21世纪 职业教育规划教材

数据结构实用教程（C语言版）

董凤服 刘 畅 王 茹 编 著

中国水利水电出版社
www.waterpub.com.cn

21世纪职业教育规划教材

数据结构实用教程（C语言版）

董凤服　刘　畅　王　茹　编　著

中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

数据结构是计算机专业的一门专业基础课，也是一门核心课程。本书是以C语言为工具编写的，学习本课程需要具备相关的C语言知识。本书介绍了各种常用的数据结构，讨论它们在计算机中的存储结构及相关操作和实用算法。

为了让学生能够应用数据结构的知识，更好地进行算法和程序的设计，本书从基本概念讲起，由浅入深，介绍各种数据结构及相关操作。在每章的课后习题里，举了很多数据结构的经典题型并给出答案，方便学生理解所学知识。为提高动手能力，在每章结束时举了很多实例，力求理论联系实际，加深学生对所学知识的理解，方便学生上机实习。

最后以线性表、栈和队列、树和图四种最基本数据结构为主，给出相关的实训内容并包括所有上机程序，极大地方便了应用型高职学生的学习及应用。

本书主要适合高等职业院校的计算机专业学生，也适合于自学计算机相关知识的人员参考使用。

本书电子教案可以从中国水利水电出版社网站免费下载，网址为：http://www.waterpub.com.cn/softdown/。

图书在版编目（CIP）数据

数据结构实用教程：C语言版 / 董凤服等编著. －北京：中国水利水电出版社，2008

21世纪职业教育规划教材

ISBN 978-7-5084-5709-3

Ⅰ. 数… Ⅱ. 董… Ⅲ. ①数据结构—高等学校：技术学校—教材②C语言—程序设计—高等学校：技术学校—教材 Ⅳ. TP311.12 TP312

中国版本图书馆CIP数据核字（2008）第111088号

书　　名	21世纪职业教育规划教材 **数据结构实用教程（C语言版）**
作　　者	董凤服　刘　畅　王　茹　编　著
出版发行	中国水利水电出版社（北京市三里河路6号　100044） 网址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）63202266（总机）、68367658（营销中心）、82562819（万水）
经　　售	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	北京市天竺颖华印刷厂
规　　格	184mm×260mm　16开本　15.5印张　379千字
版　　次	2008年9月第1版　2008年9月第1次印刷
印　　数	0001—3000册
定　　价	26.00元

序

自 1998 年教育部机构改革以后，高等职业教育、成人职业教育、中等职业教育“三教统筹”，各具特色，形成了共同发展职业教育的可喜局面。根据国务院《关于大力发展职业教育的决定》（国发[2005]35 号）和周济部长 2005 年 6 月 14 日在《全国县级职业教育中心改革与发展座谈会上的讲话》精神，根据职业教育“培养生产、服务、管理第一线需要的实用人才”和推行“半工半读、工学结合，强化实践教学”等规定文件精神，结合当前我国职业教育改革发展实际情况，对我国传统的教学模式提出了挑战，以提高人才培养质量为目的、人才培养模式改革与创新为主题的专业教学改革势在必行。

职业教育的培养目标较宽泛，其上限为技术型人才，下限为技能操作型人才，而主体则为技术应用型人才。以培养技术应用能力和提高职业素质为主线，设计学生的知识、能力和素质结构是职业教育改革的重点。在职业教育改革发展的同时，出现了许多亟待解决的问题，其中最主要的是按照职业教育培养目标的要求，培养一批“双师型”的骨干教师，编写出一批有特色的基础课程和专业主干课程教材。

教材改革是职业院校教育改革的重点，是职业院校学科建设的关键，是教学改革的基础。为解决当前职业教材匮乏的现象，由中国水利水电出版社/北京万水电子信息有限公司精心策划，与全国数十所职业院校联合组织编写了这套“21 世纪职业教育规划教材”。本套教材全面贯彻国家有关职业教育改革文件精神，从策划到主编、主审的遴选，从成立专家组反复讨论教学大纲，研究系列教材特色特点到书稿的字斟句酌、实例的选取，每一步都力争精益求精，充分考虑当前职业院校学生的特点，在编写教材中，以最新的理论为指导，以实例化操作为主线，通过案例引入、知识拓宽、综合训练等环节使学生掌握最基本的操作技能方法。

