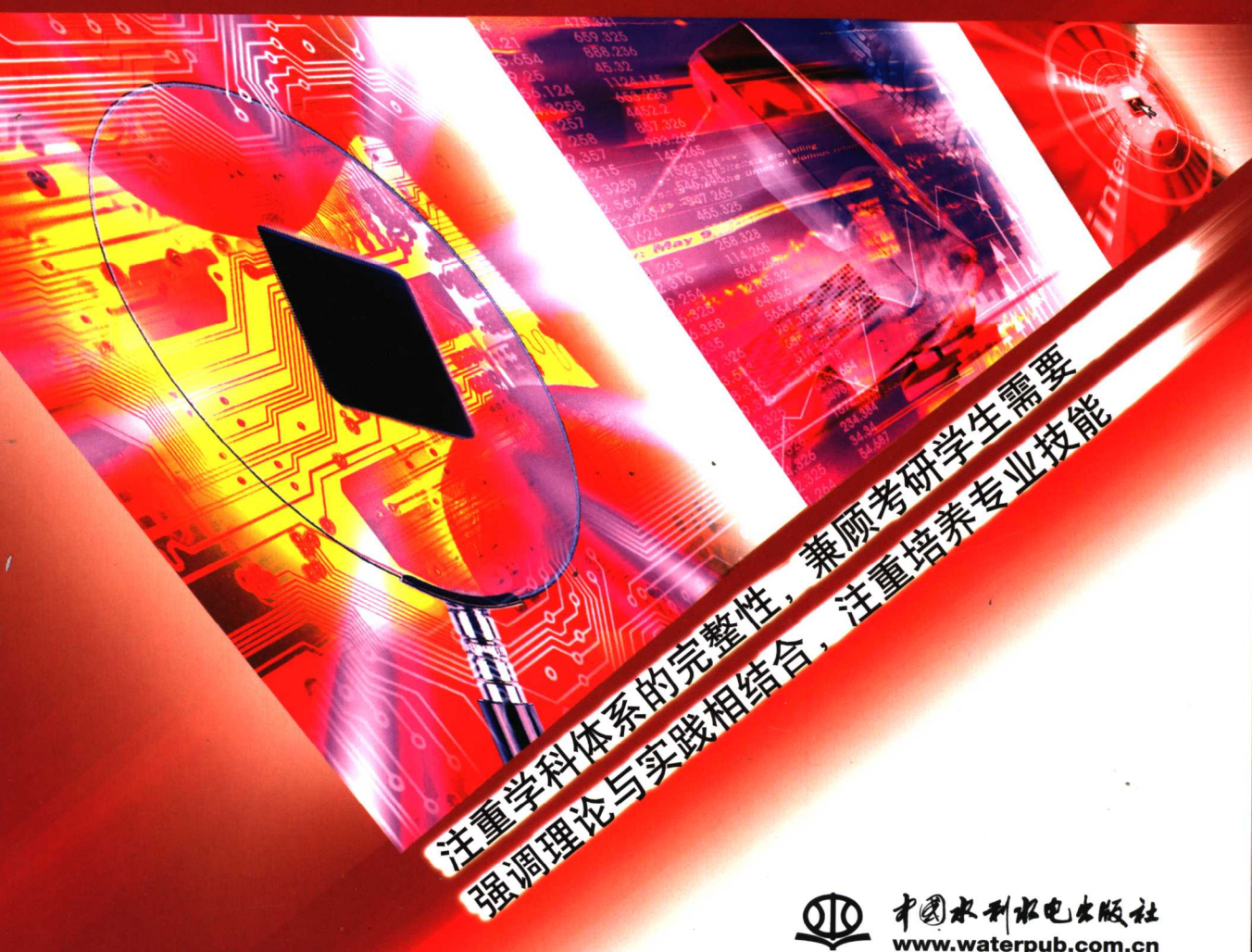
21世纪

高等院校规划教材

赵坚 姜梅 主编

数据结构(C语言版)

