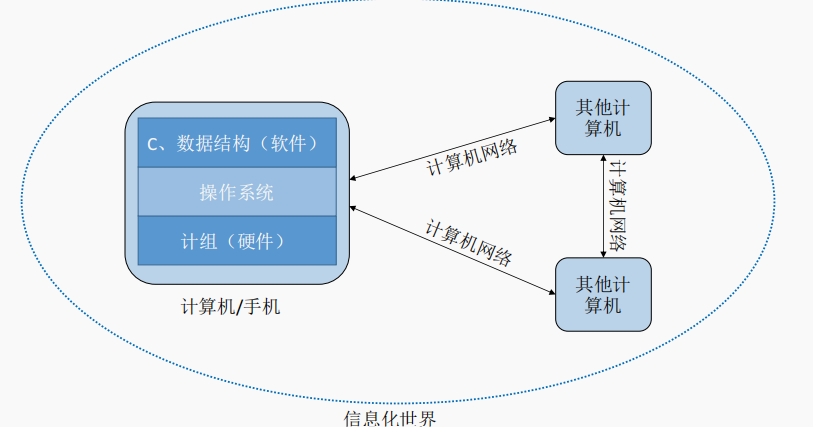
数据结构在学什么？

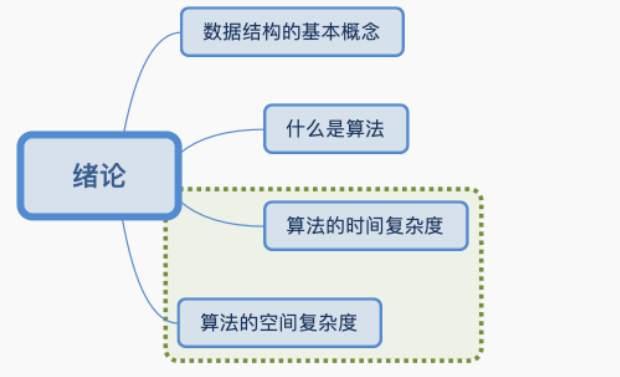
数据结构在学什么？

• 如何用**程序代码**把现实世界的**问题信息化**

• 如何用计算机**高效地处理**这些信息从而**创造价值**



绪论



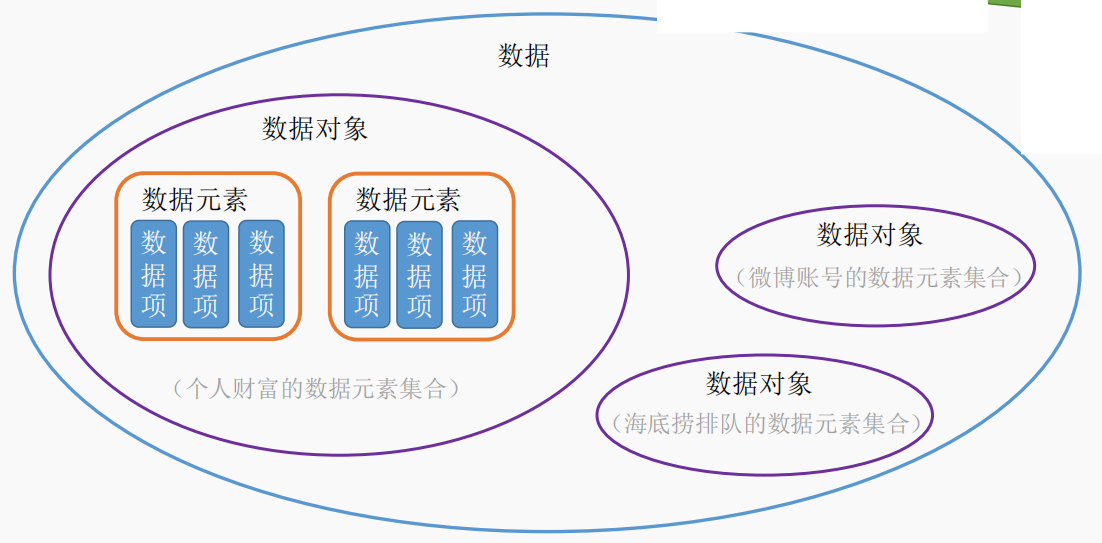
**数据**：数据是**信息的载体**，是描述客观事物属性的数、字符及所有能输入到计算机中并被**计算机程序识别和处理的符号**的集合。数据是计算机程序加工的原料。

