普通高等教育“十三五”规划教材
新工科建设之路·计算机类专业规划教材

数据结构

——C++ 语言描述

陈慧南　编著

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

普通高等教育“十三五”规划教材
新工科建设之路·计算机类专业规划教材

数据结构

——C++语言描述

陈慧南　编著

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

内 容 简 介

作者依据 ACM/IEEE 的《计算机科学课程体系规范 2013》（CS2013），参考了近年来国内外很多优秀教材，对《数据结构——使用 C++语言描述》一书从结构和内容方面都做了很大调整，编写了本教材。

本次编写保留了经典数据结构和算法知识，引入更多高级数据结构的内容。本教材注重问题求解，反映抽象、封装和信息隐蔽等现代软件设计理念，重视算法的时间和空间分析，包括查找和排序问题的时间复杂度下界分析。数据结构和算法使用 C++语言描述。

本教材突出实践性和程序设计。教材中的算法都有完整的 C++程序，构思精巧、结构清晰。所有程序都已在 VC++环境下编译通过并能正确运行。它们既是很好的学习数据结构和算法的示例，也是很好的学习 C++程序设计的示例。本教材配有大量的实例和图示，并有丰富的习题和实习题，易教易学。本教材涵盖 2019 年计算机考研大纲数据结构部分的考查内容。

本教材可作为计算机类、电子信息类及其他科学、工程和经济类等相关专业数据结构课程的教材与考研参考书，也可供使用计算机求解应用问题的工程技术人员参考。

图书在版编目（CIP）数据

数据结构：C++语言描述 / 陈慧南编著. —北京：电子工业出版社，2020.1

ISBN 978-7-121-36632-1

Ⅰ. ①数… Ⅱ. ①陈… Ⅲ. ①C++语言－数据结构－高等学校－教材 Ⅳ. ①TP311.12②TP312.8

中国版本图书馆 CIP 数据核字（2019）第 100391 号

责任编辑：冉 哲
印 刷：涿州市京南印刷厂
装 订：涿州市京南印刷厂
出版发行：电子工业出版社
北京市海淀区万寿路 173 信箱 邮编：100036
开 本：787×1 092 1/16 印张：21 字数：604.8 千字
版 次：2020 年 1 月第 1 版
印 次：2020 年 1 月第 1 次印刷
定 价：59.80 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：ran@phei.com.cn。

前　言

随着计算机技术的飞速发展，以及计算学科与其他学科的日益融合，计算机科学被应用于解决许多学科领域的计算问题，已成为其他学科不可或缺的方法和工具。讲授经典数据结构和算法知识的“数据结构”课程不仅是计算机类专业的必修课，而且正在面向更广泛的学习对象，成为很多科学、工程和经济类专业学生需要学习和选修的课程。

ACM/IEEE 的《计算机科学课程体系规范 2013》（CS2013）指出，计算机类专业的毕业生应该掌握计算机知识领域的基本知识，强调学生对知识的实际运用能力，尤其是综合运用所学知识的能力。问题求解无疑是计算机科学的根本目的。运用计算机科学的基础概念进行问题求解是计算机人才的基本能力之一。学生应该理解反复出现的知识点和通用的软件原则，如抽象和分解、模块化和信息隐蔽、时空转换等，学会在多个抽象层面上思考问题。

数据结构作为计算机科学的一门基础课，不仅讲授经典数据结构和算法，强调在现代编程环境中实现这些算法，还应该让学生学习和理解问题分解、抽象、封装与信息隐蔽、规范与实现分离、递归、算法效率等概念和方法，为学生在数据抽象和问题求解方面打下良好的基础。

作者在教学生涯中，编写出版了多本教材。其中，《数据结构——使用 C++语言描述》、《数据结构——C 语言描述》和《算法设计与分析》以及配套的《数据结构学习指导与习题解析》被列为普通高等教育“十一五”国家级规划教材。本教材是在上述教材编写的基础上，依据 CS2013 并参考了近年来国内外很多优秀教材，对《数据结构——使用 C++语言描述》一书，从结构和内容方面做了很大调整编写而成的。

