可下载教学资料
http://www.tup.tsinghua.edu.cn

21世纪普通高校计算机公共课程规划教材

数据结构实验与实训教程（第4版）

邓文华 主编

清华大学出版社

21世纪普通高校计算机公共课程规划教材

数据结构实验与实训教程（第4版）

邓文华 主编

清华大学出版社
北京

内 容 简 介

本书是配合独立学院三本各类专业的《数据结构》一书编写的《数据结构实验与实训教程》教材，根据教学内容及针对独立学院三本学生的实际情况，本书在内容编排上共分成4个部分：第一部分为预备知识，包括C语言的数据输入、结构体的概念、函数的传址调用概念及预备知识实验；第二部分为基础实验部分，给出了11个实验，包括线性结构、树形结构、图结构、查找、排序以及数组和字符串等操作；第三部分为课程设计实验部分，包括航空客运订票系统、汉诺塔游戏程序、全屏幕编辑程序设计、旅游路线安排模拟系统、停车场管理和最小生成树的Kruskal算法共6个综合性实验，此部分实验可作为数据结构的课程设计之用；为了满足教学和各类学生学习课程与考前复习的需要，第四部分安排了12套模拟试题，并给出了详细的解答。

本书内容丰富、概念清晰、实用性强。本书与"数据结构"课程的主要内容紧密结合，可供各类学生课程学习、实验、课程设计和考前复习使用，也可供教师或其他专业技术人员参考。

图书在版编目(CIP)数据

数据结构实验与实训教程/邓文华主编. —4版. —北京：清华大学出版社，2014
21世纪普通高校计算机公共课程规划教材
ISBN 978-7-302-36144-2

Ⅰ. ①数… Ⅱ. ①邓… Ⅲ. ①数据结构—教材 Ⅳ. ①TP311.12

中国版本图书馆CIP数据核字(2014)第072591号

责任编辑：魏江江　王冰飞
封面设计：何凤霞
责任校对：焦丽丽
责任印制：沈　露

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦A座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课 件 下 载：http://www.tup.com.cn，010-62795954
印 刷 者：北京富博印刷有限公司
装 订 者：北京市密云县京文制本装订厂
经　　销：全国新华书店
开　　本：185mm×260mm　　印　　张：11.75　　字　　数：285千字
版　　次：2004年8月第1版　2014年7月第4版　　印　　次：2014年7月第1次印刷
印　　数：1～2000
定　　价：25.00元

产品编号：059508-01

出版说明

随着我国改革开放的进一步深化，高等教育也得到了快速发展，各地高校紧密结合地方经济建设发展需要，科学运用市场调节机制，加大了使用信息科学等现代科学技术提升、改造传统学科专业的投入力度，通过教育改革合理调整和配置了教育资源，优化了传统学科专业，积极为地方经济建设输送人才，为我国经济社会的快速、健康和可持续发展以及高等教育自身的改革发展做出了巨大贡献。但是，高等教育质量还需要进一步提高以适应经济社会发展的需要，不少高校的专业设置和结构不尽合理，教师队伍整体素质亟待提高，人才培养模式、教学内容和方法需要进一步转变，学生的实践能力和创新精神亟待加强。

教育部一直十分重视高等教育质量工作。2007 年 1 月，教育部下发了《关于实施高等学校本科教学质量与教学改革工程的意见》，计划实施“高等学校本科教学质量与教学改革工程（简称‘质量工程’）”，通过专业结构调整、课程教材建设、实践教学改革、教学团队建设等多项内容，进一步深化高等学校教学改革，提高人才培养的能力和水平，更好地满足经济社会发展对高素质人才的需要。在贯彻和落实教育部“质量工程”的过程中，各地高校发挥师资力量强、办学经验丰富、教学资源充裕等优势，对其特色专业及特色课程（群）加以规划、整理和总结，更新教学内容、改革课程体系，建设了一大批内容新、体系新、方法新、手段新的特色课程。在此基础上，经教育部相关教学指导委员会专家的指导和建议，清华大学出版社在多个领域精选各高校的特色课程，分别规划出版系列教材，以配合“质量工程”的实施，满足各高校教学质量和教学改革的需要。

