21世纪高等学校精品规划教材

数据结构

（C语言描述）

李素若　陈万华　游明坤　编著

中国水利水电出版社
www.waterpub.com.cn

21世纪高等学校精品规划教材

数据结构（C语言描述）

李素若　陈万华　游明坤　编著

中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

本书结合编者多年教学经验，系统介绍了数据结构的基本概念和知识。在选材与编排上，本书的内容符合数据结构本科教学大纲要求，突出实用性和应用性，同时满足最新研究生考试大纲的要求。全书共9章，主要内容包括：绪论、线性表、栈和队列、串、数组和广义表、树、图、查找、排序等。全书条理清晰，概念清楚，逻辑推理严谨，内容详实，既注重数据结构和算法原理，又十分强调程序设计训练。书中算法均配有完整的C语言程序，程序结构清晰，构思精巧，且所有程序都已在Dev-C++5.0下编译通过并能正确运行，它们既是很好的学习数据结构和算法的示例，也是很好的程序设计示例。本书配套有《数据结构习题解答及上机指导》。

本书可作为普通高等院校计算机和信息技术等相关专业的学生使用的教材，也可供从事计算机工程与应用的科技工作者和其他希望学习数据结构的人员参考。

本书提供免费电子教案，读者可以从中国水利水电出版社网站以及万水书苑下载，网址为：http://www.waterpub.com.cn/softdown 或 http://www.wsbookshow.com/。

图书在版编目（CIP）数据

数据结构 : C语言描述 / 李素若, 陈万华, 游明坤编著. -- 北京 : 中国水利水电出版社, 2014.7
21世纪高等学校精品规划教材
ISBN 978-7-5170-2061-5

Ⅰ. ①数… Ⅱ. ①李… ②陈… ③游… Ⅲ. ①数据结构－高等学校－教材②C语言－程序设计－高等学校－教材 Ⅳ. ①TP311.12②TP312

中国版本图书馆CIP数据核字(2014)第104788号

策划编辑：杨庆川　　责任编辑：陈　洁　　加工编辑：宋　杨　　封面设计：李　佳

书　　名	21世纪高等学校精品规划教材 数据结构（C语言描述）
作　　者	李素若　陈万华　游明坤　编著
出版发行	中国水利水电出版社 （北京市海淀区玉渊潭南路1号D座　100038） 网址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）68367658（发行部）、82562819（万水）
经　　售	北京科水图书销售中心（零售） 电话：（010）88383994、63202643、68545874 全国各地新华书店和相关出版物销售网点
排　　版	北京万水电子信息有限公司
印　　刷	三河市铭浩彩色印装有限公司
规　　格	184mm×260mm　16开本　17.5印张　427千字
版　　次	2014年7月第1版　2014年7月第1次印刷
印　　数	0001—3000册
定　　价	32.00元

前　言

数据结构是普通高等院校计算机和信息技术等相关专业的一门主要的专业基础课，也是一门必修的核心课程。它不仅是计算机程序设计的理论基础，还是学习计算机操作系统原理、编译原理、数据库原理等课程的重要基础。

数据结构课程的主要任务是讨论数据的各种逻辑结构和数据在计算机中的存储表示，以及各种非数值运算的算法实现，其内容丰富、涉及面广，并且随着各种基于计算机的应用技术的发展而不断增加新的内容。通过学习，学生可以较全面地理解算法和数据结构概念，掌握各种数据结构和算法的实现方式，比较不同数据结构和算法特点，能够使用数据结构的基本分析方法来提高编写程序的能力和应用计算机解决实际问题的能力。

