面向21世纪

高等职业技术教育计算机类系列教材

数据结构（C语言）

主 编 刘喜勋

主 审 王 津

西安电子科技大学出版社

http://www.xduph.com

面向21世纪高等职业技术教育计算机类系列教材

数据结构(C语言)

主　编　刘喜勋

副主编　刘　肖

参　编　高晓梅　李小遐　梁　英

主　审　王　津

西安电子科技大学出版社

2003

内 容 简 介

本书共分 10 章。书中详细介绍了各种数据结构以及查找、排序的各种方法，对每一种类型的数据结构以实例为切入点，详细叙述了基本概念、逻辑结构、存储结构和常用算法。

本书专为高等职业技术学院计算机类专业学生学习“数据结构”课程而编写，本着注重应用的原则，选材精练，对基本理论的叙述深入浅出、通俗易懂。书中实例丰富，主要算法均给出了 C 语言函数。为了便于教学，每章后还配有实习和习题。

★本书配有电子教案，需要者可与出版社联系，免费索取。

图书在版编目(CIP)数据

数据结构(C 语言) / 刘喜勋主编. —西安：西安电子科技大学出版社，2003.7

(面向 21 世纪高等职业技术教育计算机类系列教材)

ISBN 7-5606-1268-7

Ⅰ. 数… Ⅱ. 刘… Ⅲ. ① 数据结构—高等学校—教材 ② C 语言—程序设计—高等学校—教材 Ⅳ. ① TP313.12 ② TP312

中国版本图书馆 CIP 数据核字(2003)第 050126 号

策　　划　马武装

责任编辑　龙晖

出版发行　西安电子科技大学出版社(西安市太白南路 2 号)

电　　话　(029)8242885　8201467　邮　编　710071

http: //www.xduph.com　　E-mail: xdupfxb@pub.xaonline.com

经　　销　新华书店

印刷单位　陕西华沐印刷科技有限责任公司

版　　次　2003 年 7 月第 1 版　2003 年 7 月第 1 次印刷

开　　本　787 毫米×1092 毫米　1/16　印张 10

字　　数　228 千字

印　　数　1～4 000 册

定　　价　11.00 元

ISBN 7 – 5606 – 1268 – 7 / TP・0666(课)

XDUP 1539001-1

序

进入21世纪后，世界已经步入知识经济发展的时期，随着我国社会主义市场经济的快速发展，各行各业越来越需要具有综合职业能力和素质全面的，直接工作在生产、技术、管理和服务第一线的应用型、技能型的高级实用人才。高等职业技术教育的任务就是面向不同岗位，培养具备一定知识和技能，具有一定职业岗位能力和跨职业、跨岗位关键能力，德、智、体全面发展的高级技术和技艺型人才。据权威机构的规划，2005年，我国高等院校在校生规模将达1600万人，其中50%是高等职业教育的学生。这说明高等职业技术教育即将和高等教育的本科教育相提并论，在我国高等教育体系中占有相当重要的地位。

高职教育作为我国高等教育的一个重要组成部分，其培养目标是具有必要理论知识和较强实践能力的高等技术应用型专门人才。它的人才培养模式应该是以培养适应生产、建设、管理、服务第一线需要的高等技术应用型专门人才为根本任务；以适应社会需要为目标；以培养技术应用能力为主线；以突出职业性、实践性、适应性和地方性为特点。计算机教学应以传授应用知识为主，强调操作使用，注重培养学生利用计算机开展专业技术分析、解决各种技术问题的意识，培养学生的自学能力和创造性学习的能力。

在我国高等职业技术教育发展的过程中，虽然部分学校已经取得了一些成功经验，并逐渐形成了自己的办学特色，但从总体上来看，高等职业技术教育尚处于起步阶段。高职教材建设明显跟不上高职发展的需要，主要表现在借用本科教材和沿用专科教材的问题上。这类教材多数在编写上以本科教材为蓝本，是“本科压缩型”，尤其在以“应用”为主旨和特征构建课程与教学内容体系上，存在着明显不足，难以符合高等职业技术教育培养目标的要求，对高职人才培养十分不利。因此，做好高职教材改革与建设工作刻不容缓。

