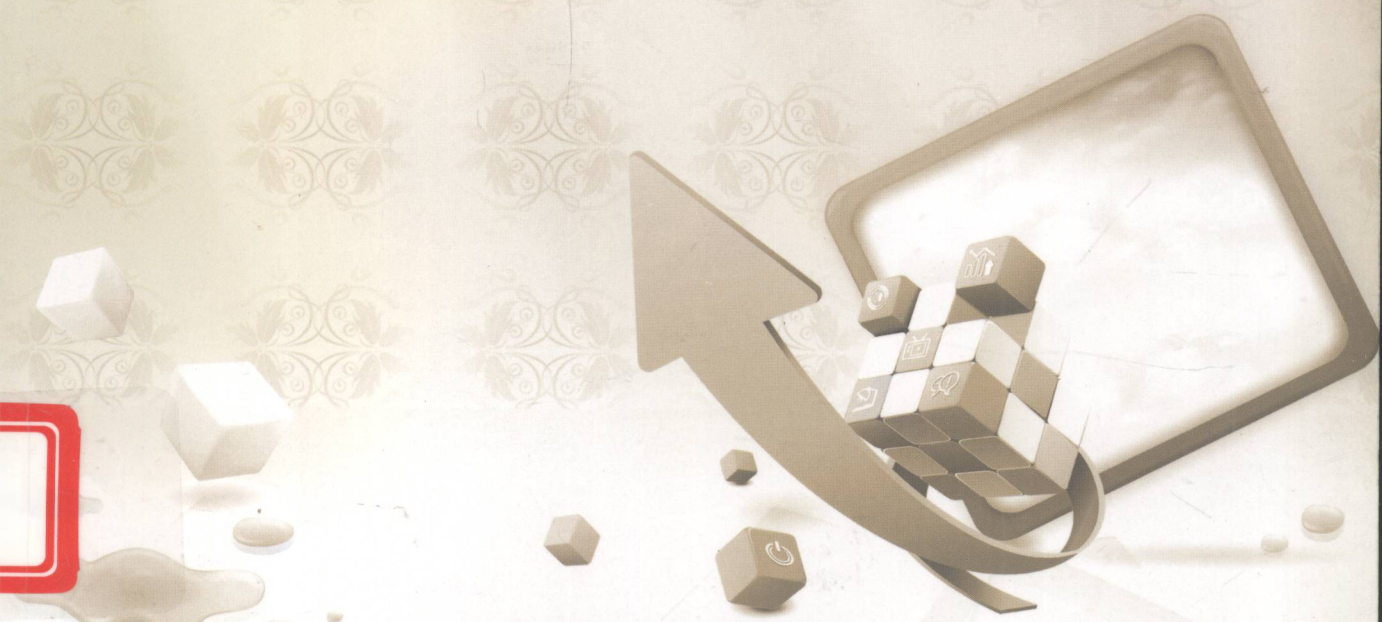
高职高专计算机实用教程系列规划教材

数据结构

（C语言版）

王桂芝　主编　　来社安　孙　凌　副主编
李　丹　吕金龙　参编



中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

高职高专计算机实用教程系列规划教材

数据结构

（C语言版）

王桂芝　主编

来社安　孙　凌　副主编

李　丹　吕金龙　参编

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

本书重点讨论了各种基本数据结构的类型描述、常用算法实现及其应用。全书共分 9 章：第 1 章主要介绍了有关数据结构的基本概念和术语；第 2～7 章分别讨论了线性表、栈和队列、串、数组和广义表、树及图等基本类型的数据结构；第 8、9 章主要讨论了查找和排序的各种实现方法及其综合分析比较。除第 1 章外，其余每章最后一节都以实训的形式给出了本章重点算法的应用实例，以便于上机验证。

本书基本理论的阐述由浅入深、算法描述清晰、内容安排合理、语言精练、逻辑推理严密，适合作为高职高专院校计算机类或信息类相关专业的教材，也可为计算机专业人员自学或参加计算机类考试提供参考。

