高职高专“十五”规划教材系列

数据结构

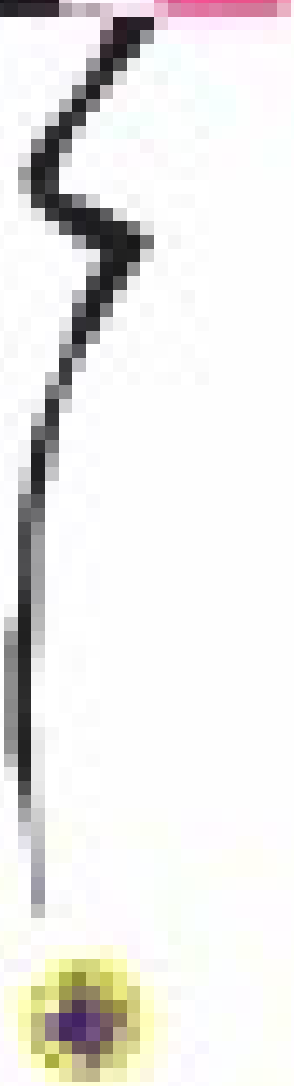
赵国玲　任文娟　编著

机械工业出版社
CHINA MACHINE PRESS

[redacted]

[redacted]

[redacted]



[redacted]

高职高专“十五”规划教材系列

数据结构

赵国玲　任文娟　编著

机械工业出版社

本书是按照教育部关于“高职高专计算机教育教学基本要求”，结合高职高专的教学特点而编写的。全书共 8 章，内容包括：数据结构在程序设计中的作用，线性表、栈、队列、数组、树和图的逻辑结构、存储结构及各种运算的实现方法，常用查找和排序算法的实现及应用等。

本书从实用的角度，对数据结构的内容进行了提炼。为提高学生的程序设计能力，培养学生的算法分析和设计能力，本书由浅入深地对每个算法都给出了完整的 C 语言函数，有些较复杂的算法还给出了相关的应用实例。每章后都配有大量的习题和实训要求及内容。

本书可作为高职高专技术学院计算机应用及相关专业的教材，也可作为各种培训班的教材和计算机爱好者的自学参考书。

图书在版编目（CIP）数据

数据结构 / 赵国玲，任文娟编著. —北京：机械工业出版社，2005.1
（高职高专“十五”规划教材系列）
ISBN 7-111-15384-7

Ⅰ. 数… Ⅱ. ①赵…②任… Ⅲ. 数据结构—高等学校：技术学校—教材
Ⅳ. TP311.12

中国版本图书馆 CIP 数据核字（2004）第 103635 号

机械工业出版社（北京市百万庄大街 22 号　邮政编码 100037）
策　　划：胡毓坚
责任编辑：李利健
责任印制：李　妍

北京蓝海印刷有限公司印刷 · 新华书店北京发行所发行

2005 年 1 月第 1 版 · 第 1 次印刷
787mm×1092mm　1/16 · 11 印张 · 271 千字
0001－5000 册
定价：16.00 元

出 版 说 明

为了贯彻国务院发［2002］16 号文件《国务院关于大力推进职业教育改革与发展的决定》的精神，进一步落实《中华人民共和国职业教育法》和《中华人民共和国劳动法》，实施科教兴国战略，大力推进高等职业教育改革与发展，我们组织力量，对实现高等职业教育培养目标和保证基本教学规格的文化基础课程、专业技术基础课程和重点建设专业主干课程的教材进行了规划和编写。

本套教材内容涵盖了普通大专院校计算机及非计算机专业的文化基础课、专业基础课、专业课以及选修课程，主要分为文化基础、编程语言、硬件技术、网络信息、数据库应用及多媒体技术等几大类。为配合高职教育关于“培养 21 世纪与我国现代化建设要求相适应的一线科技实用性人才”的最新理念，我们特为本系列教材配备了实践指导丛书，以利于老师的教学和学生的学习。

本套教材将理论教学和实践教学紧密结合，图文并茂、内容实用、层次分明、讲解清晰，其中融入了作者长期的教学经验和丰富的实践经验，是各类大专院校、职业技术学校的最佳教材，也可作为各类培训班的教材。

