高等学校计算机专业规划教材

数据结构

（AR版）

连远锋 主编
吴双元 李莉 朱丹丹 副主编

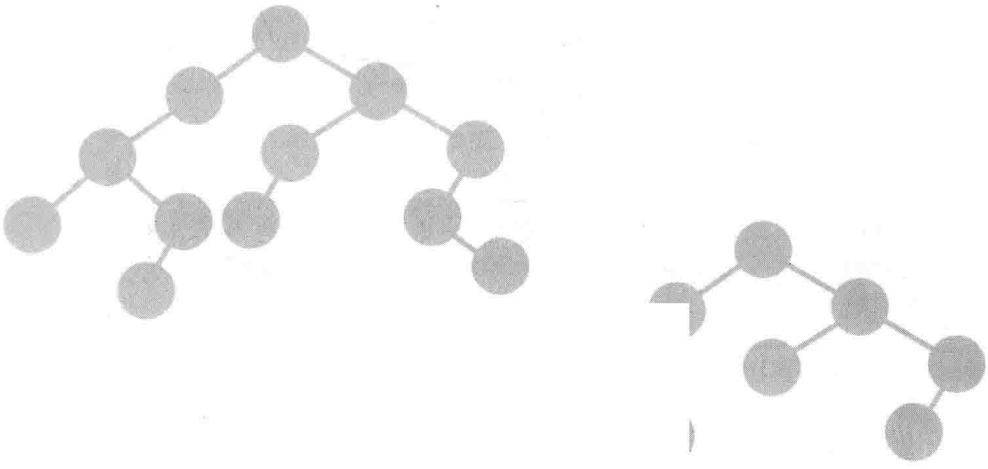
清华大学出版社

高等学校计算机专业规划教材

数据结构

（AR版）

连远锋 主编
吴双元 李莉 朱丹丹 副主编



清华大学出版社
北京

内 容 简 介

“数据结构”是计算机专业教学计划中的核心课程，也是计算机专业考研和水平等级考试的必考科目。本书共8章，内容包括绪论、线性结构、栈和队列、数组和字符串、树、图、查找和排序等。本书将数据结构知识点、实践操作和可视化交互展示融为一体，不仅强调数据结构的基本概念和程序实现，而且将增强现实(Augmented Reality，AR)技术用于数据结构教学过程，突破了传统教材的图文讲解方式，依托视觉、听觉等多模态模式来获取交互信息，弥补多媒体教学的不足，增强了学习效果。本书提供了涵盖所有知识点的AR可视化展示，给出了许多经典算法和大量C++代码，每章均附有小结和各类练习题，便于总结提高。

本书内容丰富，AR可视化效果逼真，知识点脉络清晰，代码实用性强，适合高等院校计算机专业和相关专业的本科生和研究生使用。

图书在版编目(CIP)数据

数据结构：AR版/连远锋主编. —北京：清华大学出版社，2019
(高等学校计算机专业规划教材)
ISBN 978-7-302-53524-9

Ⅰ. ①数… Ⅱ. ①连… Ⅲ. ①数据结构—高等学校—教材 Ⅳ. ①TP311.12

中国版本图书馆CIP数据核字(2019)第173232号

责任编辑：龙启铭
封面设计：何凤霞
责任校对：焦丽丽
责任印制：杨　艳

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**：24.75　　**字　　数**：572千字
版　　次：2019年10月第1版　　**印　　次**：2019年10月第1次印刷
定　　价：59.00元

产品编号：082535-01

前言

“数据结构”是计算机科学中的一门综合性专业基础课，是信息科学的核心课程，是研究非数值计算程序设计问题中计算机操作对象以及它们之间关系和操作的一门学科，也是计算机及相关专业硕士研究生入学考试的必考科目。通过对数据结构的学习，可以培养学生分析数据、组织数据的能力，对进一步学习相关专业的其他课程和开展计算机领域的各项工作，有着不可替代的作用。

“数据结构”课程的概念丰富，理论性、抽象性很强，本书是作者在总结长期教学经验基础上，通过合理规划教学内容编写而成。书中涵盖了高等学校数据结构教学实施方案和数据结构硕士研究生入学考试大纲中要求的全部知识点。全书分为8章，第1章为绪论，阐述数据、数据结构和算法等基本概念，特别强调算法分析的方法；第2～4章主要介绍线性表、栈和队列、数组和串等线性结构的定义、存储结构及相应运算方法和实现过程；第5章为树，介绍树和二叉树的概念与各种算法及其实现过程，重点介绍二叉树的各种遍历算法方法；第6章为图，介绍图的基本概念和图的各种算法及其实现过程；第7～8章讨论查找和排序的各种实现方法和性能分析。

