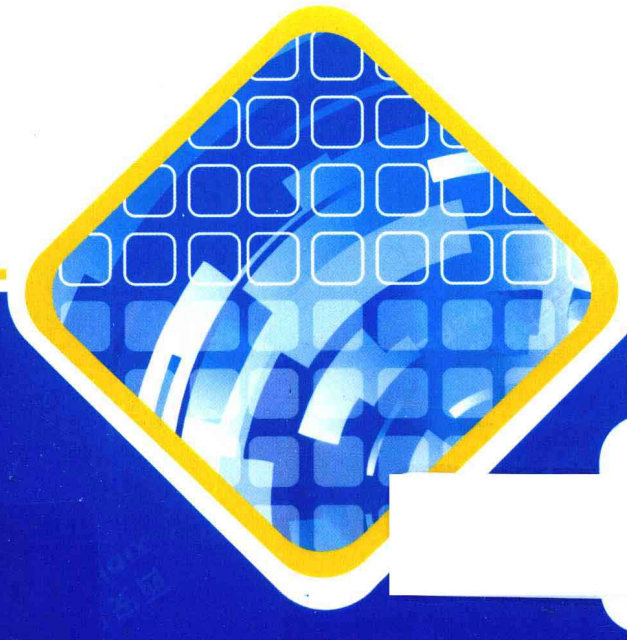
高职高专计算机专业“十二五”规划教材

数据结构

主　编　胡大威
副主编　祁淑霞　陈兴无　于谦



puter
program

西安电子科技大学出版社
http://www.xduph.com

XDUP 326200

封面设计：佳易传播 WWW.SXJYCB.COM

高职高专计算机专业“十二五”规划教材

- 计算机应用基础
- 软件技术基础
- 数据结构
- 计算机操作系统
- 计算机专业英语
- 计算机网络技术基础教程
- 微机硬件组装与维护教程
- 计算机组装与维护实训教程
- 计算机常用软件使用教程
- 计算机数据恢复技术
- C语言程序设计案例教程
- C语言程序设计教程
- C语言程序设计习题与实验指导
- C++程序设计教程
- Visual Basic程序设计案例教程
- Visual Basic.NET程序设计
- Delphi程序设计应用教程
- Java语言程序设计案例教程
- Java程序设计项目化教程
- Visual C#.NET案例教程
- 软件工程与项目管理
- 软件编码规范与文档编写
- 嵌入式软件开发
- 嵌入式系统实例教程
- ERP项目管理与实施
- 电子政务规划与建设
- 计算机网络设备与配置
- 综合布线技术
- 网络安全技术
- 计算机网络安全基础与技能训练
- Linux网络操作系统应用教程
- Windows Server 2003组网技术
- Windows Server 2003组网实例教程
- 网络故障恢复
- 数据库原理与技术
- Access数据库应用技术
- SQL Server数据库应用基础
- SQL Server数据库系统应用技术
- Oracle数据库实用技术
- Oracle数据库应用教程
- 基于C#的ASP.NET应用程序设计
- Java Web程序设计基础教程
- JSP程序设计实用案例教程
- XML案例教程
- 网页设计与制作——三剑客综合实例运用
- 网页设计与制作实例教程
- Flash 8动画制作案例教程
- 平面设计
- 三维动画案例教程
- AutoCAD 2007实用教程
- 多媒体技术与应用
- 多媒体软件开发

ISBN 978-7-5606-2970-4

9 787560 629704

定价：24.00元

高职高专计算机专业“十二五”规划教材

数据结构

主　编　胡大威

副主编　祁淑霞　陈兴无　于　谦

西安电子科技大学出版社

内 容 简 介

本书系统地介绍了各种典型数据结构的基本概念、存储结构及其各种运算的原理和算法。全书共分11章，包括绪论、线性表、栈和队列、串、数组和广义表、树、图、查找、内部排序、外部排序和文件以及实习部分等。本书的内容安排适当、浅显易懂；各章中的“基本内容”和“学习要求”可以引导读者在阅读时抓住重点；所附习题题型多样、难易恰当，便于读者理解和掌握数据结构的内容。书中介绍了丰富的算法，大部分采用C语言描述，并给出了部分完整的程序供读者上机时参考。