本套教材凝聚了数百名奋斗在职业教育第一线的教师多年的教学经验和智慧，教材内容选取新颖、实用，层次清晰，结构合理，文笔流畅，质量上乘。

本套教材涉及计算机、电子、数控、机械等专业的基础课和专业课课程，适合当前我国各类职业院校作为教材使用。

大力发展职业教育，加快人力资源开发，是落实科教兴国战略和人才强国战略，推进我国走新型工业化道路，解决“三农”问题，促进就业再就业的重大举措；是提高国民素质，把我国巨大人口压力转化为人力资源优势，提升我国综合国力，构建和谐社会的重要途径；是贯彻党的教育方针，遵循教育规律，实现教育事业全面协调可持续发展的必然要求。相信这套“21 世纪职业教育规划教材”的出版能为我国职业教育的教学改革和教材建设略尽绵薄之力。

金无足赤，人无完人，本套教材难免会有不足之处，恳请各位专家和读者批评指正。

21 世纪职业教育规划教材编委会

2006 年 6 月

前　言

数据结构是计算机专业的一门专业基础课，也是一门核心课程，在计算机专业课程的学习中起到承前启后的作用。本书是用C语言编写的，学习该课程需要先学习C语言的相关知识。本书结合计算机专业的教学大纲进行编写，介绍了各种常用的数据结构，并讨论它们在计算机中的存储结构及相关操作和实用算法。

为了让学生能够应用数据结构的知识更好地学习算法和程序设计，本书从基本概念讲起，由浅入深，介绍各种数据结构及相关操作，在每章的课后习题里，给出很多数据结构的经典题型并附答案，方便学生理解所学知识，加深学生对知识点的理解并提高学生的自学能力。

本教材共分9章。第1章“概论”主要介绍数据结构的基本概念和算法描述。第2章“线性表”介绍线性表的逻辑结构、存储结构和基本操作的实现。第3章“栈和队列”介绍栈和队列的特点及其各种存储结构与基本操作的实现，并给出了相应的实例。第4章“串和数组”介绍串的各种存储结构与基本操作的实现，数组的各种存储结构及稀疏矩阵的基本概念。第5章“树和二叉树”介绍树的定义与表示、二叉树的基本操作及哈夫曼树。第6章“图”介绍图的各种存储结构和遍历的实现，以及各种图的实际应用，如图的最小生成树、关键路径及最短路径问题等。第7章“查找”介绍了各种常用的查找方法及其实现。第8章“排序”介绍了各种常用的排序方法及其实现。第9章“上机实训”主要针对高职教育强调动手能力的要求，给出了各种重要数据类型的典型综合实例及相关的流程和分析过程，并且给出所有上机的源程序和运行结果。加深学生对所学知识的理解，体现了应用型高职院校的高技能型人才的培养目标。

本书的特色在于书中的所有算法全部在Turbo C 2.0下调试通过，对每个算法给出可以调用其运行的主函数及其相关辅助函数，并给出该算法的运行参考结果，方便学生调试运行程序，使学生更加深刻地理解程序设计过程中的算法编写技巧（如函数的调用及参数传递方式）。本书主要适合于高等职业院校的计算机专业学生，也适合于自学计算机相关知识的人员参考使用。

本书由董凤服、刘畅、王茹编著。由于水平所限，尽管编者不遗余力，仍可能存在错误和不足之处，敬请读者批评指正，本书还配套有相应的章节课件及完整的C语言源程序代码（已在Turbo C 2.0下全部调试通过），需要者可至中国水利水电出版社免费下载，网址为：http://www.waterpub.com.cn/softdown/。

编者

2008年5月

目　　录

第 1 章　概论

本章主要介绍以下内容：

- 数据结构中涉及的相关概念
- 数据结构研究的主要内容
- 算法的概念、描述方法及评价标准

随着计算机技术的飞速发展，计算机编程语言的操作对象的关系也变得更加复杂，操作形式不再是单纯的数值计算，而更多的是对这些具有一定关系的数据进行组织和管理，我们将此称为**非数值性处理**。要使计算机能够更有效地进行这些非数值性处理，就必须弄清楚这些操作对象的特点、在计算机中的表示方式以及各个操作的具体实现手段。这些就是“数据结构”这门课程研究的主要内容。