本系列教材立足于计算机公共课程领域，以公共基础课为主、专业基础课为辅，横向满足高校多层次教学的需要。在规划过程中体现了如下一些基本原则和特点。

（1）面向多层次、多学科专业，强调计算机在各专业中的应用。教材内容坚持基本理论适度，反映各层次对基本理论和原理的需求，同时加强实践和应用环节。

（2）反映教学需要，促进教学发展。教材要适应多样化的教学需要，正确把握教学内容和课程体系的改革方向，在选择教材内容和编写体系时注意体现素质教育、创新能力与实践能力的培养，为学生知识、能力、素质协调发展创造条件。

（3）实施精品战略，突出重点，保证质量。规划教材把重点放在公共基础课和专业基础课的教材建设上；特别注意选择并安排一部分原来基础比较好的优秀教材或讲义修订再版，逐步形成精品教材；提倡并鼓励编写体现教学质量和教学改革成果的教材。

（4）主张一纲多本，合理配套。基础课和专业基础课教材配套，同一门课程有针对不同层次、面向不同专业的多本具有各自内容特点的教材。处理好教材统一性与多样化，基本教材与辅助教材、教学参考书，文字教材与软件教材的关系，实现教材系列资源配套。

（5）依靠专家，择优选用。在制定教材规划时要依靠各课程专家在调查研究本课程教

材建设现状的基础上提出规划选题。在落实主编人选时,要引入竞争机制,通过申报、评审确定主题。书稿完成后要认真实行审稿程序,确保出书质量。

繁荣教材出版事业,提高教材质量的关键是教师。建立一支高水平教材编写梯队才能保证教材的编写质量和建设力度,希望有志于教材建设的教师能够加入到我们的编写队伍中来。

21世纪普通高校计算机公共课程规划教材编委会

联系人:梁颖 liangying@tup.tsinghua.edu.cn

前言

本书与独立学院三本的《数据结构》一书一起组成配套教材，结合独立学院三本的特点，在原来第3版的基础上进行了修改与补充。其任务是通过实践让学生进一步掌握常用数据结构的基本概念及不同的实现方法，并对在不同存储结构上实现不同的运算方法和技巧有所体会。本书是专门针对独立学院三本学生的实际情况，为学习“数据结构”课程而编写的实验教材。本书共分成4个部分：第一部分为预备知识，包括C语言的数据输入、结构体的概念、函数的传址调用概念及预备知识实验；第二部分为基础实验部分，在内容安排上给出了11个实验，包括线性结构、树形结构（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）、图结构（图的两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与实现）以及数组和字符串等操作；第三部分为课程设计实验部分，包括航空客运订票系统、汉诺塔游戏程序、全屏幕编辑程序设计、旅游路线安排模拟系统、停车场管理和最小生成树的Kruskal算法共6个综合性实验，这些实验的综合性比较强，可作为“数据结构”课程的课程设计之用；第四部分安排了12套模拟试题，并给出参考解答，目的是帮助学生进一步巩固所学的理论知识。

本书具有以下几个特点：

（1）每个实验题目都给出相应的C程序模板，在模板中填写关键语句或子程序即可上机通过，便于学生集中精力于主要的算法。

（2）实验内容安排多样，包括基础题和提高题，便于满足不同层次学生的需求。其形式包括给出程序框架要求填写关键算法的形式，给出类似函数要求独立编写程序的形式，给出主程序要求编写子程序的形式，以及给出算法要求编写程序的形式等。