前　言

数据结构是计算机应用及相关专业的骨干课程，主要研究数据的各种逻辑结构和存储结构，以及对数据的各种操作。它对合理、规范地进行程序设计，提高程序的运行效率起着重要的作用。因此，数据结构是程序设计人员的必修课程。

高职高专教育的培养目标是："在教育方针的指导下，培养拥护党的基本路线，适应生产、建设、管理、服务第一线需要的，德、智、体、美等全面发展的高等技术应用性人才。"针对这一培养目标，本书对数据结构的内容进行了提炼，理论上以"够用"为标准，摒弃了一些高深的难以理解的基本理论。每种数据结构的引入不是从抽象的定义出发，而是从实例出发，通过对实例的分析、理解，总结出数据结构的特点及应用。

学习数据结构的最终目的是为了提高程序设计的合理性、规范性，提高程序的运行效率。所以本教材注重引导学生理解数据结构在程序设计中的作用，提高学习的兴趣。同时注意培养学生对算法的分析、理解能力，提高算法和程序的设计能力。本教材注重启发学生解决问题的思路和方法，每一个算法总是先给出设计思想，然后再给出实现该算法的 C 语言函数。所以，要求在学习本课程之前，学生已学习过 C 语言。

本教材内容实用、精练，问题描述深入浅出，概念阐述准确，易于理解。

全书共分 8 章。

第1章 数据结构与程序设计。介绍了数据结构在程序设计中的作用，数据结构的基本概念，讨论了算法及算法的描述和算法效率的分析。

第2章 线性表。介绍了线性表的逻辑和各种存储结构，以及不同存储结构下的各种运算的实现，通过线性表的应用加深了对线性表的理解，提高学生对知识的运用能力。

第3章 数组与字符串。介绍了数组的定义及存储，矩阵压缩存储，字符串的存储及运算。

第 4 章 栈和队列。介绍了限定性线性表栈和队列的特点、存储结构及应用。

第 5 章 查找技术。主要介绍了线性表查找的各种方法及其实现，哈希表的构造及查找。

第 6 章 排序技术。分类介绍了各种常用排序方法。

第 7 章 树形结构。介绍了树与二叉树的基本概念，重点介绍了二叉树的特点、存储结构，二叉树的遍历，线索树，哈夫曼树，二叉树在查找和排序中的应用，即二叉查找树和堆排序。

第8 章 图结构。介绍了图的基本概念及存储结构，图的遍历，拓扑排序和最短路径，最小生成树及关键路径。

本书的第 1~3 章和第 7、8 章由赵国玲编写，第 4～6 章由任文娟编写，并由赵国玲完成全书的统稿工作。本书在编写过程中得到了山东电子职业技术学院各级领导及广大教师的大力支持与帮助，在此致以衷心的感谢。读者可在机械工业出版社网站（http://www.cmpbook.com）下载本书的电子教案。

由于编者水平有限，难免存在疏漏和不当之处，敬请广大读者批评指正。

编　者

目　录

第1章　数据结构与程序

本章要点

- 数据结构在程序设计中的作用
- 数据结构基本概念
- 算法及其描述

数据结构（Data Structures）是一门随计算机科学的发展而逐渐形成的学科，它是一门研究非数值计算的程序设计问题中计算机的操作对象以及它们之间的关系和运算的学科。目前数据结构已成为计算机各类专业的专业基础课、核心课之一。

本章首先介绍数据结构和算法的一些基本概念，为以后各种数据结构的学习打下一个扎实的基础。

1.1　数据结构在程序设计中的作用

众所周知，计算机所完成的任何操作都是在程序的控制下进行的，而程序的根本任务就是进行数据处理。随着计算机应用的日益深入，计算机所处理的数据对象也由纯粹的数值型发展到字符、表格、图形、图像和声音等多种形式，如果要使计算机能够对这些数据进行处理，首先要将这些数据存储在计算机内存中。如何将程序中要处理的数据进行合理地存储，以及采取何种方法能够高效地进行数据处理是程序设计的关键，也是数据结构要解决的重要问题。

