## 什么是数据结构

先确定逻辑关系,再确定存储关系然后设定运算

### 概述

1968年美国克努特教授开创了数据结构的最初体系;   
数据结构指的是数据的逻辑结构和存储结构及其操作;   
数据结构是一门综合性的专业课程，是一门介于数学、计算机硬件、计算机软件之间的一门核心课程。是设计和实现编译系统、操作系统、数据库系统及其他系统程序和大型应用程序的基础。

### 数据结构

数据结构指的是数据元素及数据元素之间的相互关系，或组织数据的形式。有人认为：按照某种逻辑关系组织起来的一批数据，应用计算机语言，按照一定的存取方式把它们存储到计算机存储器中，并为这些数据定义一个运算集合，就称为一个数据结构。

1. 数据（Data）

数据即信息的载体,是能够输入到计算机中并且能被计算机识别,存储和处理的符号总称.

2. 数据元素（Data Element）

数据元素是数据的基本单位,又称为记录(Record),一般,数据元素由若干基本项(或称为字段,域,属性)组成.

数据项,是对数据的一种描述

3. 数据类型（Data Type）

对数据元素取值范围的限定

4. 数据结构（DS Data Structure）可用形式化语言描述，即DS是一个二元组：

DS = (D,R)

D为数据元素的集合,R为D上关系的集合

eg: 将向量v(a1,a2...,an)定义为一个DS,则v=(D,R),其中:

D={ai | ai∈datatype,i,2,...n,n≥0}

R={<ai,ai+1> | ai+1s∈D,1≤i≤n-1}

符号“<ai,ai+1>”表示向量v中任意两相邻元素的先后次序关系，在这里称为有序对。如向量v=(1,3,5,7,9)，对应的D，R为:

D={2i+1 | i∈整数,i=0,1,2,,3,4}

R={<1,3>,<3,5>,<5,7>,<7,9>}

## 逻辑结构和存储结构

逻辑结构：表示数据运算之间的抽象关系（如邻接关系、从属关系等），按每个元素可能具有的直接前趋数和直接后继数将逻辑结构分为“线性结构”和“非线性结构”两大类。

存储结构：逻辑结构在计算机中的具体实现方法，分为顺序存储方法、链接存储方法、索引存储方法、散列存储方法。

数据运算：对数据进行的操作，如插入、删除、查找、排序等。

### 逻辑结构

1)线性结构 一对一

对于数据结构课程而言，简单地说，线性结构是n个数据元素的有序（次序）集合。

* + 集合中必存在唯一的一个"第一个元素”；
  + 集合中必存在唯一的一个"最后的元素”；
  + 除最后元素之外，其它数据元素均有唯一的"后继”；
  + 除第一元素之外，其它数据元素均有唯一的"前驱”。

2)树形结构（层次结构） 一对多

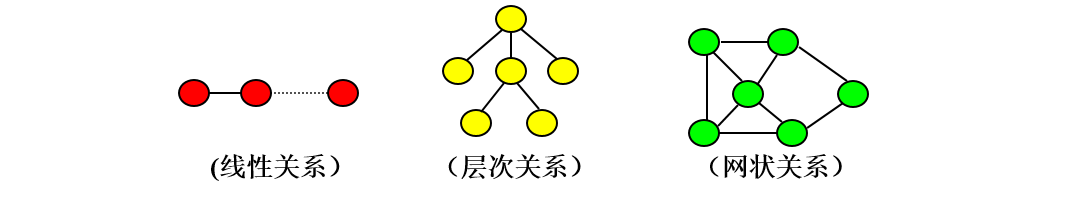
树形结构指的是数据元素之间存在着“一对多”的树形关系的数据结构，是一类重要的非线性数据结构。   
在树形结构中，树根结点没有前驱结点，其余每个结点有且只有一个前驱结点。叶子结点没有后续结点，其余每个结点的后续节点数可以是一个也可以是多个。

3)图状结构（网状结构） 多对多

图是一种比较复杂的数据结构。在图结构中任意两个元素之间都可能有关系，也就是说这是一种多对多的关系。

其他结构

除了以上几种常见的逻辑结构外，数据结构中还包含其他的结构，比如集合等。有时根据实际情况抽象的模型不止是简单的某一种，也可能拥有更多的特征。



### 存储结构

数据的**存储结构**指的是数据的**逻辑结构**在计算机存储器中的映像(或表示),存储结构是通过计算机语言所编制的程序来实现的,因而是以来与具体的计算机语言的.