**数据元素**：数据元素是数据的基本单位，通常作为一个整体进行考虑和处理，用于**描述一个个体**。

**数据项**：一个数据元素可由若干**数据项**组成，数据项是构成**数据元素的不可分割的最小单位**。

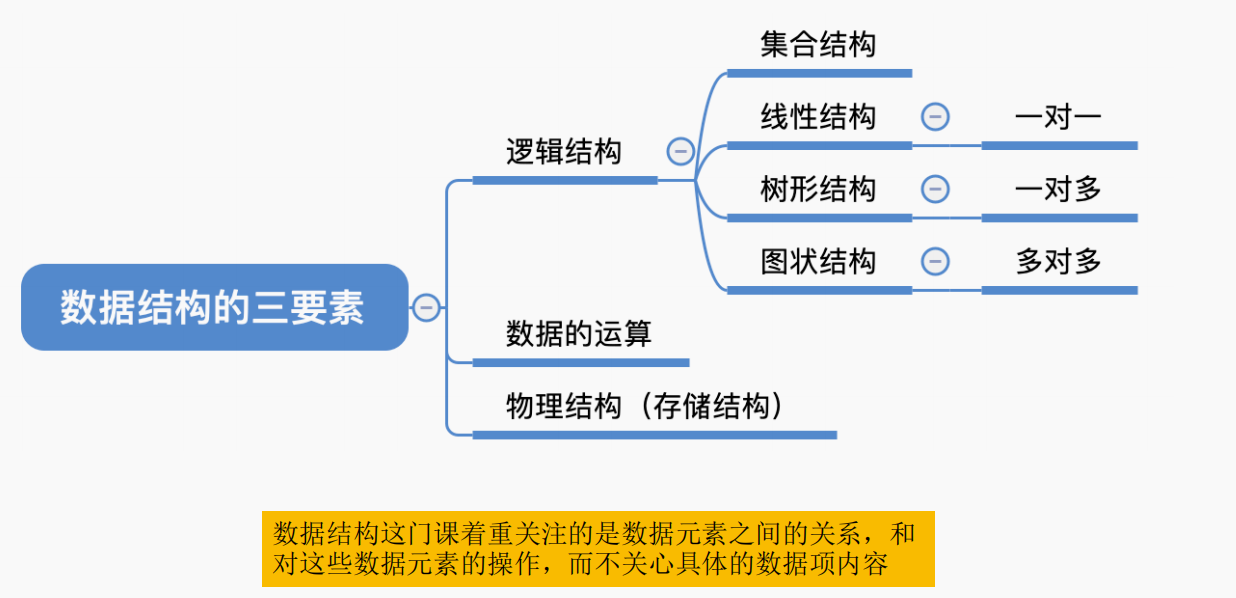
**数据对象**：数据对象是具有**相同性质**的数据元素的集合，是数据的一个子集。