学习指导与习题解答



中国水利水电出版社
www.waterpub.com.cn

21世纪高等院校规划教材

数据结构（C语言版）学习指导与习题解答

赵 坚 姜 梅 主 编

邵 明 李 兰 李传斌 李学良 副主编

中国水利水电出版社

内 容 提 要

本书是与《数据结构（C 语言版）》（赵坚、姜梅主编）一书相配套的辅助教材。全书分为三大部分：第一部分是学习指导与实训，首先给出主教材中每一章的学习指南与内容提要，然后设置了若干综合实验，通过各章的实验体现实训特色，培养学生解决实际问题的能力；第二部分是主教材中习题的参考解答；第三部分设置了 4 套模拟试题及其参考解答，目的是检验和巩固所学的理论知识。

本书既可与《数据结构（C 语言版）》一书配套使用，也可作为 C 语言描述的实训教材单独使用，还可供计算机自学人员学习参考。

图书在版编目（CIP）数据

数据结构（C 语言版）学习指导与习题解答 / 赵坚，姜梅主编. —北京：中国水利水电出版社，2005

（21 世纪高等院校规划教材）

ISBN 7-5084-3053-0

Ⅰ. 数… Ⅱ. ①赵…②姜… Ⅲ. ①数据结构－高等学校－教学参考资料 ②C 语言－程序设计－高等学校－教学参考资料 Ⅳ. ①TP311.12②TP312

中国版本图书馆 CIP 数据核字（2005）第 086390 号

书　　名	**数据结构（C 语言版）学习指导与习题解答**
作　　者	赵坚　姜梅　主编　邵明　李兰　李传斌　李学良　副主编
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）、68331835（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	北京蓝空印刷厂
规　　格	787mm×1092mm　16 开本　11.25 印张　273 千字
版　　次	2005 年 8 月第 1 版　2005 年 8 月第 1 次印刷
印　　数	0001—5000 册
定　　价	16.00 元

序

随着计算机科学与技术的飞速发展，计算机的应用已经渗透到国民经济与人们生活的各个角落，正在日益改变着传统的人类工作方式和生活方式。在我国高等教育逐步实现大众化后，越来越多的高等院校会面向国民经济发展的第一线，为行业、企业培养各级各类高级应用型专门人才。为了大力推广计算机应用技术，更好地适应当前我国高等教育的跨跃式发展，满足我国高等院校从精英教育向大众化教育的转变，符合社会对高等院校应用型人才培养的各类要求，我们成立了“21 世纪高等院校规划教材编委会”，在明确了高等院校应用型人才培养模式、培养目标、教学内容和课程体系的框架下，组织编写了本套“21 世纪高等院校规划教材”。

众所周知，教材建设作为保证和提高教学质量的重要支柱及基础，作为体现教学内容和教学方法的知识载体，在当前培养应用型人才中的作用是显而易见的。探索和建设适应新世纪我国高等院校应用型人才培养体系需要的配套教材已经成为当前我国高等院校教学改革和教材建设工作面临的紧迫任务。因此，编委会经过大量的前期调研和策划，在广泛了解各高等院校的教学现状、市场需求，探讨课程设置、研究课程体系的基础上，组织一批具备较高的学术水平、丰富的教学经验、较强的工程实践能力的学术带头人、科研人员和主要从事该课程教学的骨干教师编写出一批有特色、适用性强的计算机类公共基础课、技术基础课、专业及应用技术课的教材以及相应的教学辅导书，以满足目前高等院校应用型人才培养的需要。本套教材消化和吸收了多年来已有的应用型人才培养的探索与实践成果，紧密结合经济全球化时代高等院校应用型人才培养工作的实际需要，努力实践，大胆创新。教材编写采用整体规划、分步实施、滚动立项的方式，分期分批地启动编写计划，编写大纲的确定以及教材风格的定位均经过编委会多次认真讨论，以确保该套教材的高质量和实用性。

