计算机系列教材

数据结构与算法

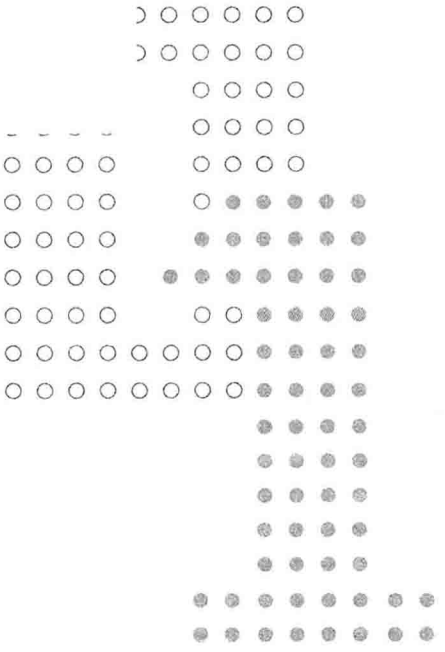
永林 周蓓 唐晓阳 编著

清华大学出版社

计算机系列教材

邹永林 周蓓 唐晓阳 编著

数据结构与算法



清华大学出版社
北京

内 容 简 介

数据结构与算法注重理论与实践相结合，不仅是计算机学科的核心基础课程，也是程序设计的重要理论基础。本书系统地讲述了数据结构与算法的基本理论和实际应用，全书分为两个部分，共 9 章，第一部分主要讨论数据结构的基础知识和表示方式，包括线性结构(线性表、栈、队列、串、数组及广义表)、树形结构、图形结构等的定义、表示和实现；第二部分讨论排序和查找两类常用算法的原理、方法及其实现技巧。

全书强调实用，注重理论指导下的实际可操作性，注重实际问题的解决。书中所有关于基本数据结构的定义和算法描述均采用标准的 C 语言格式给出，所有算法代码均在 TC 2.0、Visual C++ 6.0、Codeblocks 等开发环境中调试通过并运行正确，读者可根据各自的要求和习惯等选择使用对应的工具。

本书可作为高等学校计算机类各专业数据结构课程的教材或参考书，特别适合应用技术型本科层次的学生使用；也可供从事计算机应用相关工作的人员参考。

图书在版编目(CIP)数据

数据结构与算法/邹永林，周蓓，唐晓阳编著. —北京：清华大学出版社，2015
计算机系列教材
ISBN 978-7-302-39337-5

Ⅰ. ①数…　Ⅱ. ①邹…　②周…　③唐…　Ⅲ. ①数据结构－高等学校－教材　②算法分析－高等学校－教材　Ⅳ. ①TP311.12

中国版本图书馆 CIP 数据核字(2015)第 024961 号

责任编辑：张　玥　赵晓宁
封面设计：傅瑞学
责任校对：李建庄
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**：15.75　　**字　　数**：390 千字
版　　次：2015 年 6 月第 1 版　　**印　　次**：2015 年 6 月第1 次印刷
印　　数：1～2000
定　　价：29.50 元

产品编号：061549-01

《数据结构与算法》前言

随着计算机科学技术的不断发展和应用领域的不断扩大，在许多非数值处理的应用问题中，计算机所面对的数据结构十分复杂、数据量巨大且形式多样化，如何根据各类实际问题归纳、抽象出对象的数据特征及对象间的相互联系，从而选择合适的数据组织方法和存储方法，设计高效的求解算法，成为计算机学科需要解决的最迫切的任务。

数据结构与算法是一门实践性很强同时又十分抽象的计算机学科基础课程，本书基于 CDIO 的理念进行编写。CDIO 是源于国外的工程教育模式，体现了欧美理工类学科教育改革的全新理念。通过构思(Conceive)、设计(Design)、实现(Implement)和运行(Operate)4 个环节，引导学生积极参与“做中学”和“基于项目的教育和学习”的整个过程，达到学习效果的提高和升华，真正实现课程教学的目的。本书将这种教学理念引入到编写中，每种数据结构均以流行的抽象数据类型格式(ADT)对其进行定义，使用 C 语言函数的形式描述其对应的存储结构及基本操作算法，以典型算法设计来实现其基本应用，以应用实例分析深化对基本概念的理解和培养分析问题与解决问题的能力。