**数据结构**：数据结构是相互之间存在**一种或多种特定关系**的数据元素的集合。同一个数据对象里的数据元素，可以组成不同的数据结构。



数据结构这门课着重关注的是**数据元素之间的关系**，和对这些数据元素的**操作**，而不关心具体的数据项内容。

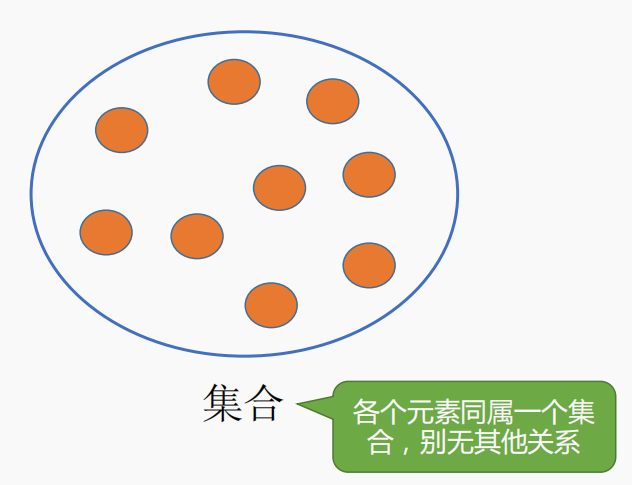
数据结构三要素



一、逻辑结构

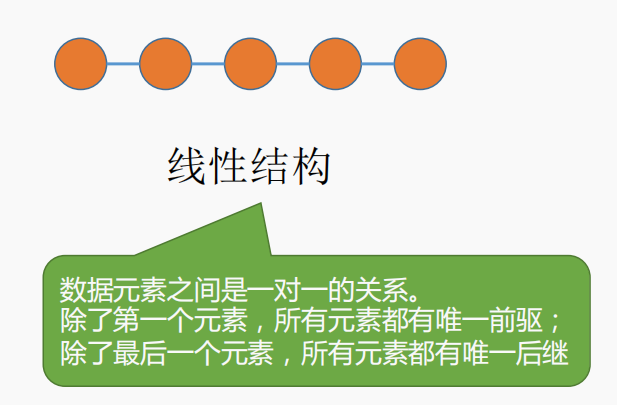
逻辑结构——数据元素之间的逻辑关系是什么？

1. 集合--各个元素同属一个集合，别无其他关系

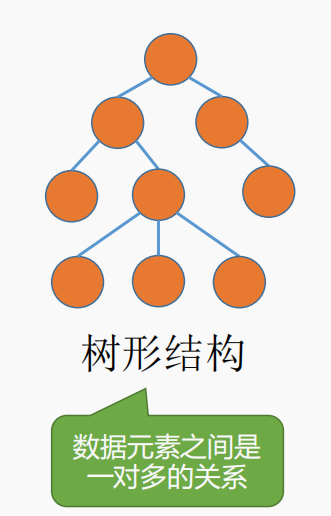


1. 线性结树--数据元素之间是**一对一**的关系。

除了第一个元素，所有元素都有**唯一前驱**；除了最后一个元素，所有元素都有**唯一后继**。



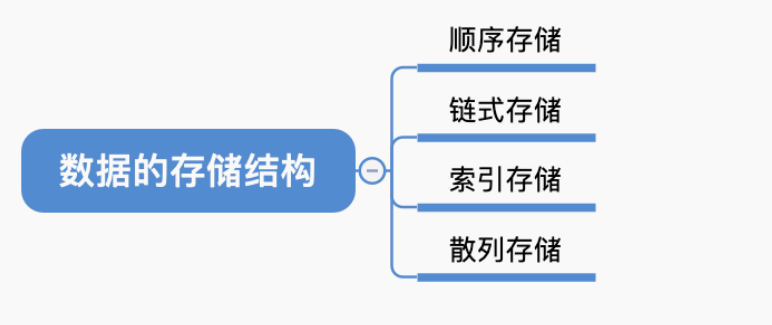
1. 树形结构--数据元素之间是**一对多**的关系



1. 图结构--数据元素之间是**多对多**的关系



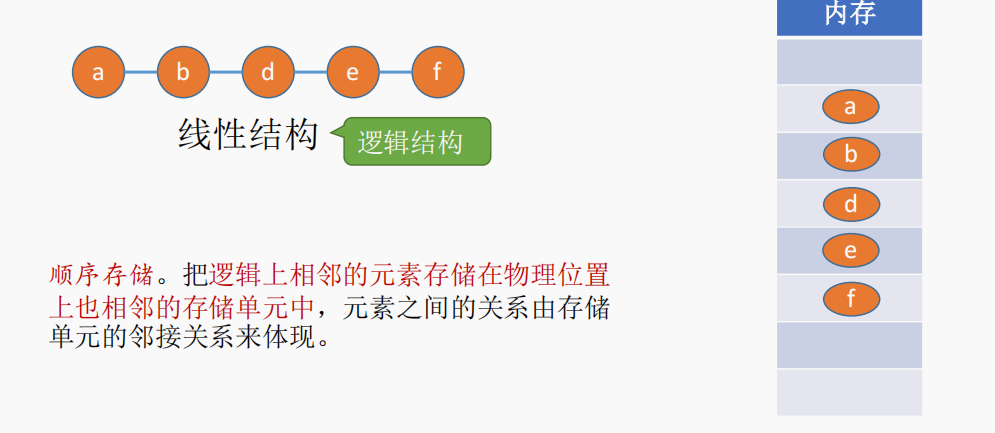
1. 数据的运算——针对于某种逻辑结构，结合实际需求，定义**基本运算**
2. 数据的物理结构（存储结构）——如何用**计算机表示**数据元素的**逻辑关系**？