图书在版编目（CIP）数据

27029°

数据结构：C 语言版 / 王桂芝主编. — 北京：中国铁道出版社，2011.8
高职高专计算机实用教程系列规划教材
ISBN 978-7-113-12943-9

Ⅰ.①数… Ⅱ.①王… Ⅲ.①C 语言－程序设计－高等职业教育－教材 ②数据结构－高等职业教育－教材 Ⅳ.①TP312②TP311.12

中国版本图书馆 CIP 数据核字（2011）第 162165 号

书　　名：数据结构（C 语言版）
作　　者：王桂芝　主编

策划编辑：秦绪好　　**读者热线：**400-668-0820
责任编辑：赵　鑫　彭立辉　　**封面设计：**付　巍
特邀编辑：李红玉　　**封面制作：**白　雪

出版发行：中国铁道出版社（北京市宣武区右安门西街 8 号　邮政编码：100054）
印　　刷：三河市华丰印刷厂
版　　次：2011 年 8 月第 1 版　　2011 年 8 月第 1 次印刷
开　　本：787mm×1092mm　1/16　**印张：**14.75　**字数：**349 千
印　　数：1～3 000 册
书　　号：ISBN 978-7-113-12943-9
定　　价：24.00 元

前　言

数据结构是计算机程序设计的重要理论基础，是高等院校计算机类、信息类及相关专业的一门必修课程。

本书主要讨论数据的逻辑结构和在计算机中的表示，以及常用的各种非数值运算的算法，使读者掌握计算机处理的数据对象的特性，学会数据的组织方法，能够选择合适的数据组织形式，把现实中的问题转换为计算机内部的数据表示和数据处理。

参与本书编写的编者具有多年数据结构教学及软件开发经验，全书采用类 C 语言作为数据结构和算法的描述语言，在对各类数据结构及算法的描述中尽量考虑 C 语言的特色。本书选用的算法思路清晰、描述精练，运用大量图示方法把抽象的理论形象化。本书注重理论与实践的统一，除第 1 章外，其余每章都安排有实训，利用完整的实例对本章所学的重要算法进行练习。

全书共分 9 章：第 1 章主要介绍了有关数据结构的基本概念和术语；第 2 章~第 7 章分别讨论了线性表、栈和队列、串、数组和广义表、树及图等基本类型的数据结构；第 8 章和第 9 章主要讨论了查找和排序的各种实现方法及其综合分析比较。

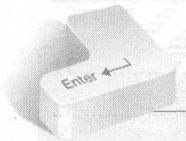
其中，正文中带“*”的小节为选学内容。

本书由王桂芝任主编，来社安和孙凌任副主编；其中，王桂芝编写第 2 章和第 8 章，来社安编写第 1 章和第 7 章，李丹编写第 3 章和第 4 章，孙凌编写第 5 章和第 6 章，吕金龙编写第 9 章。

在本书的编写过程中，我们参阅了许多有关数据结构的教材和其他大量的文献，在此对所有的编著者表示衷心的感谢。由于时间仓促，编者水平有限，书中难免存在不当之处，恳请有关专家和广大读者批评指正。

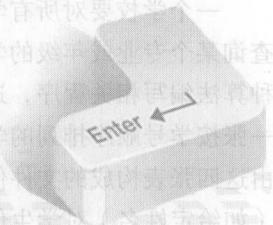
编　者

2011 年 6 月



CONTENTS

目　录



第1章 数据结构概述

教学目标

- 熟悉数据结构的基本概念和术语；
- 掌握数据的逻辑结构和存储结构及其特点；
- 掌握算法时间复杂度的计算方法；
- 了解数据结构课程的重要性。

主要知识点

- 数据结构课程的性质和地位；
- 基本概念和术语；
- 算法和算法分析。

1.1 数据结构课程的性质和地位

计算机是对各种各样的数据进行处理的机器。早期的计算机所处理的数据对象是纯粹的数值数据，程序设计者将主要的精力用在程序设计的技巧上，数据的组织并不需要花费太多的精力和时间。然而随着计算机产业的发展，计算机的功能不断丰富，所处理的数据也越来越复杂。在计算机中如何有效地组织和处理数据也成了迫切需要解决的问题，于是数据结构这门课应运而生。

