# 数据结构与算法

## 第一章

### 计算 ：借助某种工具，遵照一定规则，义明确而机械的形式进行

#### 对象：规律 技巧

#### 目标：高效 低耗

### 算法：特定计算模型下，旨在解决特定问题的指令序列

#### 输入；待处理的信息（问题）

#### 输出：经处理的信息（答案）

#### 正确性：的确可以解决指定的问题

#### 确定性：任一算法都可以描述为一个有基本操作组成的序列

#### 可行性：每一基本操作都可以实现，且在常数时间内完成

#### 有穷性：对于任何输入，经有穷次基本操作，都可以得到输出

##### 

### 计算模型

#### 度量：

##### 算法分析：

###### 正确性

###### 成本

#### TM：Turing Machine(图灵机)

##### Transition Function :(q,c;d,L/R,p)

###### 若当前状态为q且且当前字符为c，则将当前字符改为d；转向左侧/右侧的邻格；转入p状态；一旦转入特定的状态“h”，则停机；

#### RAM：Random Access Machine

##### 

#### 大O记号

##### T(n) = O( f(n) )

###### 常系数可忽略：O(f(n)) = O(c \* f(n))

###### 低次项可忽略：O(n^a + n^b) = O(n^a), a>b>0

##### O(1)

###### 常数

##### O()

###### 对数

##### O()

###### 多项式

一般得说，只看他的最高次项

##### O()

###### 指数

T(n) =

#### 其他记号

##### T(n) = Ω( f(n) )

##### T(n) =Θ( f(n) )

### 算法分析

#### 两个主要任务=正确性(不变性\*单调性)+复杂度

#### C++等高级语言的基本指令，俊等效有常数条RAM的基本指令；在渐进意义下，二者大体相当

##### 分支转向：goto //算法的灵魂；出于结构化考虑，被隐藏了

##### 迭代循环：for()、while（）、… //本质上就是“if + go”

##### 调用+递归（自我调用） //本质上也是goto

#### 复杂度分析的主要方法

##### 迭代：级数求和

##### 递归：递归跟踪+递推方程

##### 猜测+验证

#### 级数

##### 算数级数：与末项平方同阶

##### 幂方级数：比幂次高出一阶

##### 几何级数（a>1）：与末项同阶

##### 收敛级数

##### 

#### 循环 vs 级数

##### 算数级数：O()

##### 几何级数：O(n)

#### 取非极端元素

#### 起泡排序

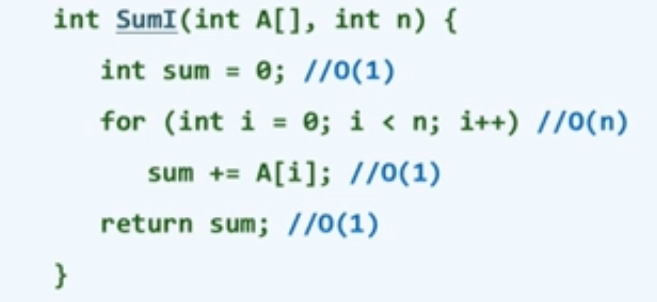
##### 不变性

##### 单调性

##### 正确性

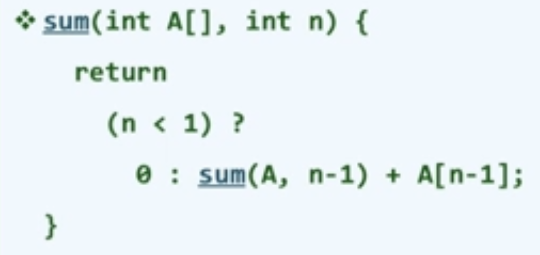
### 迭代与递归

#### 数组求和：迭代



时间复杂度 O（n）

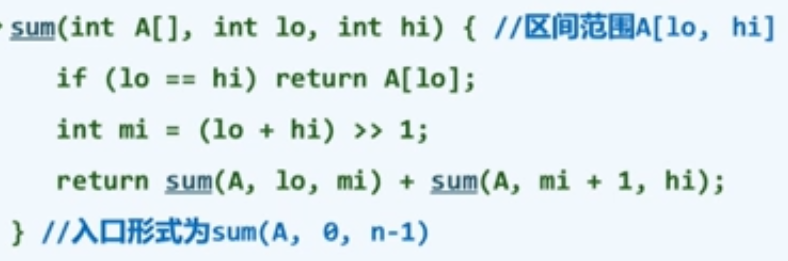
#### 数组求和：线性递归



时间复杂度：O（n）

#### 数组倒置

#### 数组求和：二分递归



### 动态规划