全国高等院校计算机基础教育研究会

“2016年度计算机基础教学改革课题”立项项目

算法与数据结构

（C语言版）

主编◎邓玉洁

北京邮电大学出版社
www.buptpress.com

全国高等院校计算机基础教育研究会
“2016年度计算机基础教学改革课题”立项项目

算法与数据结构

（C语言版）

主编◎邓玉洁

北京邮电大学出版社
www.buptpress.com

内 容 简 介

本书主要内容包括绪论、线性表、栈和队列、串、数组、树形结构、图、内部排序、查找。教材中对各类数据结构的分析按照“逻辑结构—存储结构—基本运算的实现—时空性分析—实例”的顺序进行讲述，结构规范，条理清晰。

书中给出的程序和算法都是经过仔细筛选的经典内容，便于读者理解和掌握，程序采用C语言描述并容易调试通过；每章有重点介绍和总结，总结对重要的知识点进行穿线，每章后针对本章重要知识点配有大量习题。

本书可作为高等院校计算机有关专业本科生、专科生教材，也可作为自考成人教育的教材。

图书在版编目（CIP）数据

算法与数据结构：C语言版 / 邓玉洁主编. -- 北京：北京邮电大学出版社，2017.8

ISBN 978-7-5635-5253-5

Ⅰ.①算… Ⅱ.①邓… Ⅲ.①算法分析－高等学校－教材②数据结构－高等学校－教材③C语言－程序设计－高等学校－教材 Ⅳ.①TP301.6②TP311.12③TP312.8

中国版本图书馆CIP数据核字（2017）第197095号

书　　名：算法与数据结构(C语言版)
著作责任者：邓玉洁　主编
责任编辑：刘春棠
出版发行：北京邮电大学出版社
社　　址：北京市海淀区西土城路10号(邮编:100876)
发 行 部：电话：010-62282185　传真：010-62283578
E-mail：publish@bupt.edu.cn
经　　销：各地新华书店
印　　刷：北京鑫丰华彩印有限公司
开　　本：787 mm×1 092 mm　1/16
印　　张：14.5
字　　数：355千字
版　　次：2017年8月第1版　2017年8月第1次印刷

ISBN 978-7-5635-5253-5　　定　价：29.00元

前　言

本书内容取材以数据结构为主线，算法为辅线。主要内容包括学习数据结构的基础知识、线性表、栈和队列、串、数组、树形结构、图、内部排序、查找。采取3+2+3形式展开，即3种数据结构、2种存储结构、3种基本算法加上查找和排序基本算法。

教材书写特点是每一个主题都从一个基本的概念出发，然后再逐渐深入讨论，这样做能使解释更清晰，富有启发性。学习内容由点到面、由浅入深逐渐展开，使教师在教学和学生学习时有规律可循。每一章节既相互关联又相互独立。先介绍逻辑结构，再介绍物理结构，然后讲解算法，给出具体实例。

教材侧重应用性，教学内容与应用实例有机融合，符合应用型本科特点；教材中每种数据结构都给出贴近生活的例子，使学生更好地理解和掌握数据结构，达到举一反三的目的。本书可作为高等院校计算机有关专业本科生、专科生教材，也可作为自考成人教育的教材。

本书第1章、第7章由邓玉洁编写，第2章由杨丽华编写，第3章由李芳编写，第4章、第5章由陈沛强编写，第6章由段爱玲编写，第8章由于桂玲编写，第9章由吴海燕编写，全书由邓玉洁统稿。其他参编人员还有郑凯梅、祝凯、徐艺枢、李艾静，还要感谢我的同事和我的学生们在撰写过程中提出的宝贵意见。

由于编者水平有限，不足之处欢迎各位读者指正。

目　录

第 1 章　数据结构与算法

学习目标

本章将通过对数据结构相关知识的简要介绍，描述数据结构的基本内容和主要概念，作为对本门课程内容的梗概之序。

知识要点

(1) 数据结构是什么。
(2) 数据结构及相关概念。
(3) 算法的基本概念及特性。
(4) 算法的时间复杂度求解。

1.1　学习数据结构

1.1.1　为什么学习数据结构

早期人们把计算机理解为数值计算工具，认为计算机就是用来计算的。可现实中，我们更多的不是解决数值计算问题，而是需要一些更科学有效的手段(如表、树、图等数据结构)的帮助，才能更好地处理问题。所以数据结构是一门研究非数值计算的程序设计问题中的操作对象，以及它们之间关系和操作等相关问题的学科。