为了促进高等职业技术教育教材建设，西安电子科技大学出版社组织陕西省高职院校的骨干教师共同策划编写了高职教育非计算机专业和计算机专业系列教材，现已出版。本系列教材以适应社会需要为目标，以培养技术应用能力为主线来设计学生的知识、能力、素质结构和培养方案。编写上本着重能力、严实践、求创新的总体思路；体现科学性、思维性、启发性、先进性和教学的适用性；以培养能力为主，基础理论适度，适当反映科学技术领域内的新成果来优化课程内容。本套教材突出了高职教材的特色，适合高等职业学校、高等专科学校、成人高校等高等职业技术教育和五年制高等职业技术教育以及部分中等职业技术教育的需要。

系列教材编委会

2002年8月

高等职业技术教育计算机类系列教材

编委会名单

前　言

“数据结构”是计算机专业的一门专业基础课，是核心课程之一。通过对“数据结构”课程的学习，使学生掌握有关数据的逻辑结构和物理结构的知识，提高学生分析问题、解决问题的技能。

为适应高等职业技术教育的发展，进一步提高计算机专业“数据结构”课程的教学质量，我们根据多年的教学经验，在分析国内外同类教材的基础上，博采众长，编写了这本书，奉献给广大读者。

全书共分 10 章。第 1 章叙述了数据结构的概念，并对算法描述规则及算法分析作了说明；第 2 章至第 7 章分别介绍线性表、队列、串、数组、树和图基本类型的数据结构及其应用；第 8 章和第 9 章介绍了查找和排序的分类及方法；第 10 章介绍了文件的基本概念。每章后都配有实习，并给出了完整的 C 语言程序。

本书专为高等职业技术学院计算机类专业学生而编写。

本书由刘喜勋任主编（编写第 2 章、第 3 章、第 7 章，第 8 章），刘肖任副主编（编写第 4 章、第 5 章），梁英（编写第 6 章）、李小遐（编写第 9 章、第 10 章）、高晓梅（编写第 1 章）参编，由刘喜勋统稿，王津主审。

本书编者都是多年从事本课程教学的教师，但由于编者水平有限，加之时间匆促，不妥与疏漏之处在所难免，敬请广大读者指正。

编　者

2003 年 5 月

目　录

目　录

目　录

目　录

第1章 绪 论

随着计算机科学技术、计算机产业的迅速发展，计算机应用已经深入到人类社会的各个领域。计算机不仅用于科学计算，而且更多地应用于信息处理。处理的对象不再是纯粹的数值，而是扩展到字符、表格、声音和图像等各种各样的信息。信息的处理也不再只是单纯的计算，而是一些如信息存储、信息检索等非数值计算的处理。信息的表示是计算机科学的基础。要解决某一复杂问题，设计一个高效适用的程序，就需要解决怎样合理地组织数据，建立合适的数据结构，提高程序执行的时间效率和空间效率等问题。这就是“数据结构”形成和发展的背景。

1.1 数据结构的基本概念和术语

1.1.1 引例

首先分析学籍档案类问题。设一个班级有50个学生，这个班级的学籍表如表1.1所示。

表1.1 学 籍 表

序号	学号	姓名	性别	英语	数学	物理
01	20030301	李明	男	86	91	80
02	20030302	马琳	男	76	83	85
⋮	⋮	⋮	⋮	⋮	⋮	⋮
50	20030350	刘薇薇	女	88	93	90

我们可以把表中每个学生的信息看成一个记录，表中的每个记录又由7个数据项组成。该学籍表由50个记录组成，记录之间是一种顺序关系。这种表通常称为线性表，数据之间的逻辑结构称为线性结构，其主要操作有检索、查找、插入或删除等。

又如，对于学院的行政机构，可以把该学院的名称看成树根，把下设的若干个系看成它的树枝中间结点，把每个系分出的若干专业方向看成树叶，这样就形成一个树型结构，如图1.1所示。

