01.2 - Asymptotic Notation

渐近符号（Asymptotic Notation）

$$
\begin{gathered}
f(n)=O(g(n)) \Rightarrow f(n) \leq c g(n) \\
f(n)=o(g(n)) \Rightarrow f(n)<c g(n) \\
f(n)=\Theta(g(n)) \Rightarrow c_1 g(n) \leq f(n) \leq c_2 g(n) \\
f(n)=\Omega(g(n)) \Rightarrow f(n) \geq c g(n) \\
f(n)=\omega(g(n)) \Rightarrow f(n)>c g(n) \\
f(n)=O(g(n)), f(n)=\Omega(g(n)) \Rightarrow f(n)=\Theta(g(n))
\end{gathered}
$$

Comparing Functions (比较函数)

渐近比较满足几种关系性质:

- Transitivity (传递性):
  - 如果 $f(n)=\Theta(g(n))$ 且 $g(n)=\Theta(h(n))$，则 $f(n)=\Theta(h(n))$
  - 如果 $f(n)=O(g(n))$ 且 $g(n)=O(h(n))$，则 $f(n)=O(h(n))$
  - 如果 $f(n)=\Omega(g(n))$ 且 $g(n)=\Omega(h(n))$，则 $f(n)=\Omega(h(n))$
  - 如果 $f(n)=o(g(n))$ 且 $g(n)=o(h(n))$，则 $f(n)=o(h(n))$
  - 如果 $f(n)=\omega(g(n))$ 且 $g(n)=\omega(h(n))$，则 $f(n)=\omega(h(n))$
- Reflexivity (自反性):
  - 对于任何函数 $f(n)$，都有
    - $f(n)=\Theta(f(n))$
    - $f(n)=O(f(n))$
    - $f(n)=\Omega(f(n))$
- Symmetry (对称性):
  - 如果 $f(n)=\Theta(g(n))$，则 $g(n)=\Theta(f(n))$
- Transpose Symmetry (转置对称性):
  - 如果 $f(n)=O(g(n))$，则 $g(n)=\Omega(f(n))$
  - 如果 $f(n)=o(g(n))$，则 $g(n)=\omega(f(n))$
- Sum and maximum (和与最大值):
  - $f_1(n)+f_2(n)+\ldots+f_k(n)=\Theta\left(\max\left(f_1(n), f_2(n), \ldots, f_k(n)\right)\right)$，其中 k 是常数
  - 如果 k 不是常数，则上述性质不成立

运行时间层次（Running Time Hierarchy）：

1. 对数时间（Logarithmic）：
   - 复杂度：$O(logn)$
   - 描述：算法甚至不需要读取整个输入
   - 示例：二分查找算法
2. 线性时间（Linear）：
   - 复杂度：$O(n)$
   - 描述：算法只访问输入常数次
   - 示例：线性搜索算法
3. 线性对数时间（Linearithmic）：

- 复杂度：$O(nlogn)$
- 描述：算法将输入分成相似大小的两部分，分别解决并合并结果
- 示例：快速排序和归并排序

4. 二次时间（Quadratic）：
   - 复杂度：$O(n^2)$
   - 描述：算法考虑成对的元素
   - 示例：冒泡排序、选择排序和插入排序
5. 多项式时间（Polynomial）：
   - 复杂度：$O(n^\alpha)$（其中 α 为常数）
   - 描述：算法执行许多嵌套循环
   - 示例：一般多项式时间算法
6. 指数时间（Exponential）：
   - 复杂度：$O(c^n)$（其中 c 为常数）
   - 描述：算法考虑输入元素的许多子集
   - 示例：旅行商问题的蛮力解法

- 其他常见时间复杂度：
  - 常数时间（Constant）：$O(1)$
  - 超常数时间（Superconstant）：$\omega(1)$
  - 亚线性时间（Sublinear）：$o(n)$
  - 超线性时间（Superlinear）：$\omega(n)$
  - 超多项式时间（Superpolynomial）：$\omega(n^\alpha)$
  - 亚指数时间（Subexponential）：$o(c^n)$