本次编写秉承作者以往的做法，采用抽象数据类型和面向对象方法讨论数据结构，并使用 C++语言描述。数据结构和算法直接采用程序设计语言描述，体现了计算机学科不仅是理论也是技术的特点。书中算法有完整的 C++程序，构思精巧、结构清晰。所有程序都已在 VC++环境下编译通过并能正确运行。它们既是很好的学习数据结构和算法的示例，也是很好的学习 C++程序设计的示例。

本教材强化了算法分析，对多数算法做了时空效率分析；对超出本教材范围的内容也不加证明地给出了结论；分析了查找和排序问题的时间复杂度下界；某些算法给出了正确性证明，基于学生现有数学知识来证明其正确性；引入更多高级数据结构，增加了对这些数据结构和算法实现的讨论。

上机实习在本教材中作为单独一章编写。关于经典数据结构和算法，要求学生在 C++环境下编程实现、调试和测试，并实际运行它们，评估算法效率。同时，指导学生学习书写软件文档。

全国硕士研究生招生考试计算机科学与技术学科联考计算机学科专业基础综合考试大纲（简称“计算机考研大纲”）包括数据结构、计算机组成原理、操作系统和计算机网络 4 门基础课程，其中数据结构占 30%。本教材涵盖 2019 年计算机考研大纲数据结构部分的考查内容。

全书分为16章，条理清晰，内容详实，深入浅出。书中对算法做了较详细的解释，尽可能做到可读易懂，并配有大量的实例和图示，有利于读者理解算法的实质和编程思想。每章引言和结尾处的小结可以帮助读者了解本章的要点，并配有丰富的习题，既适合课堂教学也适合自学。

本教材可作为高等学校计算机类及其他相关专业“数据结构”课程84学时的教学内容。对于学时数少于84学时的授课计划，可根据实际学时加以剪裁。对于难度较大或非基本部分的章节，作者已在目录中标上*号，供读者选取时参考。除标*号章节外的基本部分适合48～56学时的授课计划。

本教材的编写得到了电子工业出版社编辑冉哲女士的积极推荐和支持，在此衷心地表示感谢。书中若有不当之处，敬请读者批评指正。

作者

2019年于上海

目　录

第1章 概 论

数据结构是计算机学科的重要基础课程。本章介绍数据结构的基本概念，与此同时，扼要介绍问题求解、数据抽象、抽象数据类型、面向对象方法等软件设计相关概念和方法，其中涉及抽象、分解、模块化、封装与信息隐蔽、重用、继承、多态性等诸多系统设计原理。本章给出了依照抽象数据类型的观点，借助格式化自然语言与 C++模板抽象类的数据结构和算法的描述方法。本章最后讨论算法和算法分析的基本方法。

1.1 问题求解方法

进入21世纪以来，计算机技术飞速发展，计算机学科和其他学科日益融合，已成为其他学科不可或缺的方法和工具，并且渗透到社会生活的方方面面，成为对人类生活影响最大的学科，是推动社会进步的重要引擎，乃至形成了“计算思维”和“计算文化”。今天，信息技术作为现代技术的标志，已成为世界各国经济增长的主要动力。

问题求解无疑是计算机科学的根本目的。随着计算机解决问题能力的提升，很多问题都可以用计算机求解，计算机的应用非常广泛。计算机科学被应用于解决许多学科领域的计算问题。有些问题如四色问题（四色猜想），如果没有现代计算机，恐怕至今难以得解。

计算思维（computational thinking）是一种运用计算机科学的基础概念进行问题求解、系统设计和人类行为理解等的思维活动，已被列为计算机人才的专业基本能力之一。计算机科学课程的学习是培养计算思维能力的重要途径。数据结构作为计算机学科的一门专业基础课，理应在课程教学中落实计算思维能力的培养。