随着虚拟技术的普及应用，将增强现实(AR)技术应用于数据结构学习是一种满足交互式和体验式的新教学方法。本书基于AR技术，将数据结构抽象化教学案例与手机AR相结合，构建虚实交互的学习环境。实践表明，“AR＋数据结构”有助于培养学生的学习兴趣，进而激发学生的创新能力和应用能力。基于手机AR的数据结构可视化案例可以在操作对象上叠加辅助信息，引导学生完成案例学习，观察操作结果，这对于培养学生的发散思维能力起到重要作用，可以有效解决当前时空受限、教学成效不突出等问题。

为了方便学生学习和上机实验，我们还出版了与本教程配套的《基于MFC的可视化数据结构》实验指导教材，两者构成一个完成的教学系列。本书给出的所有算法和程序均在Visual C++ 2015环境下调试通过。同时，本书提供了全面丰富的教学资源，其中包括教学PPT，源程序代码和增强现实APP。

在本书编写过程中,得到了所在学院的同事的热心帮助和支持,参加本书编写、程序调试、习题收集、内容审校等工作的老师有吴双元、李莉、朱丹丹、范江波、金洲、张建兵、朱丽萍、王智广等,在此向他们表示衷心的感谢!

由于作者水平有限,本书仍可能存在错误和不足之处,敬请读者批评指正。

编　者

2019年9月

增强现实APP

目录

第1章 绪 论

1.1 数据结构的重要性

通常，人们使用计算机解决问题所遵循的步骤是“问题提出→抽象建模→解决方案→编程实现”。由问题提出到抽象建模需要分析问题，抽象建模就是对问题进行整理和抽象，建立具体的数据模型。其中最重要的是两个模型：一个是处理模型，描述问题求解的流程；另一个是数据模型，描述问题所涉及的数据以及数据之间的关系，即数据结构。解决方案是根据已建立的模型，提出求解的算法，并基于算法设计出适当的数据结构和程序框架。编程实现将程序框架用某种程序设计语言翻译成计算机可执行的程序。

例如，对高层建筑进行结构分析，目的是了解高层建筑在外来载荷下的应力应变情况。第一步，考察建筑设计方案的可行性，这就是理解问题。第二步，根据地质、水文、气流以及建筑材料的特性，建立建筑结构的数学模型。为此需借助有限元法，将整个建筑划分成网格，在每个网格结点上列出变分方程，将变分方程进行一系列变换，最后转化为包括成百上千个方程的大型线性方程组。第三步，这个方程组是一个大型稀疏系数线性方程组，其系数矩阵是一个对角块矩阵，可考虑使用分块处理的高斯消去法求解，避免存储大量的零元素。第四步，用C++语言或Java语言定义相应算法和数据结构，并在计算机上实现。上述问题属于数值计算问题。此外，还有一类属于非数值计算的问题，例如图书管理系统的开发。假设某学期为计算机系的M名本科生设置了N门可选课程，每门课程对应P本参考书。这样，学生和图书之间构成了多对多的关系，如图1.1(a)所示。在这种情况下，如果引入一个交互实体“课程”，学生和课程之间的关系就转化为两个一对多的关系，如图1.1(b)所示。

为解决此问题，第一步，要理解需求。每学期开学前，为计算机系的学生设置可选课程的列表，学生在开学的前两周可以选课和退选。另外，每门课程有P本参考书。其次，分析建模。其中，处理模型包括选课的流程和退选的流程，数据模型包括“学生”“课程”“图书”实体的构成和它们之间关系的描述。第三步，设计解决方案，包括设计各个数据实体的数据结构和存储映像、相应操作的实现逻辑、用户界面的显示和交互流程。第四步，用C++语言或Java语言实现菜单树，数据的输入、输出，以及窗口的控制和切换等。