1.1　数据结构的概念

1.1.1　基本概念及术语

在系统地学习数据结构知识之前，要先了解一些相关概念和术语。

1. *数据*（Data）

数据是指所有能输入到计算机中并被计算机程序处理的符号的总称，如整数、实数、字符、图像、声音等都是数据。

2. *数据元素*（Data Element）

数据元素（也称为结点）是数据的基本单位，在计算机程序中通常作为一个整体进行考虑和处理。一个数据元素可以由若干个数据项组成。例如，一个学生的信息为一个数据元素，而一个学生信息中的每一项（如学号、姓名、性别等）为一个数据项。数据项是数据处理中不可分割的最小单位。

3. *数据结构*（Data Structure）

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集合。这些数据元素不是孤立存在的，而是有着某种关系，这种关系称为结构。

数据结构一般包括以下三个方面的内容：

（1）数据元素之间的逻辑关系，也称数据的逻辑结构。

（2）数据元素及其关系在计算机存储器内的表示，称为数据的存储结构。

（3）数据的运算，即对数据施加的操作。

数据结构定义：按某种逻辑关系组织起来的一批数据，按一定的映像方式把它们存放在计算机存储器中，并在这些数据上定义了一个运算的集合，就叫做数据结构。

简言之，**数据结构={ 逻辑结构+存储结构+运算集合 }**。

4. *数据类型*（Data Type）

数据类型是一组性质相同的值集合以及定义在这个值集合上的一组操作的总称。

如在高级语言中，整型的取值范围为-32768～+32767，运算符集合为加、减、乘、除、取模，即+、-、*、/、%。高级语言中的数据类型分为两大类：

（1）原子类型。其值不可分解，如 C 语言中的标准类型（整型、实型、字符型）。

（2）结构类型。其值是由若干成分按某种结构组成的，因此是可以分解的，如 C 语言中的构造类型（结构体、共用体、枚举等类型）。

1.1.2 数据的逻辑结构

1. 定义

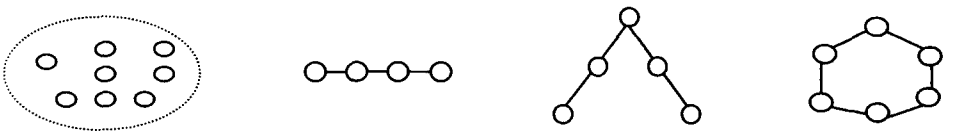
数据的逻辑结构是指数据元素之间的逻辑关系描述。可以用一个二元组表示，其形式化描述为：

$$Data\_Structure=(D,R)$$

其中，D 是数据元素的有限集合；R 是 D 上关系的有限集合。数据的逻辑结构是从逻辑关系上描述数据，与数据的存储结构无关，是独立于计算机的。

2. 数据的逻辑结构的分类

根据数据元素之间的逻辑关系的不同特性，分为如图 1-1 所示的四类基本结构。



（a）集合结构　（b）线性结构　（c）树型结构　（d）图形结构

图 1-1 数据的四种基本逻辑结构

（1）集合结构。

集合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，这是一种最简单的数据结构。

（2）线性结构。

线性结构中的数据元素之间存在着“**一对一**”的关系。

【例 1.1】学籍档案管理。

假设一个学籍档案管理系统包含如表 1-1 所示的学生信息。

表 1-1 学生信息登记表

学　号	姓　名	性　别	出生年月	……
20080101	王　刚	男	89.11	……
20080102	张振羽	男	90.01	……
20080103	刘雨婷	女	89.05	……
20080104	齐宏涛	男	88.06	……
……	……	……	……	……

特点：

表中的每一行是一个数据元素（或记录、结点），它由学号、姓名、性别及出生年月等数据项组成。

表中数据元素之间是一种先后关系，对于表中任一结点，与它相邻且在它前面的结点（称为直接前驱）最多只有一个；与表中任一结点相邻且在其后的结点（称为直接后继）也最多只有一个。我们将这种关系称为“线性结构”。