1.1.1 问题和问题求解

什么是**问题**（problem）？只要目前的情况与人们所希望的目标不一致，就会产生问题。例如，排序问题是“任意给定一组记录，排序的目的是使得该组记录按关键字值非减（或非增）顺序排列。”

问题求解（problem solving）是指寻找一种方法来实现目标。问题求解是一种艺术。没有一种通用的方法能够求解所有问题。有时，人们不得不一次又一次地尝试可能的求解方法，直到找到一种正确的求解途径。一般来说，问题求解中存在着猜测和碰运气的成分。然而，当我们积累了问题求解的诸多经验后，这种对问题解法的猜测就不再是完全盲目的，而是形成了某些问题求解的技术和策略。**问题求解过程**（problem solving process）是人们通过使用问题领域知识来理解和定义问题，并凭借自身的经验与知识去选择和使用适当的问题求解策略、技术及工具，将一个问题描述转换成问题解的过程。

1.1.2 问题求解过程

匈牙利数学家乔治·波利亚（George Polya）在1957年出版的 *How to solve it* 一书中概括了求解数学问题的四步法。这种求解问题的四步法对于大多数其他科学也是适用的，同样也可用于求解计算机应用问题。求解问题的四步法简述如下。

（1）**理解问题**（understand the problem）。毫无疑问，为了求解问题必须首先理解问题。如果不理解问题，当然就不可能求解它。此外，对问题的透彻理解有助于求解问题。这一步很重

要，它看似简单，其实并不容易。在这一步，还必须明确定义所要求解的问题并以适当的方式表示问题。对于简单问题，不妨直接用自然语言描述问题，如排序问题。

（2）**设计方案**（devise a plan）。求解问题时，首先考虑从何处着手，考虑以前是否遇到过类似的问题，是否解决过规模较小的同类问题。此外，还应选择该问题的一些特殊例子进行分析。在此基础上，考虑选择何种问题求解策略和技术进行求解，以期得到求解问题的算法。

（3）**实现方案**（carry out the plan）。实现求解问题的算法，并使用问题实例进行测试、验证。

（4）**回顾复查**（look back）。检查该求解方法是否确实求解了问题或达到了目的。评估算法，考虑该解法是否可以简化、改进和推广。

1.1.3 计算机求解问题的过程

狭义的计算思维立足于计算机学科的基本概念，研究如何将问题求解过程映射成计算机程序的方法。计算机求解问题的关键之一是寻找一种**问题求解策略**（problem solving strategy），得到求解问题的算法，从而得到问题的解。例如，求解排序问题是指设计一种排序算法，能够把任意给定的一组记录排成有序的记录序列。

借助计算机求解问题的过程与上述传统的求解问题过程本质上是类似的。

① 分析和理解问题，从待求解的问题中抽象出适当的模型，并以适当的方式表述它。

② 提取和合理组织数据，设计求解问题的有效算法。

③ 选择适当的程序设计语言编程实现数据结构与算法。

④ 评估和改进。

计算机求解问题的过程是从一个问题的描述开始，直到编写出解决问题的计算机程序从而得到问题解的全过程。对于大型复杂问题，人们必须考虑程序设计方法学、系统设计原则及其他相关理论。一个复杂系统的开发是分阶段进行的。每个阶段完成相对独立的任务，各阶段都有相应的方法和技术，每个阶段都有明确的目标，要有完整的文档资料。这种做法有助于降低程序开发和维护的困难程度，保证软件质量，提高开发大型软件的成功率和生产率。

