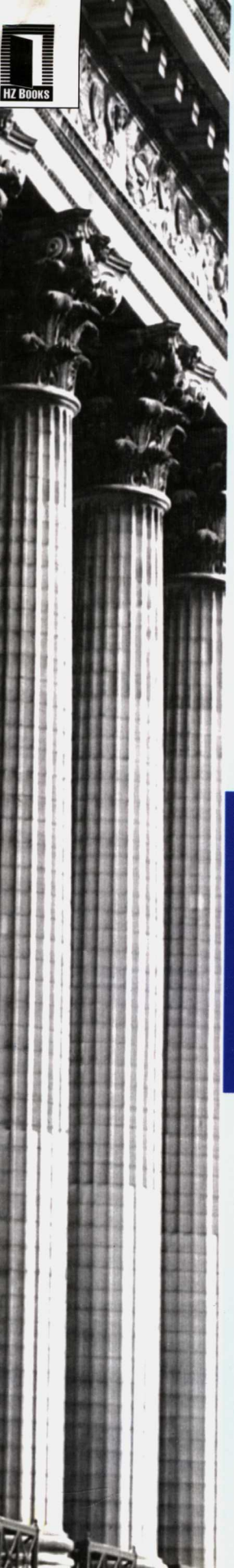


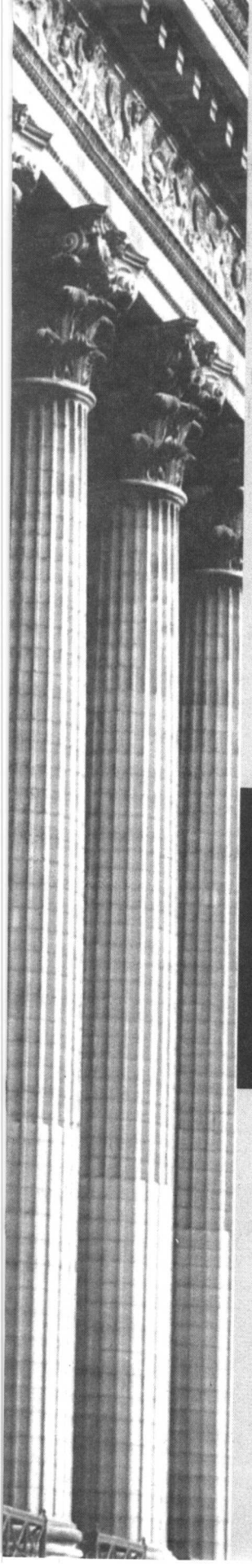
HZ BOOKS

高等院校
计算机教材系列

孟佳娜　胡潇琨　编著

算法与数据结构
实验与习题

机械工业出版社
China Machine Press



高等院校
计算机教材系列

孟佳娜　胡潇琨　编著

算法与数据结构实验与习题

机械工业出版社
China Machine Press

本书可以与《算法与数据结构（C语言版）》教材配套使用。本书主要包括两方面的内容：实验和习题。针对具体的教学，书中主要给出了13个实验；习题部分根据配套教材的主要内容和数据结构课程教学大纲的要求编写而成，既注重基础内容的练习，同时也收集了难度适中和高难度的题目。这两个方面既互相制约，又互相促进，在加深对理论知识的理解的同时又锻炼了对实际问题进行软件设计的能力，对后续课程的学习也是有好处的。

本书可以作为高等院校计算机及相关专业学习数据结构课程的参考书，对于报考计算机专业硕士研究生的考生也是极具价值的参考书，同时也适用于自学考试和计算机等级考试的应考者。

图书在版编目（CIP）数据

算法与数据结构实验与习题／孟佳娜等编著. -北京：机械工业出版社，2004.9
（高等院校计算机教材系列）
ISBN 7-111-14825-8

Ⅰ.算… Ⅱ.孟… Ⅲ.① 数据结构－高等学校－教学参考资料 ②算法设计－高等学校－教学参考资料 Ⅳ.TP311.12

中国版本图书馆CIP数据核字（2004）第064936号

机械工业出版社（北京市西城区百万庄大街22号 邮政编码 100037）
策划编辑：温莉芳
责任编辑：迟振春
北京牛山世兴印刷厂印刷·新华书店北京发行所发行
2005年6月第1版第2次印刷
787mm×1092mm 1/16·12.75印张
印数：5 001-7 000册
定价：19.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换
本社购书热线：（010）68326294