由以上事例可知，问题的最终解决取决于计算机程序，而程序的质量又取决于算法的选择和数据的组织方式。因此，学习数据结构的目的就是为了编写高质量的程序。

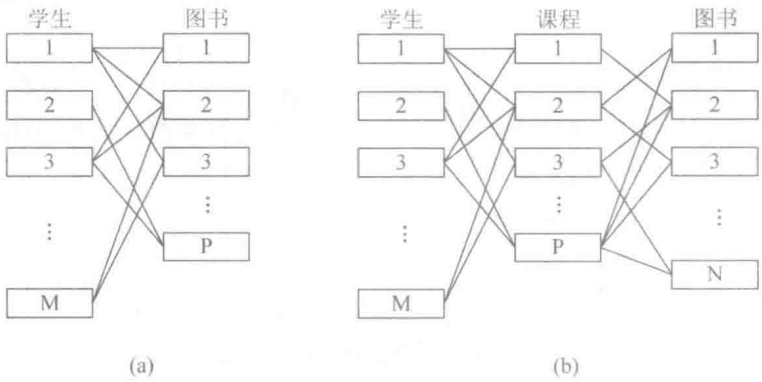


图 1.1 课程与学生之间的两种联系

1.2 数据结构基本术语

(1) 数据 (data)是信息在计算机程序中的表示形式,是描述客观事物的数、字符以及所有能输入计算机中并被计算机程序识别和处理的符号的集合。数据可分为两大类,一类是数值型数据,包括整数、浮点数、复数、双精度数等;另一类是非数值型数据,主要包括字符和字符串,以及文字、图形、图像、语音等。

(2) 数据项(data item)是构成数据元素不可分割的最小单位(称为原子)。数据元素中的数据项可以分为两种,一种称为原子项,如学生的性别、年龄等,这些数据项是在数据处理时不能再分割的最小单位;另一种称为组合项,如学生的成绩,它可以再划分为物理成绩、化学成绩等更小的项。

(3) 数据元素 (data element)是数据的基本单位。例如,考生名册中的每个学生记录、字符串中的每一个字符、数组中的每一个数组单元都是数据元素。不同场合下数据元素可以有别名,如元素、记录、结点、表项等。

(4) 数据对象(data object)是指具有一定关系的相同性质的数据元素的集合,数据对象的声明不仅包含属性,还包含定义在属性上的操作。

(5) 数据类型(data type)是一组值的集合以及定义在这个值集合上的一组操作的总称。数据类型可分为基本数据类型和结构数据类型两种。基本数据类型中的每个数据元素均为不可再分割的整体,如一个整数、浮点数、字符、指针、枚举量等,故称为原子类型。结构数据类型由基本数据类型或子结构类型按照一定的规则组合而成。例如,在 C++ 程序设计语言中,不仅规定了常用的基本数据类型,而且提供了一系列构造组合类型(如数组型、构造型、文件型等)的规则,方便用户定义基于应用问题所使用的数据类型。数据类型和数据结构是什么关系呢? 通常认为,一些基本数据类型如 char、int、float、double、string 等,是程序设计语言已经实现了的、供开发人员直接使用的数据结构。而结构数据类型是开发人员用程序设计语言表示的数据结构的存储映像。数据对象和数据结构又是什么关系呢? 从对象基本概念出发,数据对象不但包含数据结构,还定义了数据结构的存

储表示和施加于其上的运算。

(6) 抽象数据类型(abstract data type,ADT)是指数据的集合以及定义在该集合上的一组操作的总称。ADT 可以理解为对数据类型做进一步抽象。这里抽象是指抽取反映问题本质的特征,忽略具体的细节。

抽象数据类型的概念是 Liskov 和 Guttay 于 20 世纪 70 年代中期提出的,可以用三元组(D,S,P)来表示,其中 D 为数据对象,S 是定义在 D 上的关系集合,P 是定义在 D 的基本操作集,形式如下:

```
ADT 抽象数据类型名 {
    数据对象:<数据对象的定义>
    数据关系:<数据关系的定义>
    基本操作:<数据操作的定义>
        基本操作名(参数表)
        初始条件:<初始条件描述>
        操作结果:<操作结果描述>
} ADT 抽象数据类型名
```

抽象数据类型实现了封装和信息隐藏,是独立于具体的计算机的。抽象数据类型与面向对象程序设计语言中类的概念相对应,可以认为类是抽象数据类型的一个代码实体,抽象数据类型中的实现对应于类方法的接口,可以通过已有的数据类型(如整型、实型、字符型等)来表示和实现。

例如,自然数(Natural Number)的抽象数据类型定义如下:

```
ADT Natural_Number {
    数据对象: 一个整数的有序子集合,起始于 0,结束于计算机能表示的最大整数 MAXINT
    数据关系: 同属于自然数集合
    数据操作: 对于所有的 x,y∈Natural_Number,+、-、<、==、=等都是可用的操作
    Zero():NaturalNumber
        初始条件: 无
        操作结果: 返回 0
    IsZero(x):Boolean
        初始条件: 自然数 x 已存在
        操作结果: 如果(x==0) 则返回 True,否则返回 False
    Add(x,y):NaturalNumber
        初始条件: 自然数 x,y 已存在
        操作结果: 如果(x+y<=MAXINT)则返回 x+y,否则返回 MAXINT
    Equal(x,y):Boolean
        初始条件: 自然数 x,y 已存在
        操作结果: 如果(x==y)则返回 1,否则返回 0
    Successor(x):NaturalNumber
        操作结果: 如果(x==MAXINT)则返回 x,否则返回 x+1
    Subtract(x,y):NaturalNumber
        初始条件: 自然数 x,y 已存在
        操作结果: 如果(x<y)则返回 0,否则返回 x-y
} ADT Natural_Number
```