即便是在广泛采用可视化程序设计的今天，借助于集成开发环境可以很快地生成程序，但要想成为一个专业的程序开发人员，至少需要以下三个条件：

1）能够熟练地选择和设计各种数据结构和算法。

2）至少要能够熟练地掌握一门程序设计语言。

3）熟知所涉及的相关应用领域的知识。

其中，后两个条件比较容易实现，而第一个条件则需要花很多时间和精力才能够达到，它是区分一个程序设计人员水平高低的一个重要标志，数据结构贯穿程序设计的始终，缺乏数据结构和算法的深厚功底，很难设计出高水平的具有专业水准的应用程序。瑞士著名的计算机科学家沃思（N・Wirth）提出了：算法 + 数据结构 = 程序。这正说明了数据结构和算法的重要性。

例如，现有一批杂乱无章的数据，要对其进行有序化处理，应该怎样解决呢？

要解决此类问题，首先要考虑应如何将这批数据进行合理地存储；然后考虑应采用什么有效的方法对这些数据进行有序化；最后选择一种编程语言实现以上已确定的方法。其实，用计算机解决任何数据处理问题，都需要经过以上过程才能实现。

1.2 数据结构概述

数据结构主要研究和讨论以下三方面问题：

1）数据集合中各数据元素之间的固有关系，即数据和逻辑结构。

2）在对数据进行处理时，各数据元素在计算机中的存储关系，即数据的存储结构。

3）针对数据的存储结构所进行的运算。

解决好以上问题后，可以大大提高数据处理的效率。

1.2.1 数据结构基本概念

1. 数据

数据（Data）是能输入到计算机中并能被计算机处理的一切对象。它是对客观事物的符号表示。这里必须强调指出，所谓数据，绝不能仅理解为整数或实数这种狭义的“数”，必须作广义的理解。例如，一个用某种程序语言编写的程序、一篇文章、一张地图，以及图画、声音等都可视为“数据”。今后随着计算机的发展，还会不断扩充数据的范围。

2. 数据元素

数据元素（Data Element）是数据处理的基本单位，在计算机程序中要作为一个整体进行考虑和处理。由于数据范围非常广泛，因此基本单位可大可小，小到可以是一个字符，大到可以是一个国家的地图或一本书等。有时，一个数据元素可由若干个“数据项”（Data Item）组成，这种由多个数据项组成的数据元素，通常又称为记录（Record）。如一本书的目录卡片作为一个数据元素，而书目信息中的每一项如：书名、作者名、出版社名、出版日期等为数据项。总之，在数据处理中，每一个需要处理的对象可以抽象成相应的数据元素，然后加以存储。

3. 数据结构

数据结构（Data Structure）是彼此具有一定关系的数据元素的集合。事实上，这些关系反映了客观世界事物之间的联系。由于客观事物之间存在着各种不同的联系形式，所以反映在数据关系上也就各不相同。

例如，描述一年四季的季节名：春、夏、秋、冬是季节这一数据的四个数据元素，它们之间存在着必然的次序关系，即：春→夏→秋→冬，这样一种先后次序。又如，作为家庭成员的父亲、儿子或女儿这些数据元素之间，则存在着辈份关系。所以，在进行数据存储时，不仅要存储各数据元素的数据信息，还要反映出数据元素之间的关系。从这一角度考虑，数据结构又包括数据逻辑结构和数据物理结构。

逻辑结构是指数据元素之间所固有的相互关系，如上面所列举的季节、家庭成员之间的关系。物理结构又称存储结构，是数据结构在计算机中的表示（又称映像），它不仅存储了数据元素的数据信息，还存储了数据元素之间的关系信息。同一种逻辑结构可以有几种不同的物理结构来实现它。在程序设计中，研究物理结构更为重要，因为对于同一问题，数据的存储结构不同，解决问题的方法就有所不同。

至此，我们可以对数据结构的作用简单概括如下：

数据结构研究数据的逻辑结构和物理结构，并在这种结构上定义相关的运算，设计实现

这些运算的算法，分析算法的效率。

1.2.2 数据结构分类

根据数据结构中各数据元素之间的关系，常用的有以下三种基本数据结构：

1. 线性结构

线性结构中的数据元素之间存在一对一的前后次序关系。比如前面所列举的一年四季中的春、夏、秋、冬。

2. 树形结构

树形结构中的数据元素之间存在一对多的关系。如上级单位与多个下级单位的关系、家庭成员中的父亲与子女之间的关系等，就需要用所谓的“树结构”来表示。

3. 图状或网状结构

这些结构中的数据元素之间存在多对多的关系。如各城市之间的交通网络中各站点之间的关系、计算机网络中各结点之间的关系、一个大的工程项目的施工进度等，都要用复杂的“图结构”来表示。