（3）树型结构。

树型结构中的数据元素之间存在着“**一对多**”的关系。

【例 1.2】人机对弈。

人与计算机进行对弈的示意图如图 1-2 所示。

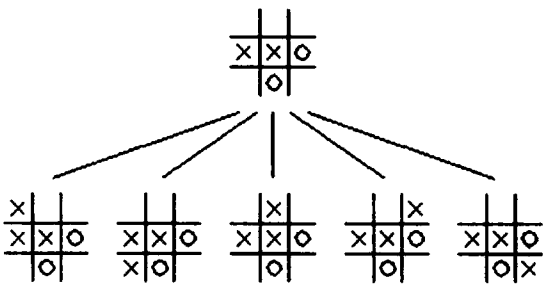


图 1-2　人机对弈图

特点：

图中将每一个棋盘看做一个数据元素，则数据元素之间的关系要比表 1-1 所示复杂许多。

图中数据元素之间是一对多关系，即一个数据元素向上和一个数据元素相连（称为双亲结点），向下和多个数据元素相连（称为孩子结点）。我们将这种关系称为“树型结构”。

（4）图形结构（或网状结构）。

结构中的任意数据元素之间都可以有关系，元素之间存在着“**多对多**”的关系。

【例 1.3】制定教学计划。

在制定教学计划时，需要考虑各门课程的开设顺序。有些课程需要先修课程，有些课程则不需要，而有些课程又是其他课程的先修课程。比如，计算机专业课程的开设情况如表 1-2 所示。

表 1-2　计算机专业学生的必修课程

课程编号	课程名称	先修课程
C_1	高等数学	无
C_2	程序设计基础	无
C_3	普通物理	C_1
C_4	离散数学	C_1,C_2
C_5	C 语言程序设计	C_2
C_6	数据结构	C_2,C_4, C_5
C_7	编译原理	C_5,C_6
C_8	计算机原理	C_3
C_9	操作系统	C_6, C_8

则上述必修课程的关系如图 1-3 所示。

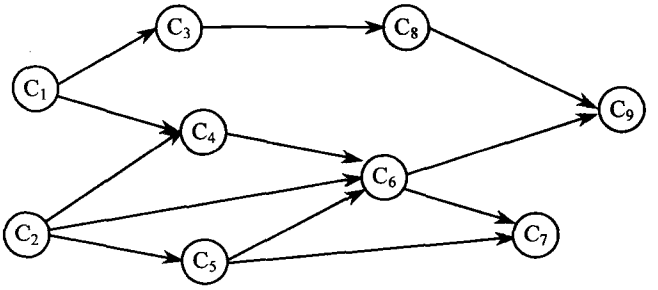


图 1-3 必修课程关系

特点：

图中数据元素存在着多对多的任意关系。一个结点可能有多个直接前驱和直接后继。

1.1.3 数据的存储结构

数据在计算机中的存储称为**数据的存储结构**，也称为**物理结构**。数据的存储结构是逻辑结构在计算机存储器中的实现。本书将介绍两种常用的基本存储结构：顺序存储结构和链式存储结构。

数据的逻辑结构和存储结构的关系是：存储结构是逻辑关系的映像与元素本身的映像，是数据结构的实现；逻辑结构是数据结构的抽象。

1. 顺序存储结构

顺序存储结构是借助元素在存储器中的相对位置来表示数据元素间的逻辑关系。

【例 1.4】对表 1-1 所示的学生信息登记表进行存储，假定每个元素占用 50 个存储单元，数据从 1000 号单元开始由低地址向高地址存放，对应的顺序存储结构如表 1-3 所示。

表 1-3 学生信息登记表的顺序存储结构

存储地址	学 号	姓 名	性 别	出生年月	……
1000	20080101	王 刚	男	89.11	……
1050	20080102	张振羽	男	90.01	……
1100	20080103	刘雨婷	女	89.05	……
1150	20080104	齐宏涛	男	88.06	……
……	……	……	……	……	……

顺序存储结构的主要特点如下：

- 可实现对各数据元素的随机访问。这是因为只要知道存储的首地址以及每个数据元素所占的存储单元，就可以计算出各数据元素的存储地址。
- 不利于修改，在对数据元素进行插入、删除运算时可能要移动一系列的数据元素。

2. 链式存储结构

链式存储结构是借助指示元素存储地址的指针来表示数据元素间的逻辑关系。

【例 1.5】对表 1-1 所示学生信息登记表进行链式存储时，在每个数据元素后边附加一个

指向“下一个结点地址”的指针字段，用于存放后继数据元素的存储地址，每个数据元素的地址是随机的，可以不连续。对应的链式存储结构如表 1-4 所示。

表 1-4　学生信息登记表的链式存储结构

存储地址	学　号	姓　名	性　别	出生年月	……	下一个结点地址
2020	20080101	王　刚	男	89.11	……	1680
1680	20080102	张振羽	男	90.01	……	3400
3400	20080103	刘雨婷	女	89.05	……	4520
4520	20080104	齐宏涛	男	88.06	……	……
……	……	……	……	……	……	……