本书可用作高等职业技术学院计算机及其相关专业的教材，也可供从事计算机应用的技术人员及自学者参阅。

图书在版编目(CIP)数据

数据结构/胡大威主编. —西安：西安电子科技大学出版社，2013.1
ISBN 978-7-5606-2970-4

Ⅰ. ① 数… Ⅱ. ① 胡… Ⅲ. ① 数据结构 Ⅳ. ① TP311.12

中国版本图书馆 CIP 数据核字(2012)第 282216 号

策　　划　陈婷
责任编辑　陈婷　曹锦
出版发行　西安电子科技大学出版社(西安市太白南路 2 号)
电　　话　(029)88242885　88201467　　邮　　编　710071
网　　址　www.xduph.com　　电子邮箱　xdupfxb001@163.com
经　　销　新华书店
印刷单位　陕西天意印务有限责任公司
版　　次　2013 年 1 月第 1 版　2013 年 1 月第 1 次印刷
开　　本　787 毫米×1092 毫米　1/16　　印　　张　15
字　　数　353 千字
印　　数　1～3000 册
定　　价　24.00 元

ISBN 978-7-5606-2970-4/TP

XDUP　3262001-1

前　言

“数据结构”是介于数学、计算机软/硬件之间的一门计算机学科的核心专业课程，主要研究如何合理地组织大量数据，如何在计算机中更有效地存储和处理数据，以及如何设计正确而高效的算法。数据结构是计算机程序设计、数据库、操作系统、编译原理及人工智能等领域的重要基础，同时也广泛地应用于信息学、系统工程等领域。数据结构不仅为程序设计提供了方法论性质的指导，而且在更高层次上总结了现实生活中多种数据的计算机处理方法。“数据结构”是一门实用性很强的课程，掌握其内容不但要在理论上打好基础，更要在实践上多下功夫。

随着计算机加工和处理的对象从简单的数值发展到一般的符号，进而发展到具有一定结构的数据，数据结构也相应地由简单变得越来越复杂。本书介绍各种常用数据结构的基本概念和逻辑特征，讨论它们在计算机中的存储表示，定义了在这些数据结构上的运算，并对算法的效率进行了简要的分析。

全书共分 11 章，第 1 章介绍了数据结构和算法的基本概念；第 2～4 章介绍了线性表、栈和队列及串等几种基本的线性结构；第 5 章介绍了数组和广义表；第 6、7 章介绍了树和图这两种非线性结构；第 8、9 章介绍了数据处理中广泛使用的技术——查找和内部排序；第 10 章对外部排序和文件作了简要的介绍；第 11 章实习部分给出了实习指导、实习题和一个数据结构的应用实例——迷宫问题，以便读者上机实习。

本书的内容取舍适当、浅显易懂，既注重原理又重视实践，加大了算法实现和实习的分量。每章中的“基本内容”和“学习要求”可以引导读者在学习本章内容时抓住重点，所介绍的算法大部分采用 C 语言描述成可直接上机执行的程序，并作了较详细的注释，有利于读者理解算法的实质内容和基本思想。书中的习题题型多样、题量丰富、难易恰当，便于学生理解和掌握数据结构的内容。

书中带“*”部分可以酌情作为选修内容，全书建议教学参考学时为 64 学时。

本书第 1、2、3、4、6、8、11 章由胡大威编写，第 5、7 章由陈兴无编写，第 9、10 章由祁淑霞编写，于谦参与了部分章节的编写。全书由胡大威统稿。

限于编者水平和经验，书中不足之处在所难免，惟望广大读者不吝赐教。联系邮箱：hdw9678@sina.com。

作　者

2012 年 6 月

目　录

第1章 绪 论

基本内容：

本章将介绍数据结构的基本概念、基本术语、算法及算法时间复杂度的分析方法。

学习要求：

(1) 熟悉各种基本术语、数据结构的逻辑结构与存储结构、数据结构的四种基本存储方法，特别要了解数据的逻辑结构与存储结构、数据结构与数据类型之间的区别。

(2) 了解算法的性质、算法与程序的区别和联系、数据结构与算法之间的关系。

(3) 掌握算法的书写规范。

(4) 结合实例了解估算算法时间复杂度的方法。

数据结构在计算机科学中有着十分重要的地位，它不仅是一般程序设计的基础，而且是编译原理、数据库系统、操作系统等专业课程的重要基础，并且还为软件开发和程序设计提供必需的技能训练。在众多的计算机系统软件和应用软件中都要用到各种数据结构，例如，操作系统中的存储管理、作业调度要用到链表及队列等结构，数组、链表及查找和排序算法在数据库技术中都得到了直接的应用；而文件的组织方式、检索技术则是操作系统和数据库的重要组成部分。