1. 顺序存储：

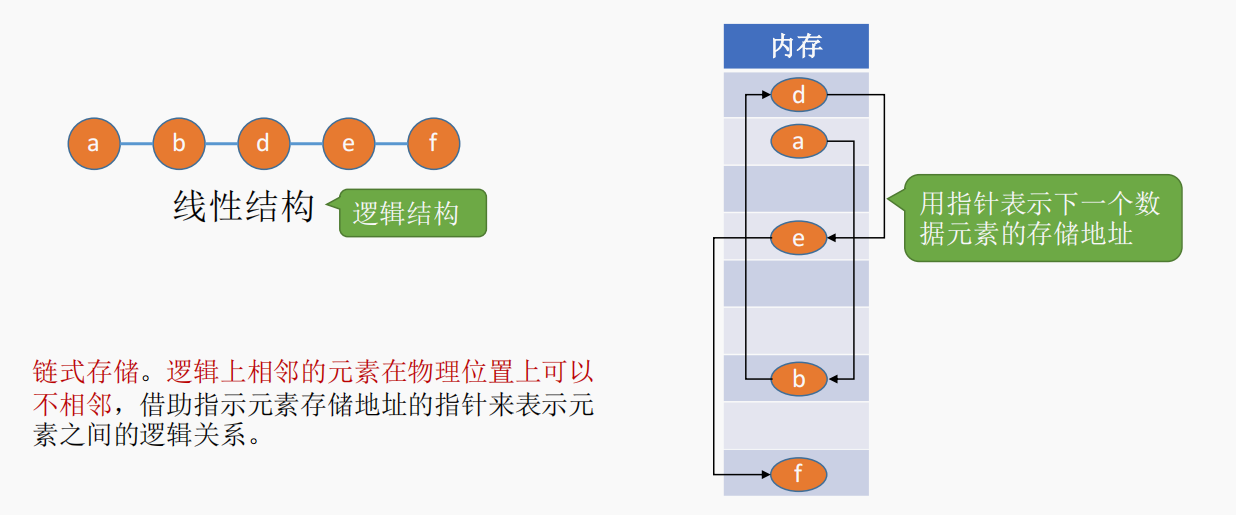
把**逻辑上相邻的元素存储在物理位置上也相邻的存储单元中**，元素之间的关系由存储