以上三种基本数据结构又分为线性结构和非线性结构两大类，其中树和图两种结构合称为非线性数据结构。

图 1-1 为上述各结构的关系图，其中一个“○”表示一个数据元素。

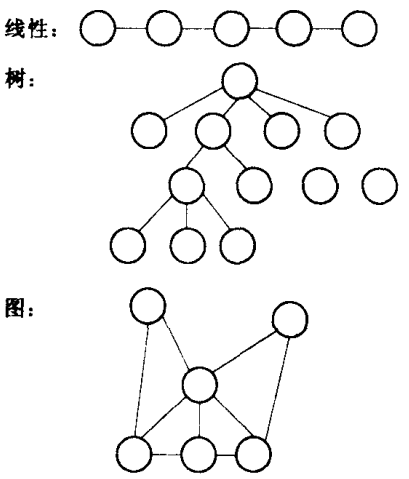


图 1-1 基本数据结构关系图

1.3 算法及其描述

算法（Algorithm）是程序设计的精髓，程序设计的实质就是构造解决问题的算法，将其解释为计算机语言。算法的设计取决于数据的逻辑结构，算法的实现取决于数据的物理存储结构。

1.3.1 什么是算法

算法是对解决某个特定问题而采取的方法和步骤的准确而完整的描述。做任何事情都必须事先想好进行的步骤，然后按部就班地进行，才不会发生错误，计算机也是如此。对于一些常用算法应熟记，在解决实际问题时，可参考已有的类似算法，设计出符合自己需要的算法。

【例 1-1】 求某一正整数 n 的阶乘，即计算 1×2×3×4×…×n。

按照传统计算方法，计算步骤应该是：

第一步：计算 1×2 得到结果 2。

第二步：将第一步得到的乘积 2 再乘以 3，得到结果 6。

第三步：将 6 再乘以 4，得 24。

依照上面的步骤重复下去，每次计算时用上一步得到的结果作被乘数，乘数增 1，直到乘数大于 n 为止。

用计算机解决此问题时，其基本思路与人工计算基本一致。具体实现时，要设两个变量，如 p 和 j，用变量 p 代表被乘数，用变量 j 代表乘数，计算步骤可描述如下：

S1：1⇒p

S2：2⇒j

S3：如果 j≤n，执行以下操作，否则转到 S4

S3.1：p×j⇒p

S3.2：j+1⇒j

S3.3：转到 S3

S4：输出 p 的值

S5：结束

一个算法应该具有以下五个重要的特征：

1．有穷性

一个算法应包含有限个操作步骤。即一个算法在执行若干个操作步骤之后应该能够结束，而且每一步都在合理的时间内完成。

2．确定性

算法中的每一步都必须有确切的含义，不能是含糊的、模棱两可的。

3．可行性

算法中的每一个步骤都应该是能够有效地执行，并得到确定结果的操作。只要算法中有一个操作是不能执行的，整个算法就不具有可行性。

4．输入

所谓输入，是指在算法执行时，从外界取得必要的数据（一般是通过键盘输入）。计算机运行程序的目的是为了进行数据处理，在大多数情况下，这些数据需要通过输入得到。有些情况下，数据已经包含在算法中，算法执行时不需要输入任何数据，这样，算法中也可以没有输入。所以一个算法中可以有零个或多个输入。

5．输出

一个算法有一个或多个输出，以反映对输入数据处理后的结果。没有输出的算法是毫无意义的。

算法的这些特性可以约束程序设计人员正确地书写算法，并使之能够正确无误地执行，达到求解问题的预期效果。

1.3.2 算法的描述

为了描述一个算法，可以用多种不同的表示方法。例如有自然语言表示法、流程图、N-S 图、伪代码、PAD 图等。

自然语言表示法就是用自然语言描述算法，在以上例题所介绍的算法中，均是用自然语言表示的。这种表示方法虽然通俗易懂，但文字冗长，且易产生歧义性。对于大型的复杂算法，用自然语言很难清楚、准确地描述出来。下面主要介绍两种常用的算法描述工具——流程图和 N-S 图。