（3）第四部分安排了12套模拟试题，并给出详细的参考解答，有利于学生的自学、复习。

本书由邓文华任主编，其中，第二、三部分由邓文华修改、编写；第四部分由胡智文修改、编写；第一部分由毕保祥编写。全书由邓文华统稿。

本书的所有程序都在Turbo C或Win-TC软件开发环境下调试运行通过。

编写独立学院三本的教材是一项新的尝试，难免存在疏漏，恳请读者赐教指正。

编　者

2014年4月

目 录

第一部分 预备知识

数据结构预备知识

数据结构不仅具有较强的理论性，更具有较强的实践性。在数据结构的教学过程中，如果学生的程序设计语言基础薄弱，会在很大程度上影响正常的教学进度。因此，为了方便学生进行后续实验，这里将数据结构所必需的C语言语法做一下简单介绍。编者根据多年的教学实践，发现学生完成上机实验时遇到的问题主要包括不能正确地输入数据，对结构体概念陌生，对函数的传址调用概念不清，对指针与链表比较生疏。由于篇幅所限，这里仅对前3个问题加以介绍，关于指针与链表的知识请学生参阅相应的C语言程序设计教程。

一、基本输入和输出

上机实践离不开数据的输入和输出。对于一个算法程序，如果数据不能正确地输入，算法设计得再好也无法正常运行。虽然输入和输出看起来简单，但是往往在上机实验时最容易出错，尤其是输入。

（一）scanf()输入函数

C语言的输入由系统提供的输入函数来实现，例如scanf()函数等。因为标准的输入和输出函数都在头文件stdio.h中声明，所以只有在程序中将其包含进来才能使用。在程序的首部一般要求写入：

```
# include < stdio.h >
```

scanf()函数的功能很多，输入格式也多种多样，这里只给出使用时需要注意的几个问题。

(1) 如果一条scanf()语句中有多个变量且都是数值型(int、float、double)，在输入数据时可在一行之内输入多个数据，数据之间用空格分隔。如果语句中在%d和%f之间有一个逗号“,”，则在数据之间也要以“,”分隔。例如：

```
int x; float y;
scanf ("%d %f", &x, &y);
```

正确的输入应该是“整数 空格 实数 回车”，例如：

```
50  78.3
```

就是在两个数据之间用空格键作为分隔符，最后按回车键。

再如：

```
scanf ("%d, %f", &x, &y);
```

在%d和%f之间有一个逗号,那么正确的输入方法是"整数 逗号 实数 回车",例如:

```
50,78.3
```


(2) 不要将字符型变量或字符串与数值型变量混合在一条scanf()语句中,而应该各自单独写一条输入语句,这样不容易出错。"空格键"在数值型数据之间起"分隔符"的作用,而在字符或字符串之间,"空格"则被当做一个输入的字符,不能起到"分隔符"的作用。

(3) scanf()函数中必须传递变量的地址,因为数组名本身就是数组的首地址,所以在scanf语句中的字符数组名之前不得冠以取地址符"&",而在整型变量、实型变量和字符型变量之前必须加上"&"。

(二) printf()输出函数

C语言的输出是由系统提供的printf()等输出函数来实现的,与使用scanf()函数一样,在程序的首部也要写入:

```
# include <stdio.h>
```

关于printf()函数,这里强调以下几点。

(1) 在scanf()之前,应该先使用printf()语句进行必要的提示。例如:

```
char name[10], x;
int y;
printf("\n 请输入学生的姓名、性别和成绩: ");
scanf( "%s",name);
scanf( "%c",&x);
scanf( "%d",&y);
```

(2) 在该换行的地方要及时换行。例如

```
int i;
printf("数据输出如下: \n");                    /* 输出并换行 */
for (i=1; i<=15; i++) {
  printf("%5d", i*i );
  if(i%5==0) printf("\n");                     /* 每行输出5个数据 */
}
```

(3) 在调试程序时可以适当添加一些输出语句,以便监视程序的运行状态,在程序调试成功后再将这些输出语句去掉。

二、结构体的运用

在C语言中,结构体的定义、输入与输出是数据结构程序设计的重要语法基础。

(一) 定义结构体类型的一般格式

定义结构体类型的一般格式如下:

```
struct 结构体类型名
{   类型名1     成员名1;
    类型名2     成员名2;
    ...
```

```
    类型名 n      成员名 n;
};
```