顺序存储

顺序存储（Sequential Storage）：将数据结构中各元素按照其逻辑顺序存放于存储器一片连续的存储空间中。

链式存储

链式存储（Linked Storage）：将数据结构中各元素分布到存储器的不同点，用地址（或链指针）方式建立它们之间的联系，由此得到的存储结构为链式存储结构。

索引存储

索引存储（Indexed Storage）：在存储数据的同时，建立一个附加的索引表，即索引存储结构=数据文件+索引表。

散列存储

散列存储(Hash Structure)： 根据数据元素的特殊字段(称为关键字key)，计算数据元素的存放地址，然后数据元素按地址存放，所得到的存储结构为散列存储结构(或Hash结构)。

### 运算

数据元素包含的许多值是非数值性的（如文字、日期等数据），对其进行的操作（或运算）也不再是加、减、乘、除等数学运算，而是诸如：查询（查找一条记录），插入（增加一条记录），修改（修改元素中的某些信息），删除（删除一条记录），分类（按某数据项的值建立索引）等这样的运算。

## 算法基础

### 算法

瑞士科学家沃思（N.Wirth）的著名公式：

数据结构 + 算法 = 程序

数据的逻辑结构与存储结构密切相关:

算法设计: 取决于选定的逻辑结构

算法实现: 依赖于采用的存储结构

### 算法基础

算法(Algorithm)是一个有穷规则(或语句,指令)的有序集合,确定了解决某一个问题的运算序列,对于问题的初始输入,通过算法有限步骤的运行,产生一个或多个输出.

**算法的特性**

* **有穷性** —— 算法执行的步骤（或规则）是有限的；
* **确定性** —— 每个计算步骤无二义性；   
  **可行性** —— 每个计算步骤能够在有限的时间内完成；
* **输入** —— 算法有一个或多个外部输入；
* **输出** —— 算法有一个或多个输出。

**评价算法好坏的方法**

* **正确性**：运行正确是一个算法的前提。
* **可读性**：容易理解、容易编程和调试、容易维护。
* **健壮性**：考虑情况全面，不容易出现运行错误。
* **时间效率高**：算法消耗的时间少。
* **储存量低**：占用较少的存储空间。

**时间复杂度计算**   
 **算法效率**——用依据该算法编制的程序在计算机上执行所消耗的时间来度量。（算法时间是由控制结构和原操作的决定的）。记号——引用了“O”。“O”表示一个数量级的概念。根据算法中语句执行的最大次数（频度）来 估算一个算法执行时间的数量级。

**时间复杂度**：算法中基本操作重复执行的次数是问题规模n的某个函数f（n），

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

它表示随问题规模n的增大，算法执行时间的增长率和f（n）的增长率相同。

语句的频度：是该语句重复执行的次数。

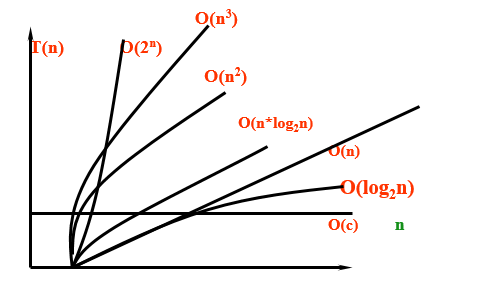
时间复杂度计算方法:

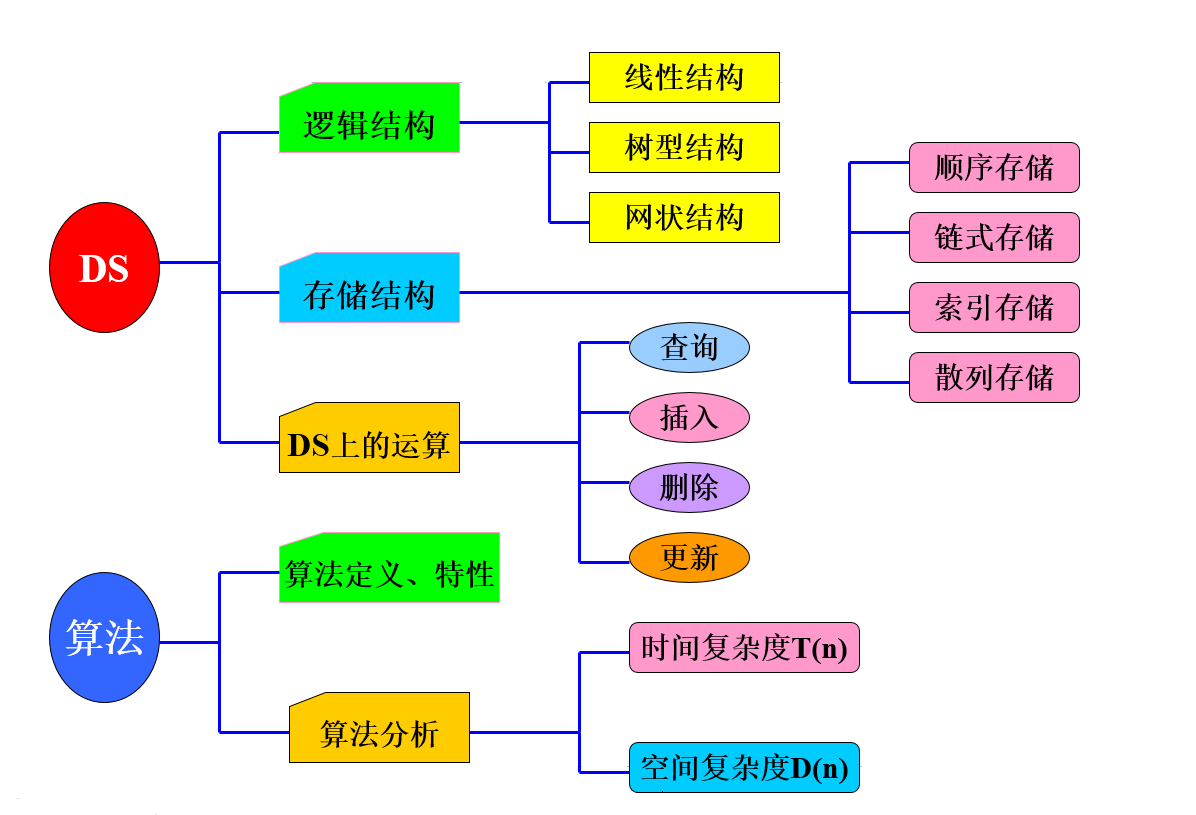
1) 将表达式中每一步骤需要运行的次数进行加和,

2) 如果没有不定项,那么时间复杂度为常量级别,记为O(1)

3) 如果有不定项,则使用最高项,其他项数不计,

4) 如果有不定项,最高次幂的系数不计,剩余的为时间复杂度





备注:空间复杂度 S(n)

## 线性表

线性表是信息表的一种形式，表中数据元素之间满足线性关系（或线性结构），是一种最基本、最简单的数据结构类型。线性表的定义是描述其逻辑结构，而运算是讨论在线性表上进行的查找、插入、删除等操作。   
 线性表作为一种基本的数据结构类型，在计算机存储器中的映象（或表示）一般有两种形式，一种是顺序映象，一种是链式映象。

**我们通常对一个数据模型的分析也是从逻辑结构、存储结构、运算三个部分来进行分析。**

## 线性表的顺序存储

若将线性表L=(a0,a1, ……,an-1)中的各元素依次存储于计算机一片连续的存储空间，这种机制表示为线性表的顺序存储结构。

### 顺序存储结构的特点：

1. 逻辑上相邻的元素 ai, ai+1，其存储位置也是相邻的；   
2. 对数据元素ai的存取为随机存取或按地址存取。   
3. 存储密度高。

**顺序存储的优点:**

1 容易查找,只要知道其任何一个元素的位置,就可以知道其他元素的位置

2 存储密度高

3 在需要频繁查找和遍历时,采用顺序存储

### 顺序存储结构的不足：

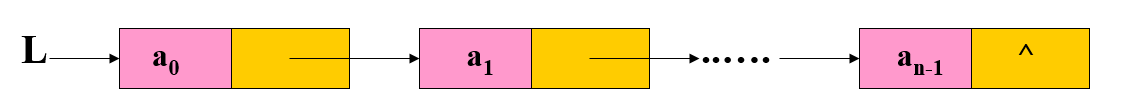
不利于在中间部分频繁的进行插入和删除操作

在Python中，list存放于一片单一连续的内存块，故可借助于列表类型来描述线性表的顺序存储结构，而且列表本身就提供了丰富的接口满足这种数据结构的运算。Python的源码是用c语言实现的，内部接口的实现过程并不是我们关注的重点，事实上在Python中也很少讨论内存空间的问题。我们在这里只是对这种“现成”的内建数据结构使用即可。

eg: append insert remove 修改, index

## 线性表的链式存储

相对于顺序存储，链式存储更加接近我们传统意义的列表，通常也叫做链表。   
将线性表L=(a0,a1,……,an-1)中各元素分布在存储器的不同存储块，称为结点，每个结点（尾节点除外）中都持有一个指向下一个节点的引用，这样所得到的存储结构为链表结构。



红色:数字域 黄色:指针域(或地址域)