1．用流程图表示算法

流程图是算法的图形描述工具，它用一些几何图形表示各种类型的操作。美国国家标准

化协会 ANSI 规定了一些常用的流程图符号，如图 1-2 所示，这些符号已为世界各国程序设计人员普遍采用。

图 1-2a 表示程序的开始或结束；图 1-2b 表示输入或输出操作；图 1-2c 表示条件判断，它只有一个入口，但有两个出口供选择；图 1-2d 表示处理功能，框内可注明简要功能；图 1-2e 表示子程序、子模块等；图 1-2f 中的直线表示控制流的流线，箭头指示流程方向；图 1-2g 为流程图间断处使用的连接符号，在转入和转出的圆框内应填入相同的文字、数字或名称。

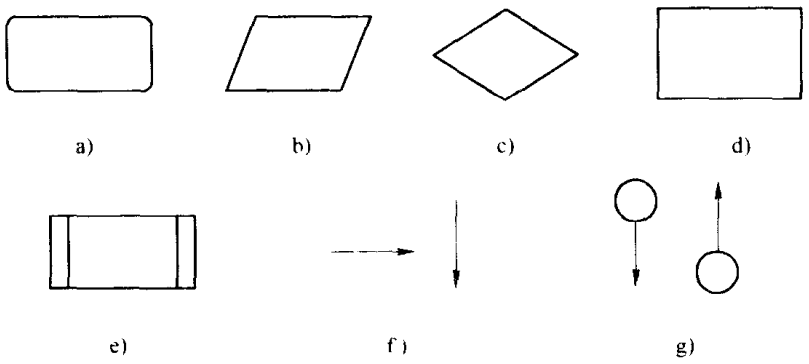


图 1-2　流程图符号

a) 起始/终止框　b) 输入/输出框　c) 判断框　d) 处理框　e) 子程序框　f) 流程方向符号　g) 连接符号

流程图独立于任何一种程序设计语言，具有直观、清晰、易于学习掌握的特点，至今仍是程序设计者普遍使用的一种算法描述工具。

2. 用 N-S 图表示算法

1973 年，美国学者 Nassi 和 Shneiderman 提出了一种符合结构化程序设计原则的图形描述工具——N-S 图。它完全去掉了流程图中会引起麻烦的流程线，全部算法写在一个矩形框内，在该框内还可以包含其他的从属于它的框，所以也称为盒图。

N-S 图中规定了五种图形构件，如图 1-3 所示。

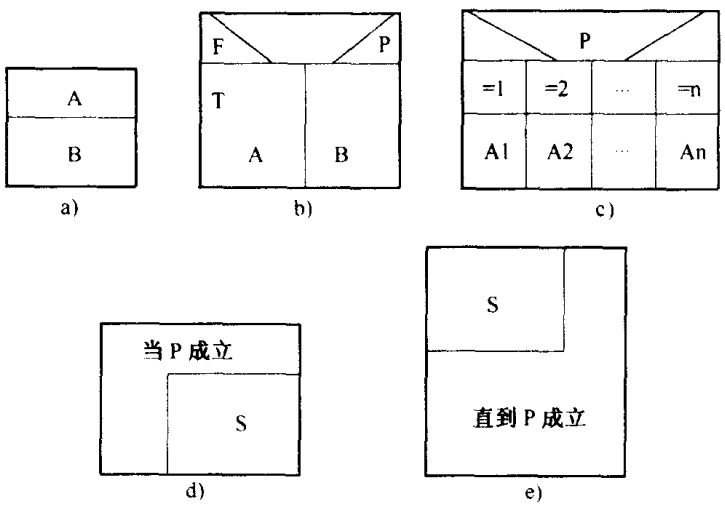


图 1-3　N-S 图基本图形构件

a) 顺序型　b) 选择型　c) 多分支选择型　d) 当型重复　e) 直到型重复

图 1-3a 为顺序型，表示按顺序先执行 A 操作，再执行 B 操作；

b 为选择型，表示若条件 P 成立，执行 A 框的操作，否则执行框 B 的操作；

c 为多分支选择型，为多出口判断的图形表示，P 为控制条件，根据 P 的取值，相应地执行其值下面各框的操作。

d 为当型重复，e 为直到型重复，表示两种类型的循环，P 为循环条件，S 为重复执行的操作。d 是先判断 P 的取值，再执行 S；e 是先执行 S，再判断 P 的取值。