1.2 什么是数据结构

1.2.1 算法与数据结构

如前所述，计算机求解问题必须先把对客观事物的描述抽象为数据，把问题求解步骤抽象为算法，才能由计算机处理。每个程序都会隐式或显式地使用数据，程序中的数据常常需要组织成数据结构，由算法操纵和管理从而得到结果。粗略地说，算法一词用于描述一个可用计算机实现的问题求解方法。算法是程序的灵魂。此外，通常还需要根据问题组织数据。数据的组织结构方式将直接影响求解问题的算法及其效率。算法与数据结构最终必须使用某种程序设计语言实现，才能在计算机中实际运行并得到问题的解。算法、数据结构与程序三者的关系恰如 Niklaus Wirth 教授所著的一本书的书名《算法+数据结构=程序》所描述的。因此，算法与数据结构的理论和方法对计算机学科是至关重要的。

数据结构和算法的内容主要包含在 ACM/IEEE 的《计算机科学课程体系规范 2013》（computer science curricula 2013，简称 CS2013）的**软件开发基础**（software development fundamentals，SDF）和**算法与复杂性**（algorithms and complexity，AL）两个知识领域中。面向对象程序设计的内容包含在**程序设计语言**（programming languages，PL）知识领域中。本书中关于经典数据结构与算法的知识点和要求见“SDF/基本数据结构”和“AL/基本数据结构和算法”知识单元。“AL/高级数据结构和算法”涵盖平衡树、B 树、拓扑排序、键树等内容。算法分析

基本方法的相关知识点见“AL/基本分析”知识单元。“SDF/算法与设计”知识单元涉及问题求解、算法及其效率，以及抽象、封装与信息隐蔽、行为与实现分离等诸多知识点。递归技术见“SDF/程序设计基本概念”知识单元。

1.2.2 数据结构基本概念

数据结构是计算机科学与技术领域中广泛使用的术语。然而，究竟什么是数据结构，在计算机科学界至今没有标准的定义。

Sartaj Sahni 在《数据结构、算法与应用》（第二版）一书中称：“数据结构是一个数据对象，该对象的实例和组成实例的数据元素间存在联系，这些联系可由相关的函数来规定”。该书中，Sartaj Sahni 将**数据对象**定义为“一个**数据对象**（data object）是**实例**（instance）或**值**（value）的集合。”

Clifford A. Shaffer 在《数据结构与算法分析》（第三版）一书中的定义是：“数据结构是 ADT[①]的物理实现。”“ADT 定义了数据类型的逻辑形式，数据结构是实现数据类型的物理形式。”

Robert L. Kruse 则在《数据结构与程序设计》一书中，将一个数据结构的设计过程分成抽象层、数据结构层、实现层和应用层。其中，抽象层讨论数据如何彼此相关和需要执行什么操作，这一层关注的是问题求解，而不是如何解决问题。数据结构层需要考虑较多的细节，使得能分析各个方法的行为从而做出适当的选择。实现层确定数据结构在计算机内的表示。应用层实现一个应用所需的全部细节。作者认为，最初两层可称为概念层，在这一层上主要关心问题求解，而不是程序设计；中间两层可称为算法层，这两层关心表示数据和操纵数据的精确方法；最后两层专门考虑程序设计。

Frank M. Carrano 和 Timothy Henry 在他们的《数据结构与抽象》（第四版）和《C++数据抽象和问题求解》（第六版）中说：“数据结构是一种可以用程序设计语言定义的，用于存储数据的结构。”“数据结构是使用程序设计语言对 ADT 的实现。”