本书强调实用性，注重理论指导下的可操作性，注重提高分析问题、解决问题的能力。各章均配有小结，目的在于引导读者复习该章内容；各章课外习题和实验课题由配套教材《数据结构与算法习题解析和实验指导》提供，以期通过典型习题与实践指导使读者更全面、更透彻地掌握数据结构与算法这门课程。

本书第 1 章介绍数据结构的概念，第 2～第 5 章介绍各种线性结构的知识，第 6 章介绍树形结构，第 7 章介绍图形结构，第 8 章介绍查找算法，第 9 章介绍排序算法。

与本书配套的课件及相关电子资料，需要的读者可与作者联系，作者的 E-mail：zyl@cslg.cn。

参加编写的有邹永林(第 1、第 4、第 7 和第 9 章)、周蓓(第 2、第 6 和第 8 章)、唐晓阳(第 3 和第 5 章)，周思林、朱奭、沈健、洪蕾等参与讨论和算法的设计与调试；邹永林负责全书的统稿。

由于作者水平有限，书中难免存在不足之处，恳请广大读者批评指正。

编　者

2015 年 5 月

FOREWORD

《数据结构与算法》目 录

第1章 绪 论

计算机科学是研究用计算机进行信息表示和处理的科学。从1946年世界上诞生了第一台电子计算机以来，计算机科学与技术的发展日新月异，计算机的应用也从最初的数值计算，扩展到非数值计算的各个领域，其表示和处理的对象也由单纯的数值数据扩展到字符数据、图形图像数据、声音数据等相对比较复杂、带有一定结构、彼此存在一定关联的数据形式，这就给各类程序设计带来了新的问题和困难。为了编写出能很好地完成处理各种具体问题的程序，必须对所要处理的问题中包含的各种数据对象的特性及数据对象之间存在的各种关系进行深入分析和研究，从而更好地进行程序设计。这就是数据结构课程需要研究的主要目标和任务。

1.1 引言

数据的表示是计算机科学的基础。在实际工程应用领域中，大多数计算机程序的实现目标与其说是完成运算，不如说是组织、存储和检索数据。因为从运行时间和存储空间两方面分析，这些程序都必须合理地组织数据，以支持高效的信息处理过程。

在用计算机求解某个特定问题时，一般需要经过以下三步：

(1) 建立数学模型。对该特定问题进行分析，从中抽象出能准确地描述此问题的数学表示形式。

(2) 设计问题求解的算法。对该数学模型进行分析，寻找求解此模型的方法，并用某种方式将问题的求解过程加以描述。

(3) 编写程序，完成问题的求解。

在使用数学模型对具体的问题进行描述时，由于问题本身的多样性和复杂性，其数学模型的表示形式也千变万化，有些问题的数学模型可以用数学方程(组)或微积分方程(组)等相对简单的形式加以描述；但更多的非数值计算问题，它们无法用相对简单的数学方程表示和描述。

1.1.1 几个实例

【例1-1】 在实际工作中，人们面临各种数据管理问题，如人事档案管理、财务工资管理、客户关系管理、库存物资管理、学生学籍管理等。以学生的学期课程成绩管理为例，假设某班共有40位同学，某学期共开设英语、高等数学、微机基础和哲学4门必修课程，为了实现对该班学生的各门课程成绩的管理，可以通过设计如表1-1所示的表，在表中至少应包括学号、姓名、英语、高等数学、微机基础和哲学共6个数据项，分别记录每个学生的学号、姓名和各门课的成绩，其中学号项的值必须针对每个学生都是唯一的。这样，可

以根据表中的学号值查询对应于该学号的学生的所有课程成绩；也可以通过设置查询条件如“英语课程成绩大于等于80分”来获取该课程成绩高于80分的所有学生的名单；或设置某种组合形式的查询条件如“英语、高等数学、微机基础、哲学的成绩均大于等于60分”得到确定该学期所有课程全部及格的学生名单以备作为评选奖学金等活动的依据等。可用一张二维表描述这类数据管理问题，表中的记录（数据元素）按照学号依次组织，记录之间存在一一对应的关系，人们将这类数学模型称为“线性”的数据结构。

表1-1　学生学期成绩表

学　号	姓　名	英　语	高等数学	微机基础	哲　学
99001	李伟	75	78	65	83
99002	刘可欣	71	82	85	77
99003	刘强	84	76	78	91
99004	钱立新	75	56	61	71
⋮	⋮	⋮	⋮	⋮	⋮
99039	徐红	72	68	55	73
99040	万新	88	78	84	86

