算法的4个定义。

1. 如果存在正常数和使得当时，则记为。。表示，的增长率小于等于的增长率。大表示法。判断算法的时间复杂度时，经常用它表示。
2. 如果存在正常数和使得当时，则记为。的增长率大于等于的增长率。
3. 当且仅当和。的增长率等于的增长率。
4. 如果对于所有的常数存在使得当时，则记为。非正式的定义：且，则。的增长率小于的增长率。

当时，意味着函数是在以不快于的速度增长；因此是一个上界。也意味着，因此是的一个下界。

增长率的法则：

1. 如果且，那么
2. 
3. 
4. 如果是一个次多项式，那么。
5. 对于任意常数，。（对数如非特别指明，都是以2为底）

在大表示法中，不要将常数项和低阶项放入表达式中。

计算极限来确定和的相对增长率。

1. 极限是0，。
2. 极限是，。
3. 极限是∞，。
4. 极限摆动，二者无关。

就排序算法而言，不好的排序算法是。

算法分析，要分析的最重要的资源是运行时间。

：输入个数据所花费的平均情形的运行时间。

：输入个数据所花费的最坏情形的运行时间。

计算运行时间的一般法则：

1. for循环：一个for循环的运行时间至多是该for循环内语句的运行时间乘以迭代的次数。
2. 嵌套循环：从里向外分析这些循环。在一组嵌套循环内部的一条语句总的运行时间为该语句的运行时间乘以该组的所有循环的大小的乘积。

for (int i = 0; i != N; ++i)

{

for (int j = 0; j != N; ++j)

{

++k;

}

}

运行时间为。

1. 顺序语句：将各个语句的运行时间求和即可（各个语句运行时间的最大值就是所求的运行时间）。

for (int i = 0; i != N; ++i)

{

a[i] = 0;

} 运行时间为

for (int i = 0; i != N; ++i)

{

for (int j = 0; j != N; ++j)

{

++k;

}

} // 运行时间为。

总运行时间为：

在算法分析中，我们通常考虑下面两个情形：

：算法对输入所花费的平均情形；

：算法对输入所花费的最坏情形。

算法分析结果的约定：不存在特定的时间单位，抛弃低阶项，计算大运行时间，其是一个上界。