1.1.1 数据结构课程所讨论的内容

计算机解决一个问题时，一般需要以下几个步骤：①分析问题从中抽象出适当的数学模型；②设计解决此数学模型的算法；③编制出程序使问题得到解决。数值计算问题的数学模型一般可由数学方程或数学公式来描述，然而更多的非数值计算问题无法用数学方程加以描述。这类问题所处理的对象更多是由字符、表格和图像等构成的信息，这就需要把处理的各种信息按照其逻辑特性组织起来，然后再进行存储及算法的设计。

一般来说，非数值计算问题中所处理的对象可以按照其逻辑特性描述成表、树和图3种形式。下面看几个例子。

【例1.1】学生信息检索系统。

一个学校要对所有学生的信息进行管理，利用学生信息检索系统来实现某个学生情况的查询、查询某个专业或年级的学生等。这就需要把学生的有关信息进行有效的组织并存储，然后按照某种算法编写相关程序，这样可以用计算机实现自动检索。由此，可以在学生信息检索系统中建立一张按学号顺序排列的学生信息表和分别按姓名、专业、年级顺序排列的索引表，如图1-1所示。由这四张表构成的文件便是学生信息检索的数学模型，计算机的主要操作便是按照某个特定要求（如给定姓名）对学生信息文件进行查询。

诸如此类的还有电话自动查号系统、考试查分系统、仓库库存管理系统等。在这类文档管理的数学模型中，计算机处理的对象之间通常存在着一种简单的线性关系，这类数学模型可称为线性的数据结构。

学号	姓名	性别	专　业	年 级
20100001	吴承志	男	计算机应用	2010级
20100002	廖诗佳	女	计算机网络	2010级
20100301	周　晶	女	软件技术	2010级
20110302	张会友	男	计算机网络	2011级
20110303	石宝国	男	计算机应用	2011级
20110801	何文颖	女	计算机应用	2011级
20110802	周　鑫	男	软件技术	2011级
20120803	崔文靖	男	计算机网络	2012级
20120601	周　晶	女	计算机应用	2012级
20120602	施子健	男	软件技术	2012级

（a）学生信息表

崔文靖	8
何文颖	6
廖诗佳	2
周　晶	3，9
石宝国	5
施子健	10
吴承志	1
周　鑫	7
张会友	4

（b）姓名索引表

计算机应用	1，5，6，9
计算机网络	2，4，8
软件技术	3，7，10

（c）专业索引表

2010级	1，2，3
2011级	4，5，6，7
2012级	8，9，10

（d）年级索引表

图1-1　学生信息查询系统中的数据结构

【例1.2】计算机和人对弈问题。

要实现计算机与人对弈，就要将对弈的策略事先存入计算机。由于对弈的过程是在一定规则下随机进行的，所以为使计算机能灵活对弈，就必须对对弈过程中所有可能发生的情况以及相应的对策都考虑周全，并且一个“好”的棋手在对弈时不仅要看棋盘当时的状态，还应能预测棋局发展的趋势，甚至最终结局。因此，在对弈过程中，计算机操作的对象是对弈过程中可能出现的棋盘状态（称为格局）。格局之间的关系是由比赛规则决定的，通常这个关系不是线性的，因为从

一个棋盘格局可以派生出几个格局。如图 1-2 所示，最上方所示为井字棋的一个格局，由这一格局可以派生出五个格局，而从某一个新的格局又可派生出四个可能出现的格局等。因此，若将从对弈开始到结束的过程中所有可能出现的格局都画在一张图上，则可得到一棵倒长的“树”。“树根”是对弈开始之前的棋盘格局，所有的“叶子”就是可能出现的结局，对弈的过程就是从树根沿“树叉”到某个“叶子”的过程。