现实世界各领域中的大量信息都必须转换成数据才能在计算机中存储和处理。数据是信息的载体，应用程序处理各种各样的数据。笼统地说，所谓**数据**（data），就是计算机加工处理的对象。数据一般分两类：**数值数据**（numerical data）和**非数值数据**（non-numerical data）。数值数据是一些整数、实数或复数，主要用于工程计算、科学计算和商务处理等。非数值数据包括字符、文字、图形、图像、语音、表格等。这类数据的特点是量大，而且往往有着复杂的内在联系。单纯依靠改进程序设计技巧，已无法编制出高效可靠的程序，而必须对数据本身的结构加以研究。虽然有些算法并不要求精心组织输入数据，但另一些算法的确依赖于精心设计的数据结构。对问题实例的数据进行恰当组织与重构，有助于设计和实现高效的算法。算法设计通常建立在所处理的数据的一定组织形式之上。对于相同数据的同样处理要求，如果选择不同的数据结构，则会有不同的处理效率：操作时间和存储空间。数据结构主要研究与解决如何使用计算机组织和处理非数值问题而产生的理论、技术及方法，是计算机学科研究的基本课题之一。

前面提到，数据是计算机加工处理的对象。一个数据对象是实例或值的集合。一个数据对象的实例可以是不可分割的（即原子的），也可以是由其他数据对象的实例组合而成的。我们将组成数据对象实例的成分数据称为**数据元素**（data element）。数据元素是数据的基本单位。数据元素可以是简单类型的，如整数、实数、字符等，也可以是结构类型的，如记录。例如，字符串数据对象 String 是所有可能的字符串实例的集合。每个字符串实例均由字符组成，每个数据

① ADT：abstract data type，抽象数据类型。

元素均是字符数据对象 Letter 的一个实例。如果把每个学生的记录看作一个数据元素，则它包含学号、姓名、性别等**数据项**（data item）。某班学生的记录组成了表 1-1 中给出的学生情况表。

表 1-1　学生情况表

学　号	姓　名	性　别	其 他 信 息
B02040101	王小红	女	……
B02040102	林悦	女	……
B02040103	陈菁菁	女	……
B02040104	张可可	男	……
……	……	……	……

一般地，一个数据对象的实例及组成实例的数据元素间存在着相关性。研究发现，从应用问题得到的数据对象的实例，数据元素间常常存在着某种结构上的相关性。对数据元素间逻辑结构关系的描述称为数据的逻辑结构。数据的逻辑结构是面向应用问题的，是从用户角度看到的数据结构。数据必须在计算机内存储，数据的存储结构是数据在计算机内的表示形式，是逻辑数据的存储映像。它是面向计算机的。实际上，数据元素间的相关性除逻辑结构上的关系外，有时还需考虑数据元素值间的相关性。例如，一个有序表，除表现为数据元素排成一列的结构关系外，还要求按数据元素值的大小排列。这类相关性由数据对象上的操作保证。一个数据对象都有一组操作（或函数）与其相关联。这些操作可以是将一个对象的某个实例转换成该对象的另一个实例，或者转换成其他对象的一个实例，或者同时执行这两种转换。一个操作可以仅仅创建一个新的实例，而对于创造新对象的原对象不做任何改变。例如，两个自然数相加操作产生一个新的两数之和的自然数，但两个加数没有任何改变。

一个数据结构是一个数据对象及与之相关联的一组操作，用以规定可以对该对象实例及组成实例的数据元素实施的操作。当我们研究数据结构时，还需要关心数据对象的表示（实际上是实例的表示），以及与数据对象相关的操作的算法实现。每个数据对象的表示都应当有利于操作的高效实现。

由此看来，数据的逻辑结构及在数据上执行的操作应该是数据结构研究的重要方面。另外，数据的存储结构及操作的实现也是数据结构研究不可缺少的另一个重要方面。有效地使用存储空间，并具有高的时间效率，是判定一种数据结构设计优劣的主要标准。

本书中，我们将一个数据结构的讨论分为两个层次，即抽象层和实现层。抽象层讨论数据的结构方式，即数据是如何彼此相关的及需要执行什么操作。实现层讨论数据的存储表示及操作的算法实现。