数据结构内容多、理论深、概念抽象。因此，在本书的编写中，编者结合自己的教学和编程实践经验，精选了基础理论内容，降低了概念的抽象性和理论难度。一方面力图用生动、通俗易懂的语言并结合编程实例来讲解各个知识点，便于读者理解和掌握；另一方面加强了数据结构设计、算法设计等实践应用环节。全书共 9 章，第 1 章主要讲述数据结构和算法的基本概念；第 2～7 章分别讲述线性表、栈和队列、多维数组和广义表、树和图这几种基本数据结构的特点、存储方法和基本运算，作为本书的重点，书中使用大量的篇幅来介绍这些基本数据结构的实际应用；第 8 章和第 9 章讲述查找和排序的基本原理与方法。另外，书中所涉及的有关概念及背景知识皆做出详细交代；对相关的定理和性质给出简单证明；对所有算法，都详细讨论其设计思想和实现方法，最后给出完整的 C 语言代码；书中所有算法和程序代码均在 Dev-C++5.0 环境下调试通过。

本书配套有《数据结构习题解答及上机指导》，内含与主教材各章内容相配合的习题解答参考和 7 套模拟考试试题和 10 个精心设计的实验，每个实验均包括实验目的、实验内容、实验说明、实验指导等，两本书配套使用可以更为全面地掌握数据结构这门课程。

本书第 1～5 章由李素若编写，第 6、8、9 章由陈万华编写，第 7 章由游明坤编写；全书由李素若负责审核和统稿；参加本书编写大纲讨论的教师还有严永松、胡玉荣、任正云、武永成、张牧等。

由于编者水平有限，加之时间仓促，书中难免有疏漏之处，敬请广大读者批评指正，以使本书质量得到进一步提高。

编　者

2014 年 4 月

目　　录

第1章　绪论

“数据结构”是计算机及相关专业的专业基础课程之一，是一门十分重要的核心课程。本课程主要研究数据结构的逻辑结构、存储结构以及定义在该结构上的操作及操作实现三个方面的内容。

本章主要介绍数据、数据元素、数据结构、数据的逻辑结构及数据的物理结构等基本概念；还将介绍算法的概念与特征、算法性能分析的方法等。通过本章的学习，读者应该掌握以下内容：

- 数据结构的概念；
- 数据类型和抽象数据类型；
- 算法和算法分析。

1.1　数据结构的概述

什么是数据结构？这是一个难以直接回答的问题。一般来说，用计算机解决一个具体问题时，大致需要经过下列几个步骤：首先要从具体问题中抽象出一个适当的数学模型，然后设计一个解此数学模型的**算法**（algorithm），最后编出程序，进行测试，调整直至得到最终解答。寻求数学模型的实质是分析问题，从中提取操作的对象，并找出这些操作对象之间含有的关系，然后用数学的语言加以描述。为了说明这个问题，我们首先举一个例子，然后再给出明确的含义。

假定有一个学生通讯录，记录了某校全体学生的姓名和相应住址，现在要写一个算法，要求当给定任何一个学生的姓名时，该算法能够查出该学生的住址。这样一个算法的设计将完全依赖于通讯录中的学生姓名及相应的住址是如何组织的，以及计算机是怎样存储通讯录中的信息的。

如果通讯录中的学生姓名是随意排列的，其次序没有任何规律，那么当给定一个姓名时，则只能在通讯录从头开始逐个与给定的姓名比较，顺序查对，直至找到所给定的姓名为止。这种方法相当费时间，效率很低。然而，若我们对学生通讯录进行适当的组织，按学生所在班级来排列，并且再造一个索引表，这个表用来登记每个班级学生姓名在通讯录中的起始位置，这样一来，情况将大为改善。这时，当要查找某学生的住址时，则可先从索引表中查到该学生所在班级的学生姓名是从何处起始的，然后就从此起始处开始查找，而不必去查看其他部分的姓名。由于采用了新的结构，于是就可以写出一个完全不同的算法。

