# 1.4 基本概念和术语

## 1.4.1 数据

是描述客观事物的符号，是计算机中可以操作的对象，是能被计算机识别，并输入给计算机处理的符号集合。

## 1.4.2 数据元素

是组成数据的有一定意义的基本单位，在计算机中通常作为整体处理，也被称为记录。

## 1.4.3 数据项

一个数据元素可以由诺干个数据项组成。（数据项是数据不可分割的最小单位）

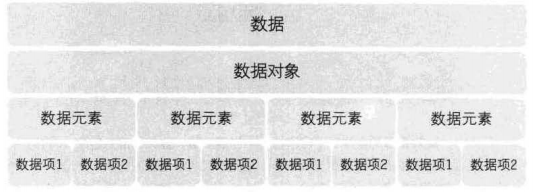
## 1.4.4 数据对象

是性质相同的数据元素的集合，是数据的子集。

## 1.4.5 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结合

结构：不同数据间存在着特定的关系，这种关系被称为结构



# 1.5 逻辑结构和物理结构

## 1.5.1逻辑结构

逻辑结构：是指数据对象中数据元素之间的相互关系。

逻辑结构分以下四种：

1. 集合结构

集合结构：集合结构中的数据元素除了同属一个集合外，它们之间没有其他关系。

1. 线性结构

线性结构：线性结构中的数据元素之间的关系是一对一的关系。

1. 树形结构

树形结构：树形结构中的数据元素之间存在一种一对多的层次关系。

1. 图形结构

图形结构：图形结构的元素是多对多的关系

用示意图表示数据的逻辑结构时，要注意两点：

* 将每个数据元素当作一个节点，用圆圈表示
* 元素之间的逻辑关系用节点之间的连线表示，如果这个关系是有方向的，需要用带头的连线表示

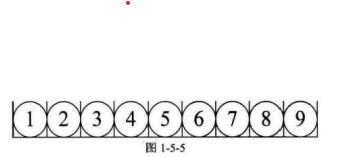
## 1.5.2 物理结构（存储结构）

物理结构：是指数据的逻辑关系在计算机中的存储形式。

数据的存储方式有两种：顺序存储和链式存储

### 1.5.2.1顺序存储结构

顺序存储结构：是把数据存放在地址连续的存储单元里，其数据间的逻辑关系和物理关系是一致的。



### 1.5.2.1链式存储结构

链式存储结构：是把数据元素放在任意的存储单元，这组存储单元可以是连续的也可以是不连续的。（用指针存放数据的地址，通过地址找到元素的位置）

a

逻辑结构是面向问题的，而物理结构是面向计算机的，其基本的目标就是将数据及其逻辑关系存储到计算机的内存中



# 1.6抽象数据类型

## 1.6.1 数据类型

数据类型：是指一组性质相同的值的集合及定义在此集合上的一些关系的总称

原子类型：是不可以再分解的基本类型，包括整型、实型、字符型等。

结构类型：由诺干个类型组合而成，是可以再分解的，例如整型数组是由诺干个整型组成的

抽象是指抽出事物具有的普遍性的本质。它是抽出问题的特征而忽略非本质的 细节，是对具体事物的一个概括。抽象是一种思考问题 方式，它隐藏了繁杂的细 节，只保留实现目标所必需的信息。

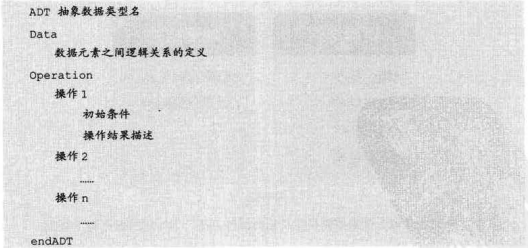
## 1.6.2 抽象数据类型

抽象数据类型（Abstract Data type, ADT）是指一个数据模型及定义在该模型上的一组操作。

"抽象'的 意义在 数据类型的数学抽象特性。不仅包括已存在的，还包括未实现的。

实际上，抽象数据类型体现了程序设计中的问题分解、抽象和信息隐藏的特性

描述抽象 数据类型的标准格式:



# 2.1 算法

算法：算法是解决特定问题求解步骤的描述，在计算机表现为指令的有限序列，并且每条指令表示一个或者多个操作

## 2.1.1算法的特性

算法是什么？

算法是解决特定问题求解步骤的描述，在计算机中表现为指令的有限序列，并且每条指令表示一个或者多个操作

算法的五个基本特性：输入、输出、有穷性、确认性、可行性。

输入输出:

算法具有零个或多个输入，至少有一个或多个输出

有穷性:

算法在执行有限的步骤之后，自动结束而不会出现无限循环，并且每一个步骤在可接受的时间内完成。

确定性:

算法的每一步骤都具有确定的含义，不会出现二义性

可行性:

算法的每一步都必须是可行的，也就是说，每一步都能够通过执行有限次数完成

## 2.1.2算法的设计要求

### 2.1.2.1 正确性

算法的正确性是指算法至少应该具有输入、输出和加工无歧义性、能正确反应问题的需求、能够得到问题的正确答案。

算法的“正确”在用法上的四个层次

1. 算法程序没有语法错误
2. 算法程序对于合法的输入数据能够产生满足要求的输出结果
3. 算法程序对于非法的输入数据能够得出满足规格的说明结果
4. 算法程序对于精心选择的，甚至刁难的测试数据都有满足要求的输出结果

### 2.1.2.2可读性

算法的另一目的是为了便于阅读、理解和交流

### 2.1.2.3健壮性

当输入数据不合法时，算法也能做出相关处理，而不是产生异常或莫名其妙的结果

### 2.1.2.4时间效率高和存储量低

算法的执行时间越短效率越高，算法在执行过程中所需最大内存空间越小的存储量越低

## 2.1.3算法效率的度量方法

### 2.1.3.1事后统计方法：

这种方法主要是通过设计好的测试程序和数据，利用计算机计时 器对不同算法编制的程序的运行时间进行比较，从而确定算法效率的高低

但这种方法显然是有很大缺陆的:

• 必须依据算法事先编制好程序，这通常需要花费大量的时间和精力 如果 编制出来发现宫根本是很糟糕的算法，不是竹篮打水一场空吗?

• 时间的比较依赖计算机硬件和软件等环境因素，有时会掩盖算法本身的优 要知道，现在的一台四核处理器的计算机， N~ 286 386 486 老爷爷辈的机器相比，在处理算法的 算速度上，是不能相提并论的 所用的操作系统、编译器、 行框架等软件的不同，也可以影响官们的结 果;就算是同一台机器， CPU 使用率和内存占用情况不 样，也会造成细 微的差异

• 算法的测试数据设计困难，并且程序的运行时间往往还与测试数据的规模 有很大关系，效率高的算法在才、的测试数据面前往往得不到体现 比如 10 个数字的排序，不管用什么算沽， 异几乎是零 而如果有 百万个随机 数字排序，那不同算法的差异就非常大了。那么我们为了比较算沽，到底 用多少数据来测试，这是很难判断的问题

基于事后统计 法有这样那样的缺陷，我们考虑不 采纳