在数据处理领域中，建立数学模型有时并不十分重要，事实上，许多实际问题是无法表示成数学模型的。人们最感兴趣的是数据集合中各数据元素之间存在什么关系，应如何组织它。简单地说，数据结构是指相互有关联的数据元素的集合。例如，向量和矩阵就是数据结构，在这两种数据结构中，数据元素之间有着位置上的关系。又如，图书馆中的图书卡片目录就是一个较为复杂的数据结构，对于列在卡片上的各种书，可能在主题、作者等问题上相互关联，甚至一本书本身也有不同的相关成分。数据元素具有广泛的含义。一般来说，现实世界中客观存在的一切个体都可以是数据元素。

数据结构并不是要教你怎样编程，编程语言的精练也不在数据结构的管辖范围之内，数据结构是教你如何在现有程序的基础上把它变得更优(运算更快，占用资源更少)，它改变的是程序的存储运算结构而不是程序语言本身。

如果把程序看成一辆汽车，那么程序语言就构成了这辆车的车身和轮胎，而算法则是这辆车的核心——发动机。这辆车跑得是快是慢，关键就在于发动机的好坏(当然轮胎太烂了

也不行），而数据结构就是用来改造发动机的。可以这么说，数据结构并不是一门语言，它是一种思想、一种方法、一种思维方式。

再举一个简单的例子，我们要写文章的时候，一定要先构思文章的结构，然后才能下笔去写，写的时候可能用英文，也可能用法文或中文，在这里文章构思的结构就相当于数据结构，而用的语言就是实现数据结构算法的编程语言。可以这样理解，数据结构是思想，实现数据结构算法的语言（如 C 语言）是工具。工具一定是依赖思想存在的，可见数据结构的重要性。

数据结构就是教你怎样用最精简的语言，利用最少的资源（包括时间和空间）编写出最优秀、最合理的程序。

1968 年，美国的高德纳（Donald E. Knuth）教授在其所写的《计算机程序设计艺术》第一卷《基本算法》中，较系统地阐述了数据的逻辑结构和存储结构及其操作，开创了数据结构的课程体系。同年，数据结构作为一门独立的课程，在计算机科学的学位课程中出现。

20 世纪 70 年代初，出现了大型程序，软件也开始相对独立，结构程序设计成为程序设计方法学的主要内容，人们越来越重视数据结构，认为程序设计必须是好的数据结构加上好的算法。

1.1.2 如何学习数据结构

1. “数据结构”课程的地位

“数据结构”课程较系统地介绍了软件设计中常用数据结构以及相应的存储结构和算法，系统介绍了常用的查找和排序技术，并对各种结构与技术进行分析和比较，内容非常丰富。“数据结构”涉及多方面的知识，如计算机硬件范围的存储装置和存取方法，在软件范围中的文件系统、数据的动态管理、信息检索，数学范围的集合、逻辑的知识，还有一些综合性的知识，如数据类型、程序设计方法、数据表示、数据运算、数据存取等，是计算机专业一门重要的专业技术基础课程。

“数据结构”的内容将为“操作系统”“数据库原理”“编译原理”等后续课程的学习打下良好的基础（如图 1-1 所示），“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，同时也训练高效地解决复杂问题程序设计的能力，因此“数据结构”是数学、计算机硬件、计算机软件三者共同的一门核心课程，“数据结构”课程是计算机专业提高软件设计水平的一门关键性课程。