在计算机发展的初期，计算机主要用于数值计算，所处理的对象是纯数值性的信息，程序设计者无需重视数据结构。随着计算机软/硬件的飞速发展和应用领域的不断扩大，计算机所处理的对象也从简单的纯数值信息逐步发展到字符、图形、图像、声音等各种复杂且具有一定结构的非数值性数据，并且它们在实际应用中所占的比例越来越大。据统计，如今计算机中90%以上的机器时间都是用来处理这类非数值性数据信息的。新的计算机应用领域的不断扩大，要求人们不断探索新的数据表示形式，从而使数据结构的内容得到进一步丰富和发展。

为了设计出效率高、可靠性好的程序，人们不但要掌握一般的程序设计方法，而且还必须研究计算机程序加工的对象，即研究各种数据的特性及数据之间的关系。数据是现实世界中的对象的某些特性和特征的部分抽象。针对每一种新的应用领域的处理对象，如何选择合适的数据表示形式，如何有效地组织计算机存储，在此基础上又如何有效地实现对象之间的运算关系，就是数据结构要研究和解决的问题。

1.1 基本术语和基本运算

1.1.1 基本术语

在大多数情况下，计算机程序处理的信息(数据)之间往往具有重要的结构关系，这就

是数据结构研究的内容。

以电话号码查询系统为例，设有一个电话号码簿，它记录了 N 个人的名字和其对应的电话号码，假定按如下形式安排：$(a_1, b_1)(a_2, b_2)\cdots(a_n, b_n)$，其中 a_i、$b_i(i=1, 2, \cdots, n)$ 分别表示某人的名字和对应的电话号码。要求设计一个算法，当给定任何一个人的名字时，该算法能够打印出此人的电话号码，如果该电话簿中没有此人，则该算法能够报告查无此人。

上述问题就是一种数据结构问题。算法的设计，依赖于计算机如何存储人的名字和对应的电话号码，或者说依赖于名字和其电话号码的结构。此例中，可将名字和对应的电话号码设计成二维数组、表结构、向量等。而采用不同的数据结构，将直接影响对算法的选择和执行的效率。

1. 数据

数据是承载信息的物理符号。描述现实对象的数、字符以及所有能输入到计算机中进行加工、处理的信息统称为数据，它是计算机程序加工的“原料”，其含义极为广泛。

2. 数据元素

数据元素是数据的基本单位。它可以是单一数据项，也可以是若干逻辑上有联系的相关数据项的组合，数据项是数据不可分割的最小单位。

数据元素视不同场合有其他同义语，在顺序结构中称为元素，在链式结构中称为结点，在图中称为顶点，在文件中称为记录，等等。

3. 数据对象

数据对象是具有相同性质的数据元素的集合，是现实世界实体对象的数据抽象，是数据的一个子集。例如，为了描述数据化的“学生”这个实体对象，假定姓名、学号、年龄、性别、系别等这些数据项是你所关心的，那么，某一具体的学生姓名“张三”、学号“995341”、年龄“20”、性别“男”、系别“计算机系”就是构成数据对象“学生”的一个数据元素。

4. 数据结构

1) 数据结构的概念

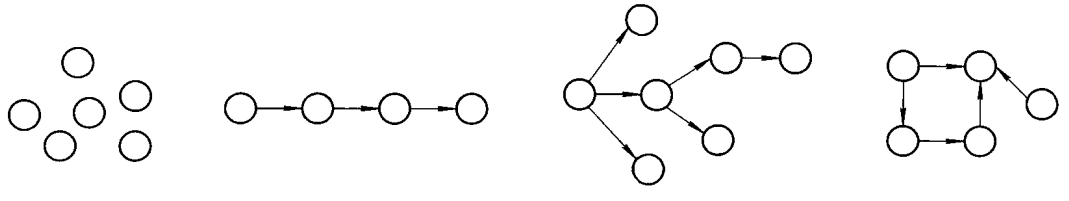
数据对象中的数据不是孤立的，而是彼此相关的，数据元素相互之间的关系称为结构。根据数据元素之间关系的不同特征，通常有以下四类基本数据结构。