链式存储结构的主要特点如下：

- 利于修改，在对数据元素进行插入、删除运算时，仅需修改数据元素的指针字段值，而不必移动数据元素。
- 由于逻辑上相邻的数据元素在存储位置中不一定相邻，因此，链式存储结构不能对数据元素进行随机访问。

1.1.4　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。

2. 抽象数据类型的表示

抽象数据类型实际上就是对该数据结构的定义。因为它定义了一个数据的逻辑结构以及在此结构上的一组算法。可以用一个三元组表示为：

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

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。

抽象数据类型通常是指由用户定义，用以表示应用问题的数据类型，抽象数据类型由基本的数据类型组成，并包括一组相关的服务（或称操作）。

抽象数据类型有些类似于 Pascal 语言中的记录（record）类型和 C 语言中的构造（struct）类型，但它增加了相关的服务。

3. 抽象数据类型的两个重要特征

（1）数据抽象用抽象数据类型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。

1.2　算法和算法分析

1.2.1　算法的概念

1. 算法的定义

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”，所谓算法，

就是为解决特定问题而采取的步骤和方法。

2. 算法的特性

（1）有穷性：一个算法必须（对任何合法的输入值）在执行有限步之后结束。

（2）确定性：算法中的每一条指令必须有确切的含义，不会产生二义性。

（3）可行性：算法中描述的操作都可以通过执行有限次基本操作来实现。

（4）输入：一个算法有零个或多个输入。

（5）输出：一个算法必有一个或多个输出。

3. 算法的评价

要设计一个好的算法，通常需要考虑以下几个方面的要求。

（1）正确性：要求算法能够正确地执行预先规定的功能，并达到所期望的性能要求。

（2）可读性：为了便于理解、测试和修改算法，算法应该具有良好的可读性。

（3）健壮性：当输入非法的数据时，算法应能恰当地做出反应或进行相应处理，而不是产生莫名奇妙的输出结果。并且处理出错的方法不应是中断程序的执行，而是返回一个表示错误或错误性质的值，以便在更高的抽象层次上进行处理。

（4）高效性：对同一个问题，执行时间越短，算法的效率越高。

（5）低存储量：完成相同的功能，执行算法所占用的存储空间应尽可能少。

4. 算法的描述

为了表示一个算法，可以用多种不同的方法，常用的有自然语言、传统流程图、结构化流程图、N-S 流程图等。本书采用 C 语言实现对各种数据结构及算法的操作描述，算法是以函数形式描述，语法格式如下：

```
类型标识符  函数名(形式参数表)
/*算法说明*/
{ 语句序列 }
```

1.2.2 算法分析

对于求解同一个问题，可以设计出若干个算法，对于不同的算法进行性能分析是数据结构的一个重要内容。在算法满足正确性的前提下，如何评价不同算法的优劣呢？通常主要考虑算法的时间复杂度和空间复杂度这两个方面。一般情况下，鉴于运算空间（内存）较为充足，所以把算法的时间复杂度作为重点分析。

1. 时间复杂度（Time Complexity）

一个算法所需的运算时间通常与所解决问题的规模大小有关。问题规模是一个和输入有关的量，用 **n** 表示问题规模的量，把算法运行所需的时间 T 表示为 n 的函数，记为 **T(n)**。不同的 T(n)算法，当 n 增长时，运算时间增长的快慢很不相同。一个算法所需的执行时间就是该算法中所有语句执行次数之和。当 n 逐渐增大时 T(n)的极限情况，一般简称为时间复杂度。

当讨论一个程序的运行时间时，注重的不是 T(n)的具体值，而是它的增长率。T(n)的增长率与算法中数据的输入规模紧密相关，而数据输入规模往往用算法中的某个变量的函数来表示，通常是 **f(n)**。随着数据输入规模的增大，f(n)的增长率与 T(n)的增长率相近，因此 T(n)同 f(n)在数量级上是一致的。记为：

$$\mathbf{T(n)=O(f(n))}$$

其中，大写字母 O 为 Order(数量级)的字头，f(n)为函数形式，如 $T(n)=O(n^2)$。

注意，当 T(n)为多项式时，可只取其最高次幂项并省略其系数，其他的次幂项及系数均略去不写。一般地，对于足够大的 n，常用的时间复杂度存在以下顺序：

$$O(1)<O(\log_2 n)<O(n)<O(n\log_2 n)<O(n^2)<O(n^3)<O(2^n)$$

算法时间复杂度的数量级越大，表示该算法的效率越低；反之越高。例如，O(1)为常数数量级，即算法的时间复杂度与输入规模 n 无关。

【例 1.6】 分析以下算法的时间复杂度。

```
x=0;y=0;
for(k=1;k<=n;k++)
x++;                          (1) 执行 n 次
for(i=1;i<=n;i++)
for(j=1;j<=n;j++)
y++;                          (2) 执行 n² 次
```