可见，“树”可以是某些非数值计算问题的数学模型，它也是一种数据结构。

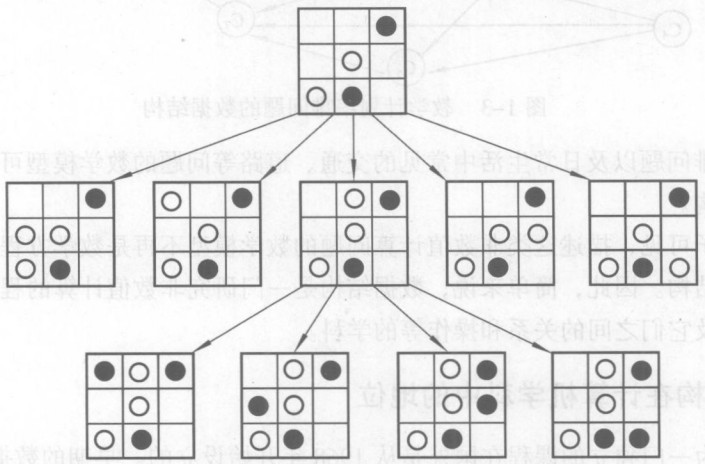


图 1-2　井字棋对弈“树”

【例 1.3】教学计划编排问题。

一个教学计划包含许多课程，在教学计划包含的许多课程之间，有些必须按规定的先后次序进行，有些则没有次序要求，如表 1-1 所示。在教学计划编排过程中，就要考虑课程之间先修和后续的关系。这种各个课程之间的次序关系可用一个称做图的数据结构（见图 1-3）来表示。有向图中的每个顶点表示一门课程，如果从顶点 v_i 到 v_j 之间存在有向边$<v_i, v_j>$，则表示课程 i 必须先于课程 j 进行。

表 1-1　软件专业的课程设置

课 程 代 号	课 程 名 称	先 修 课 程
C_1	计算机基础	无
C_2	微机原理	C_1
C_3	HTML 语言	C_1
C_4	C 语言程序设计	无
C_5	数据结构	C_1 C_4
C_6	数据库原理及应用	C_1 C_4
C_7	面向对象技术	C_4 C_5 C_6
C_8	Java 程序设计	C_5 C_7
C_9	ASP.NET	C_3
C_{10}	软件工程	C_9 C_7

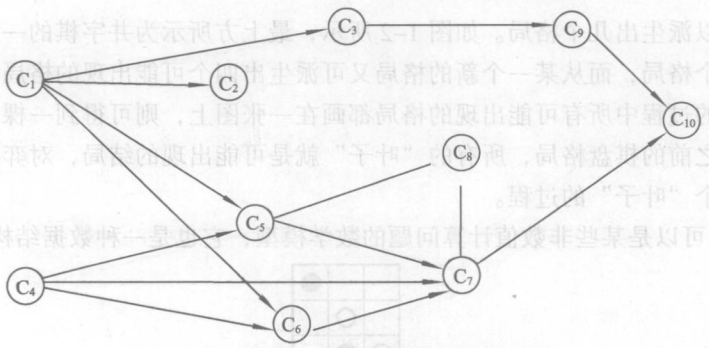


图 1-3　教学计划编排问题的数据结构

这类课程编排问题以及日常生活中常见的交通、道路等问题的数学模型可以抽象为一种所谓“图”的数据结构。

综上三个例子可见，描述这类非数值计算问题的数学模型不再是数学方程，而是诸如表、树和图之类的数据结构。因此，简单来说，数据结构是一门研究非数值计算的程序设计问题中计算机的操作对象以及它们之间的关系和操作等的学科。

1.1.2　数据结构在计算机学科中的地位

数据结构作为一门独立的课程在国外是从 1968 年开始设立的。早期的数据结构几乎是图论，特别是表、树的理论的同义词。随后，数据结构这个概念被扩充到包括网络、集合代数论、格、关系等方面，从而变成了现在称为离散数学的内容。然而，由于数据必须在计算机中进行处理，因此，不仅要考虑数据本身的数学性质，还必须考虑数据的存储结构，这就进一步扩大了数据结构的内容。从 20 世纪 60 年代末到 70 年代初，出现了大型程序，从软件工程的角度出发，为了提高程序的可维护性、可调试性等性能要求，程序和数据必须分离。于是，人们就越来越重视数据结构，认为程序设计的实质是对确定的问题选择一种好的结构，加上设计一种好的算法，正如瑞士著名的计算机科学家 N.Wirth 所提出的著名公式“程序=算法+数据结构”所阐述的那样。随着数据库系统技术的不断提高，大型数据库程序设计越来越多，对于数据处理的技术和范围要求越来越高，因此在数据结构理论中又增加了文件管理（特别是大型文件）的内容。今天，虽然数据结构理论已经比较成熟，但对于它在面向各个专业应用领域的研究还在不断发展，如多维图形数据结构、动画数据结构、音频数据结构，特别是它在实际工作和生活中的应用。