上述的学生通讯录就是一个数据结构问题。由此看到，计算机算法与数据的结构密切相关，算法无不依附于具体的数据结构，数据结构直接关系到算法的选择和效率。

下面再对学生通讯录做进一步讨论。众所周知，当有新学生进校时，通讯录需要添加新

学生的姓名和相应的住址；在学生毕业离校时，应从通讯录中删除毕业学生的姓名和住址。这就要求在已安排好的结构上进行插入（insert）和删除（delete）。对于一种具体的结构，如何实现插入和删除？是把要添加的学生姓名和住址插入到前头，还是末尾，或是中间某个合适的位置上？插入后，对原有的数据是否有影响？有什么样的影响？删除某学生的姓名和住址后，其他数据（学生的姓名和住址）是否要移动？若需要移动，则应如何移动？这一系列的问题说明，为适应数据的增加和减少的需要，还必须对数据结构定义一些运算。上面只涉及两种运算，即插入和删除运算。当然，还会提出一些其他可能的运算，如学生搬家后，住址变了，为适应这种需要，就应该定义修改（modify）运算等。

这些运算显然是由计算机来完成，这就要设计相应的插入、删除和修改的算法。也就是说，数据结构还需要给出每种结构类型所定义的各种运算的算法。

通过以上讨论，可以直观地认为：数据结构是研究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一门学科。

1.2 基本概念和常用术语

本节将讲解与数据结构相关的一些重要的基本概念和常用术语。这些概念和术语将在后续章节中多次出现。

1. *数据*

数据是描述客观事物的数值、字符以及能输入机器且能被处理的各种字符的集合，即计算机化的信息。换句话说，数据就是对客观事物采用计算机能够识别、存储及处理的形式所进行的描述。

在计算机科学中，数据的含义非常广泛，把一切能够输入到计算机中并能被计算机程序处理的信息，包括文字、表格、图像等，统称为数据。例如，一个学生成绩管理程序所要处理的数据，如表 1-1 所示。

表 1-1 学生成绩表

学号	姓名	数据结构	大学物理	高等数学	平均成绩
0232101	王刚	95	90	85	90
0232102	李娟	90	80	85	85
0232103	赵平	99	95	91	95
0232104	王强	86	70	84	80
0232105	张雪	92	91	84	89

2. *数据元素*

数据元素也称为结点，它是组成数据的基本单位，是一个数据整体中相对独立的单元。例如，在表 1-1 中，为了便于处理，把其中的每一行（代表一个学生成绩）作为一个基本单位来考虑，故该数据由 5 个结点构成。

一般情况下，一个结点还可以分割成若干具有不同属性的字段（也称为数据项）。例如，在表 1-1 中，每个结点都由学号、姓名、数据结构、大学物理、高等数学和平均成绩 6 个字段构成。字段是构成数据的最小单位。

3. 数据对象

在数据结构中，将性质相同的数据元素的集合称为数据对象，它是数据的一个子集。上例：一个班级的学生成绩表可以看作一个数据对象。

4. 数据结构

数据结构由某一数据元素集合及该集合中所有数据元素之间的关系组成。具体来说，数据结构包含 3 个方面的内容，即数据的逻辑结构、数据的存储结构和对数据所施加的操作。例如，在表 1-1 中，除了有 5 个学生成绩的数据外，这 5 条记录还存在着一对一的关系（逻辑结构）以及这些记录在存储器中以怎样的方式进行存储（存储结构）。