图 1-4 和图 1-5 分别给出了【例 1-1】的流程图和 N-S 图。

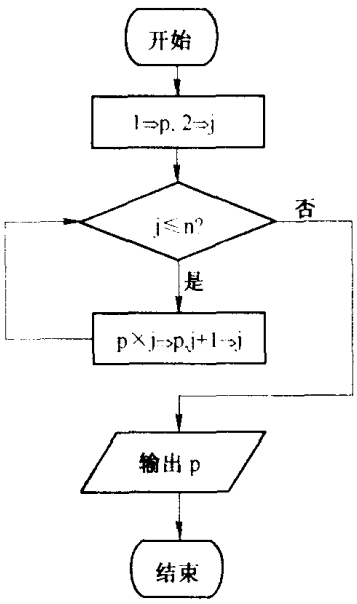


图 1-4　求 n 的阶乘的流程图

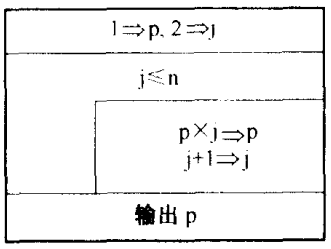


图 1-5　求 n 的阶乘的 N-S 图

3. 用计算机语言描述算法

用流程图或 N-S 图描述算法，是为了满足人们阅读和交流算法的需要，一个真正能上机运行的算法，必须是严格按照语法规则采用某种编程语言编写的。在本书中，为了便于大家对算法的理解和调用，所有算法都是严格按照 C 语言的语法规则编写的。对于每个算法，都给出了实现该算法的完整的 C 语言函数，大家在实际上机运行时，只要配上相应的主函数进行调用就行了。所有算法函数都可以在 Turbo C 环境下调用运行。

如，对于【例 1-1】中求 n!的问题，给出的 C 语言函数如下：

```
long fac(n)
int n;
{int j;
 long p=1;
 for (j=2;j<=n;j++)
    p=p*j;
 return p;
}
```

读者在上机运行该算法时，只要配上如下类似的主函数，就构成了一个完整的 C 程序，完成任意正整数阶乘的运算。

```
main()
{int n;
 long m;
 scanf ("%d",&n);
 m=fac(n);
 printf("\n%d!= %ld\n",n,m);
}
```

1.3.3 算法的复杂度

解决一个问题可以有多种算法，那么该怎样判断它们的优劣呢？判断算法优劣的标准很多，本书不作深入讨论。这里主要想讨论的是：一个算法除了正确性必须保证以外（一个有错误的算法根本不能称为算法），一个主要指标是它的效率。

算法的效率包括时间与空间两个方面，分别称为时间复杂度与空间复杂度。所谓时间复杂度，是指一个算法执行时所需运算次数的多少；空间复杂度是指一个算法执行时所需的辅助内存空间的大小。这两种度量都不表示为绝对的量，而是用量级的概念来表示，并且在对一个算法进行分析时主要考虑其时间效率。这种量级通常用大写字母“O”来表示。如：O(1)、O(n)、$O(n^2)$、$O(2^n)$、O(log n) 分别表示常量阶、线性阶、平方阶、指数阶、对数阶。显然对时间复杂度为 O(1)的算法执行时间最省，对于指数阶的算法，当 n 较大时，其增长率达到惊人的程度。因此我们应避免使用那些复杂度达到指数阶的算法。

例如，有一算法的程序如下：

```
for (m=0 ; m<= n ; m++)
  for (j=0; j<=n; j++)
    {r[m][n]=0;
     for (k=0 ; k<=n; k++)
       r[m][j]=r[m][j]+a[m][k]*b[k][j] ;
     }
```

根据程序的执行，该算法的时间复杂度量级为 O（n^3）。

1.4 小结

本章主要介绍了数据结构的基本概念、作用及算法的描述和分析。通过本章的学习，首先要对数据结构在程序设计中的重要作用有一个充分的认识，并需掌握数据结构的基本概念，了解数据结构的分类及算法的设计、描述方法。

为了更好地完成后续课程的学习，请在进入下一章学习之前，充分复习 C 语言的相关知识，特别是函数、数组及指针部分。

1.5 实训

1．熟悉 Turbo C 环境。