单元的邻接关系来体现。



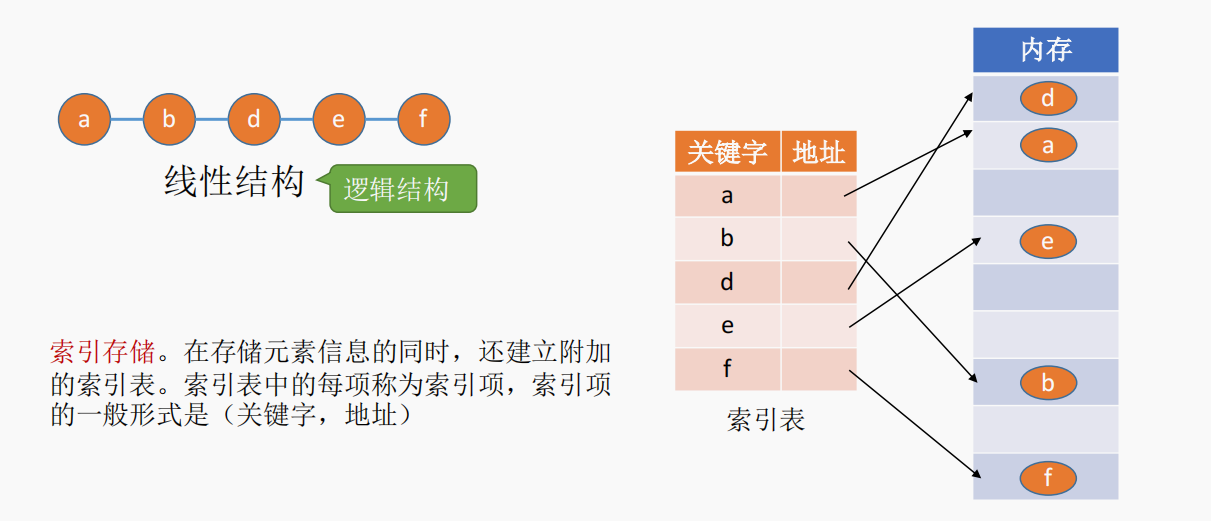
（2）链式存储：

**逻辑上相邻的元素在物理位置上可以不相邻**，借助指示元素存储地址的**指针**来表示元素之间的逻辑关系。



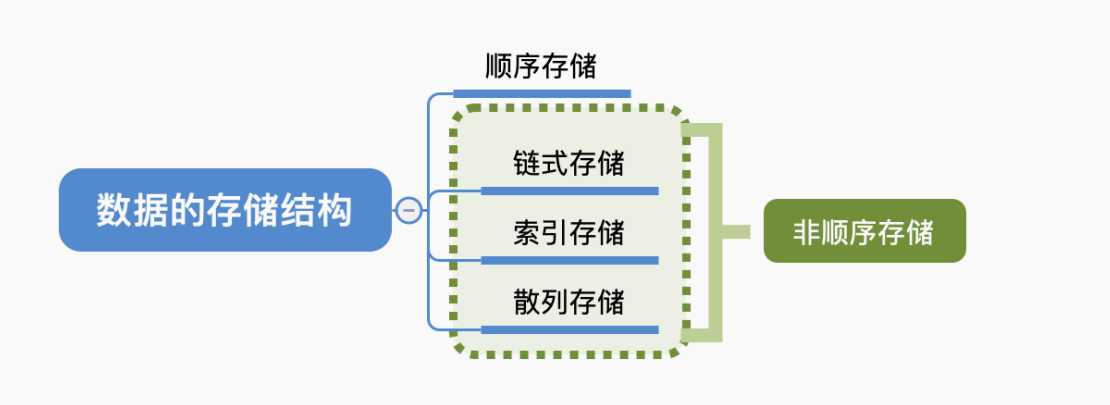
（3）索引存储：

在存储元素信息的同时，还建立**附加的索引表**。索引表中的每项称为索引项，索引项的一般形式是（**关键字，地址**）



（4）散列存储：

根据元素的**关键字直接计算出该元素的存储地址**，又称哈希（Hash）存储



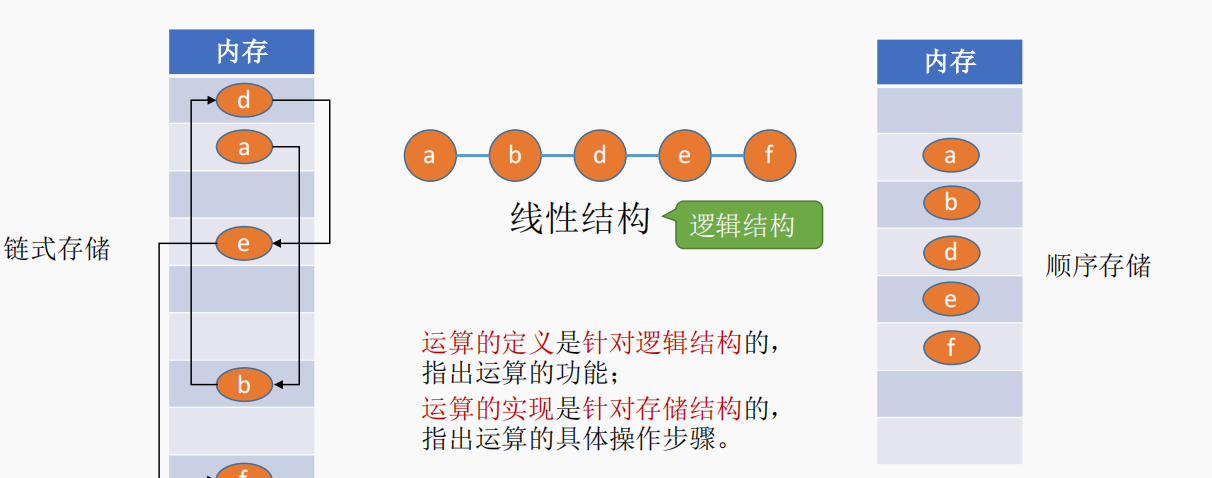
1. 若采用**顺序存储**，则各个数据元素在**物理**上必须是**连续**的；若采用非顺序存储，则各个数据元素在**物理**上可以是**离散**的。