现在，数据结构是计算机专业的一门专业基础课，在计算机学科中起着承上启下的作用，在计算机技术的各个领域中也有着广泛的应用。学习这门课程需要程序设计语言作为基础，学习前要有扎实的程序设计基本功，同时又为后续的数据库等一系列课程提供知识基础。数据结构的研究不仅涉及计算机硬件（特别是编码理论、存储装置和存取方法等）的研究范围，而且和计算机软件的研究有着密切的关系，无论是编译程序还是操作系统，均涉及数据元素在存储器中的分配问题。因此，可以认为数据结构是介于数学、计算机硬件和计算机软件三者之间的一门核心课程。

学习数据结构的目的是为了了解计算机处理对象的特性，将实际问题中所涉及的处理对象在计算机中表示出来并对它们进行处理。与此同时，通过算法训练来提高学生的思维能力，通过程序设计的技能训练来促进学生的综合应用能力和专业素质的提高。

1.2　基本概念和术语

数据结构的基本概念和术语是学习数据结构的基础，在这里先给出其确切的定义。

1.2.1　数据结构的相关术语

数据（Data）：数据即信息的载体，是对客观事物的符号表示，指能输入到计算机中并能被计算机程序处理的符号的总称。例如，整数、实数、字符、文字、声音、图形、图像等都是数据。

数据元素（Data Element）：数据元素是数据的基本单位，它在计算机处理和程序设计中通常作为一个整体进行考虑。例如，上节中的“树”中的一个棋盘格局。一个数据元素一般由一个或多个数据项组成。例如，一个学生的信息为一个数据元素，而一个学生的信息中的每一项（如姓名、专业、年级等）为一个数据项。数据项是数据不可分割的最小单位。

数据对象（Data Object）：数据对象是具有相同特征的数据元素的集合，是数据的一个子集。例如，整数数据对象是集合 $N=\{0, \pm 1, \pm 2, \cdots\}$，字母字符数据对象是集合 $C=\{'A', 'B', \ldots 'Z'\}$。

数据结构（Data Structure）：数据结构简称 DS，是数据元素的组织形式，即相互之间存在一种或多种特定关系的数据元素的集合。数据结构可以用二元组来定义：

$$\text{Data\_Structures} = (D, S)$$

其中，D 是数据元素的有限集，即数据对象；S 是 D 上所有数据元素之间的关系的有限集。

例如，在计算机科学中，复数是一种数据结构，可定义如下：

$$\text{Complex} = (C, R)$$

其中，C 是含两个实数的集合$\{c_1, c_2\}$；$R=\{P\}$，而 P 是定义在集合 C 上一种关系$\{<c_1, c_2>\}$，有序偶 $<c_1, c_2>$表示 c_1 是复数的实部，c_2 是复数的虚部。

一般地，数据结构包括以下三个方面的内容：

① 数据的逻辑结构：数据元素之间的逻辑关系。

② 数据的存储结构：又称数据的物理结构，是指数据元素及其关系在计算机内存中的表示（又称映像）。它包括数据元素的表示和数据元素之间关系的表示。

③ 数据的运算及实现：对数据元素可以施加的操作以及这些操作在相应存储结构上的实现。

1.2.2　数据的逻辑结构

数据的逻辑结构是指数据元素之间的逻辑关系。它是从逻辑关系上描述数据，与数据存储无关，是独立于计算机的。因此，数据的逻辑结构可以看做从具体问题中抽象出来的数学模型。

数据的逻辑结构可分为四种基本类型：集合结构、线性结构、树形结构和图状结构。

1. 集合结构

结构中的数据元素之间除了“同属于一个集合”的关系外，别无其他关系。例如，认定一个学生是否为班级的成员。

2. 线性结构

结构中的数据元素之间存在一对一的关系，即所谓的线性关系。例如，以学生入学报到的时间先后顺序排列数据元素。

线性结构中数据元素之间的逻辑关系是：有且仅有一个开始（第一个）元素和一个终端（最后一个）元素；所有元素最多只有一个直接前驱（前面紧相邻）和一个直接后继（后面紧相邻）；开始元素没有直接前驱，终端元素没有直接后继。