教材编委会分析研究了应用型人才与研究型人才在培养目标、课程体系和内容编排上的区别，分别提出了 3 个层面上的要求：在专业基础类课程层面上，既要保持学科体系的完整性，使学生打下较为扎实的专业基础，为后续课程的学习做好铺垫，更要突出应用特色，理论联系实际，并与工程实践相结合，适当压缩过多过深的公式推导与原理性分析，兼顾考研学生的需要，以原理和公式结论的应用为突破口，注重它们的应用环境和方法；在程序设计类课程层面上，把握程序设计方法和思路，注重程序设计实践训练，引入典型的程序设计案例，将程序设计类课程的学习融入案例的研究和解决过程中，以学生实际编程解决问题的能力为突破口，注重程序设计算法的实现；在专业技术应用层面上，积极引入工程案例，以培养学生解决工程实际问题的能力为突破口，加大实践教学内容的比重，增加新技术、新知识、新工艺的内容。

本套规划教材的编写原则是：

在编写中重视基础，循序渐进，内容精炼，重点突出，融入学科方法论内容和科学理念，反映计算机技术发展要求，倡导理论联系实际和科学的思想方法，体现一级学科知识组织的层次结构。主要表现在：以计算机学科的科学体系为依托，明确目标定位，分类组织实施，兼容互补；理论与实践并重，强调理论与实践相结合，突出学科发展特点，体现

学科发展的内在规律；教材内容循序渐进，保证学术深度，减少知识重复，前后相互呼应，内容编排合理，整体结构完整；采取自顶向下设计方法，内涵发展优先，突出学科方法论，强调知识体系可扩展的原则。

本套规划教材的主要特点是：

（1）面向应用型高等院校，在保证学科体系完整的基础上不过度强调理论的深度和难度，注重应用型人才的专业技能和工程实用技术的培养。在课程体系方面打破传统的研究型人才培养体系，根据社会经济发展对行业、企业的工程技术需要，建立新的课程体系，并在教材中反映出来。

（2）教材的理论知识包括了高等院校学生必须具备的科学、工程、技术等方面的要求，知识点不要求大而全，但一定要讲透，使学生真正掌握。同时注重理论知识与实践相结合，使学生通过实践深化对理论的理解，学会并掌握理论方法的实际运用。

（3）在教材中加大能力训练部分的比重，使学生比较熟练地应用计算机知识和技术解决实际问题，既注重培养学生分析问题的能力，也注重培养学生思考问题、解决问题的能力。

（4）教材采用“任务驱动”的编写方式，以实际问题引出相关原理和概念，在讲述实例的过程中将本章的知识点融入，通过分析归纳，介绍解决工程实际问题的思想和方法，然后进行概括总结，使教材内容层次清晰，脉络分明，可读性、可操作性强。同时，引入案例教学和启发式教学方法，便于激发学习兴趣。

（5）教材在内容编排上，力求由浅入深，循序渐进，举一反三，突出重点，通俗易懂。采用模块化结构，兼顾不同层次的需求，在具体授课时可根据各校的教学计划在内容上适当加以取舍。此外还注重了配套教材的编写，如课程学习辅导、实验指导、综合实训、课程设计指导等，注重多媒体的教学方式以及配套课件的制作。

（6）大部分教材配有电子教案，以使教材向多元化、多媒体化发展，满足广大教师进行多媒体教学的需要。电子教案用 PowerPoint 制作，教师可根据授课情况任意修改。相关教案的具体情况请到中国水利水电出版社网站 www.waterpub.com.cn 下载。此外还提供相关教材中所有程序的源代码，方便教师直接切换到系统环境中教学，提高教学效果。

总之，本套规划教材凝聚了众多长期在教学、科研一线工作的教师及科研人员的教学科研经验和智慧，内容新颖，结构完整，概念清晰，深入浅出，通俗易懂，可读性、可操作性和实用性强。本套规划教材适用于应用型高等院校各专业，也可作为本科院校举办的应用技术专业的课程教材，此外还可作为职业技术学院和民办高校、成人教育的教材以及从事工程应用的技术人员的自学参考资料。

我们感谢该套规划教材的各位作者为教材的出版所做出的贡献，也感谢中国水利水电出版社为选题、立项、编审所做出的努力。我们相信，随着我国高等教育的不断发展和高校教学改革的不断深入，具有示范性并适应应用型人才培养的精品课程教材必将进一步促进我国高等院校教学质量的提高。