根据数据结构中数据元素之间的结构关系的不同特征，通常将数据结构分为如下四种基本结构：

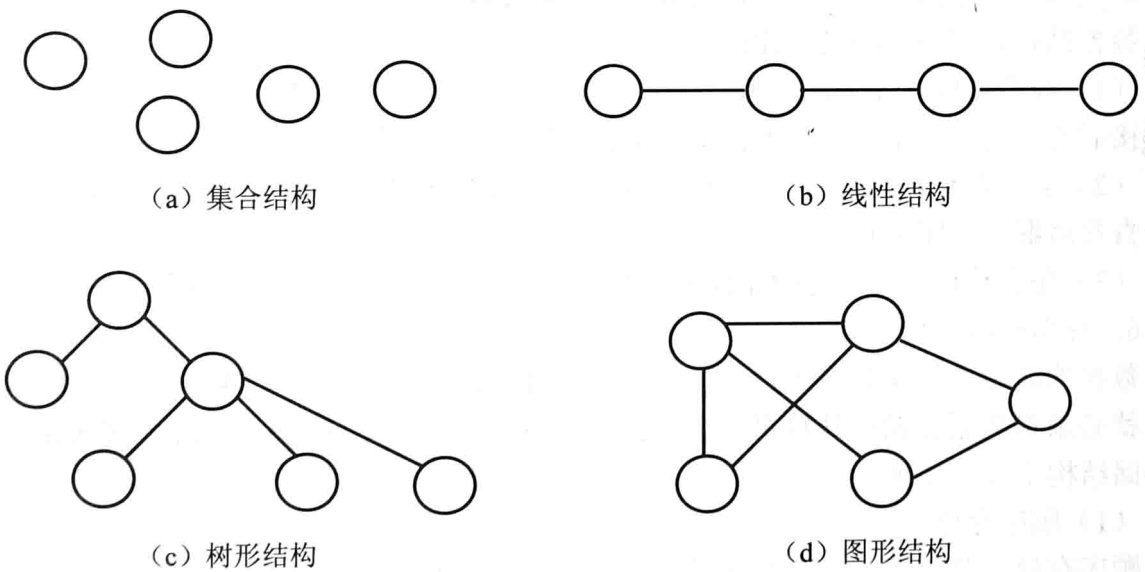
（1）集合结构（set）：数据元素的有限集合。数据元素之间除了“属于同一个集合”的关系之外没有其他关系。元素顺序是随意的。

（2）线性结构（linear）或称为序列（sequence）结构：数据元素的有序集合。数据元素之间形成一对一的关系。

（3）树形结构（tree）：树是层次结构，树中数据元素之间存在一对多的关系。

（4）图形结构（graph）：图中数据元素之间的关系是多对多的。

图 1-1 给出了上述四种基本结构的示意图。



（a）集合结构　（b）线性结构

（c）树形结构　（d）图形结构

图 1-1　四种基本结构示意图

数据结构是一个二元组，其形式定义为：

Data_Structure=(D,S)

其中：D 是数据元素的有限集，S 是 D 上关系的有限集。下面举两个简单例子说明之。

【例 1-1】部门的上级领导下级的数据结构（见图 1-2）：a 领导 b，a 领导 c，b 领导 d，b 领导 e。可以如下定义其数据结构：

T=(D,R)

其中：D 是数据元素的集合 D={a,b,c,d,e}

R 是 D 上的关系集合 R={<a,b>,<a,c>,<b,d>,<b,e>}

【例 1-2】一小组有 a,b,c 三个学生，一个导师 A 和一个辅导员 B，此小组的数据结构如图 1-3 所示。可以定义如下数据结构：

T=(D,R)