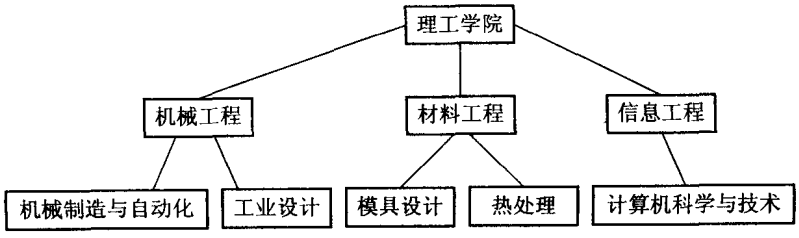

图1.1 专业设置

树中的每个结点可以包含较多的信息，结点之间的关系不再是顺序的，而是分层、分叉的结构。树型结构的主要操作有遍历、查找、插入或删除等。

最后分析交通问题。如果把若干个城镇看成若干个顶点，再把城镇之间的道路看成边，它们可以构成一个网状的图(如图 1.2 所示)，这种关系称为图型结构或网状结构。在实际应用中，假设某地区有 5 个城镇，有一调查小组要对该地区每个城镇进行调查研究，并且每个城镇仅能调查一次，试问调查路线怎样设计才能以最高的效率完成此项工作？这是一个图论方面的问题。交通图的存储和管理确实不属于单纯的数值计算问题，而是一种非数值的信息处理问题。

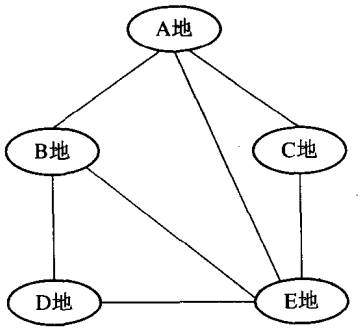

图 1.2　交通示意图

1.1.2　数据结构有关概念及术语

一般来说，数据结构研究的是一类普通数据的表示及其相关的运算操作。数据结构是一门主要研究怎样合理地组织数据，建立合适的数据结构，提高计算机执行程序所用的时间效率和空间效率的学科。1968 年，美国的 D.E.Knuth 教授开创了数据结构的最初体系，他的名著《计算机程序设计技巧》较为系统地阐述了数据的逻辑结构和存储结构及其操作。

“数据结构”是计算机专业的一门专业基础课。它为操作系统、数据库原理、编译原理等后继专业课程的学习奠定了基础。数据结构涉及到各方面的知识，如计算机硬件范围中的存储装置和存取方法，计算机软件范围中的文件系统、数据的动态存储与管理、信息检索，数学范围中的许多算法知识。

在计算机科学中，数据(Data)是描述客观事物的数字、字符以及所有能够输入到计算机中并被计算机处理的信息的总称。除了数字、字符之外，还有用英文、汉字或其他语种字母组成的词组、语句，以及表示图形、图像和声音等的信息，这些也可称为数据。

数据元素(Data Element)是数据的基本单位，在计算机中通常作为一个整体进行考虑和处理。例如，图 1.1“专业设置树”中的一个专业，图 1.2“交通图”中的一个城镇都可称为一个数据元素。数据元素除了可以是一个数字或一个字符串以外，它也可以由一个或多个数据项组成。例如，表 1.1 中每个学生的学籍信息作为一个数据元素，在表中占一行。每个数据元素由序号、学号、姓名、性别、英语成绩等 7 个数据项组成。数据项(Data Item)是有独立含义的数据的最小单位，有时也称为字段(Field)。

数据对象(Data Object)是具有相同性质的数据元素的集合，是数据的一个子集。例如，整数数据对象是集合 N={0，±1，±2,…}，字母字符数据对象是集合 C={'A', 'B', …, 'Z'}。

本节表 1.1 中的学籍表也可看成一个数据对象。

数据结构(Data Structure)是带有结构的数据元素的集合，它是指数据元素之间的相互关系，即数据的组织形式。我们把数据元素间的逻辑上的联系，称为数据的逻辑结构。常见的数据结构有前文所介绍的线性结构、树型结构、图型结构。数据的逻辑结构体现数据元素间的抽象化相互关系，并不涉及数据元素在计算机中具体的存储方式，是独立于计算机的。

然而，讨论数据结构的目的是为了在计算机中实现对数据的操作，因此还需要研究如何在计算机中表示数据。数据的逻辑结构在计算机存储设备中的映像被称为数据的存储结构，也可以说数据的存储结构是逻辑结构在计算机存储器中的实现，又称物理结构。数据的存储结构是依赖于计算机的。常见的存储结构有顺序存储结构、链式存储结构等。关于它们的详细解释将在以后的章节中逐步给出。