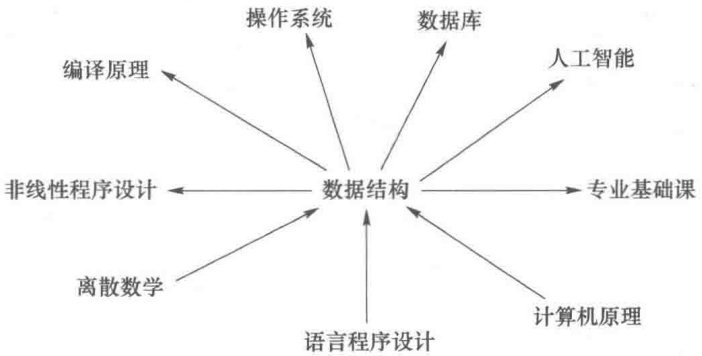

图 1-1 “数据结构”与其他课程的关系

2. “数据结构”课程的学习特点

“数据结构”课程教学目标要求学生学会分析数据对象特征，掌握数据组织方法和计算机的表示方法，以便为应用所涉及数据选择适当的逻辑结构、存储结构及相应算法，初步掌握算法时间空间分析的技巧，培养良好的程序设计技能。

“数据结构”的学习过程是进行复杂程序设计训练的过程。技能培养的重要程度不亚于知识传授。数据结构从某种意义上说，是程序设计的后继课程。如同学习英语一样，学习英语不难，学好英语不易，要提高程序设计水平必须经过艰苦的磨炼。因此，学习数据结构，仅从书本上学习是不够的，必须经过大量的实践，在实践中体会构造性思维方法，掌握数据组织与程序设计的技术。

3. “数据结构”课程的学习方法

在学习数据结构的时候，采用“323”模式来学习，脉络会非常清楚。所谓“323”指的是3种数据结构、2种存储方法、3种重要算法。3种数据结构即线性结构、树结构、图结构；2种存储方法即顺序存储和链式存储；3种重要算法即查找、插入、删除，如图1-2所示。

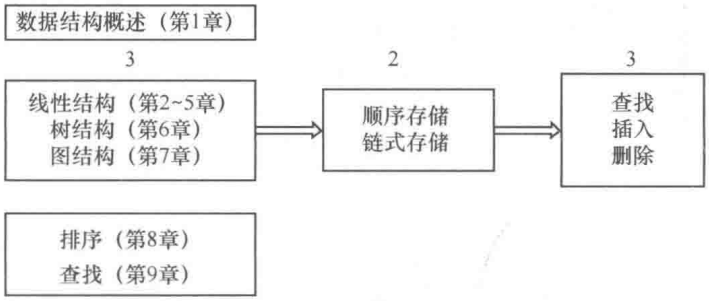

图1-2　数据结构模式

我们可以参看书中目录，脉络就会非常清晰。第1章即数据结构基本概念的介绍，第2～5章线性结构、第6章树、第7章图是重点章节，第8章和第9章排序和查找是重要的数据运算技术，也是非常重要的。

1.2　基本概念和术语

1.2.1　基本概念

1. 数据

数据是描述客观事物的数值、字符以及能输入机器且能被处理的各种符号的集合。换句话说，数据是对客观事物采用计算机能够识别、存储和处理的形式所进行的描述。简言之，数据就是计算机化的信息。数据不仅包括整型、实型等数值类型，还包括字符、声音、图像、视频等非数值类型。

例如，对C源程序，数据不仅仅是源程序所处理的数据。相对于编译程序来说，C编译程序相对于源程序是一个处理程序，它加工的数据是字符流的源程序(.c)，输出的结果是目标程序(.obj)；对于链接程序来说，它加工的数据是目标程序(.obj)，输出的结果是可执行

程序(.exe)。

2. 数据元素

数据元素是组成数据的、有一定意义的基本单位，是数据集合的个体，在计算机中通常作为一个整体进行考虑和处理。一组数据元素可由一个或多个数据项(data item)组成，数据项是有独立含义的最小单位，此时的数据元素通常称为记录(record)。