我们期待广大读者对本套规划教材提出宝贵意见，以便进一步修订，使该套规划教材不断完善。

21 世纪高等院校规划教材编委会

2004 年 8 月

前　言

本书是与《数据结构（C 语言版）》（赵坚、姜梅主编）一书相配套的辅助教材。全书共分三大部分：第一部分给出了主教材中每一章的知识要点及综合实验，包括线性结构（线性结构的定义、组织形式、结构特征和类型说明以及在两种存储方式下实现的插入、删除、查找的算法，循环链表、双（循环）链表的结构特点和在其上的插入、删除等操作），树型结构（二叉树的二叉链表存储方式、结点结构和类型定义，二叉树的基本运算及应用），图状结构（图的各种存储结构的表示方法），查找（顺序查找、树表查找、散列表查找的基本思想及存储、运算的实现），排序（插入排序、冒泡排序、快速排序、直接选择排序、堆排序、归并排序和基数排序的基本思想及实现），以及数组和字符串的操作。这一部分体现实训特色，突出实训重点，培养学生应用理论知识解决实际问题的能力。第二部分是主教材中习题的参考解答。第三部分给出了 4 套模拟试题及其参考解答，目的是检验和巩固所学的理论知识。

数据结构是一门实践性很强的课程。对于自己编写的每一个算法，不仅要尽量符合算法评价的各项指标，更重要的是上机验证，在反复调试的过程中，通过典型的数据输入使得算法中的每条语句都被执行通过。若调试过程发现语法或逻辑错误，则要及时修改。通过上机运行程序能够加深对所学知识的理解和掌握，进而获得书本上学不到的知识。

解决一个算法问题通常要经过以下几步：①根据题目要求分析出设计思路或建立起数学模型；②根据设计思路或数学模型画出相应的流程图；③根据流程图用一种计算机语言（如 C 语言）编写出详细算法；④编写出能够调用该算法的完整程序；⑤上机调试和运行该程序。通过反复调试和修改，直到获得满意的结果为止。

对于要解决的同一个问题，由于所采用的数据结构可能不同，所选择的计算方法（即算法）可能不同，则编写出的程序就可能不同。但只要程序正确并且有效（即具有较好的时间和空间复杂度）即可。因此，每个人按照习题编写出的算法程序不要求与本书所给的解答完全一致，也许读者编写出的算法具有更好的性能。

本书由赵坚、姜梅主编，邵明、李兰、李传斌、李学良任副主编。本书主要编写人员及分工如下：赵坚负责编写第 1 章和第 6 章，李兰负责编写第 2 章和第 8 章，邵明负责编写第 3 章和第 9 章，姜梅负责编写第 4 章和第 7 章，李传斌负责编写第 5 章和第 10 章，李学良负责编写模拟试题、程序编辑和调试。参加本书编写（包括大纲讨论）的还有王红、王成端、刘永华、沈祥玖、相伟、周朋红、肖孟强、李禹生、安志远、杨立等。

书中所有算法和程序都在 C 语言或 Borland C++语言环境下调试通过，但由于编写时间仓促，作者水平有限，错误和不足之处在所难免。恳请专家和读者指正，以便进一步提高本书的质量。

编　者

2005 年 5 月于青岛

目　　录

序
前言

第一部分　学习指导与实训

第 1 章　绪论 1
1.1　学习指南 1
1.2　内容提要 1
第 2 章　线性表 3
2.1　学习指南 3
2.2　内容提要 3
2.3　实训概要 4
第 3 章　栈和队列 13
3.1　学习指南 13
3.2　内容提要 13
3.3　实训概要 19
第 4 章　串 28
4.1　学习指南 28
4.2　内容提要 28
4.3　实训概要 29
第 5 章　数组和广义表 34
5.1　学习指南 34
5.2　内容提要 34
5.3　实训概要 35
第 6 章　树和二叉树 41
6.1　学习指南 41
6.2　内容提要 41
6.3　实训概要 43
第 7 章　图 54
7.1　学习指南 54
7.2　内容提要 54
7.3　实训概要 56