3．树形结构

结构中的数据元素之间存在一对多的关系。例如，班级中的管理体系，班长管理多个班委，每个班委管理多个成员。

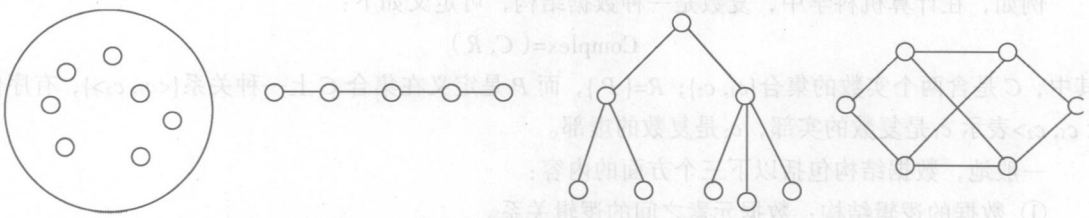
树形结构实际上是元素之间有分支并具有层次关系的结构，类似于自然界的树。在这种结构中，每个元素允许有多个直接后继。

4．图状结构

图状结构也称网状结构，结构中的数据元素之间存在多个对多个的关系。例如，同学之间的朋友关系。

在图状结构中，元素的前驱和后继个数都不加限制，元素之间的关系是任意的，图中任意两个元素之间都可以相关。

图 1-4 为上述四类基本结构的关系图。有时也可将集合结构、树形结构和图状结构归纳为非线性结构，此时数据的逻辑结构就可分为线性结构和非线性结构两种类型。由于“集合”是数据元素之间关系极为松散的一种结构，也可用其他结构来表示它，因此数据结构中通常只讨论后 3 种逻辑结构。



（a）集合结构　（b）线性结构　（c） 树形结构　（d）图状结构

图 1-4　四类基本逻辑结构

1.2.3　数据的存储结构

讨论数据结构的目的是为了在计算机中实现对它的操作，这就要研究数据的存储结构，即数据的逻辑结构在计算机内存中的存储方式。数据的存储结构又称物理结构，有顺序存储和链式存储两种。

1．顺序存储结构

这种存储结构是把数据元素依次存储在一组地址连续的存储单元中，元素的逻辑关系由存储单元的位置直接体现。在高级语言中，常用一维数组来实现顺序存储结构。

顺序存储结构的特点是：所有元素存储在一组连续的存储单元中，逻辑上相邻的元素存放在计算机内存后仍然相邻。对于线性结构自然可以采用顺序存储，而非线性结构也可以通过线性化处理后顺序存储。

2. 链式存储结构

将数据元素存储在一组任意的存储单元中，而用附加的指针域表示数据元素之间的逻辑关系，由此得到的存储结构称为链式存储结构。在链式存储结构中每个结点（数据元素）由两部分组成：数据域和指针域。其中，数据域存放元素本身的数据，指针域存放有关地址。

链式存储的特点是：所有元素存储的内存单元不要求连续，元素之间的关系通过结点的指针域来表示，从而逻辑上相邻的数据元素物理上不一定相邻。这种存储结构比顺序存储结构更灵活、方便。

数据的逻辑结构和物理结构是密不可分的两个方面，对于任何一种算法的设计都取决于数据的逻辑结构，而算法的实现则依赖于所采用的存储结构。

1.2.4　数据类型和抽象数据类型

虽然数据的存储结构涉及的是数据元素及其关系在内存中的具体表示，但对于不同的硬件其存储方式不同，而且对用户来说不需要了解全部的实现细节。本书在高级语言的层次上讨论数据结构的操作，使用高级语言中提供的“数据类型”来描述存储结构。例如，用一维数组来描述顺序存储结构。

1. 数据类型

数据类型（Data Type）是一个值的集合和定义在这个值集上的一组操作的总称。数据类型反映了数据的取值范围和对这类数据可以施加的运算。例如，C 语言中的整型变量，其值集为某个区间上的整数，在典型的 16 位计算机中一般取-32 768～32 768，定义在其上的操作为加、减、乘、除和取模等运算。