前　言

随着计算机科学的不断发展，计算机在社会各个方面的应用在逐步加强，很难想象信息社会没有计算机将会是什么样子。

计算机所加工和处理的对象是数据，数据具有一定的组织结构，而数据结构正是研究如何组织、存储数据和对数据进行操作的一门科学。数据结构课程是计算机专业的一门核心课程，并且逐步发展成为其他理工专业的选修课，因此具有相当重要的地位和作用。

通过多年的教学实践证明，理论和实践的有机结合是学好数据结构的基础。也就是说，要学好数据结构课程只通过在课堂上获取相关的理论知识是远远不够的，必须加强动手的能力，即上机进行实践和习题的练习。通过对课程中的典型算法和自己编写的算法进行上机调试，既能够加强对理论的理解同时又锻炼了对实际问题进行软件设计的能力，对后续课程的学习也是有好处的。而习题的练习对于加深对理论知识的理解也相当重要，在教学中我们发现学生理解授课内容并不困难，但一接触习题，特别是算法设计题目便无从下手，造成“眼高手低”的现象。这既需要深刻理解课堂内容，更需要加大习题练习的力度。两方面既互相制约，又互相促进，才能达到充分掌握这门课程的目的。

为了更好地学习数据结构课程，我们编写了这本书，本书可以与用类C语言编写的《算法与数据结构（C语言版）》教材配套使用。本书主要包括两方面的内容：实验和习题。

针对具体的教学，书中主要给出了13个实验，分别为：顺序表基本操作、顺序表其他操作、链表基本操作、链表其他操作、表达式计算、数组的建立和使用、二叉树基本操作、二叉树其他操作、图的基本操作、图的其他操作、二叉排序树操作、哈希表操作、各种内部排序方法。实验部分的主要特点为：针对数据结构课程的特点，实验1、3、7、9都是各自存储结构的一些基本操作，同时又是其他实验题目上机实践的基础。在用到这种存储结构的具体实验题目中，可以把这些基本操作作为C语言的头文件包含进去，从而省去很多重复的定义和操作。

习题部分根据配套教材的主要内容和教学大纲的要求，从内容上主要分11章：第1章是概论，分析数据结构的基本概念和算法设计的相关习题；第2章是线性表，分析顺序表和链表的相关习题；第3章是栈和队列，分析栈和队列的逻辑和存储结构的相关习题；第4章是串，分析串的各种存储结构的相关习题；第5章是数组和广义表，分析数组的压缩存储和广义表的存储的相关习题；第6章是树，分析树的逻辑和存储结构的相关习题；第7章是图，分析图的逻辑和存储结构的相关习题；第8章是动态存储管理，分析系统如何分配和回收内存的相关习题；第9章是集合，分析基本的查找方法的相关习题；第10章是排序，分析基本的内部排序和外部排序方法的相关习题；第11章是文件，分析文件的组织和结构的相关习题。

本书习题既注重基础内容的练习，同时也收集了难度适中和高难度的题目，因此适用面比较广。本书每一章都列举了一些典型习题并且做了解答，使读者在做完练习题目后及时了解自己的答案是否正确，当然对于算法设计题目并没有标准答案，我们所做的解答只是抛砖

引玉，读者可以尝试融合自己思想的解答。精选的例题力图达到以下三个目的：1）帮助理解数据结构课程的内容，强化基本概念；2）培养良好的算法设计风格，训练程序设计技术；3）正确掌握为简单的应用合理选择数据结构的方法。同时每一章最后还挑选了一些综合练习题，这部分没有答案，可以作为读者检验自己对知识理解程度的手段，教师也可以在这部分内容中挑选一些作为课堂作业。

本书可以作为计算机及相关专业学生学习数据结构课程的参考书，对于报考计算机专业硕士研究生的考生也是极具价值的参考书，同时也适用于自学考试和计算机等级考试的应考者。

最后需要说明的是，学习主要靠自己多总结、多思考、多练习、多上机实践，这是学好数据结构课程的关键。

本书实验部分和习题部分的第1、2、3、5、6、7、8、9、11章由烟台大学孟佳娜编写，习题部分的第4、10章由烟台大学胡潇琨编写。烟台大学陈守孔老师给予了很多帮助和建议，在此表示感谢。