2. 数据的存储结构会影响存储**空间分配的方便程度**

3. 数据的存储结构会影响**对数据运算的速度**

运算的**定义**是针对**逻辑结构**的，指出**运算的功能**；

运算的**实现**是针对**存储结构**的，指出**运算的具体操作步骤**。



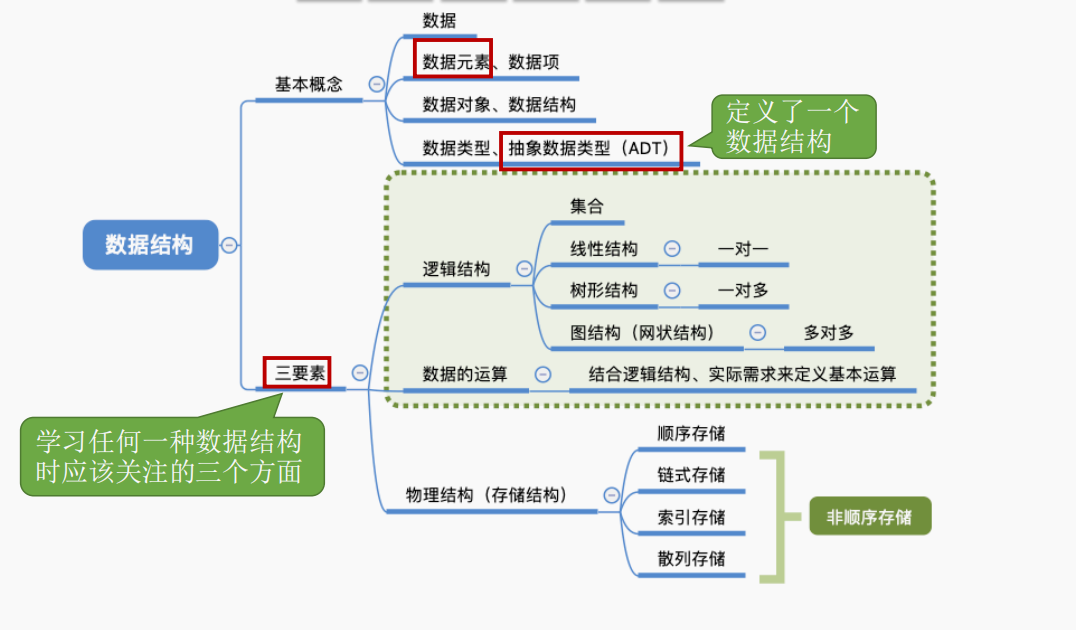
数据类型、抽象数据类型：

数据类型是一个值的集合和定义在此集合上的一组操作的总称。

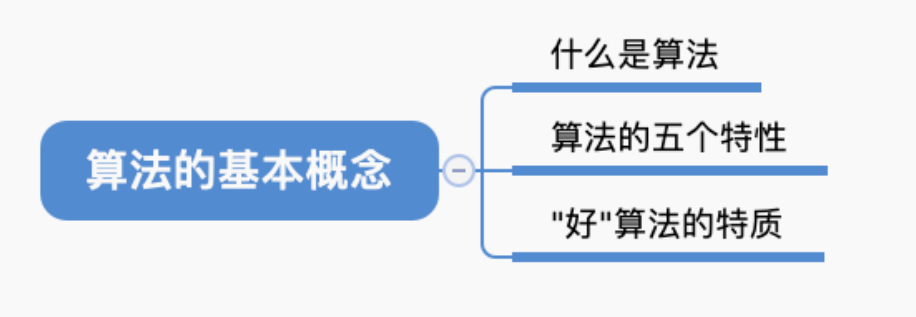
1）原子类型。其值不可再分的数据类型。

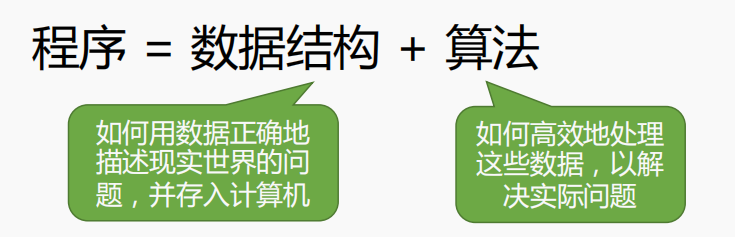
2）结构类型。其值可以再分解为若干成分（分量）的数据类型。

抽象数据类型（Abstract Data Type，ADT）是抽象数据组织及与之相关的操作。ADT 用数学化的语言定义数据的逻辑结构、定义运算。与具体的实现无关。