按“值”是否可以分解，可以把数据类型分为两类：

① 原子类型：每个对象仅由单值构成的类型，其值不可分解。例如，C 语言的基本类型（整型、实型、字符型）、指针类型和空类型。

② 结构类型：每个对象可由若干成分按某种结构构成的类型，其值可分解成若干成分（或分量）。例如，C 语言的数组是一种结构类型，它由固定个数的同一类型的数据顺序排列而成。记录也是一种结构类型，它由固定个数的不同（也可以相同）类型顺序排列而成，记录类型中的每个记录包含有固定个数的不同类型数据，每个数据也都可以直接访问。

在某种意义上，数据结构可以看成一组具有相同结构的值，结构类型可以看成由一种数据结构和定义在其上的一组操作组成。

2. 抽象数据类型

抽象数据类型（Abstract Data Type，ADT）可以看做数据的逻辑结构及在此结构上定义的一组操作，是对数据类型的进一步抽象。

抽象数据类型不仅包括数据类型的定义，而且为这个新的类型说明了一个有效的操作集合。抽象类型强调数据类型的数据特性，而不管它们在不同处理器上的具体实现。例如，整数类型就是一个抽象类型，其数据对象是整数，基本操作有加、减、乘、除等运算。人们在程序设计中大量使用这些运算，但没有人去追究这些运算到底是如何实现的。使用抽象数据类型可以提高软件的复用程度，使得使用它的人可以只关心它的逻辑特征，不需要了解它的存储方式；定义它的人

同样也不需要关心它是如何进行存储的。

抽象数据类型一般可以由数据元素、数据元素之间的关系及定义在其上的操作三个要素来定义。可以用三元组表示：

$$ADT=(D, S, P)$$

其中，D 是数据对象；S 是 D 上所有数据元素之间的关系的有限集；P 是对 D 的基本操作表。

抽象数据类型可通过固有数据类型来表示和实现，即利用处理器中已存在的数据类型来说明新的结构，用已经实现的操作来组合新的操作。为了简单起见，本书不再给出每种数据结构的完整抽象数据类型描述，而是采用介于伪码和C语言之间的类C语言作为工具来描述数据结构及其算法。这使得数据结构与算法的描述和讨论简明清晰，不拘泥于C语言的细节，又能容易地转换成C或C++程序。

1.3 算法及算法分析

1.3.1 算法的概念

算法（Algorithm）是对特定问题求解步骤的描述，是指令的有限序列，其中每条指令表示一个或多个操作。对于实际问题，不仅要选择合适的数据结构，还要有个好的算法，这样才能更好地求解问题。一个算法必须具有以下五个特征：

① 有穷性：一个算法必须在执行有限步骤之后结束，且每一步骤都可在有穷时间内完成。

② 确定性：算法中每一条指令的含义必须明确，无二义性。在任何情况下对于相同的输入，必须能得出相同的输出。

③ 可行性：算法中描述的操作均可通过已经实现的基本运算的有限次执行来实现。

④ 输入：一个算法可以有零个或多个输入，这些输入取自于某个特定的对象集合。

⑤ 输出：一个算法至少有一个输出，这些输出是与输入有着某些特定关系的值。没有输出的算法是没有实际意义的。

1.3.2 算法的设计要求

同一个问题可能有许多求解的算法，在利用计算机解决问题时就需要对这些求解算法进行筛选，从中选择一个最好的算法来解决问题。算法选择是否恰当将直接影响到问题解决的效果。一个好的算法通常应考虑以下目标：

1. 正确性

算法应当满足具体问题的要求，即具备正确性，否则就谈不上解决实际问题。“正确”的含义在通常情况下有很大的差别，大体可分为以下四个层次：

① 程序不含语法错误。

② 程序对于随意的几组合法输入数据能够得出符合要求的结果。

③ 程序对于精心设计的典型合法数据输入能够得出符合要求的结果。

④ 程序对于所有合法的输入数据都能得出符合要求的结果。

显然，输入所有的合法数据来检验算法的正确性是极其困难的，而对于①、②两个层次的检