例如，学生登记表是数据，每一个学生的记录就是一个数据元素。

3. 数据项

一个数据元素可由若干个数据项组成。比如学生登记表中的学号、姓名、出生日期都是数据元素的数据项。

数据项是数据不可分割的最小单位。在“数据结构”课程中，数据项是最小单位，有助于我们更好地解决问题，但真正讨论问题时，数据元素才是数据结构中建立数据模型的着眼点。

4. 数据对象

数据对象是性质相同的数据元素的集合，是数据的一个子集。例如，整数数据对象是集合 $N=\{0,\pm1,\pm2,\cdots\}$，字母字符数据对象是集合 C={'A','B',…,'Z'}。不论数据元素集合是无限集(如整数集)、有限集(如字符集)，还是由多个数据项组成的复合数据元(如学籍表)，只要性质相同，都是同一个数据对象。

5. 数据类型

数据类型是一组性质相同的值集合以及定义在这个值集合上的一组操作的总称。数据类型中定义了两个集合，即该类型的取值范围，该类型中可允许使用的一组运算。例如，高级语言中的数据类型就是已经实现的数据结构的实例。从这个意义上讲，数据类型是高级语言中允许的变量种类，是程序语言中已经实现的数据结构(即程序中允许出现的数据形式)在高级语言中的整型类型，则它可能的取值范围是－32 768～＋32 767，可用的运算符集合为加、减、乘、除、乘方、取模(如 C 语言中＋、－、*、/、%)。

6. 数据结构

数据结构是指相互之间存在一种或多种特定关系的数据元素集合，是带有结构的数据元素的集合，它指的是数据元素之间的相互关系，即数据的组织形式。由此可见，计算机所处理的数据并不是数据的杂乱堆积，而是具有内在联系的数据集合。我们关心的是数据元素之间的相互关系与组织方式，及其施加运算及运算规则，并不涉及数据元素内容具体是什么值。

数据结构定义中提到了一种或多种特定关系，具体是什么关系呢，这正是我们下面要讨论的问题。

1.2.2 逻辑结构和存储结构

在上一小节中已给出数据结构的概念，数据结构是指相互之间存在一种或多种特定关系的数据元素集合。这个描述是一种非常简单的解释。数据元素间的相互关系具体应包括三个方面：数据的逻辑结构、数据的物理结构、数据的运算集合。

1. 逻辑结构

数据的逻辑结构是指数据元素之间逻辑关系的描述。数据结构的形式定义为：数据结构是一个二元组 Data_Structure=(D,R)，其中 D 是数据元素的有限集，R 是 D 上关系的有限集。

根据数据元素之间关系的不同特性，通常有下列四类基本的结构。

(1) 集合结构

结构中的数据元素之间除了同属于一个集合的关系外，无任何其他关系。各个数据元素是“平等”的，它们的共同属性是“同属于一个集合”，类似于数学中的集合，如图 1-3 所示。

(2) 线性结构

结构中的数据元素之间存在着一对一的线性关系，如图 1-4 所示。

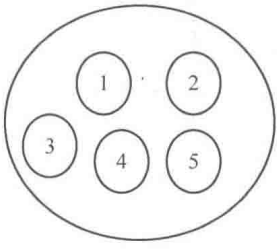

图 1-3　集合结构

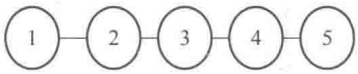

图 1-4　线性结构

(3) 树形结构

结构中的数据元素之间存在着一对多的层次关系，如图 1-5 所示。

(4) 图状结构

结构中的数据元素之间存在着多对多的任意关系，如图 1-6 所示。

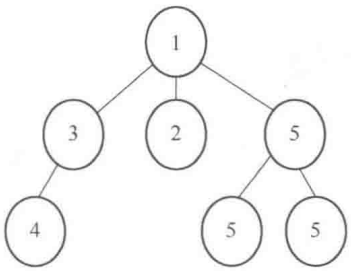

图 1-5　树形结构

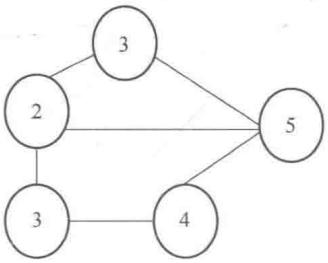

图 1-6　图状结构

如果一个数据结构不是线性结构，可称为非线性结构。显然，在非线性结构中，各数据元素之间的前后继关系要比线性结构复杂，因此，对非线性结构的存储与处理比线性结构要复杂得多。线性结构与非线性结构都可以是空的数据结构。一个空的数据结构究竟是属于线性结构还是属于非线性结构，这要根据具体情况来确定。如果对该数据结构的运算是按线性结构的规则来处理的，则属于线性结构；否则属于非线性结构。