(1) 集合：结构中的数据元素除了同属于一种类型外，别无其他关系，如图 1-1(a)所示。

(2) 线性结构：结构中的数据元素之间存在一对一的关系，如图 1-1(b)所示。

(3) 树型结构：结构中的数据元素之间存在一对多的关系，如图 1-1(c)所示。

(4) 图状结构或网状结构：结构中的数据元素之间存在多对多的关系，如图 1-1(d)所示。



(a) 集合　(b) 线性结构　(c) 树型结构　(d) 图状(或网状)结构

图 1-1　四类基本结构关系图

将相互间存在一种或多种特定关系的数据元素的集合称为**数据结构**，即数据结构是带有结构的元素的集合。虽然至今尚未有一个关于数据结构的标准定义，但直观上可以理解为：数据结构是一门研究在程序设计中计算机操作的对象以及它们之间的关系和运算的学科。它包含了三个方面的内容：① 研究计算机需要处理的数据对象和对象之间的关系；② 刻画应用中数据的逻辑组织；③ 描述数据在计算机中存储、传送和转换的方式。

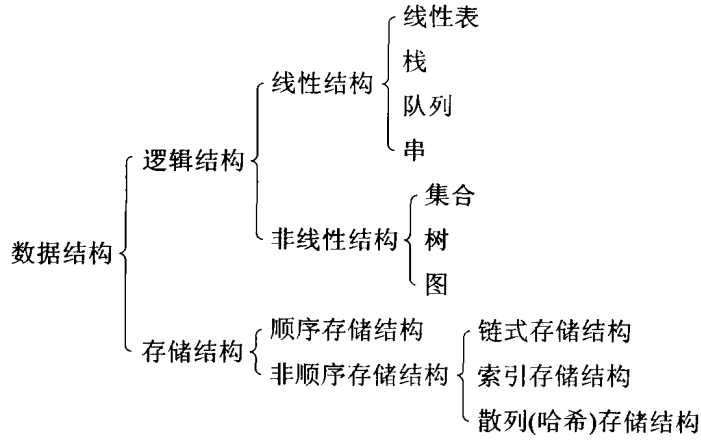
数据结构研究的是数据元素之间的逻辑关系(逻辑结构)和这种关系在计算机中的存储表示(存储结构)，并对每种结构定义各自的运算和设计出相应的算法。

数据结构不仅包含数据的逻辑结构和数据的存储结构，而且还包含数据的运算，所以应该将数据的逻辑结构、数据的存储结构和数据的运算看成一个整体。

数据结构和数据对象是不同的，在描述一种数据结构时，不但要描述数据对象，还要描述数据元素之间的关系。

2) *数据结构的分类*

数据结构按其内容可作如下分类：



线性结构的逻辑特征是：如果结构是非空集，则有且仅有一个开始结点和一个终端结点，且每一个结点最多只有一个直接前驱和一个直接后继。

非线性结构的逻辑特征是：一个结点可能有多个直接前驱和直接后继。

顺序存储结构是把逻辑上相邻的结点存储在物理上相邻的存储单元中，结点的逻辑关系由存储单元的邻接关系来体现。其特点是借助元素在存储器中的相对位置来表示数据元素之间的逻辑关系。通常用高级语言中的数组类型来描述顺序存储结构。

链式存储结构是指结点间的逻辑关系由结点的指针来表示的存储结构，逻辑上相邻的结点不一定存储在物理上相邻的存储单元中。其特点是借助指示元素存储地址的指针来表示数据元素之间的逻辑关系。通常用高级语言中的指针类型来描述链式存储结构。

索引存储结构是在存储结点数据的同时还建立索引表，通过索引表来查找结点所在的存储位置的一种存储结构。

散列存储结构(也称为哈希存储结构)是根据结点关键字直接计算出结点的存储地址的。

各数据元素之间的逻辑关系是用户按使用需要建立起来，并呈现在用户面前的数据元素的结构形式，这种形式称为数据的逻辑结构。在不易混淆的情况下，常常将数据的逻辑结构简称为数据结构。

