算分 Ch-01 基础知识

1. 算法 有限步内停机

2. 算法的时间复杂度

- 最坏情况下时间复杂度 $W(n)$
- 平均情况下时间复杂度 $A(n)$

3. 函数的渐进的界 设 f 和 g 是定义域在自然数集 $\mathbb{N}$ 上的函数。

- $\exists\, c > 0$ 和 n_0，对 $\forall\, n \geq n_0$，有 $0 \leq f(n) \leq cg(n)$ 成立，称 $f(n)$ 的渐进的上界为 $g(n)$，记作 $f(n) = O(g(n))$。
- $\exists\, c > 0$ 和 n_0，对 $\forall\, n \geq n_0$，有 $0 \leq cg(n) \leq f(n)$ 成立，称 $f(n)$ 的渐进的下界为 $g(n)$，记作 $f(n) = \Omega(g(n))$。
- 若对 $\forall\, c > 0$ 都 $\exists\, n_0$，当 $n \geq n_0$ 时有 $0 \leq f(n) < cg(n)$ 成立，记作 $f(n) = o(g(n))$。
- 若对 $\forall\, c > 0$ 都 $\exists\, n_0$，当 $n \geq n_0$ 时有 $0 \leq cg(n) < f(n)$ 成立，记作 $f(n) = \omega(g(n))$。
- 若 $f(n) = O(g(n))$ 且 $f(n) = \Omega(g(n))$，记作 $f(n) = \Theta(g(n))$。

定理 1.1 设 f 和 g 是定义域为自然数集合的函数

- 如果 $\lim\limits_{n\to\infty} \dfrac{f(n)}{g(n)}$ 存在且等于某个常数 $c > 0$，那么 $f(n) = \Theta(g(n))$。
- 如果 $\lim\limits_{n\to\infty} \dfrac{f(n)}{g(n)} = 0$，那么 $f(n) = o(g(n))$。
- 如果 $\lim\limits_{n\to\infty} \dfrac{f(n)}{g(n)} = +\infty$，那么 $f(n) = \omega(g(n))$。

定理 1.2 设 f、g 和 h 是定义域为自然数集合的函数

- 如果 $f = O(g)$ 且 $g = O(h)$，那么 $f = O(h)$。
- 如果 $f = \Omega(g)$ 且 $g = \Omega(h)$，那么 $f = \Omega(n)$。
- 如果 $f = \Theta(g)$ 且 $g = \Theta(h)$，那么 $f = \Theta(h)$。

定理 1.3 设 f 和 g 是定义域为自然数集合的函数，若对某个其他的函数 h，有 $f = O(h)$ 和 $g = O(h)$，那么 $f + g = O(h)$。

定理 1.4 $\forall\, b > 1, \alpha > 0, \log_b n = o(n^\alpha)$.

定理 1.5 $\forall\, r > 1, d > 0, n^d = o(r^n)$.

$(\log n)^{\log n} = n^{\log\log n}$、$a^{\log_b n} = n^{\log_b a}$、$\log(n!) = \Theta(n \log n)$、$n^{\frac{1}{\log n}} = 1$、$\log_k n = \Theta(\log_l n)$

4. 斯特林公式

$$n! = \sqrt{2\pi n}(\frac{n}{e})^n(1+\Theta(\frac{1}{n}))$$

递推方程求解方法　公式法、迭代归纳法、尝试法、递归树、主定理

5. 主定理　设 $a \geq 1, b > 1$ 为常数， $f(n)$ 为函数，$T(n)$ 为非负整数，且

$$T(n) = aT(n/b) + f(n)$$

则有以下结果：

- 若 $f(n) = O(n^{\log_b a-\epsilon})$，$\epsilon > 0$，那么 $T(n) = \Theta(n^{\log_b a})$.
- 若 $f(n) = \Theta(n^{\log_b a})$，那么 $T(n) = \Theta(n^{\log_b a}\log n)$.
- 若 $f(n) = \Omega(n^{\log_b a+\epsilon})$，$\epsilon > 0$，且对某个常数 $c < 1$ 和所有充分大的 n，有 $af(n/b) \leq cf(n)$，那么 $T(n) = \Theta(f(n))$.

6. 插入排序 *InsertSort*

```
InsertSort(A,n)
输入：n个数的数组A
输出：按照递增顺序排好序的数组A
for j <- 2 to n do
    x <- A[j]
    i <- j-1
    while i>0 and x<A[i] do
        A[i+1] <- A[i]
        i <- i-1
    A[i+1] <- x
```

7. 二分归并排序 *MergeSort*

```
MergeSort(A,p,r)
输入：数组A[p..r], 1 <= p <= r <= n
输出：从A[p]到A[r]按照递增顺序排好序的数组A
if p < r
then q <- (p+r)/2
    MergeSort(A,p,q)
    MergeSort(A,q+1,r)
    MergeSort(A,p,q,r)
```

```
Merge(A,p,q,r)
输入：按照递增顺序排好序的数组A[p..q]与A[q+1..r]
输出：按照递增顺序排序的数组A[p..r]
 x <- q-p+1, y <- r-q
 将A[p..q]复制到B[1..x]，将A[q+1..r]复制到C[1..y]
 i <- 1, j <- 1, k <- p
```

```
while i<=x and j<=y do
    if B[i] <= C[j]
    then A[k] <- B[i]
         i <- i+1
    else A[k] <- C[j]
         j <- j+1
    k <- k+1
if i>x then 将C[j..y]复制到A[k..r]     // B已空
else 将B[i..x]复制到A[k..r]            // C已空
```