2. 存储结构

存储结构(又称物理结构)是逻辑结构在计算机中的存储映像，是逻辑结构在计算机中的实现，它包括数据元素的表示和关系的表示。

数据的逻辑结构在计算机存储空间中的存放形式称为数据的存储结构（也称数据的物理结构）。由于数据元素在计算机存储空间中的位置关系可能与逻辑关系不同，因此，为了表示存放在计算机存储空间中的各数据元素之间的逻辑关系（即前后继关系），在数据的存储结构中，不仅要存放各数据元素的信息，还需要存放各数据元素之间前后继关系的信息。

逻辑结构与存储结构的关系为：存储结构是逻辑关系的映像与元素本身映像。逻辑结构是抽象，存储结构是实现，两者综合起来建立了数据元素之间的结构关系。

数据元素存储结构形式有两种：顺序存储和链式存储。

（1）顺序存储结构

顺序存储结构是把数据元素存放在地址连续的存储单元里，其数据间的逻辑关系和物理关系是一致的，如图 1-7 所示。

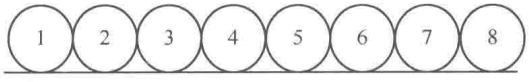

图 1-7　顺序存储结构

顺序存储就排队按顺序站好，每个人占一小段空间。在学习 C 语言时，数组就是这样的顺序存储结构。要建立一个 8 个整型元素的数组，计算机就会在内存中开辟一段连续的能存储 8 个整型数据的空间，数组一个一个存放到空间里。

（2）链式存储结构

链式存储结构是把数据元素放在任意的存储单元里，这组存储单元可以是连续的，也可以是不连续的。数据元素的存储关系并不能反映其逻辑关系，因此需要用一个指针存放数据元素的地址，这样通过地址就可以找相关联的数据元素的位置，如图 1-8 所示。

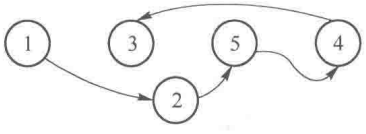

图 1-8　链式存储结构

显然，链式存储就灵活多了，数据存在哪里不重要，只要有一个指针存放了相应的地址就能找到它。

逻辑结构是面向问题的，物理结构是面向计算机的，基本目标就是将数据及其逻辑关系存储到计算机中。存好之后做什么呢，就是运算。

3. 数据的运算

数据的存储不同决定了运算不同。通常，一个数据结构中的元素结点可能是在动态变化的。根据需要或在处理过程中，可以在一个数据结构中增加一个新结点，也可以删除数据结构中的某个结点（称为删除运算）。插入与删除是对数据结构的两种基本运算。除此之外，对数据结构的运算还有查找、分类、合并、分解、复制和修改等。在对数据结构的处理过程中，不仅数据结构中的结点（即数据元素）个数在动态地变化，而且各数据元素之间的关系也有可能在动态地变化。

如果在一个数据结构中一个数据元素都没有，则称该数据结构为空的数据结构。在一个空的数据结构中插入一个新的元素后就变为非空；在只有一个数据元素的数据结构中，将该元素删除后就变为空的数据结构。

1.3　算　　法

数据结构与算法之间存在本质联系，在某一类型数据结构上，总要涉及其上施加的运算，只有通过对定义运算的研究，才能清楚理解数据结构的定义和作用；在涉及运算时，总要联系到该算法处理的对象和结果的数据。

在本门课程中，我们将遇到大量的算法问题，因为算法联系着数据在计算过程中的组织方式，为了描述实现某种操作，常常需要设计算法，因而，算法是研究数据结构的重要途径。

现在我们来写一个小程序，求 1＋2＋3＋…＋100 结果的程序，大多数人马上会写出下面的 C 语言代码：

```
int i,sum = 0,n = 100;
for (i = 1;i <= n;i++)
    {
sum = sum + i;
}
      printf("%d",sum);
```