第 8 章　排序 63
8.1　学习指南 63
8.2　内容提要 63
8.3　实训概要 64
第 9 章　查找 69
9.1　学习指南 69
9.2　内容提要 69
9.3　实训概要 73
9.4　参考程序 75
第 10 章　文件 81
10.1　学习指南 81
10.2　内容提要 81

第二部分　习题参考解答

第 1 章　绪论 83
第 2 章　线性表 84
第 3 章　栈和队列 90
第 4 章　串 97
第 5 章　数组与广义表 103
第 6 章　树和二叉树 108
第 7 章　图 115
第 8 章　排序 126
第 9 章　查找 136
第 10 章　文件 141

第三部分　模拟试题及参考答案

模拟试题一 144
模拟试题二 147
模拟试题三 150
模拟试题四 153
模拟试题参考答案 158
模拟试题一参考答案 158
模拟试题二参考答案 160
模拟试题三参考答案 164
模拟试题四参考答案 167
参考文献 170

第一部分　学习指导与实训

第1章　绪　论

1.1　学习指南

（1）掌握数据结构的常用术语，基本概念以及学习数据结构的意义。

（2）理解和掌握集合、线性结构、树型结构和图形结构等常用结构的表示和实现方法，掌握评价算法的一般规则以及算法描述和分析的方法。

（3）本章重点是理解数据的逻辑结构、存储结构以及数据运算3方面的概念及相互关系。难点是算法时间复杂度的分析方法。

1.2　内容提要

（1）数据结构研究的是数据的表示和数据之间的关系。这些关系反映了客观世界事物之间的联系，一般包括3方面的内容：逻辑结构、存储结构及算法。从逻辑上讲，数据有集合结构、线性结构、树结构和图结构4种。从物理实现上讲，数据有顺序结构、链接结构、索引结构和散列结构等4种。由它们的组合可以构成任何更为复杂的存储结构。从理论上讲，任何一种数据的逻辑结构都可以用任一种存储结构来实现。

（2）在集合结构中，不考虑数据之间的次序关系，它们处于无序的、各自独立的状态。在线性结构中，数据之间是1对1的关系。在树结构中，数据之间是1对多的关系。在图结构中，数据之间是多对多的关系。

（3）数据的存储结构是逻辑结构在计算机中的表示或实现。顺序存储结构是把数据元素按逻辑次序依次存放在一组连续的存储单元中，这种存储方法是用存储结点间的位置关系表示数据元素之间的逻辑关系，具体实施时可用高级语言中的数组来实现。一个数组占有一片连续的存储空间，每个元素的存储单元是按下标位置从0开始连续编号的，逻辑上相邻的元素其存储位置也相邻。对于任一种数据的逻辑结构，若能够把元素之间的逻辑关系对应地转换为数组下标位置之间的物理关系，就能够利用数组来实现其顺序存储结构。

链式存储结构就是用一组任意的存储单元——结点来存储结构中的数据元素。每一个存储结点都由1个数据域和至少1个指针域（指向逻辑上相邻对象的物理存储地址）组成。数据元素的逻辑次序是通过链表中的指针链接实现的。

索引存储结构是用结点的索引号来确定结点的存储地址。通常在存储结点的同时，附加索引表。结点间的逻辑关系由索引项来表示。

散列存储结构是用散列函数来确定和计算数据元素的存储位置。

（4）实现一个运算的方法和步骤称为此运算的算法。它一般是为解决一个或一类问题所给出的一个确定的、有限长的操作序列。描述一个算法有许多形式，如采用自然语言、程序流程图、类 C 语言或其他高级语言等。在数据结构中主要使用类语言或某种高级语言。本书采用通用性强的 C 语言来描述。

（5）一个算法的性能主要可以从 4 个方面进行评价：正确性、可读性、健壮性、有效性。其中有效性又包括时间复杂度和空间复杂度两个方面。一个算法的时间复杂度和空间复杂度越低，表明其所需的时间代价及空间代价越低，说明算法的效果越好。