数据结构在计算机内的实际存储形式称为数据的存储结构，也称为数据的物理结构。它是逻辑结构到计算机存储器的映射。

数据的逻辑结构和存储结构是密切相关的，二者可能一致，也可能不一致。在有的结构类型中，如一维数组中，这二者是一致的；而在有的结构类型中，如线性表中，这二者则可能是不一致的。在线性表中数据的逻辑关系表现为数据元素的有序性，即除了第一个和最后一个元素外，每个中间元素均有确定的前驱和后继。其存储结构若是顺序的，则同逻辑结构保持一致；若是链接的，则同逻辑结构不一致。另外，即使是线性表，若其元素的插入、删除运算被限制在表的一端进行，则该线性表便成为栈结构；若限制元素的插入在表的一端而删除在表的另一端进行，则该线性表便成为队列结构。由此可见，数据的逻辑结构和运算方式也密切相关。

5. 数据类型

数据类型是和数据结构密切相关的一个概念，它最早出现在高级程序设计语言中，用来刻画程序操作对象的特性。数据类型是程序设计语言中已经实现了的数据结构，它规定了数据的取值范围、存取方式及允许进行的运算。数据类型的引进基于数据的两个特征：一是数据有值，并取值于一定的范围内；二是对值可以施行给定的运算。数据的这两个特征用来区分不同种类的数据，因此可把数据类型定义为对数据对象值域和关于值域可施行的运算的一种分类。类型不同，值的存储表示、解释及运算方式也不同。如 C 语言中的 int 型变量一般占用两个字节，可进行算术、关系运算等。

程序设计语言允许用户直接使用的数据类型由具体语言决定，数据类型反映了程序设计语言的数据描述和处理能力。C 语言除了提供整型、实型、字符型等基本类型数据外，还允许用户自定义各种类型数据，例如数组、结构和指针等。

数据结构是在程序设计语言的基础上由用户建立起来的，它依靠程序设计语言提供的数据类型来描述数据的逻辑结构，也依靠程序设计语言提供的各种设施来定义、描述运算及算法。这些运算可由用户按实际需要自己定义，而不是由程序设计语言系统事先规定。因此，数据类型和数据结构的主要区别是：前者面向系统；后者面向对象，是高一层的数据抽象。

6. 抽象数据类型

1) 抽象数据类型的定义

抽象数据类型(Abstract Data Type，ADT)是指一个数学模型以及定义在该模型上的一组操作。抽象数据类型的定义取决于它的一组逻辑特性，而与其在计算机内部的表示和实现无关，即不论其内部结构如何变化，只要它的数学特性不变，都不会影响其外部的使用。例如，“整数”是一个抽象数据类型，其数学特性和具体使用的计算机或语言无关。

抽象数据类型不仅仅局限于计算机中已经定义并实现的数据类型，还包括用户在设计软件系统时自己定义的数据类型。为了提高软件的复用率，在构造软件系统的各个相对独立的模块时，定义一组数据和施于这些数据之上的一组操作，并在模块内部给出它们的表示和实现细节，而在模块外部使用的只是抽象的数据和抽象的操作。例如，矩阵的抽象数据类型定义为：矩阵是一个由 $m \times n$ 个数排成 m 行 n 列的表，它可以进行初等变换、相加、相乘、求逆等运算。又例如某电影院需要一个售票软件，要求能够查询哪些票被售出、哪些票还未售出，则：

① 数据对象和关系：由表示排和号的两个整数限制的一个固定长度的元素序列，以及一

个表示票总数的整数。

② 基本操作：

a. 售票：销售票。

b. 预订：预订票或取消预订。

c. 查询余票：查询还有多少未售出的票。

d. 是否售出：查询指定的票是否售出。

抽象数据类型有两个重要特性：

(1) 数据抽象：当用 ADT 描述程序处理的实体时，强调的是其本质的特征、其所能完成的功能以及它和外部用户的接口(即外界使用它的方法)。

(2) 数据封装：将实体的外部特性和其内部实现细节分离，并且对外部用户隐藏其内部实现细节。

抽象数据类型可以使我们更容易描述现实世界，比如用线性表描述学生的成绩，用树或图描述遗传关系等。使用它的人，可以只关心它的逻辑特征而不需要了解它的存储方式；定义它的人，同样不必要关心它是如何存储的。