其中：D={A,B,a,b,c}

R={P1,P2}

P1={<A,a>,<A,b>,<A,c>}

P2={<B,a>,<B,b>,<B,c>}

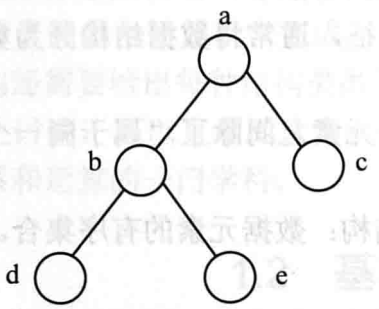


图 1-2 部门上下级领导关系图

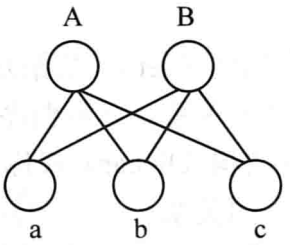


图 1-3 导师和辅导员与学生关系图

5. 逻辑结构

结点和结点之间的逻辑关系称为数据的**逻辑结构**。

数据结构从逻辑结构划分为：

（1）线性结构。元素之间为一对一的线性关系，第一个元素无直接前驱，最后一个元素无直接后继，其余元素都有一个直接前驱和直接后继，见图 1-1（b）。

（2）非线性结构。元素之间为一对多或多对多的非线性关系，元素可有多个直接前驱或多个直接后继，见图 1-1（c）、（d）。

（3）集合结构。元素之间无任何关系，元素的排列无任何顺序，见图 1-1（a）。

6. 存储结构

数据的逻辑结构是独立于计算机的，它与数据在计算机中的存储无关，要对数据进行处理，就必须将数据存储在计算机中。数据在计算机中的存储方式称为数据的**存储结构**。数据的存储结构主要有 4 种。

（1）顺序存储。

顺序存储通常用于存储具有线性结构的数据。将逻辑上相邻的结点存储在连续存储区域 M 的相邻的存储单元中，使得逻辑相邻的结点一定是物理位置相邻。这种映像通过物理上存储单元的相邻关系来体现结点间相邻的逻辑关系。

例如对于一个数据结构 B=(K,R)，其中：

K={k1,k2,k3,k4,k5,k6,k7,k8,k9}

R={r}

r={<k1,k2>,<k2,k3>,<k3,k4>,<k4,k5>,<k5,k6>,<k6,k7>,<k7,k8>,<k8,k9>}

它的顺序存储方式如图 1-4 所示。

（2）链式存储。

链式存储方式是给每个结点附加一个指针段，一个结点的指针所指的是该结点的后继的

存储地址，因为一个结点可能有多个后继，所以指针段可以是一个指针，也可以是多个指针。在链式存储中逻辑相邻的结点在连续存储区域 M 中可以不是物理相邻的。前面讲到，数据的存储结构一定要体现它的逻辑结构，这里是通过指针来体现的。

例如数据的逻辑结构 B=(K,R)，其中：

K={k1,k2,k3,k4,k5}

R={r}

r={<k1,k2>,<k2,k3>,<k3,k4>,<k4,k5>}

这是一个线性结构，它的链式存储如图 1-5 所示。很明显，在这种存储方式下，必须要知道开始结点的存储地址。

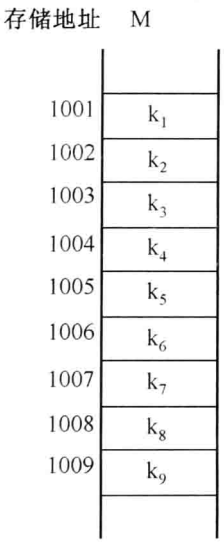


图 1-4　顺序存储的图示

存储地址	info	next
1000		
1001	k_1	1003
1002		
1003	k_2	1007
1004		
1005	k_4	1006
1006	k_5	∧
1007	k_3	1005
1008		

图 1-5　一个线性结构的链式存储的图示

例如数据的逻辑结构 B=(K,R)，其中：

K={k1,k2,k3,k4,k5}

R={r}

r={<k1,k2>,<k1,k3>,<k2,k4>,<k4,k5>}

这是一种树型结构，它的链式存储表示如图 1-6 所示。

存储地址	数据	指针1	指针2
1000	k_1	1001	1006
1001	k_2	1005	∧
1002			
1003	k_5	∧	∧
1004			
1005	k_4	1003	∧
1006	k_3	∧	∧
1007			
1008			

图 1-6　一个树型结构的链式存储的图示

（3）索引存储。

在线性结构中，设开始结点的索引号为1，其他结点的索引号等于其前驱结点的索引号加1，则每一个结点都有唯一的索引号，根据结点的索引号可确定该结点的存储地址。例如，一本书的目录就是各章节的索引，目录中每个章节后面标示的页码就是该章节在书中的位置。如果某本书的每个章节所占页码相同，那么可以由一个线性函数来确定每个章节在书中的位置。