通常所谓的“数据结构”是指数据的逻辑结构、数据的存储结构以及定义在它们之上的一组运算。不论是存储结构的设计，还是运算的算法设计，都必须考虑存储空间的开销和运行时间的效率。因此，“数据结构”课程不仅讲授数据信息在计算机中的组织和表示方法，同时也训练学生高效地解决复杂问题程序设计的能力。

1.2 算法描述与分析

1.2.1 什么是算法

在解决实际问题时，当确定了数据的逻辑结构和存储结构之后，需进一步研究与之相关的一组操作(也称运算)，主要有插入、删除、排序、查找等。为了实现某种操作(如查找)，常常需要设计一种算法。算法(Algorithm)是对特定问题求解步骤的一种描述，是指令的有限序列。描述算法需要一种语言，可以是自然语言、数学语言或者是某种计算机语言。算法一般具有下列 5 个重要特性：

(1) 输入：一个算法应该有零个、一个或多个输入。

(2) 有穷性：一个算法必须在执行有穷步骤之后正常结束，而不能形成无穷循环。

(3) 确定性：算法中的每一条指令必须有确切的含义，不能产生多义性。

(4) 可行性：算法中的每一个指令必须是切实可执行的，即原则上可以通过已经实现的基本运算执行有限次来实现。

(5) 输出：一个算法应该至少有一个输出，这些输出是同输入有某种特定关系的量。

1.2.2 算法描述工具——C 语言

如何选择描述数据结构和算法的语言是十分重要的问题。传统的描述方法是用 PASCAL 语言。在 Windows 环境下涌现出一系列功能强大、面向对象的描述工具，如 Visual C++，Borland C++，Visual Basic，Visual FoxPro 等。近年来在计算机科学研究、系统开发、教学以及应用开发中，C 语言的使用越来越广泛。因此，本教材采用 C 语言进行算法描述。为了能够简明扼要地描述算法，突出算法的思路，而不拘泥于语言语法的细节，本书有以下约定：

(1) 问题的规模尺寸用 MAXSIZE 表示，约定在宏定义中已经预先定义过，例如：

```
#define MAXSIZE 100
```

(2) 数据元素的类型一般写成 ELEMTP，可以认为在宏定义中预先定义过，例如：

```
#define ELEMTP int
```

在上机实验时根据需要，可临时用其他某个具体的类型标识符来代替。

(3) 一个算法要以函数形式给出：

```
类型标识符　函数名(带类型说明的形参表)
  {语句组}
```

例如：

```
int add (int a,int b)
{int c;
  c=a+b;
  return(c);
}
```

除了形参类型说明放在圆括号中之外，在描述算法的函数中其他变量的类型说明一般省略不写，这样使算法的处理过程更加突出明了。

(4) 关于数据存储结构的类型定义以及全局变量的说明等均应在写算法之前进行说明。

下面的例子给出了书写算法的一般步骤。

例 1.1　有 n 个整数，将它们按由大到小的顺序排序，并且输出。

分析：n 个数据的逻辑结构是线性表(a_1,a_2,a_3,…,a_n)；选用一维数组作存储结构。每个数组元素有两个域：一个是数据的序号域，一个是数据的值域。

```
struct node
{int num;                                    /*序号域*/
 int data;                                   /*值域*/
}
/*算法描述 1.1*/
void simsort(struct node a [MAXSIZE], int n)  /*数组 a 的数据由主函数提供*/
{int i,j, m;
for(i=1;i<n;i++)
 for(j=1;j<=n;j++)
  if(a[i].data<a[j].data)
   {m=a[i];a[i]=a[j];a[j]=m;}
  for(i=i;i<=n;i++)
  printf("%8d %8d %10d\n",i,a[i].num,a[j].data);
}
```

1.2.3　算法分析技术初步

著名的计算机科学家 N.沃思提出了一个有名的公式：算法+数据结构=程序。由此可见，数据结构和算法是程序的两大要素，二者相辅相成，缺一不可。一种数据结构的优劣是在

实现其各种运算的算法中体现的。对数据结构的分析实质上也就是对实现其多种运算的算法的分析。评价一个算法应从四个方面进行：正确性、简单性、运行时间、占用空间。但主要看这个算法所要占用机器资源的多少。而在这些资源中时间和空间是两个最主要的方面，因此算法分析中最关心的也就是算法所需的时间代价和空间代价。