（6）算法分析首先是考虑正确性，此外，还要考虑执行算法所耗费的时间和空间代价。从时间方面的分析称为时间复杂度；从空间方面的分析称为空间复杂度。讨论时通常以时间复杂度为主，因为节约时间往往比节省空间更重要。算法的时间复杂度和空间复杂度通常用数量级的形式表示出来。

常用的时间复杂度从低到高的级别依次为：常量级 $O(1)$、对数级 $O(\log_2 n)$、线性级 $O(n)$、平方级 $O(n^2)$、指数级 $O(2^n)$等。当处理的数据量较大时，处于前面级别的算法比处于后面级别的算法更有效。

第 2 章　线性表

2.1　学习指南

（1）理解线性表的类型定义，掌握顺序表和链表的结构差别。
（2）熟练掌握顺序表的结构特性，熟悉顺序表的存储结构。
（3）熟练掌握顺序表的各种运算，并能灵活运用各种相关操作。
（4）熟练掌握链式存储结构特性，掌握链表的各种运算。

2.2　内容提要

线性表的特点：线性表由一组数据元素构成，表中元素属于同一数据对象。在线性表中，数据元素之间的相对位置是线性的，数据按照前后顺序进行有限排列，第一个数据元素有且仅有一个直接后继，最后一个数据元素有且仅有一个直接前驱，其他所有数据元素都有且仅有一个直接后继和一个直接前驱。

1. 线性表

（1）线性表 L 是由 n（n≥0）个数据元素 a_1，a_2，a_3，…，a_n 组成的有限序列，其中数据元素的个数 n 定义为线性表的长度，n=0 的线性表称为空表。

（2）线性表的逻辑结构：一个非空(n≠0)的线性表记为：L＝(a_1，a_2，a_3，…，a_n)。a_i（1≤i≤n）称作线性表的第 i 个数据元素，下标 i 为 a_i 元素在线性表中的位序，称其前面的元素 a_{i-1} 为 a_i(2≤i≤n)的直接前驱，称其后面的元素 a_{i+1} 为 a_i 的直接后继。

2. 线性表的顺序存储

（1）线性表的顺序存储特点：把逻辑相邻的数据元素存储在物理相邻的存储单元中，也就是在内存中用一组地址连续的存储空间顺序存放线性表的各元素。用顺序存储方法存储的线性表称为顺序表。第一个数据元素称为开始结点，最后一个数据元素称为终端结点。

线性表的顺序存储结构具有以下两个基本特点：

1）线性表的所有元素所占的存储空间是连续的。

2）线性表中各数据元素在存储空间中是按逻辑顺序依次存放的。

由此可以看出，在顺序存储线性表结构中，其前后两个元素的存储空间是紧邻的，且前驱元素一定存储在后继元素的前面。

（2）线性表的顺序存储结构。

```
#define MaxLen 100                          /*定义线性表的容量为 100*/
typedef ElemType suitable data_type         /*数据元素类型*/
typedef struct{
ElemType data[MAXLEN];                      /*定义存储表中元素的数组*/
```

```
int length;                    /*线性表的实际长度 */
}sqlist;
```

（3）顺序表的基本运算主要包括：顺序表的插入、顺序表的删除、顺序表的查找等。

3. 线性表的链式存储

（1）线性表的链式存储特点。在链表存储方式中，任意两个在逻辑上相邻的数据元素在物理上不一定相邻，数据元素的逻辑次序是通过链表中的指针链接实现的。链表的长度是动态的、可扩充的，在链表中插入和删除时不需移动元素。链式存储结构适用于插入和删除频繁，存储空间需求不定的情形。

（2）线性表的链式存储结构。为了在链表中表示元素之间的线性关系，每一个数据元素在存储时除了存储自身的数据之外，还需要存储其后继或前驱元素的地址，因此数据元素需要一个复合的数据类型表示：

```
typedef struct LNode
{ ElemType data;          /*结点值*/
  Struct Lnode* next;     /*链接下一个结点的地址*/
}LNod,*LinkList;
```

