

21
世纪

高等学校计算机应用型本科规划教材精选



数据结构



辛运伟 主编
朱耀庭 主审



清华大学出版社

21 世纪高等学校计算机应用型本科规划教材精选

数 据 结 构

辛运伟 主编
朱耀庭 主审

清 华 大 学 出 版 社
北 京

内 容 简 介

本书是数据结构与算法设计的教科书,将数据结构与算法设计有机地结合起来,向读者系统介绍了数据结构的基本概念及主要的算法设计方法。

全书共分7章,第1章介绍了数据结构的基本概念及主要的数学方法,第2章至第7章分别介绍了线性表、栈和队列及数组,树和图等重要的数据结构及基本操作的实现过程,以及查找和排序等数据结构的相关知识。本书在内容讲授过程中辅以大量的实例,旨在帮助读者更好地理解概念并了解如何使用这些概念去解决实际问题。书中主要算法都用C++语言写出,并给出了详细的注解。

本书概念清楚,选材精练,叙述深入浅出,用了大量的例子和图表来说明基本概念和方法,直观易懂。每章后面都附有习题,读者可以通过练习复习来检验所学知识。本书可以作为大专院校计算机专业学生的教材,也可以作为广大计算机科学与工程领域从业人员的一本参考书。

本书的电子课件可到清华大学出版社网站(<http://www.tup.com.cn>)下载。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

数据结构/辛运伟主编. —北京:清华大学出版社,2010.6

(21世纪高等学校计算机应用型本科规划教材精选)

ISBN 978-7-302-22181-4

I. ①数… II. ①辛… 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(2010)第028912号

责任编辑:索梅 李玮琪

责任校对:焦丽丽

责任印制:何芊

出版发行:清华大学出版社

地 址:北京清华大学学研大厦A座

<http://www.tup.com.cn>

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

印 刷 者:北京宏伟双华印刷有限公司

装 订 者:北京国马印刷厂

经 销:全国新华书店

开 本:185×260 印 张:16 字 数:382千字

版 次:2010年6月第1版 印 次:2010年6月第1次印刷

印 数:1~4000

定 价:25.00元

产品编号:032925-01

21 世纪高等学校计算机应用型本科规划教材精选

编写委员会成员

(按姓氏笔画)

王慧芳 朱耀庭 孙富元

高福成 常守金



“教

育部财政部关于实施高等学校本科教学质量与教学改革工程的意见”(教高[2007]1号)指出:“提高高等教育质量,既是高等教育自身发展规律的需要,也是办好让人民满意的高等教育、提高学生就业能力和创业能力的需要”,特别强调“学生的实践能力和创新精神亟待加强”。同时要求将教材建设作为质量工程的重要建设内容之一,加强新教材和立体化教材的建设;鼓励教师编写新教材,为广大教师和学生提供优质教育资源。

“21世纪高等学校计算机应用型本科规划教材精选”就是在实施教育部质量工程的背景下,在清华大学出版社的大力支持下,面向应用型本科的教学需要,旨在建设一套突出应用能力培养的系列化、立体化教材。该系列教材包括各专业计算机公共基础课教材;包括计算机类专业,如计算机应用、软件工程、网络工程、数字媒体、数字影视动画、电子商务、信息管理等专业方向的计算机基础课、专业核心课、专业方向课和实践教学的教材。

应用型本科人才教育重点是面向应用、兼顾继续深造,力求将学生培养成为既具有较全面的理论基础和专业基础,同时也熟练掌握专业技能的人才。因此,本系列教材吸纳了多所院校应用型本科的丰富办学实践经验,依托母体校的强大教师资源,根据毕业生的社会需求、职业岗位需求,适当精选理论内容,强化专业基础、技术和技能训练,力求满足师生对教材的需求。

本丛书在遴选和组织教材内容时,围绕专业培养目标,从需求逆推内容,体现分阶段、按梯度进行基本能力→核心能力→职业技能的培养;力求突出实践性,实现教材和课程系列化、立体化的特色。

突出实践性。丛书编写以能力培养为导向,突出专业实践教学内容,为有关专业实习、课程设计、专业实践、毕业实践和毕业设计教学提供具体、翔实的实验设计,提供可操作性强的实验指导,完全适合“从实践到理论再到应用”、“任务驱动”的教学模式。

教材立体化。丛书提供配套的纸质教材、电子教案、习题、实验指导和案例,并且在清华大学出版社网站(<http://www.tup.com.cn>)提供及时更新的数字化教学资源,供师生学习与参考。



课程系列化。实验类课程均由“教程+实验指导+课程设计”三本教材构成一门课程的“课程包”，为教师教学、指导实验以及学生完成课程设计提供翔实、具体的指导和技术支持。

希望本丛书的出版能够满足国内对应用型本科学生的教学要求，并在大家的努力下，在使用中逐渐完善和发展，从而不断提高我国应用型本科人才的培养质量。

丛书编委会

2009年7月

前言

FOREWORD



数据结构历来是计算机科学与技术专业的核心课程之一。自 1968 年将数据结构列为一门独立的课程至今已经过去了 40 多年,但它在计算机专业课程体系中的地位以及它在计算学科的多个主要分支中的地位丝毫没有减弱,反而愈加重要了。随着计算机性能的不断提升,速度越来越快,人们对使用计算机求解问题的需求也越来越高,对高效率程序的渴望日益迫切。本书的目的是要教给学生良好的程序设计技巧和必要的算法分析方法,提高他们的程序设计能力和解决问题的能力。

目前,计算机应用已经涉及到人们活动的各个领域,高级程序设计语言和编程工具的推出及普遍使用,为各行各业的人员学习程序设计提供了可行的条件,现在为计算机编写程序已经不仅仅是计算机专业人员的工作。因此,数据结构与算法课程早已不仅仅是计算机专业独有的课程,本书是为所有需要学习计算机程序设计技术的读者编写的。