1. 空间

所谓算法的空间代价(或称空间复杂性)，是指当问题的规模以某种单位由 1 增至 n 时，解决该问题的算法实现所占用的空间也以某种单位由 1 增至 f(n)，并称该算法的空间代价是 f(n)。

2. 时间

(1) 语句频度(Frequency Count)：指的是在一个算法中该语句重复执行的次数。

(2) 算法的渐近时间复杂度(Asymptotic Time Complexity)：算法中基本操作重复执行的次数依据算法中最大语句频度来估算，它是问题规模 n 的某个函数 f(n)，算法的时间量度记作 T(n)=O(f(n))，表示随着问题规模 n 的增大，算法执行时间的增长率和 f(n)的增长率相同，称作算法的渐近时间复杂度，简称时间复杂度。时间复杂度往往不是精确的执行次数，而是估算的数量级。它着重体现的是随着问题规模的增大，算法执行时间增长的变化趋势。

1.3 实习：常用算法实现及分析

例如，在下列三个程序段中：

(a) x=x+1;

(b) for(i=1;i<=n;i++) x=x+1;

(c) for(j=1;j<=n;j++)
　　for(k=1;k<=n;k++) x=x+1;

语句 x=x+1 的频度分别为 1、n 和 n^2，则(a)、(b)、(c)的时间复杂度分别是 O(1)、O(n)、$O(n^2)$。由此可见，随着问题规模的增大，其时间消耗也在增大。

下面以(c)程序段为例，进行时间复杂度的分析。

步骤 1：先把所有语句改为基本操作。

```
    j=1;                    1
a: if j<=n                  n+1
   {k=1;                    n*1
    b: if k<=n              n*(n+1)
       {x=x+1;              n*n
        k++;                n*n
        goto b;
       }
     j++;                   n*1
    goto a;
   }
```

步骤 2：分析每一条语句的语句频度，标到每条语句后边，如上。

步骤 3：统计总的语句频度：$1+n+1+n+n(n+1)+n^2+n^2+n=3n^2+4n+2$。

步骤 4：判断最大语句频度为 n^2，所以时间复杂度为 $O(n^2)$。其中 O 表示等价无穷小。

现在来分析例 1.1 中算法 1.1 的时间复杂度。算法中有一个二重循环，if 语句的执行频度为

$$n+(n-1)+(n-2)+\cdots+3+2+1=\frac{n(n+1)}{2}$$

数量级为 $O(n^2)$。算法中输出语句的频度为 n，数量级为 O(n)。该算法的时间复杂度以 if 语句的执行频度来估算(忽略输出部分)，则记为 $O(n^2)$。算法还可能呈现的时间复杂度有指数阶 O(lbn)等。

习　题　1

1. 简述下列术语：数据元素，数据，数据对象，数据结构，存储结构。

2. 试写一算法，自大至小依次输出顺序读入的 3 个整数 x、y 和 z 的值。分析算法的元素比较次数和元素移动次数。

3. 举出一个数据结构的例子，叙述其逻辑结构、存储结构、运算等三方面的内容。

4. 叙述算法的定义及其重要特性。

5. 分析并写出下面的各语句组所代表的算法的时间复杂度。

(1)
```
{for(i=1;i<=n;i++)
   for(j=1;j<=i;j++)
     for(k=1;k<=j;k++)
       {s=i+k;printf("%d",s);}
}
```

(2)
```
{i=1;k=0;
  while(i<=n-1)
  {k=k+10*i;
    i++;
  }
}
```

(3)
```
{i=1;k=0;n=100;
  do{k=k+10*i;
      i++;
    }while(i= =n);
}
```

(4)
```
{i=1;j=0;
  while(i+j<=n)
  {if(i>j) j++;
```

```
        else i++;
      }
    }
(5) {x=n;              /*n>1*/
     y=0;
     while(x>=(y+1)*(y+1)) y++;
    }
(6) {m=91;n=100;
     while(n>0)
     {if(m>0){m=m-1;n--;}
      else m++;
     }
    }
```