算法基本概念





算法（Algorithm）是**对特定问题求解步骤的一种描述**，它是指令的有限序列，其中的每条指令表示一个或多个操作

算法的特性：

（1）**有穷性**。一个算法必须总在执行有**穷步之后结束**，且每一步都可在**有穷时间内完成**。

（2）**确定性**。算法中每条指令必须有确切的含义，对于**相同的输入**只能得出**相同的输出**。

（3）**可行性**。算法中描述的操作都可以通过已经实现的**基本运算执行有限次**来**实现**。

（4）**输入**。一个算法有**零个或多个输入**，这些输入取自于某个特定的对象的集合。

（5）**输出**。一个算法有**一个或多个输出**，这些输出是与输入有着某种特定关系的量。

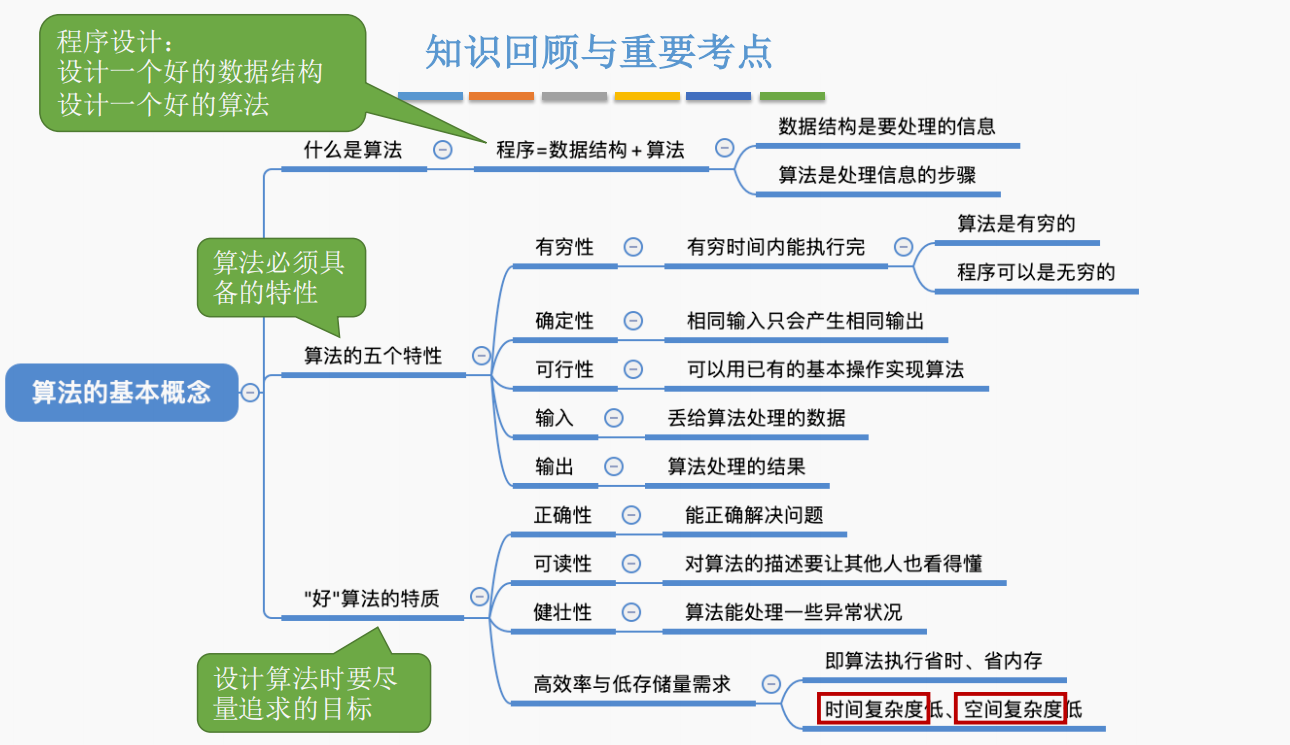
“好”算法的特质--设计算法时要尽量追求的目标

（1）**正确性**。算法应能够正确地解决求解问题。

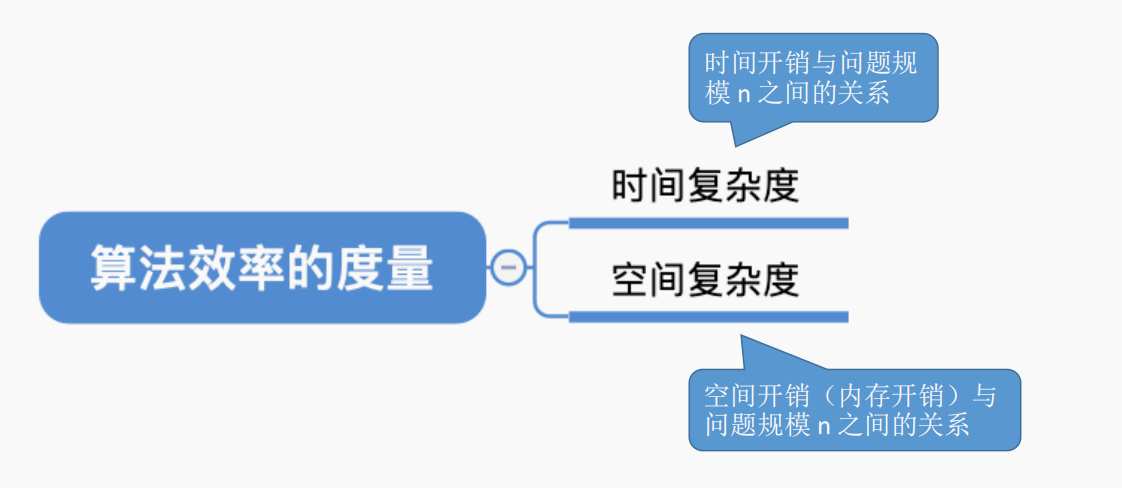
（2）**可读性**。算法应具有良好的可读性，以帮助人们理解。

（3）**健壮性**。输入非法数据时，算法能适当地做出反应或进行处理，而不会产生莫名其妙的输出结果。

（4）**高效率**与**低存储量需求（时间复杂度**和**空间复杂度）**