2) 抽象数据类型的描述和实现

抽象数据类型的定义可以用三元组描述为

(D，S，P)

其中，D 是数据对象；S 是 D 上的关系集；P 是 D 的基本操作集。

在本书各章中均采用以下格式来定义抽象数据类型：

```
ADT 抽象数据类型名 {
    数据对象：〈数据对象的定义〉
    数据关系：〈数据关系的定义〉
    基本操作：〈基本操作的定义〉
} ADT 抽象数据类型名
```

其中，数据对象和数据关系的定义用伪码描述，基本操作的定义格式为

```
基本操作名(参数表)
    初始条件：〈初始条件描述〉
    操作结果：〈操作结果描述〉
```

基本操作有两种参数：赋值参数和引用参数。其中赋值参数只为操作提供输入值；引用参数以“&”打头，除可提供输入值外，还将返回操作结果。

初始条件描述了操作执行之前数据结构和参数应满足的条件，若不满足，则操作失败，并返回相应的出错信息。在上、下文关系明确的情况下，我们可以省略初始条件限制。

操作结果说明了在操作正常完成之后，数据结构的变化状况和应返回的结果。若初始条件为空，则可省略之。

例如，抽象数据类型“复数”的定义为

```
ADT Complex {
  数据对象：D = {e1,e2 | e1,e2∈RealSet }
  数据关系：R1 = {<e1,e2> | e1 是复数的实部，e2 是复数的虚部 }
  基本操作：
```

InitComplex(&Z, v1, v2)
操作结果：构造复数 Z，其实部和虚部分别被赋以参数 v1 和 v2 的值
DestroyComplex(&Z)
初始条件：复数已存在
操作结果：复数 Z 被销毁
GetReal(Z, &RealPart)
初始条件：复数已存在
操作结果：用 RealPart 返回复数 Z 的实部值
GetImag(Z, &ImagPart)
初始条件：复数已存在
操作结果：用 ImagPart 返回复数 Z 的虚部值
Add(z1,z2, &sum)
初始条件：z1、z2 是复数
操作结果：用 sum 返回两个复数 z1、z2 的和值
} ADT Complex

其中，用两个实数来表示复数，将复数定义为两个实数的有序对，并约定复数的实部是前驱、虚部是后继。

抽象数据类型需要通过高级编程语言中已经实现的数据类型(通常称之为固有数据类型)来实现。例如，对以上定义的复数类型可以利用 C 语言实现如下描述：

```
// 存储结构的定义
typedef struct
{
   float realpart;
   float imagpart;
} complex;
// 基本操作的函数原型说明
void InitComplex(complex &Z,float realval,float imagval);
  // 构造复数 Z，其实部和虚部分别被赋以参数 realval 和 imagval 的值
void DestroyComplex( complex &Z)
  // 销毁复数 Z
float GetReal(complex Z );
  // 返回复数 Z 的实部值
float GetImag(complex Z );
  // 返回复数 Z 的虚部值
void Add(complex z1,complex z2,complex &sum );
  // 以 sum 返回两个复数 z1、z2 的和
  // 基本操作的实现
⋮
void Add(complex z1,complex z2,complex &sum )
```

```
{
  // 以 sum 返回两个复数 z1、z2 的和
  sum.realpart = z1.realpart + z2.realpart;
  sum.imagpart = z1.imagpart + z2.imagpart;
}
⋮
```

1.1.2 基本运算

数据结构必须给出每种结构类型所定义的各种运算的相应算法。数据结构的基本操作有如下几种：

(1) 建立与消除：建立和消除一个数据结构。

(2) 访问：访问数据结构中的某个数据元素。

(3) 插入：在数据结构中的指定位置上增添新的数据元素。

(4) 删除：删除数据结构中指定的某个数据元素。

(5) 修改：修改数据结构中指定的某个数据元素的值。

(6) 查找：在数据结构中寻找满足某个特定要求的数据元素的值。

(7) 排序：重新安排数据元素之间的逻辑顺序关系，使其值按由小到大或由大到小的次序排列。

为了增加对数据结构的感性认识，下面举一个例子来具体说明。

假设要在某大学的全体学生中查找一位叫张三的学生的个人资料，要求设计一个算法，若找到，则给出其姓名、学号、班级和系别；若没有找到，则显示“查无此人”。

通过分析可以发现，这个算法的设计依赖于学生个人资料的有关数据的逻辑结构及在计算机内的存储方式。在这个问题中，数据元素是学生，全校学生构成数据对象，全校学生按不同的方式组织起来就形成了不同的逻辑结构，查找的方法随之也会有所不同。