这是最简单的计算机程序之一，它就是一种算法，问题在于这个算法是不是真的好呢？是不是高效呢？让我们看下面一个例子，高斯求 1＋2＋3＋…＋100 方法：

```
sum = 1 + 2 + 3 + … + 100
sum = 100 + 99 + 98 + … + 1
2 * sum = 101 + 101 + 101 + … + 101
```

共 100 个 101，所以 sum＝1 050，用程序来实现如下：

```
int i,sum = 0,n = 100;
sum = (1 + n) * n/2;
printf("%d",sum);
```

这个方法要比刚刚的方法快得多，不仅可以用于 1 加到 100，就是加到 1 000、10 000，也是瞬间之事。如果用刚才的程序，计算机要循环 1 000 次、10 000 次的加法运算。可见人脑比计算机快得多，似乎成为现实。

1.3.1　算法的定义

算法(algorithm)是一组有穷的规则，它们规定了解决某一特定类型问题的一系列运算，是对解题方案的准确与完整的描述。

算法是解题的步骤，可以把算法定义成解一确定类问题的任意一种特殊的方法。在计算机科学中，算法要用计算机算法语言描述，算法代表用计算机解一类问题的精确、有效的方法。算法＋数据结构＝程序，求解一个给定的可计算或可解的问题，不同的人可以编写出不同的程序，来解决同一个问题，这里存在两个问题：一是与计算方法密切相关的算法问题；二是程序设计的技术问题。算法和程序之间存在密切的关系。

1.3.2 算法的特性

作为一个算法，一般应具有以下几个基本特征。

1. 确定性

算法的每一种运算必须有确定的意义，该种运算应执行何种动作应无二义性，目的明确；这一性质反映了算法与数学公式的明显差别。在解决实际问题时，可能会出现这样的情况：针对某种特殊问题，数学公式是正确的，但按此数学公式设计的计算过程可能会使计算机系统无所适从，这是因为根据数学公式设计的计算过程只考虑了正常使用的情况，而当出现异常情况时，此计算过程就不能适应了。

2. 可行性

要求算法中有待实现的运算都是基本的，每种运算至少在原理上能由人用纸和笔在有限的时间内完成；针对实际问题设计的算法，人们总是希望能够得到满意的结果。但一个算法又总是在某个特定的计算工具上执行的，因此，算法在执行过程中往往要受到计算工具的限制，使执行结果产生偏差。

3. 输入

一个算法有 1 个或多个输入，在算法运算开始之前给出算法所需数据的初值，这些输入取自特定的对象集合。

4. 输出

作为算法运算的结果，一个算法产生一个或多个输出，输出是同输入有某种特定关系的量。

5. 有穷性

一个算法总是在执行了有穷步的运算后终止，即该算法是可达的。数学中的无穷级数，在实际计算时只能取有限项，即计算无穷级数值的过程只能是有穷的。因此，一个数的无穷级数表示只是一个计算公式，而根据精度要求确定的计算过程才是有穷的算法。算法的有穷性还应包括合理的执行时间的含义。因为如果一个算法需要执行千万年，显然失去了实用价值。

满足前四个特性的一组规则不能称为算法，只能称为计算过程，操作系统是计算过程的一个例子。在一个算法中，有些指令可能是重复执行的，因此指令的执行次数可能是远远大于算法中的指令条数。由有穷性可知，对于任何输入，一个算法在执行了有限条指令后一定要终止并且必须在有限的时间内完成，因此，一个程序如果对任何输入都不会陷入无限循环，即是有穷的，则它就是一个算法。

1.3.3 算法效率度量方法

一种数据结构的优劣是由实现其各种运算的算法具体体现的，对数据结构的分析实质上就是对实现运算算法的分析，除了要验证算法是否正确解决该问题外，需要对算法的效率作性能评价。

性能评价分正确性、可读性、健壮性、高效率与低存储量需求几个方面。

在计算机程序设计中，对算法分析是十分重要的。通常对于一个实际问题的解决，可以提出若干个算法，那么如何从这些可行的算法中找出最有效的算法呢？或者有了一个解决