其中，每个成员都是结构体的一个数据子域。数据子域的类型一般是基本数据类型（int、char 等），也可以是数组，还可以是已经定义的另一个结构体。

（二）定义结构体变量的方法

定义结构体变量的方法如下：

```
struct 结构体类型名   结构体变量名 1,结构体变量名 2,…,结构体变量名 m;
```

例如：

```
struct  ElemType                          /* 定义结构体类型 */
{ int  num;
  char  name[10];
};
struct  ElemType  x;                      /* 定义结构体变量 x */
```

（三）用 typedef 语句为结构体类型定义新的类型名

例如：

```
typedef  struct
      { int  num;
        char  name[10];
      } ElemType;
```

其中，ElemType 就是结构体类型的一个新类型名。这样，定义变量 x 的语句就可以写为：

```
ElemType  x;
```

（四）结构体变量的引用

（1）用“.”引用。

```
<结构体变量名>.<成员名>
```

例如：

```
x.num = 10;
scanf("%s",x.name);
```

其使用方法和同类型的简单变量一样。

（2）用“->”引用。

```
<指针变量名> -> <成员名>
```

其中，<指针变量名>必须是指向同类型结构体的指针变量。例如：

```
ElemType  *pt;
    pt = &x;                              /* pt 保存 x 的地址,常称为使 pt 指向变量 x */
```

形如 pt->num 和 pt->name 的表示读做“pt 所指的 num”和“pt 所指的 name”。其使用方法和同类型的简单变量一样。

(五)一个简单的结构体应用程序

设一个人的记录如下:

姓名:Zhang San
年龄:20
性别:男
身高:175厘米
体重:68千克

下面用struct定义这位男生的记录结构,并用一个结构体变量保存其记录的值。另外,用两个变量分别演示用"."和"->"方法引用结构体的成员。

```
main(){
struct person {
  char name[20];
  int age;
  char sex;
  int height;
  int weight;
} stu1, * p;
/ * 初始化 stu3 * /
  struct person stu3 = {"Zhang San",20,'M',175,68};
  printf("   % s     % d     % c     % d     % d\n", stu3. name, stu3. age = 20, stu3. sex, stu3.
height,     stu3. weight);
/ * 初始化 stu1 * /
  strcpy(stu1. name,"zhang San");
  stu1. age = 20;  stu1. sex = 'M';
  stu1. height = 175;  stu1. weight = 68;
  printf("   % s     % d     % c     % d     % f\n", stu1. name, stu1. age = 20, stu1. sex, stu1.
height, stu1. weight);
/ * 实际上,用一条赋值语句"stu1 = stu3;"即可 * /

/ * 初始化 person2 * /
p = &stu1;
strcpy(p -> name,"Wang Wu");
p -> age = 19;
p -> sex = 'M';
p -> height = 170;
p -> weight = 65;
printf("   % s     % d     % c     % d     % d\n", p -> name, p -> age = 20, p -> sex, p -> height,
p -> weight);
}//结束 main 函数
```

三、函数的参数传递

函数是组成C语言程序的基本单位,函数具有相对独立的功能,可以被其他函数调用,也可以调用其他函数。直接或间接调用自身的函数称为递归函数。C语言的源程序中必须有一个主函数main(),还可以包含若干个自定义函数。函数的设计和调用是程序设计必不可少的技能,是程序设计最重要的基础,能否熟练地设计和使用函数是体现学生程序设计能

力高低的基本条件。这里对C语言函数的基本概念进行简单的总结。

（一）函数的定义

函数定义的一般格式如下：

```
类型名   函数名(形式参数表)
{   函数体; }
```

其中，类型名是函数返回值的数据类型，例如int、float等。返回值是在函数运行时所得到的某个结果，例如：

```
float  funx(形参表)
{ 函数体; }
```