1.2.3　数据的逻辑结构

从概念上讲，一个数据对象的实例是由数据元素依据某种逻辑联系组织起来的。对数据元素间逻辑关系的描述被称为数据的逻辑结构，它可以用一个二元组表示，即

$$DS=(D,R)$$

其中，D 是数据元素的有限集合，R 是 D 中数据元素序偶的集合。

例如，$DS=\{D,R\}$，$D=\{a,b,c,d\}$，$R=\{<a,b>,<b,c>,<c,d>\}$，其中，序偶$<a,b>$表示数据元素 a 和 b 间的关系，称 a 是 b 的**直接前驱**（predecessor），b 是 a 的**直接后继**（successor）。在不引起混淆的情况下，常简称直接前驱为**前驱**，直接后继为**后继**。

本书中以小圆圈表示数据元素，用带箭头的线表示数据元素间的关系。两个不同数据元素的序偶称为**边**。图 1-1 是一个包含 4 个数据元素的线性数据结构的示意图。

图 1-1　DS 示意图

根据数据结构中数据元素间逻辑关系的不同特征，可划分为以下 4 种基本逻辑结构。

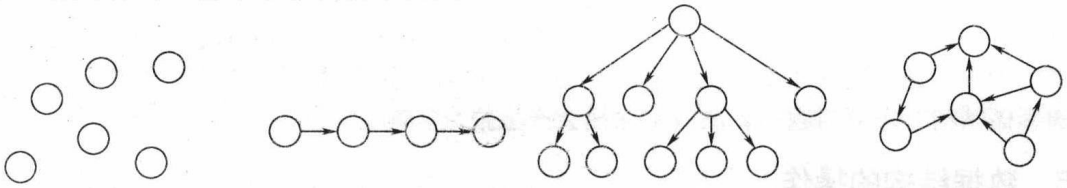
（1）**集合结构**（set）。在集合结构中，数据元素间的顺序是随意的。数据元素间除“属于同一个集合”的联系之外没有其他关系。由于集合结构的数据元素间没有固有的关系，因此往往需要借助其他结构才能在计算机中实际表示此结构。

（2）**线性结构**（linear）。线性结构是数据元素的有序序列，其中，第一个数据元素没有前驱只有后继，最后一个数据元素只有前驱没有后继，其余数据元素有且仅有一个前驱和一个后继。数据元素间形成一对一的关系。

（3）**树状结构**（tree）。在树中，除一个特殊数据元素称为根，它没有前驱只有后继外，其余数据元素都有且仅有一个前驱，但后继的数目不限。对于非根数据元素，都存在着一条从根到该元素的**路径**。数据元素间存在一对多的关系，树是层次数据结构。

（4）**图结构**（graph）。图是最一般的数据结构，图中每个数据元素的前驱和后继的数目都不限。图中数据元素间的关系是多对多的关系。

上述 4 种基本的结构关系又可分为两类：**线性结构**（linear structure）和**非线性结构**（non-linear structure）。我们把除线性结构以外的几种结构关系——树、图和集合归入非线性结构一类。图 1-2 为 4 种基本结构的示意图。



(a) 集合结构　(b) 线性结构　(c) 树状结构　(d) 图结构

图 1-2　4 种基本的结构关系

1.2.4　数据的存储表示

计算机内存是由有限个存储单元组成的一个连续存储空间。这些存储单元是字节编址或者是字编址的。从存储器角度看，内存中是一堆二进制数据。它们可以被机器指令解释为指令、整数、字符、布尔值等，也可以被数据结构的算法解释为具有某种结构的数据。

对一个数据结构，找到一种有效的存储表示方式，使它适于计算机表示是十分重要的。**顺序**表示和**链接**表示是两种最基本的存储表示方式。

顺序（或称**连续**）表示方式需要一块连续的存储空间，并把逻辑上相关的数据元素依次存储在该连续的存储区中。例如，由 4 个数据元素组成的线性数据结构(a_0,a_1,a_2,a_3)，存储在某个连续的存储区内，设存储区的起始地址是 102，假定每个数据元素占 2 个存储单元，则其顺序表示如图 1-3(a)所示。