（4）散列存储。

散列存储的思想是构造一个从集合K到存储区域M的函数h，该函数的定义域为K，值域为M，K中的每个结点K_i在计算机中的存储地址由$h(K_i)$确定。

一个数据结构存储在计算机中，整个数据结构占的存储空间一定不小于数据本身所占的存储空间，通常把数据本身所占存储空间的大小与整个数据结构所在存储空间的大小之比称为数据结构的存储密度。显然，数据结构的存储密度不大于1。顺序存储的存储密度为1，链式存储的存储密度小于1。

7. 数据处理

数据处理是指对数据进行查找、插入、删除、合并、排序、统计以及简单计算等操作的过程。在早期，计算机主要用于科学和工程计算。20世纪80年代以来，计算机主要用于各种非数值数据的处理。据有关统计资料表明，现在计算机用于数据处理的时间的比例达到80%以上，而且随着时间的推移和计算机应用进一步的普及，计算机用于数据处理的时间比例必将进一步增大。

8. 数据类型

数据类型是指程序设计语言中各变量可取的数据种类，它是高级程序设计语言中的一个基本概念，和数据结构的概念密切相关。一方面，在程序设计语言中，每一个数据都属于某种数据类型。类型明显或隐含地规定了数据的取值范围、存储方式以及允许进行的运算，可以认为，数据类型是在程序设计中已经实现了的数据结构。另一方面，在程序设计过程中，当需要引入某种新的数据结构时，总是借助编程语言所提供的数据类型来描述数据的存储结构。

9. 算法

简单地说，算法就是解决特定问题的方法。特定问题可以是数值的，也可以是非数值的。

解决数值问题的算法称为数值算法。科学和工程计算方面的算法都属于数值算法，如求解数值积分、求解线性方程组、求解代数方程以及求解微分方程等。解决非数值问题的算法称为非数值算法。数据处理方面的算法都属于非数值算法，如各种排序算法、查找算法、插入算法、删除算法以及遍历算法等。数值算法和非数值算法并没有严格的区别。一方面，在数值算法中主要进行算术运算，而在非数值算法中主要进行比较和逻辑运算；另一方面，特定的问题可能是递归的，也可能是非递归的，因而解决它们的算法就有递归算法和非递归算法之分。从理论上讲，任何递归算法都可以通过循环或堆栈等技术转化为非递归算法。

1.3 数据抽象和抽象数据类型

1.3.1 数据抽象

抽象（abstraction）可以被理解为一种机制，其实质是抽取共同的和实质的东西，忽略非

本质的细节。抽象可以使求解问题过程以自顶向下的方式分步进行：首先考虑问题的最主要方面，然后再逐步细化，进一步考虑问题的某些细节，并最终实现之。

在程序设计中，数据和运算是两个不可缺少的因素。所有的程序设计活动都是围绕着数据和其上的相关运算而进行的。从机器指令、汇编语言中的数据没有类型的概念，到现在的面向对象程序设计语言中抽象数据类型概念的出现，程序设计中的数据经历了一次次抽象，数据的抽象经历了三个发展阶段。

