1+2+3+\cdots+i$

$$= \sum_{i=1}^{n}\frac{i^2}{2} + \sum_{i=1}^{n}\frac{i}{2}$$

for $(i=1;\ i\le n;\ i{+}{+})$
循环 $\frac{i(i+1)}{2}$

$$= \frac{1}{2}\left[\sum_{i=1}^{n} i^2 + \sum_{i=1}^{n} i\right]$$

$$= \frac{1}{2}\,\frac{n(n+1)(2n+1)}{6} + \frac{1}{2}\,\frac{n(n+1)}{2}$$

☆ $\boxed{1^2+2^2+3^2+\cdots+n^2 = \frac{n(n+1)(2n+1)}{6}}$

$$= \frac{1}{2}\left[\frac{n(n+1)(2n+1)}{6} + \frac{3n(n+1)}{6}\right]$$

$$= \frac{1}{2}\,\frac{n(n+1)[(2n+1)+3]}{6} = \frac{n(n+1)(n+2)}{6} = O(n^3)$$