表明函数funx()在被调用并执行完成时有一个float型数值返回。

函数的类型还可以是void，表明函数不需要返回值。例如：

```
void  funy(形参表)
{ 函数体; }
```

例1.1 下面程序中有两个自定义函数和一个主函数，其中，自定义函数sumabc()计算3个整数之和，自定义函数displayLine()仅输出一条线，主函数main()调用两个自定义函数。

程序如下：

```
#include  <stdio.h>
int   sumabc(int a, int b, int c)                    /*求3个整数之和*/
{  int  s;
    a=b+c;
    s=a+b+c;
    return s;
}
void  displayLine(void)
 { printf("----------------------\n");
 }
void  main( )
{   int  x,y, z ,sabc;
    x=y=z=8;
    display();                                       /*画一条线*/
    printf("\n sum= %d",sumabc(x,y,z));              /*在输出语句中直接调用函数sumabc()*/
    printf("\n  %6d%6d%6d",x,y,z);
    display();                                       /*画一条线*/
    x=2; y=4; z=6;
    sabc =sumabc(x, y, z);                           /*在赋值语句中调用函数sumabc()*/
    printf("\n   sum= %d", sabc);
    printf("\n  %6d%6d%6d",x,y,z);
    display();                                       /*画一条线*/
}
```

运行结果：

```
----------------------

sum = 32
      8     8     8
----------------------
sum = 20
      2     4     6
----------------------
```

(二) 函数的参数传递

函数在被调用时，由主调函数提供实参，并传递给形参。在调用结束后，有时可通过参数返回新的数据给主调函数，以实现数据在主调函数和被调函数之间的双向传递。C语言实现参数传递的方法分为传值和传址两大类。

例1.1中的函数sumabc()被主函数main()调用就是传值调用。也就是说，将x、y和z的值分别赋给a、b和c。传值调用只能单向传递，即从主调函数到被调函数。被调函数返回后，在主函数中输出的实参的值仍与调用之前相同，即不论在调用期间形参值是否改变，在调用结束返回到主调函数之后实参值不会改变，例如x的值没有改变。

在C语言中采用指针变量做形参来实现传址。传址调用的主要特点是可以实现数据的双向传递。在调用时实参将地址传给形参，通过地址将对应的数据代入被调函数。如果在调用期间形参值发生改变(即该地址对应的单元中的数据发生变化)，那么调用结束返回主调函数之后，因为实参地址没有改变，所以其对应的单元中被改变的数据得以返回，从而实现了数据的双向传递。

例1.2 将例1.1中的函数sumabc()修改为如下：

```
int   sumabc(int *a, int b, int c)
{
      int  s;
      *a=b+c;
      s=*a+b+c;
      return s;

}
```

在main()中第一次调用sumabc()函数的语句如下：

```
x=y=z=8;
printf("\n sum= %d",sumabc(&x,y,z));
printf("\n   %6d%6d%6d",x,y,z);
```

第二次调用如下：

```
x=2; y=4; z=6;
sabc =sumabc(&x, y, z);
printf("\n   sum= %d", sabc);
printf("\n   %6d%6d%6d",x,y,z);
```

则x的值在调前后发生了改变。

运行结果：

```
----------------------

sum = 32
      16     8     8
----------------------
sum = 20
      10     4     6
----------------------
```

函数的设计和运用是C语言程序设计课程中重要的一部分内容，也有一定的难度，尤其值得大家重点关注。

预备知识实验

一、实验目的

1. 了解抽象数据类型(ADT)的基本概念及描述方法。

2. 通过对复数抽象数据类型ADT的实现熟悉C语言语法及程序设计，为以后章节的学习打下基础。

二、实例

复数抽象数据类型ADT的描述及实现。

[复数ADT的描述]

```
ADT complex{
  数据对象：D={ c1,c2  | c1,c2∈FloatSet }
  数据关系：R={ <c1,c2> | c1   c2    }
  基本操作：创建一个复数          creat(a);
            输出一个复数          outputc(a);
            求两个复数相加之和    add(a,b);
            求两个复数相减之差    sub(a,b);
            求两个复数相乘之积    chengji(a,b);
            等等;
          } ADT complex;
```