如果学生的姓名是随意排列、毫无规律的话，则只能从学生名单中第一个姓名开始，逐个与“张三”比较，直到找到为止；若查找完整个名单后还没有找到此人，则输出“查无此人”。这种组织方法采用了线性结构，相应的查找方法是顺序查询，这种查找方法很简单，却相当费时，效率太低。

如果将学生的姓名进行适当的组织，将它们按汉语拼音字母的顺序排列起来，并且建立一个索引表，就可先从索引表中找到以字母“Z”开头的姓名在学生名单中的起始位置，然后从该处开始往下查找，而不用去查找其余 25 个字母开头的姓名。由于采用了这种索引结构来组织数据，相应地就使用了另一种完全不同的查找方法——索引查找，显然它的查找效率比前一种高。

如果将学生的名单按系别进行划分，每个系又按专业再细分，每个专业又按班级组织起来，则只要知道了张三所在的系别、专业和班级就很容易查明是否有此人。这里采用了树形结构来组织数据，相应地就可以使用树表的查询方法来查找。

学生名单的组织方式就是一个数据结构问题，上述三种不同的数据结构可以得出完全不同的三种查找算法。由此可见，算法与数据结构密切相关，每一种算法依赖于具体的数据结构，数据结构直接影响着对算法的选择和执行的效率。

1.2　算法描述和算法分析

同一个问题的数据集合可以组织成不同的数据结构。例如，学生成绩档案在逻辑上属于线性数据结构，可以组织成线性表，也可以组织成二叉树，具体选择哪种结构，要视具体情况而定。此外，同为线性表，在存储时既可以选择顺序存储，也可以选择链式存储。由于数据的运算是通过算法来描述的，因此就产生了对数据结构和算法如何选择和评价的问题。

1.2.1　算法

1. 定义

算法是对特定问题求解步骤的一种描述，它是指令的有限序列，其中每一条指令表示一个或多个操作。

算法具有如下性质：

(1) 动态有穷性。算法中每条指令的执行次数必须是有限的，最后一定要终止。

(2) 确定性。算法中每条指令的含义都必须是明确的、无二义性的。

(3) 输入。一个算法可以有 0 个或 0 个以上的输入，这些输入取自于某个特定对象的集合。

(4) 输出。一个算法至少产生一个输出，这些输出是同输入有着某些特定关系的量。

(5) 可行性。每条指令的执行时间都是有限的。

2. 算法与程序的区别

算法经过有限步计算后一定会终止，而程序不一定能满足动态有穷性。例如，操作系统是程序而不是算法，这是因为只要系统不遭到破坏，它永远也不会终止。此外，程序一般是用某种计算机语言书写的，而算法既可以采用计算机语言来描述又可以采用自然语言来描述。

3. 算法与数据结构的关系

算法和数据结构密切相关，同一应用问题采用不同的数据结构，其运算的算法也不相同。

著名的瑞士计算机科学家 N.Wirth 教授指出：算法 + 数据结构 = 程序。这里的算法指的是对数据运算顺序的描述，数据结构是指数据的逻辑结构和存储结构。程序设计的实质是对确定的问题选择一种好的数据结构，从而设计出一种好的算法，而好的算法在很大程度上取决于所采用的数据结构。

4. 算法设计要求

(1) 正确性。设计出的算法应当能满足具体问题的要求，至少应包括对于输入、输出、加工和处理等的正确描述。

(2) 可读性。设计出的算法应当有助于读者对算法的理解。

(3) 健壮性。当输入的数据非法时，算法能作出适当的反应。

(4) 高效率与低存储量需求。效率指的是算法执行的时间，存储量需求是指算法执行