（3）各种链表形式。

1）单链表：数据元素之间只存在单方向的指向，即可以由链表头直接查询到链表尾，但这种链表不能反方向访问。

2）循环链表：将单链表中最后一个结点的指针指向链表的头结点，使整个链表形成一个环形，称这种头尾相边的线性链表形式为循环链表，这样从表中任一结点出发，顺着指针链可以找到表中其他的任何结点。

3）双链表：用两个指针表示结点间的逻辑关系。

（4）线性链表的基本运算：链表的插入、链表的删除、链表的查找等。

2.3 实训概要

本章实训主要是在线性表的两类存储结构（顺序结构和链式结构）上实现基本操作。通过本章实训能够更好地理解和掌握线性表的相关知识。

2.3.1 实验 1：线性表的基本操作

1. 实验目的

（1）掌握线性表的基本运算，熟悉对线性表的一些基本操作和具体的函数定义。

（2）掌握顺序存储的概念，学会定义线性表的顺序存储类型。

（3）熟悉 C 语言程序的基本结构，掌握程序中的用户头文件、实现文件和主文件之间的相互关系及各自的作用。

（4）熟悉 C 语言环境的使用以及多文件程序的输入、编辑、调试和运行的全过程。

（5）加深对顺序存储结构的理解，逐步培养解决实际问题的编程能力。

2. 实验要求

（1）熟练掌握线性表的存储结构及其基本操作。

（2）理解实训案例中的算法，掌握线性表在实际中的应用。

（3）将上机程序全部调试通过。

（4）独立完成一至二个实训项目，保存程序运行结果，并结合程序进行分析。

3. 实验内容

（1）线性表的基本操作。

第一步：定义线性表的存储结构。

第二步：编写线性表操作的具体函数定义。

第三步：使用定义的线性表并调用线性表的一些操作，实现具体运算。

1）初始化线性表。

2）创建一个线性表。

3）在线性表中查找指定的元素。

4）在线性表中插入指定值的元素。

5）在线性表中删除指定位置的元素。

6）输出线性表。

注意：每完成一个步骤，必须及时输出线性表元素，以便于观察操作结果。

sqList.h

```
#define TRUE            1
#define FALSE           0
#define OK              1
#define ERROR           0
#define OVERFLOW        -2
#define ML 10
/*定义 ElemType 为 int 类型*/
typedef int ElemType;
/*线性表顺序存储类型*/
typedef struct sqList                    /*定义线性表结构*/
  {
    ElemType  list[ML];                  /*线性表元素，存储空间位址*/
    int size;                            /*当前线性表长度*/
    int MAXSIZE;                         /*当前 list 数组元素个数*/
}sqList;
```