抽象数据类型为数据类型的定义和实现提供了两个视图，即外部视图和内部视图。外部视图包括抽象数据类型名称、数据对象说明及可供用户使用的操作集合，其目的是为用户提供使用抽象数据类型的“说明书”。内部视图包括数据对象的存储结构定义和基于存储结构的操作实现。为了提高软件的复用性，软件系统的框架应该基于数据而不是基于操作之上。因为软件定义的操作通常会随着用户需求的变化而不断更新，相对而言，系统定义的数据对象则较为稳定。如果软件系统将对象作为基本构件进行搭建，在系统功能面临升级时，通常只需做局部改写即可满足需求。在这种情况下，软件系统的所有基本构件一定要符合“封装”和“信息隐藏”的规范，防止外部用户直接存取构件内部的数据和调用构件内部非公开的操作，而只能通过构件提供的指定外部调用操作来存取构件内部的数据。这种数据的组织访问方法体现了抽象数据类型的概念。

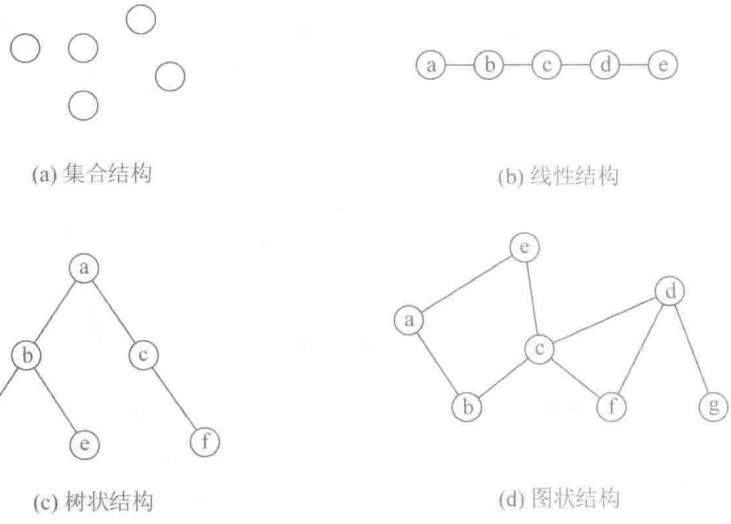
(7) 数据结构(data structure)是指相互之间存在一定关系的数据元素集合。需要注意的是，数据元素是讨论数据结构时涉及的最小数据单位，数据项一般不予以考虑。

1.3 数据的逻辑结构

数据的逻辑结构是指数据元素间逻辑关系的整体，即从逻辑关系上描述数据。数据结构在形式上可定义为一个二元组：

```
Data_Structure=(D,R)
```

其中，D 是数据元素的有限集合；R 是定义在 D 上的关系有限集合。数据的逻辑结构分为线性结构和非线性结构。线性表属于的线性结构；集合、树、图属于非线性结构。按照元素之间逻辑关系的不同，数据的逻辑结构分为 4 类，如图 1.2 所示。



(a) 集合结构　(b) 线性结构

(c) 树状结构　(d) 图状结构

图 1.2　不同数据结构中各数据元素间的联系

(1) 集合结构：数据元素属于同一集合，除此之外，数据元素之间没有联系。

(2) 线性结构：数据元素之间存在一对一的线性关系。

(3) 树状结构：数据元素之间存在一对多的关系。

(4) 图形结构：数据元素之间存在多对多的关系。

1.4 数据的存储结构

数据的存储结构是指数据的逻辑结构在计算机中的表示或实现，又称为物理结构，它包括数据元素的表示及关系的表示。存储结构依赖于计算机语言，通常有4种存储方式：

(1) 顺序存储结构：将逻辑上相邻的元素存放到物理位置上相邻的存储单元中，数据元素之间的逻辑关系由存储单元的邻接位置关系来体现。由此得到的存储结构称为顺序存储结构。通常，顺序存储结构可由程序设计语言中的一维数组来描述。