第一个发展阶段是从无类型的二进制数到基本数据类型的产生。

第二个发展阶段是从基本数据类型到用户自定义类型的产生。

第三个发展阶段是从用户自定义类型到抽象数据类型的出现。

抽象数据类型是用户自己定义类型的一个机制。数据和运算（即对数据的处理）是程序设计的核心，数据表示的复杂性决定了其上运算实现的复杂性，这也是整个系统复杂性的关键所在。60 年代末期出现的“软件危机”就是因为在软件开发中不能有效地控制数据表示，无法控制整个软件系统的复杂性，最终导致软件系统的失败。“软件危机”使人们认识到，在基于功能抽象的模块化设计方法中，模块间的连接是通过数据进行的，数据从一个模块传送到另一个模块，每一个模块在其上施加一定的操作，并完成一定的功能。尽管 Pascal、C 等语言的用户自定义类型机制使得用户可以将这些连接数据作为某种类型的对象直接处理，然而，由于这些用户自定义类型的表示细节是对外部公开的，没有任何保护措施，程序设计人员可以随意地修改这种类型对象的某些成分，添加一些不合法的操作，而处理这个数据对象的其他模块却一无所知，从而危害整个软件系统。这些不利因素将会给由多人合作进行的大型软件系统开发带来致命的危害。为了有效地控制大型程序系统的复杂性，必须从两个方面加以考虑：一方面是更新程序设计语言中的类型定义机制，使类型的内部表示细节对外界不可见，程序设计中不需要依赖于数据的某种具体表示；另一方面要寻求连接模块的新方法，尽可能缩小模块间的界面。面向对象程序设计语言 C++中的类就是实现抽象数据类型的机制。

1.3.2　抽象数据类型

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仅取决于它的一组逻辑特性，而与其在计算机内部如何表示和实现无关，即不论其内部结构如何变化，只要它的数学特性不变，都不影响其外部的使用。

抽象数据类型是数据类型的进一步抽象，是大家熟知的基本数据类型的延伸和发展。众所周知，整数类型是整数值数据模型（或简单理解为整数）和加、减、乘、除四则运算等的统一体，人们在程序设计中大量使用整数类型及其相关的四则运算，而没有人去追究这些运算是如何实现的。如果人们问到此问题，有人可能会简单回答：计算机自动进行处理。实际上，每一种基本数据类型都有一组与其相关的运算，而这组运算的具体实现都被封装起来，人们不知道也确实不必去关心这些细节。试想一下，如果程序设计人员在程序设计中要考虑基本数据类型的相关运算是如何具体实现的，那将是多么繁杂和令人头痛的事情，程序中的错误也会增加许多。这其中就孕育着抽象数据类型的思想。将基本数据类型的概念进行延伸，程序设计人员可以在进行问题求解时，首先确定该问题所涉及的数据模型以及定义在该数据模型上的运算集合，然后求出从数据模型的初始状态到达目标状态所需的运算序列，就成为该问题的求解算法，这样做就是一种抽象，它使得用户不必去关心数据模型中的数据具体表示及相关运算的具体实

现，这将大大提高软件开发中的开发效率和程序的正确性等。

抽象数据类型是与表示无关的数据类型，是一个数据模型及定义在该模型上的一组运算。对一个抽象数据类型进行定义时，必须给出它的名称及各运算的运算符名，即函数名，并且规定这些函数的参数性质。一旦定义了一个抽象数据类型及具体实现，程序设计中就可以像使用基本数据类型那样，十分方便地使用抽象数据类型。

1.3.3 抽象数据类型的描述和实现

抽象数据类型的描述包括给出抽象数据类型的名称、数据的集合、数据之间的关系和操作的集合等方面的描述。抽象数据类型的设计者根据这些描述给出操作的具体实现，抽象数据类型的使用者依据这些描述使用抽象数据类型。

抽象数据类型形式化定义可以用以下三元组表示：

ADT=(D,S,P)

其中，D 是数据对象，S 是 D 上的关系集，P 是对 D 的基本操作集。

抽象数据类型描述的一般形式如下：

```
ADT 抽象数据类型名称{
        数据对象：
            ……
        数据关系：
            ……
        操作集合：
            操作名 1：
            ……
            操作名 n：
}ADT 抽象数据类型名称
```