由于作者水平有限，书中难免存在一些缺点和错误，诚恳地欢迎广大读者提出批评意见和建议。

编　者

2004年4月

目　录

第一部分 实 验 部 分

实验1 顺序表基本操作

一、实验目的

1. 熟悉C语言的上机环境，掌握C语言的基本结构。
2. 会定义线性表的顺序存储结构。
3. 熟悉对顺序表的一些基本操作和具体的函数定义。

【注意事项】在做第一次“数据结构”课程实验之前，要在硬盘上建立好自己的工作目录，专门用来存储你所做的实验程序及相关信息，以后每次做实验都采用这个目录。

二、实验内容

该程序的功能是对元素类型为整型的顺序表进行一些操作。该程序包括顺序表结构类型的定义以及对顺序表操作的具体的函数定义和主函数。

```
/* 定义DataType为int类型 */
typedef int DataType;

/*顺序表存储空间的总分配量*/
#define   MAXSIZE 100

/* 顺序存储类型 */
typedef struct
    {DataType data[MAXSIZE];      /*存放线性表的数组*/
     int length;                  /*length是顺序表的长度*/
    }SeqList;

/* 初始化顺序表 */
SeqList SeqListInit( )

/* 清空顺序表 */
void ListClear(SeqList L)

/* 求顺序表长度 */
int ListLength(SeqList L)

/* 检查顺序表是否为空 */
int ListEmpty(SeqList L)
```

```
/*检查顺序表是否为满 */
int ListFull(SeqList L)

/* 遍历顺序表 */
void ListTraverse(SeqList L)

/* 从顺序表中查找元素 */
DataType ListGet(SeqList L ,int i)

/* 从顺序表中查找与给定元素值相同的元素在顺序表中的位置 */
int ListLocate(SeqList L, DataType x)

/* 向顺序表中插入元素 */
void ListInsert(SeqList L, int i, DataType x)

/* 从顺序表中删除元素 */
void ListDelete(SeqList L,int i)

/*求顺序表中元素的前驱*/
DataType ListPrior (SeqList L,DataType e)

/*求顺序表中元素的后继*/
DataType ListNext(SeqList L,DataType e)
```

实验2　顺序表其他操作

一、实验目的

进一步掌握在线性表的顺序存储结构上的一些其他操作。

二、实验内容

程序1

已知一个线性表，用另辟空间和利用原表两种方法把线性表逆置。

【设计要求】在程序中构造三个子程序分别为:

```
SeqList reverse(SeqList A)            /* 顺序表的就地逆置 */
void ListTraverse(SeqList L)          /* 遍历顺序表 */
SeqList create(int n)                 /* 建立顺序表 */
```

程序2

已知两个非递减有序的线性表*La*和*Lb*，将*La*和*Lb*合并成一个线性表*Lc*，*Lc*也非递减有序。

【设计要求】在程序中构造三个子程序分别为:

```
SeqList MergeSeqList(SeqList La,SeqList Lb)     /* 合并顺序表 */
void ListTraverse(SeqList L)                    /* 遍历顺序表 */
SeqList create()                                /* 建立顺序表 */
```

程序3

已知两个非递减有序的线性表*La*和*Lb*，长度分别为*m*和*n*，假设*La*的空间足够大，利用原表*La*，将*La*和*Lb*合并成一个仍然非递减有序的线性表。要求时间复杂度为$O(m+n)$。

【设计要求】在程序中构造三个子程序分别为:

```
SeqList MergeSeqList(SeqList La,SeqList Lb,int m,int n)
                                        /* 合并顺序表 */
void ListTraverse(SeqList L)            /* 遍历顺序表 */
SeqList create()                        /* 建立顺序表 */
```

程序4

约瑟夫环问题：任给正整数*N*和*K*，按下述方法可以得到1,2，…，*n*的一个置换：将数字1，2,…，*n*环形排列，按顺时针方向自1开始报数，报到*K*时输出该位置上的数字，并使其出列。然后从它在顺时针方向的下一个数字继续报数，如此下去，直到所有的数字全部出列为止。例如，*N*=10、*K*=3时，则正确的出列顺序应为3，6，9，2，7，1，8，5，10，4。