[复数ADT实现的源程序]

```
#include <stdio.h>
#include <stdlib.h>
/* 存储表示,结构体类型的定义 */
typedef  struct
   { float x;                              /* 实部子域 */
    float y;                               /* 虚部的实系数子域 */
   }comp;
/* 全局变量的说明 */
comp a,b,a1,b1;
int z;
/* 子函数的原型声明 */
```

```
void creat(comp *c);
void outputc(comp a);
comp add(comp k,comp h);
/*主函数*/
main()
{ creat(&a);   outputc(a);
  creat(&b);   outputc(b);
  a1 = add(a,b); outputc(a1);
}                                                    /*main*/
/*创建一个复数   */
void creat(comp *c)
{ float c1,c2;
  printf("输入实部 real x = ?");scanf("%f",&c1);
  printf("输入虚部 xvpu y = ?");scanf("%f",&c2);
  (*c).x = c1;  c ->y = c2;
}                                                    /*creat*/
/*输出一个复数*/
void outputc(comp a)
  { printf("\n   %f + %f i \n\n",a.x,a.y);
   }
/*求两个复数相加之和*/
comp add(comp k,comp h)
{ comp l;
  l.x = k.x + h.x;   l.y = k.y + h.y;
  return(l);
  }                                                  /*add*/
```

三、练习题

1. 将上面的源程序输入计算机,并进行调试。
2. 编写求两个复数的差、积、商的函数。
3. 编写求一个复数的实部、虚部、模的函数。
4. 编写 main()分别调用以上函数。
5. 输入测试数据,输出结果(数据自定)。

第二部分 基础实验

实验1 线性表的基本操作

一、线性表的基本概念

线性表是一种线性结构，线性结构的特点是数据元素之间是一种一对一的线性关系，数据元素“一个接一个的排列”。在一个线性表中数据元素的类型是相同的，或者说线性表是由同一类型的数据元素构成的线性结构。在实际问题中线性表的例子很多，例如学生情况信息表是一个线性表，表中数据元素的类型为学生类型。一个字符串也是一个线性表，表中数据元素的类型为字符型，等等。

综上所述，线性表的定义如下：

线性表是具有相同数据类型的 $n(n\geqslant 0)$ 个数据元素的有限序列，通常记为：

$$(a_1,a_2,\cdots,a_{i-1},a_i,a_{i+1},\cdots,a_n)$$

其中，n 为表长，当 $n=0$ 时线性表称为空表。

表中相邻元素之间存在着顺序关系，将 a_{i-1} 称为 a_i 的直接前趋，将 a_{i+1} 称为 a_i 的直接后继。也就是说，对于 a_i，当 $i=2,\cdots,n$ 时，有且仅有一个直接前趋 a_{i-1}，当 $i=1,2,\cdots,n-1$ 时，有且仅有一个直接后继 a_{i+1}，a_1 是表中的第一个元素，它没有前趋，a_n 是最后一个元素，它没有后继。

线性表是最简单、最基本、最常用的一种线性结构，它有两种存储方法，即顺序存储和链式存储，主要的基本操作是插入、删除和检索等。

二、实验目的

1. 掌握线性表的基本运算。
2. 掌握顺序存储的概念，学会对顺序存储数据结构进行操作。
3. 加深对顺序存储数据结构的理解，逐步培养解决实际问题的编程能力。

三、实验内容

（一）基础题

1. 编写线性表基本操作函数：

(1) InitList(LIST *L, int ms)初始化线性表；

(2) InsertList(LIST *L, int item, int rc)向线性表的指定位置插入元素；

(3) DeleteList1(LIST *L, int item)删除指定元素值的线性表记录；