抽象数据类型的具体实现依赖于程序设计语言。面向对象的程序设计语言中的“类”支持抽象数据类型、支持信息隐藏。本书设定读者使用 C 语言进行程序设计，书中使用 C 语言进行算法的描述和实现，而 C 语言在对抽象数据类型的支持上是欠缺的，一个抽象数据类型的不同实现，如一个使用顺序存储，一个使用链式存储，尽管在不同的实现中可以使同一操作对应的函数名相同，但是抽象数据类型的操作所对应的函数原型一般是不同的。因此，要用 C 语言完整地实现抽象数据类型是不现实的。本书采用介于伪码和 C 语言之间的类 C 语言作为描述工具，有时也用伪码描述一些只含抽象操作的抽象算法。这使得数据结构与算法的描述和讨论简明清晰，不拘于 C 语言的细节，又能容易转换成 C 或者 C++程序。

本书采用类 C 语言精选了 C 语言的一个核心子集，同时做了若干扩充修改，增强了语言的描述功能，以下对其做简要说明。

（1）预定义常量和类型。

```
#define TRUE 1
#define FALSE 0
#define OK 1
#define ERROR 0
```

#define INFEASIBLE -1
#define OVERFLOW -2
typedef in Status; //Status 是函数的类型，其值是函数结果状态代码

（2）数据结构的存储结构。

typedef ElemType first;

（3）基本操作的算法。

函数类型 函数名（函数参数表）{
//算法说明
语句序列
}//函数名

（4）赋值语句。

- 简单赋值：
  变量名=表达式；
- 串联赋值：
  变量名 1=变量名 2=…=变量名 k=表达式；
- 成组赋值：
  （变量名 1，…，变量名 k）=（表达式 1，…，表达式 k）；
  结构名=结构名；
  结构名=（值 1，…，值 k）；
  变量名[]=表达式；
  变量名[起始下标…终止下标]=变量名[起始下标…终止下标]；
- 交换赋值：
  变量名<-->变量名；
- 条件赋值：
  变量名=条件表达式？表达式？表达式 T：表达式 F

（5）选择语句。

- if（表达式）语句；
- if（表达式）语句；
  else 语句；
- switch（表达式）{
  case 值 1：语句序列 1；break；
  …
  case 值 n：语句序列 n；break；
  default：语句序列 n+1；break；
  }
- switch{
  case 条件 1：语句序列 1；break；
  …
  case 条件 n：语句序列 n；break；

default：语句序列 n+1；break；

}

（6）循环语句。

for（赋初值表达式；条件；修改表达式序列）语句；

while（条件）语句；

do{语句序列}while（条件）；

（7）结束语句。

return [表达式]；

return；//函数结束语句

break；//case 结束语句

exit（异常代码）；//异常结束语句

（8）输入和输出语句。

scanf（[格式串]，变量 1，…，变量 n）；

（9）注释。

//文字序列

（10）基本函数。

max（表达式 1，…，表达式 n）

min,abs,floor,ceil,eof,eoln

（11）逻辑运算。

&&与运算；||或运算

1.4 算法和算法分析

1.4.1 算法及性能标准

为了求解某问题，必须给出一系列的运算规则，这一系列的运算规则是有限的，表达了求解问题的方法和步骤，这就是一个算法。

一个算法可以用自然语言描述，也可以用高级程序设计语言描述，也可以用伪代码描述。本书采用 C 语言对算法进行描述。算法具有五个基本特征：

① 有穷性，算法的执行必须在有限步内结束。

② 确定性，算法的每一个步骤必须是确定的无二义性的。

③ 输入，算法可以有 0 个或多个输入。

④ 输出，算法一定有输出结果。

⑤ 可行性，算法中的运算都必须是可以实现的。

衡量一个算法的性能，主要有以下几个标准：

① 正确性（correctness）：算法的执行结果应当满足预先规定的功能和性能要求。

② 简明性（simplicity）：一个算法应当思路清晰、层次分明、简单明了、易读易懂。

③ 健壮性（robustness）：当输入不合法数据时，应能做适当处理，不至于引起严重后果。

④ 效率（efficiency）：有效使用存储空间，并有高的时间效率。