解： $T(n)= n+n^2$

$T(n)=O(n^2)$

上述算法中的基本运算是语句（2），其执行频率为 n^2。则 $T(n)=n^2=O(n^2)$。

【例 1.7】 分析以下算法的时间复杂度。

```
i=1;
while(i<=n)
   i=2*i;                     (1) 执行 f(n)次
```

解： 设语句（1）执行次数是 f(n)，则 $2^{f(n)}\leqslant n$。

得到 $T(n)=O(\log_2 n)$。

【例 1.8】 求两个矩阵相乘的函数的时间复杂度。

```
void mult(int a[], int b[], int c[])
{/*以二维数组存储矩阵元素，c 为 a 和 b 的乘积*/
  for(i=1;i<=n;++i)                    (1) 执行 n 次
    for(j=1;j<=n;++j)                  (2) 执行 n² 次
    {  c[i,j]=0;
       for(k=1;k<=n;++k)               (3) 执行 n³ 次
          c[i,j]+=a[i,k]*b[k,j];
    }
}
```

解： 嵌套循环为每层循环次数的乘积，因为该函数为三重循环，所以时间复杂度为 $O(n^3)$。

2. 空间复杂度（Space Complexity）

一个算法的空间复杂度是指程序运行开始到结束所需要的存储空间。包括算法本身所占用的存储空间、输入/输出数据占用的存储空间以及算法在运行过程中的工作单元和实现算法所需辅助空间。类似于算法的时间复杂度，算法所需存储空间的量度记为：

$$S(n)=O(f(n))$$

表示随着问题规模 n 的增大，算法运行所需存储量的增长率与 f(n)的增长率相同。

1.2.3 相关 C 语言知识回顾

1. 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，用以描述操作对象的特性。高级语言中的数据类型可分为两类：一类是原子类型，如 C 语言中的基本类型（整型、实型、字符型等）、指针和空类型等不可再分的值；另一类是结构类型，如数组、结构体等是通过若干成分（可以是原子类型或结构类型）构造而成的。

（1）数组类型。

数组中的每一个元素都属于同一个数据类型。数组有一维数组和多维数组，数组名标识一个数组，下标指示一个数组元素在该数组中的顺序位置，下标从 0～n-1（n 为数据元素个数）。例如，int a[10];定义了包含 10 个整数的数组 a，数组下标范围 0～9。

（2）结构体类型。

结构体是一种复合的数据类型，该结构体类型内可以有多个成员，每个结构体成员可以有不同的数据类型。基本语法格式如下：

【格式】

```
struct  结构体名                    /*定义结构体类型*/
{
   成员列表;
};
struct 结构体名 变量名表;          /*定义结构体变量*/
```

如：

```
struct  student                 /*声明 struct  student 结构体类型*/
{  int   num;
   char  name[20];
   int    age;
   float  score;
};
struct  student  stu1;          /*定义 struct  student 结构体类型变量 stu1*/
struct  student  *p;            /*定义 struct  student 结构体类型指针变量 p*/
p=&stu1;                        /*将指针变量 p 指向结构体变量 stu1*/
```

（3）结构体类型指针对结构体成员的引用方法。

当结构体类型的指针指向某一结构体变量时，就可以使用该指针对其指向的结构体变量内的各成员进行引用。引用的方法为：

指针名->成员名 等价于：(*指针名).成员名 等价于：结构体变量名.成员名

在本书的程序内大量使用结构体类型的指针，所以大都采用第一种写法。

例如：在前面执行 p=&stu1;语句后，使 p 指向结构体变量 stu1，就可以有以下使用方法，且都是等价的。

```
stu1.age=25;          /*以下三种形式引用结构体成员是等价的*/
(*p).age=25;
p->age=25;
```

（4）用 typedef 定义类型。