【例1-2】 本例分析棋类对弈游戏，如五子棋、围棋、象棋等。以最简单的井字棋对弈为例，其初始状态是一个空的棋盘格局。对弈开始后，每下一步棋，则构成一个新的棋盘格局，且相对于上一个棋盘格局的可能选择可以有多种形式……因而整个对弈过程就像如图1-1所示的“一棵倒长的树”的形态。在这棵“树”中，从初始状态（根）至某一最终的格局（叶子）的一条路径，描述的就是一次具体的对弈过程实例。描述对于这类问题的数学模型，“树”状图是最简单直观的方式，树中的棋盘格局（数据元素）之间存在一个对多

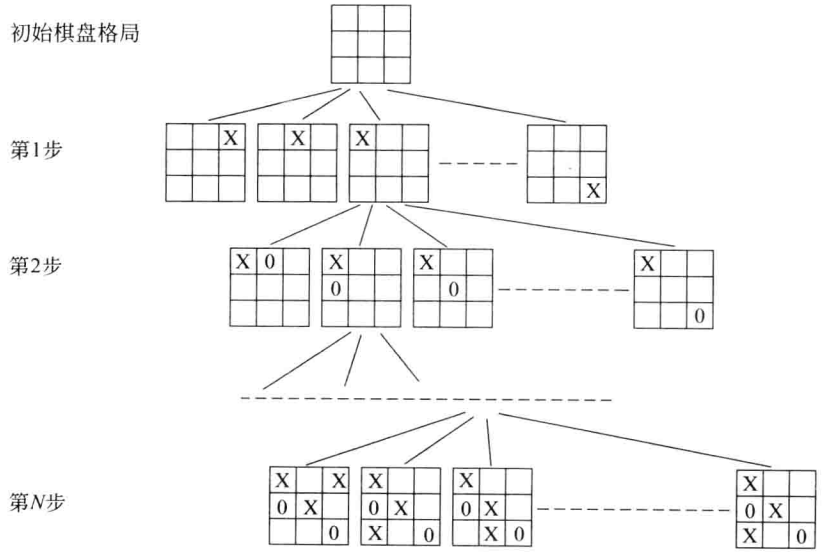


图1-1　井字棋的对弈过程

个的关系，这种数学模型称为“树形”的数据结构。

【例 1-3】 本例讨论工程规划问题，如通信线路架设、工程施工计划、交通通行控制等。以专业课程实施计划的制订为例，假设计算机应用专业在 4 年中共需要开设高等数学、普通物理、程序设计、离散数学、计算机原理、数据结构、编译技术和操作系统等 8 门课程，其中某些课程之间彼此构成先修课程和后续课程的关系（见图 1-2），因此在具体落实学年学期课程安排时，必须根据这些关系来确定各门课程的开设时间，使之符合课程教学的实际情况。图 1-3 中的这类规划问题，课程（数据元素）与课程（数据元素）之间存在多个对多个的关系，这类数学模型由于可用一张图来描述，因而称为“图形”的数据结构。

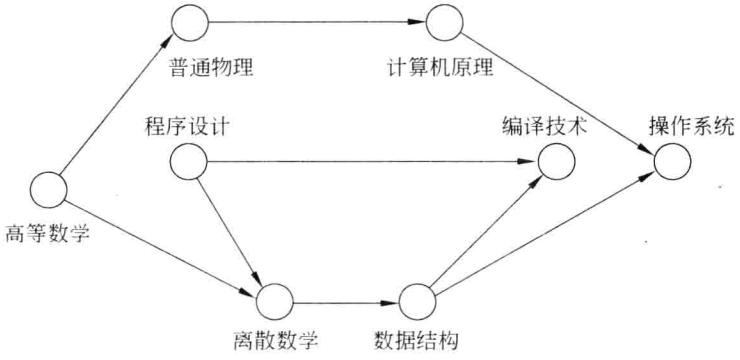


图 1-2 专业课程之间的关系

由此可知，对于非数值型数据处理问题的数学模型的描述形式不再表现为相对简单的数学方程（组），而是诸如二维表、树和图等更为复杂多变的形式。对于这类数学模型的求解，必须通过深入研究和探索，才能正确实现。