【设计要求】在程序中构造一个子程序为:

```
void Js(int n,int k)                    /*按正确的输出次序输出约瑟夫环中的元素*/
```

实验3　链表基本操作

一、实验目的

1. 定义单链表的结点类型。
2. 熟悉对单链表的一些基本操作和具体的函数定义。
3. 通过单链表的定义掌握线性表的链式存储结构的特点。
4. 掌握循环链表和双链表的定义和构造方法。

二、实验内容

该程序的功能是实现单链表的定义和操作。该程序包括单链表结构类型以及对单链表操作的具体的函数定义和主函数。程序中的单链表（带头结点）结点为结构类型，结点值为整型。

```
/* 定义DataType为int类型 */
typedef int DataType;

/* 单链表的结点类型 */
typedef struct LNode
    {DataType data;
     struct LNode *next;
    }LNode,*LinkedList;

/* 初始化单链表 */
LinkedList LinkedListInit()

/* 清空单链表 */
void LinkedListClear(LinkedList L)

/* 检查单链表是否为空 */
int LinkedListEmpty(LinkedList L)

/* 遍历单链表 */
void LinkedListTraverse(LinkedList L)

/* 求单链表的长度 */
int  LinkedListLength(LinkedList L)

/* 从单链表中查找元素 */
LinkedList  LinkedListGet(LinkedList L,int  i)

/* 从单链表中查找与给定元素值相同的元素在链表中的位置 */
```

```
LinkedList  LinkedListLocate(LinkedList  L, DataType  x)

/* 向单链表中插入元素 */
void  LinkedListInsert(LinkedList L,int i,DataType  x)

/* 从单链表中删除元素 */
void LinkedListDel(LinkedList  L,DataType x)

/* 用尾插法建立单链表 */
LinkedList  LinkedListCreat( )
```

实验4　链表其他操作

一、实验目的

1. 熟悉对单链表的一些其他操作。
2. 掌握循环链表和双链表的一些操作，理解与单链表操作的不同。

二、实验内容

程序1

设单链表L是一个非递减有序表，写一算法将x插入其中后仍保持L的有序性。

【设计要求】在程序中构造三个子程序分别为:

```
LinkedList  LinkedListCreat( )                    /*建立链表*/
void InsertList(LinkedList L,int x)               /*插入结点*/
void print(LinkedList L);                         /*输出链表中的结点*/
```

程序2

利用原空间，将两个单链表合并成一个单链表。

【设计要求】在程序中构造三个子程序分别为:

```
LinkedList  LinkedListCreat( )                          /*建立链表*/
LinkedList ListConcat(LinkedList La,LinkedList Lb)      /*合并链表*/
void print(LinkedList L);                               /*输出链表中的结点*/
```

程序3

已知两个非递减有序的单链表La和Lb，将La和Lb合并成一个线性表Lc，Lc也非递减有序。

【设计要求】在程序中构造三个子程序分别为:

```
LinkedList  LinkedListCreat( )                    /*建立链表*/
LinkedList union(LinkedList La,Lb)                /*合并链表*/
void print(LinkedList Lc);                        /*输出链表中的结点*/
```

程序4

已知一个单链表，利用原表把单链表逆置。

【设计要求】在程序中构造三个子程序分别为:

```
LinkedList  LinkedListCreat( )                    /*建立链表*/
void List_reverse(LinkedList L)                   /*逆置链表*/
void print(LinkedList L);                         /*输出链表中的结点*/
```

程序5

在计算机上先输入一串正整数的序列。请编写一个程序，首先用链表存储该序列。然后执行删除操作，即先从链表中找出最小的结点，删除它。然后再在剩余的链表中，找出最小的结点，再删除之。直至表空为止。

【设计要求】在程序中构造四个子程序分别为:

```
LinkedList  LinkedListCreat( )                    /*建立链表*/
int min(LinkedList head);                         /*求链表中的最小结点*/
LinkedList del(LinkedList head, int num);         /*删除结点*/
void print(LinkedList L);                         /*输出链表中的结点*/
```