(2) 链接存储结构：该存储结构不要求逻辑上相邻的元素在物理位置上也相邻，元素之间的逻辑关系由指针指示。通常，链表存储结构可以借助程序设计语言中的指针类型来描述。

(3) 索引存储结构：该存储结构在存储元素信息的同时还建立索引表。索引表中的每一项称为索引项，索引项的形式是(关键字，地址)。关键字是能够标识一个元素的数据项。若每个元素在索引表中都有一个索引项，则该索引表称为**稠密索引**(Dense Index)；若一组相邻的元素在索引表中共有一个索引项，则该索引表称为**稀疏索引**(Sparse Index)。稠密索引中索引项中的地址指示元素所在的物理位置，稀疏索引中索引项中的地址指示一组相邻元素的起始存储位置。

(4) 散列存储结构：该存储结构根据元素的关键字，根据元素的关键字直接计算出该元素的存储地址，也称为Hash存储。

上述4种基本的存储结构既可以单独使用，也可以组合起来建立数据结构的存储结构。同一种逻辑结构采用不同的存储方法，可以得到不同的存储结构。在具体实现时选择何种存储结构来表示相应的逻辑结构，要视不同的需求而定，需着重考虑运算的时间和空间要求。

1.5 算法的基本概念

算法(Algorithm)一词来源于公元9世纪波斯数学家AI-Khowarizmi。通俗地说，算法是对特定问题求解步骤的具体描述，它是关于某种运算规则的集合。算法能够解决某个特定问题，需要满足以下5个特性：

(1) 确定性：算法的每一条指令必须具有确切的定义，无二义性。在任何条件下，对于相同的输入，算法仅有唯一的一条执行路径得到相同的输出。

(2) 可行性：算法中描述的操作均可以通过已经实现的基本运算的有限次执行得以实现。

(3) 有穷性：一个算法必须在执行有穷步之后结束，且每一步必须在有限时间内完成。

(4) 输入：一个算法具有零个或多个输入，这些输入取自特定的数据对象集合。

(5) 输出：一个算法具有一个或多个输出，这些输出同输入之间存在某种特定的关系。

算法与数据结构是密切结合、相辅相成的。解决某一特定类型问题的算法可以选用不同的数据结构，而方案选择的恰当与否直接影响算法的效率。反之亦然，一种数据结构的优劣由各种算法的执行来体现。

计算机程序是完成某一特定任务的一组计算机指令序列，是对一个算法使用某种程序设计语言的具体实现。算法可以用不同的编程语言来实现，它们遵循的逻辑步骤是相同的，但是算法不等于程序，算法必须可终止。这意味着不是所有的计算机程序都是算法。例如，操作系统是一个在无限循环中执行的程序而不是一个算法，然而可以将操作系统的各种任务看成一个单独的问题，每个问题由操作系统中的一个子程序通过有穷步之后得以解决。

1.6 算法分析与度量

在应用程序中，数据结构一旦确定，接下来就需要描述算法的设计和实现细节。通常，一个问题的表示可采用多种数据结构；类似地，一个问题的解决也可以有多个算法供选择。因此，需要对算法进行分析，以确定采用什么数据结构和算法最合适。

1.6.1 算法的评价标准

对具体问题而言，所选用的数据结构是否合适与算法直接相关。数据结构的可用性实际是由求解问题的算法来体现的。因此，对数据结构的分析本质上就是对算法的分析。度量一个算法的好坏，主要有以下几个标准：

(1) 正确性：要求算法能够正确地实现预定的功能和性能要求。这是最重要的标准，这要求算法的设计者对问题有正确的理解，能正确地、无歧义地描述算法，对于任何合法的输入，算法都会得出正确的结果。

(2) 鲁棒性：也称健壮性，要求算法具有对输入参数、打开文件、读文件记录、子程序调用状态进行自动检错、报错并通过与用户对话来纠错的功能。即当输入数据不合法时，算法也能够识别并做出处理，而不是产生错误动作或陷入瘫痪。

(3) 可读性：算法应当是可读的。这是阅读、理解和交流的需要。为了达到这一要求，算法的逻辑必须是清晰的、简单的和结构化的，所有的变量和函数的命名必须有实际含义。在算法中必须加入注释。

(4) 简单性：算法的简单性是指一个算法所采用数据结构和方法的简单程度。算法的简单性与算法的出错率直接相关。算法越简单，其出错率越低，可靠性越高。

(5) 高效性：算法的高效性也称为效率，包括算法执行的时间效率和空间效率。时间效率表示算法运行得有多快，空间效率表示算法需要多少额外的存储空间。下面将重点讨论算法的效率。