1.1.2 数据结构的产生和发展

数据结构课程的产生起源于 20 世纪 60 年代。

此前，随着各种计算机程序设计语言出现，允许使用的数据类型逐渐增多，计算机需要处理的数据的组成成分日趋复杂，如 FORTRAN 语言中允许使用复数类型的数据，用两个实型数分别表示复数的实部和虚部，开始形成“结构”的雏形；COBOL 语言允许使用记录类型，是可以用多种不同类型的数据组合而成的更复杂结构；SNOBOL 语言能十分方便地进行串（非数值）类型数据的操作；LISP 语言中还定义了带层次性的表结构等。对这些数据及数据之间的关系的描述、表示和操作，迫切需要通过专门的、系统的分析研究来解决，从而导致了“数据结构”作为一门独立课程的形成和发展。

20 世纪 60 年代中期以后，在数据结构的发展过程中，陆续发生了几个重大的事件。首先，在美国计算机界提出了信息结构（后改名为数据结构）的概念，1968 年美国计算机协会（ACM）颁发了建议性的计算机教学计划，首次将数据结构列为一门独立的课程；同年，著名计算机科学家 Knuth 教授发表了《计算机程序设计的技巧》第一卷《基本算法》，

全面而又系统地论述了数据的逻辑结构和存储结构，并给出了各种典型的算法，为“数据结构”课程奠定了理论基础；20世纪70年代以后，逐渐出现大型的程序和大规模的文件系统，结构化程序设计成为程序设计方法学的主要研究方向，人们对数据结构的研究越来越重视，普遍认为程序设计的实质就是对所处理的问题选择一种好的数据结构，并在此结构基础上施加一种好的算法，著名科学家Wirth教授的著作《算法＋数据结构＝程序》正是这种观点的集中体现。此后，有关数据结构的研究不断深入，内容也逐渐固定和规范，并相继出现了各种数据结构著作和教材。

数据结构在计算机科学中是一门综合性的专业基础课，不仅涉及计算机硬件，也与计算机软件有关。因此，可以认为数据结构是介于数学、计算机硬件和计算机软件三者之间的一门核心课程，它不仅是一般程序设计的基础，也是设计和实现编译程序、操作系统、数据库系统及其他系统程序和大型应用程序的重要基础。

有关数据结构的研究仍在不断发展，一方面，面向各专门领域中特殊问题的数据结构正在研究和深入；另一方面，从抽象数据类型的观点出发，对数据结构进行探讨已成为新趋势。

1.2 数据结构

1.2.1 基本概念和术语

1. 数据

数据(Data)是描述客观事物的文字、声音、图形图像等所有能输入计算机中并被计算机程序加工处理的符号(信息)的集合。一般意义下的数据形式如整数和实数等，只是数据的特例。一个编译程序或文字处理程序处理的数据是文件中的字符串，而多媒体处理程序处理的是通过特殊编码定义后的图形、图像、声音、动画等数据。对于计算机而言，数据的含义非常广泛，一切通过各种编码形式输入计算机中并进行处理的对象均可归属到数据的范畴。

2. 数据元素

数据元素(Data Element)是数据的基本单位，是数据集合中的一个个体。通常，在计算机程序中它作为一个整体进行考察和处理。所谓基本单位，指其大小可变，可根据描述数据个体的性质的需要确定。每个数据元素既可以只包含一个数据项(Data Item或Field)，也可以由若干数据项组合而成。数据项是构成一个数据元素具有独立现实意义、不可分割的最小单位。

3. 数据对象

数据对象(Data Object)是性质相同的数据元素的集合，是数据集合中的一个子集。例如，整型数据对象是集合{0，±1，±2，…}，字母字符数据对象是集合{A，B，…，Z，a，

b,…,z}等。

1.2.2 数据结构定义

数据结构是相互之间存在一种或多种特定关系的数据元素的集合。在各类实际应用问题中,数据元素之间总是存在着各种关系,描述数据元素之间关系的方法称为结构。通常,可根据数据元素之间所存在的关系的不同特征,用4类基本结构予以描述:

(1) 集合:指结构中的数据元素之间只存在"同属一个集合"的关系。

(2) 线性结构:指结构中的数据元素之间存在"一个对一个"的关系。

(3) 树形结构:指结构中的数据元素之间存在"一个对多个"的关系。

(4) 图形结构:指结构中的数据元素之间存在"多个对多个"的关系。图1-3所示为上述4类基本结构示意图。

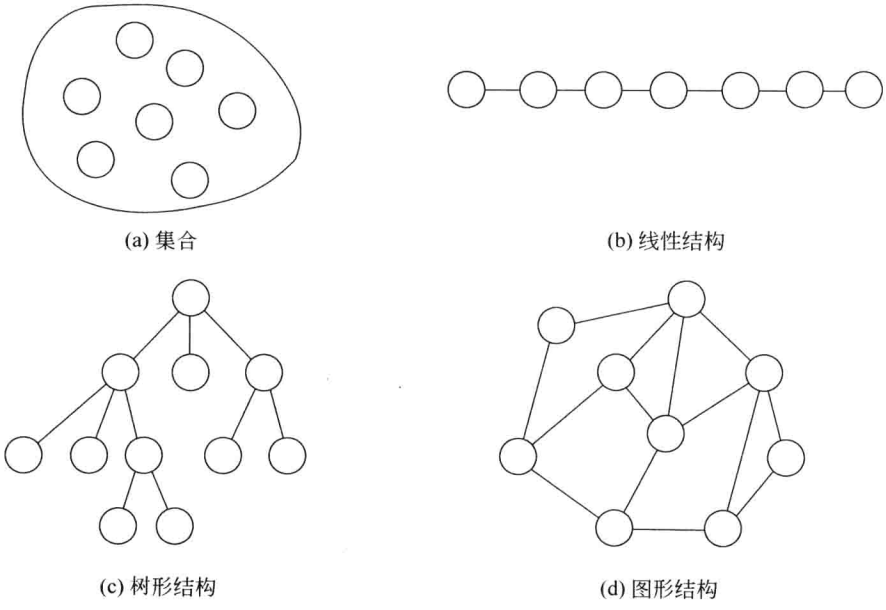


图1-3 4类基本结构示意图

1. 数据结构的形式定义

借助集合论,数据结构可描述为一个二元组

$$\text{Data\_Structure} = (D,S) \tag{1-1}$$

其中,D是数据元素的有限集合;S是D集上关系的有限集合。可通过以下例子加以理解。

【例1-4】 复数的数据结构表示。

在计算机科学中,复数可定义为

$$\text{Complex} = (C,R) \tag{1-2}$$

其中C是包含两个实数的集合{ r_1,r_2}；R是定义在集合C上的一种关系{<r_1,r_2>}，其中有序对<r_1,r_2>分别代表复数的实部和虚部。

【例1-5】 某城市电话号码簿的数据结构表示。

某城市的电话号码簿可定义为

$$\text{Telephone\_Book} = (T,R) \tag{1-3}$$

其中，$T=\{(u_1,t_1),(u_2,t_2),\cdots,(u_n,t_n)\}$，n表示用户个数，$u_i$表示用户名，$t_i$表示对应的电话号码；$R=\{<N_i,N_{i+1}>|N_i,N_{i+1}\in T,i=1,2,\cdots,n-1\}$，$N_i=(u_i,t_i)$。

【例1-6】 成绩管理系统中某班级学生的学期成绩表的数据结构表示。

某班级学生的学期成绩表可定义为

$$\text{Score\_Table} = (D,R) \tag{1-4}$$

其中，$D=\{(r_1,n_1,s_{11},\cdots,s_{1k}),(r_2,n_2,s_{21},\cdots,s_{2k}),\cdots,(r_m,n_m,s_{m1},\cdots,s_{mk})\}$，m为该班学生数，k为学期课程数，$r_i$为学生序号，$n_i$为学生姓名，$s_{i1}$为第1门课程成绩，…，$s_{ik}$为第k门课程成绩；$R=\{<N_i,N_{i+1}>|N_i,N_{i+1}\in T,i=1,2,\cdots,m-1\}$，$N_i=(r_i,n_i,s_{i1},\cdots,s_{ik})$。

数据结构的形式定义是从数学角度对操作对象的描述，也就是说，是通过抽象得到的数学模型。这种数学模型描述的是数据元素之间的逻辑关系，因此，又称为数据的逻辑结构。相对于数据的逻辑结构，为了在计算机中实现对应的操作，还必须讨论它在计算机中的表示方式。

2. 数据的存储结构(物理结构)