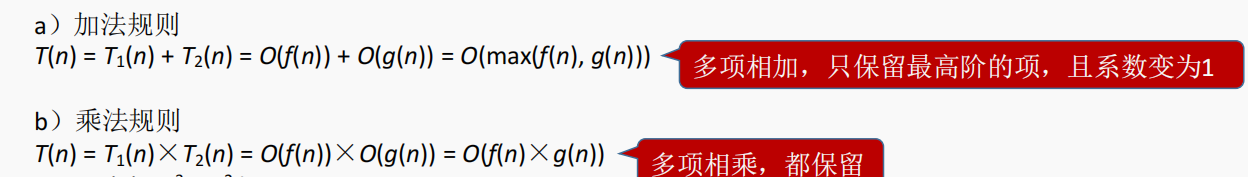
算法--效率的度量

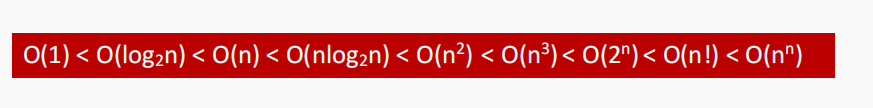


算法**时间复杂度**：事前预估**算法时间开销T(n)**与**问题规模n**的关系（T 表示 “time”）

结论1：可以只考虑**阶数高的部分**

结论2：问题规模足够大时，**常数项系数**也可以**忽略**





结论1：**顺序执行的代码**只会影响常数项，**可以忽略**

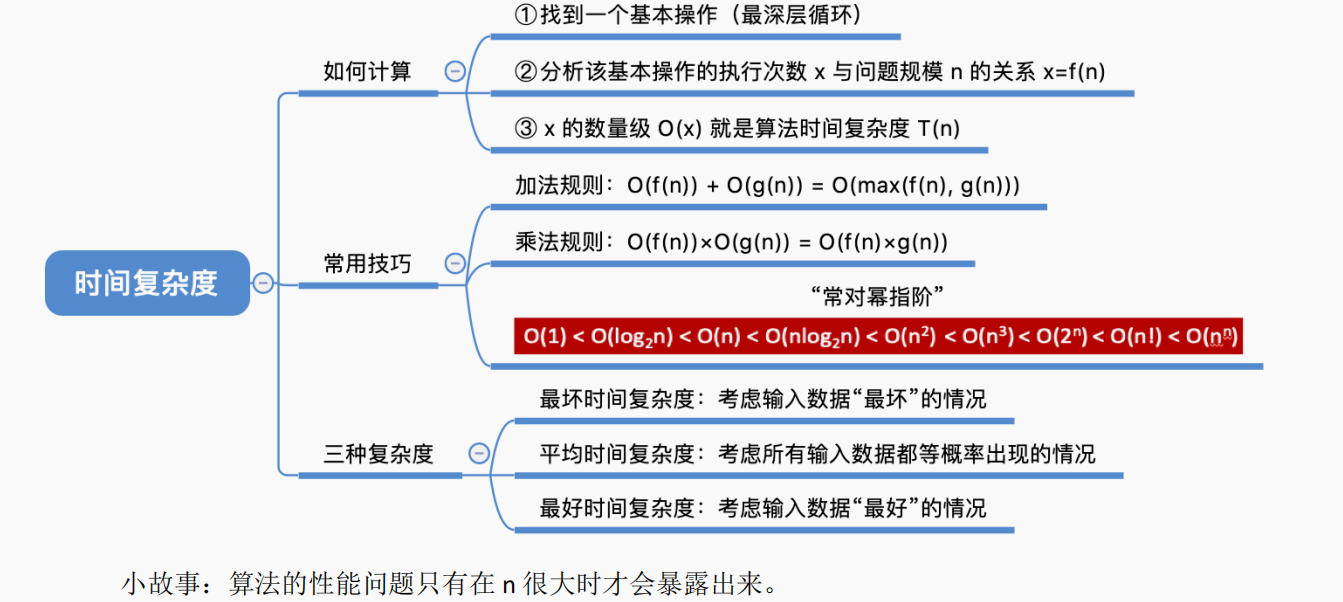
结论2：只需挑**循环中**的**一个基本操作**分析它的**执行次数与n的关系**即可

结论3：如果有**多层嵌套循环**，只需关注**最深层循环**循环了几次

**最坏时间复杂度**：最坏情况下算法的时间复杂度

**平均时间复杂度**：**所有输入示例等概率出现的情况**下，算法的期望运行时间

最好时间复杂度：最好情况下算法的时间复杂度

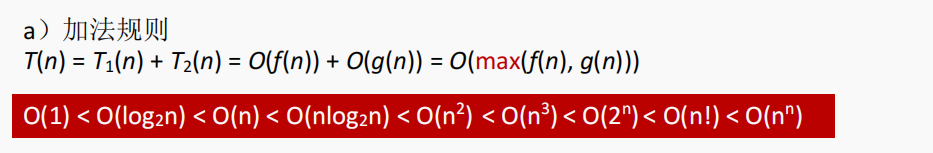


算法**空间复杂度：**

程序运行时的内存需求：

算法**原地工作**——算法所需内存空间为常量

只需关注**存储空间大小**与**问题规模相关**的变量



函数递归调用带来的内存开销：