程序6

利用单循环链表作为存储结构，实现约瑟夫环问题。

【设计要求】在程序中构造三个子程序分别为:

```
CiLinkList  CiLinkListCreat( )                    /*建立不带头结点的单循环链表*/
void del(CiLinkList last, int N, int K)           /*依次输出符合要求的结点*/
void print(CiLinkList L);                         /*输出链表中的结点*/
```

程序7

在双链表上实现下列操作:

a）建立双链表

```
DLinkedList  creat()
```

b）插入元素

```
void  DlistInsert(DLinkedList L,int x,int i)
```

c）删除元素

```
void  DlistDelete(DLinkedList L,int i)
```

程序8

设有一个双链表，每个结点中除有prior、next及data（可设为正整数）三个域之外，还有一个专门记录访问该结点次数的数据域freq，其值在初始化时为零。每当在链表中进行一次*Search* (l, *key*)时，则数据域data值等于*key*的结点，其freq域值将加1。并使该双链表中结点按freq值的递减顺序排列，freq值越大的结点越靠近表头。请编写符合上述要求的*Search* (l, *key*)程序。

【设计要求】在程序中构造三个子程序分别为:

```
DLinkedList Creat()                               /*建立链表*/
void Search(DLinkedList head,int key)             /*查找链表中符合要求的结点*/
void print(DLinkedList head);                     /*输出链表中的结点*/
```

实验5　表达式求值

一、实验目的

1. 会定义顺序栈和链栈的结点类型。
2. 掌握栈的插入和删除结点操作的特点。
3. 熟悉对栈的一些基本操作和具体的函数定义。

二、实验内容

程序1

该程序的功能是实现顺序栈的定义和操作。该程序包括定义的栈结构类型以及对每一种栈操作的具体的函数定义和主函数。

```
/* 定义DataType为int类型 */
typedef int DataType;

/* 栈的结点类型 */
#define MAXSIZE  1024
typedef  struct
    {DataType  data[MAXSIZE];
     int  top;
    }SeqStack;

/* 初始化顺序栈 */
SeqStack  SeqStackInit()

/* 检查顺序栈是否为空 */
int SeqStackEmpty(SeqStack S)

/* 把S置为空栈 */
void ClearStack(SeqStack  S)

/* 把元素x压入栈，使其成为新的栈顶元素 */
void SeqStackPush(SeqStack S,DataType x)

/* 把栈顶元素弹出 */
DataType SeqStackPop(SeqStack S)

/* 取栈顶元素 */
DataType SeqStackGetTop(SeqStack S)

/*输出顺序栈中的元素*/
```

```
void SeqStackPrint(SeqStack S)
```

程序2

用顺序栈实现算术表达式求值。将表达式看成字符串序列，输入语法正确、不含有变量的整数表达式（表达式中的数字限为单位数），利用运算符的优先关系，把中序表达式转换为后序表达式后输出，然后求出该后序表达式的值。

例如：输入的表达式为2*(6-4)+8/4，转换后得到的后序表达式为264-*84/+。

【设计要求】在程序中构造六个子程序分别为：

```
int empty(SeqStack stack); /*检查栈是否为空*/
int operation(char op); /*判断是否为运算符*/
int priority(char op); /*判断运算符的优先权*/
SeqStack push(SeqStack stack,char value); /*入栈*/
SeqStack pop(SeqStack stack,char *value); /*出栈*/
double count(char *backorder); /*计算逆波兰表达式的值*/
```

程序3

用链栈实现算术表达式求值（与程序2的基本要求相同）。

链栈结点的类型描述如下：

```
typedef int DataType;
typedef  struct StackNode
    {DataType data;
    struct StackNode *next;
    }StackNode, *LinkedStack;
```

实验6　数组的建立和使用

一、实验目的

1. 掌握C语言中数组的类型定义。
2. 掌握数组的建立和使用的特点。

二、实验内容

程序1

在计算机上以字符串的形式输入了两个任意长的整数，编写求这两个整数的积的程序。

程序2

若矩阵$A_{m\times n}$中的某个元素a_{ij}是第i行的最小值，同时又是第j列中的最大值，则称此元素为该矩阵中的一个马鞍点。假设以二维数组存储矩阵$A_{m\times n}$，试编写求出矩阵中所有马鞍点的算法。