数据的逻辑结构在计算机中的表示(映像)称为数据的存储结构或物理结构。这个映象包括数据元素的表示和数据元素之间关系的表示两方面。

在计算机中，数据信息的表示(存储)单位有位(b)、字节(B)、千字节(KB)、兆字节(MB)等，不同类型的数据在存储时需要的空间大小是不同的。一般地，一个整型数据需要2字节，一个字符型数据需要1字节，一个实型数据则需要4字节……数据的逻辑结构中包含的数据元素在计算机中的存储形式称为元素(Element)或结点(Node)，若此数据元素由多个数据项组成，则每个数据项对应在元素或结点中的部分称为域(Field)，根据数据项中的数据的类型，分为数据域(Data Field，存放普通类型的数据)和链域(Link Field，存放指针型数据，也称指针域)。因此，所谓元素或结点就是指数据元素在计算机中的映像(存储结构)。

数据元素之间的关系在计算机中表示时，通常可采用顺序映像和非顺序映像两种不同的方式实现。所谓顺序映像，指数据元素之间的逻辑关系借助对应结点的存储单元的相对位置关系来表示，也就是说，逻辑关系相邻的数据元素用物理位置相邻的结点顺序表示；而非顺序映象指数据元素之间的逻辑关系借助对应结点中的链域(表示结点存储单元地址的指针)中的数据来具体描述，与各结点的实际存储位置无关。

由上可知，数据的逻辑结构中描述数据元素之间的关系与数据存储结构中结点的物理位置之间虽然没有必然的对应关系，但在逻辑结构基础上定义的基本操作则需要依赖于数据的存储结构，因此，研究数据结构，一定涉及有关的基本操作及实现方法。

3. 基本操作及实现

对于任何一种数据结构，当其包含的数据元素之间的关系确定以后，对应的存储结构也可以根据具体问题的需要而确定。若求解过程需要涉及结构中元素的值或存储位置的改变、元素的增加或删除等，这些对原来存储结构的修改行为，就是所谓的基本操作。通常，在定义某种数据结构时，所对应的各种基本操作也同时予以定义；将它表示成物理结构时，也同时给出对应的基本操作函数。

因而，完整的数据结构概念可认为是由数据的逻辑结构、存储(物理)结构及基本操作集三个部分组成，可用以下形式加以描述：

$$\text{Data\_Structure} = (D, S, P) \tag{1-5}$$

其中，D 是数据元素的有限集合；S 是 D 集上关系的有限集合；P 是对 D 的基本操作集。

1.2.3 数据类型和抽象数据类型

1. 数据类型

数据类型(Data Type)是程序设计语言中对于常量、变量、表达式在程序执行期间所有可能取值的集合，以及在这些值上可以进行的操作。例如，在 C 语言中定义为 int 类型的整型变量的取值范围及其可以参与的加减乘除和取模等运算。

每一种程序设计语言都有一组固有的或基本的数据类型。对于 FORTRAN 语言，基本数据类型有 INTEGER、REAL、LOGICAL、CHARACTER 和 COMPLEX 等；对于 C 语言，则为 int、float、long int、double、char、enum 和指针等；许多现代的程序设计语言中允许定义新的类型，这些新的类型，既可以是对基本数据类型的限制，也可以是将若干基本类型的组合形式。

如果某个数据对象是仅由单值构成形式组成的类型，则称为原子类型，如整型、字符型等；相反，若是由一组值构成形式组成的类型，则称为结构类型或组合类型，它可进一步分解为若干成分，每一个成分既可以是原子类型，也可以是另一结构类型。在程序设计语言中提供的基本操作都是在原子类型上进行的。

从某种意义上说，数据类型可理解成在程序设计语言中已经实现的数据结构。

2. 抽象数据类型

抽象数据类型(Abstract Data Type，ADT)是一种数据类型及在这种数据类型上定义的一组操作。抽象数据类型不仅包括数据类型的定义，同时也为这种类型说明了一个有效的操作集合。虽然，从某种意义上看，抽象数据类型与数据类型在本质上相通。例如，在各种计算机上都具有“整型”数据类型，它所定义的数学特性是一致的，由于它在不同处理器中的实现方法可以有差别，因此它就是一种抽象数据类型。但是，在抽象数据类型层次分析中，数据类型的范畴更广，它不仅可以包括在处理器中已经定义并实现的数据类型，也可以包括用户自己设计的数据类型。“抽象”的意义就在于数据类型的数学抽象