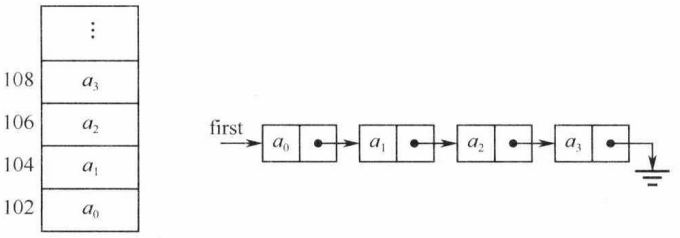
在顺序表示下，可以容易地用一个数学公式来确定每个数据元素的存储地址。对于图 1-3(a)的顺序存储结构，可使用式（1-1）计算数据元素 a_k 的存储位置 $\mathrm{Loc}(a_k)$，即

$$\mathrm{Loc}(a_k)=102+2\times k \tag{1-1}$$

顺序表示并不仅仅限于存储线性数据结构。对于非线性数据结构，如树状结构，有时也可采用顺序表示。这将在以后讨论。

另一种基本的存储表示方式是链接表示。在链接表示下，为了在计算机内存储一个数据元

素，除需要存放该数据元素本身的信息外，还需要存放与该数据元素相关的其他数据元素的地址信息。这两部分信息组成存放一个数据元素的**结点**(node)。图 1-3(b)所示为线性结构(a_0,a_1,a_2,a_3)的链接表示。其中，每个结点的存储块分成两部分，一部分存放数据元素自身，另一部分包含该数据元素逻辑上的后继结点的存储地址。这种关于相关结点的地址信息被称为**链**（link）。图 1-3(b)中，用电路的接地符号表示空链（即不代表任何具体结点的存储地址）。注意，一个结点的存储地址通常是指存放该结点的存储块的起始存储单元的地址。



（a）顺序表示　　（b）链接表示

图 1-3　两种基本的存储表示方式

注意：在以后的讨论中，在不会引起混淆的场合，可不明确区分结点和数据元素（元素）这两个术语。但必要时，将包括地址信息在内的存储块整体称为结点，而将其中的元素信息部分称为该结点中的元素。

除顺序表示和链接表示外，还可以有其他存储数据的方式，如**索引**方式和**散列**方式。关于这两种方式，请参看以后相关章节。这些基本的存储表示方式及它们的组合，可衍生出数据的更多形式的存储结构。对于数据的某种组织方式往往有多种存储方式可以选择，而选择和构建不同的存储结构对求解问题的算法效率往往会产生很大影响。

1.2.5　数据结构的操作

如果说数据的逻辑结构描述了数据的静态特性，那么在数据的逻辑结构上定义的一组操作给出了数据被使用的方式，即数据的动态特性。通过使用数据结构上定义的操作，用户可对数据结构的实例或组成实例的元素实施所需的操作。操作的结果可使数据改变状态。

研究数据结构是为了解决应用问题。数据抽象强调将数据和数据上执行的一组操作组合在一起。数据结构包含的最常见的操作有：

创建——创建数据结构的一个实例；

撤销——撤销数据结构的一个实例；

查找——在数据结构的一个实例中查找满足条件的元素；

更新——在数据结构的一个实例中修改指定元素的值；

插入——在数据结构的一个实例中插入一个新元素；

删除——从数据结构的一个实例中删除指定元素；

遍历——按照某种顺序，系统地访问数据结构的一个实例的各元素，使得每个元素恰好被访问一次。

栈是一种线性数据结构，可以向栈中加入元素，但只允许访问和删除最后入栈的元素。例 1-1 给出了在栈上定义的若干操作。

例 1-1　栈数据结构。

（1）向栈中加入一个元素（Push 操作）；

（2）从栈中删除最后加入的元素（Pop 操作）；

（3）访问最后加入栈中的元素（Top 操作）。