sqList.c

```
#include <stdio.h>
#include <stdlib.h>
#include <alloc.h>
#include "sqList.h"

/*初始化线性表*/
sqList *Init_List(sqList *L,int ms)
{
    L=(sqList *)malloc(ms*sizeof(sqList));
    if(!L){
        printf("Memory allocation failure!\n");
        exit(OVERFLOW);
```

```
        }
else
        L->size=0;
    L->MAXSIZE=ms;
    return L;
}

/*输出一个线性表*/
void Disp_List(sqList *L)
{
int  i;
for(i=0;i<L->size;i++)
printf("%d\t",L->list[i]);
printf("\n");
}
/*在线性表中查找元素值为 x 的元素*/
int LocateElem_List(sqList *L, ElemType x)
/*寻找线性表元素，其中返回≥0 为元素位置，-1 为没找到*/
{
int  i=0;
for(i=0;i<=L->size;i++)
if (L->list[i]==x)                    /*找到相同的元素，返回位置*/
    return i;
if (i>L->size) return -1;             /*没找到*/
}

int Insert_List(sqList *L, ElemType x ,int mark)
/* x 为记录值,mark 为插入位置*/
{
int i=1;
if(L->size>=L->MAXSIZE)               /*线性表已满*/
   return -1;
if(mark>0){                           /*插入位置为表头*/
  for(i=L->size+1; i>=mark ; i--)     /*将线性表元素后移*/
    L->list[i+1]=L->list[i];
    L->list[i]=x;
}
else if(mark<0)
   L->list[L->size]=x;                /*插入位置为表尾*/
   L->size++;
return  FALSE ;
}

/*在线性表中删除指定元素值*/
int  Delete_List1(sqList *L , int item)
/*删除的线性表元素，返回≥0 为删除成功*/
```

```
{
    int i,j;
    for(i=0;i<L->size;i++)
      if(item==L->list[i])            /*找到相同的元素*/
        break;
    if(i<L->size){
      for(j=i+1;j<L->size-1;j++)
        L->list[j]=L->list[j+1];
      L->size--;
    return i;
    }
    return  FALSE;
}

/*删除指定位置的线性表元素*/
int Delete_List2(sqList *L,int mark)
{
int i, item;
    if(mark>0){
  item=L->list[mark];
       for(i=mark+1;i<L->size-1;i++)
         L->list[i]=L->list[i+1];
       L->size--;
      return i;
    }
  return FALSE;
}

sqListmain.c
#include"sqList.c"
void main ()
{
int p,n;
ElemType x=0;
sqList a,*b;
b=Init_List(&a,ML);
printf("listaddr=%p\tsize=%d\tMaxSize=%d\n",b->list,b->size,b->MAXSIZE);
while(1)
   {
     printf("\n 请输入元素值,0 为结束输入: ");
     scanf("%d",&x);
     if(!x) break;
     printf("请输入插入位置: ");
     scanf("%d",&p);
     Insert_List(b,x,p);
     printf("线性表为: \n");
```

```
    Disp_List(b);
  }
  while(1)
  {
   printf("请输入查找元素值，输入 0 结束查找操作：");
   scanf("%d",&x);
   if(!x)    break;
   n=LocateElem_List(b,x);
   if(n<0)    printf("没找到\n");
  else
   printf("有符合条件的元素,位置为：%d\n",n+1);
}
while(1)
{
   printf("请输入删除元素值,输入 0 结束查找操作：");
   scanf("%d",&x);
   if(!x)
     break;
   n= Delete_List1(b,x);
   if(n<0)
     printf("没找到\n");
   else{
     printf("删除元素成功，线性表为:");
   Disp_List(b);
   }
}
while(1)
{
 printf("请输入删除元素位置，输入 0 结束查找操作：");
   scanf("%d",&p);
   if(!p)    break;
   n= Delete_List2(b,p);
   if(p<0)   printf("位置越界\n");
   else{
     printf("线性表为：");
     Disp_List(b);
   }
}
}
```

2.3.2 实验 2：链表的操作

1. 实验目的

（1）学会单链表结点的定义。

（2）熟悉单链表的一些基本操作和具体的函数定义。

（3）加深对链表的理解，逐步培养解决实际问题的编程能力。

2. 实验内容

第一步：定义单链表的存储结构。

第二步：编写单链表操作的具体函数定义。

第三步：用定义单链表和调用单链表的一些操作，实现对单链表的具体运算。

（1）初始化单链表。

（2）创建一个单链表。

（3）在单链表中查找指定的元素。

（4）在单链表中删除指定值的元素。

（5）在单链表中删除指定位置的元素。

（6）输出单链表。

注意：每完成一个步骤，必须及时输出单链表元素，以便于观察操作结果

```
linkList.h
#define TRUE    1
#define FALSE   0
#define OK  1
#define ERROR   0
#define flag    0                   /*按下 0 代表输入结束*/
typedef int ElemType;               /*定义 ElemType 为 int 类型*/
/*定义单链表存储结构*/
typedef struct linkList
{
ElemType   data;                    /*结点值 */
struct linkList  *next;             /*指向下一个结点的指针 */
}LinkList;

linkList.c
#include<stdio.h>
#include<stdlib.h>
#include<alloc.h>
#include"linkList.h"

/*初始化线性表*/
void init_LinkList(LinkList *head)
{
head=NULL;
}

/*清空单链表*/
void cls_LinkList(LinkList *head)
{
LinkList  *cp ,*np;
cp=head;
while(cp!=NULL)
 {
```