本书共分 7 章,全面介绍了解决从简单到复杂的各类应用问题的数据结构及其编程实现方法。第 1 章绪论,介绍了数据结构中用到的一些基本概念,也介绍了抽象数据类型及面向对象的概念。此外,作为预备知识,本章还简述了本书中将要用到的数学方法以及算法的概念和算法分析的基本方法。

第 2 章介绍了线性表的定义及基本操作的实现,分析了这些实现方法的时空复杂度,比较了静态及动态两种实现方式的特点。这些内容也为读者分析更复杂方法的时空复杂度奠定了基础。本章还介绍了线性表的一个重要应用,帮助读者更好地理解线性表,了解如何使用线性表来解决实际问题。

第 3 章介绍了两种受限的线性表,分别是栈和队列。栈和队列都属于线性表的范畴,但是它们的操作位置具有特殊性。与一般的线性表相比,它们的实现要简单许多。此外,本章还介绍了数组的概念。

第 4 章介绍了树的概念。重点介绍了树与二叉树的存储方式及二叉树相关操作的具体实现。读者需要好好理解层次结构与线性结构的差异。以二叉树为例,本章还介绍了遍历的概念,并实现了具体的遍历算法。在这一章中,读者除了要掌握这些基本概念和基本实现方法之外,更需要深入了解递归的概念并具备使用递归处理问题的能力。本章还介绍了二叉树与森林的对应关系和转换机制,以及哈夫曼树和哈夫曼编码的概念。

第 5 章介绍了图结构。图是比线性结构和树结构更复杂的一种数据结构,本章除介绍图的基本概念外,还介绍了图的两种存储方式,给出了两种存储方式下主要操作的实现方法。本章中还介绍了图的多个经典问题,包括图的遍历、图的拓扑排序、图的最小生成树、图的单源最短路径等。

第 6 章介绍了查找。查找是一种非常重要的操作,基于不同的结构有不同的查找方法。

本章介绍了基于数组存储结构的顺序查找方法及折半查找方法和基于树形存储结构的二叉排序树的查找方法及 B 树的查找方法,另外,还介绍了一种应用广泛的哈希查找方法。

第 7 章介绍了内部排序的概念,并给出了多种排序算法,分析了各排序算法的特点,讨论了各自的时空复杂度。

在本书各章的介绍中,既包含了数据结构的基本概念,也介绍了算法的设计与实现。实际上,数据结构与算法的设计是程序设计的核心,二者是密不可分的。掌握了初步的程序设计方法之后,面对实际的应用问题,最重要的就是学习如何选择和设计有效的数据结构和算法,因为编写程序的目的是解决问题。本书采用标准 C++ 语言作为全书的基本描述语言,书中对各段程序代码都尽可能地给出了说明,相信即使没有学习过 C++ 的读者,也能很快理解书中的内容。

本书的编者参阅了国内外经典著作和最新教材,结合编者多年讲授数据结构和算法分析课程的经验,力求在内容方面深入浅出,既先进严谨,又通俗易懂,希望能受到我国相关专业的老师和同学的欢迎。本书各章均使用实例来帮助读者深入理解概念,各章最后都给出习题,习题的类型分为三类,读者可以以此来检验自己的学习成果。

本书由辛运帏编写,侯林峰、于文平、王春韶、赵伟调试了大部分例程。在本书的编写过程中,一直得到南开大学陈有祺教授、朱耀庭教授、卢桂章教授、刘璟教授、周玉龙教授的鼓励和支持。在此对一直以来关心我们、帮助我们所有老师、朋友、同事表示诚挚的感谢。

虽然我们希望能够奉献出一本优秀的数据结构与算法教材,但限于水平和能力,书中难免有错误和不妥之处,敬请读者提出宝贵意见,携手共同提高我们的教学水平。

编者
于南开园



第 1 章 绪论	1
1.1 数据结构的基本概念和术语	1
1.2 抽象数据类型及面向对象概念	3
1.2.1 抽象数据类型.....	3
1.2.2 面向对象的概念.....	4
1.3 有关的预备知识	5
1.3.1 集合.....	5
1.3.2 递归.....	6
1.4 算法和算法分析	8
1.4.1 算法的基本概念.....	9
1.4.2 算法的评估和复杂性度量	10
本章小结	12
习题 1	13
第 2 章 线性表	14
2.1 线性表的定义和基本运算.....	14
2.1.1 线性表的定义	14
2.1.2 线性表的操作	16
2.2 线性表的实现.....	18
2.2.1 顺序存储结构	18
2.2.2 链式存储结构	22
2.2.3 两种基本实现方式的比较	30
2.2.4 循环链表	31
2.2.5 双向链表	32
2.3 线性表的应用.....	36
本章小结	40
习题 2	40
第 3 章 栈、队列和数组	44
3.1 栈.....	44
3.1.1 栈的定义	44

3.1.2	栈的实现	45
3.2	队列	50
3.2.1	队列的定义及基本运算	50
3.2.2	队列的实现	51
3.3	数组	57
3.3.1	数组的抽象数据类型	57
3.3.2	数组的存储方式	58
3.3.3	特殊数组	60
3.3.4	数组的应用实例	65
	本章小结	68
	习题 3	68
第 4 章	树与二叉树	71
4.1	树	71
4.1.1	树的基本概念	71
4.1.2	树的抽象数据类型	74
4.2	二叉树	74
4.2.1	二叉树的定义及其主要特性	75
4.2.2	二叉树的实现	77
4.2.3	二叉树的遍历	81
4.3	树与森林	85
4.3.1	树的存储结构	85
4.3.2	森林与二叉树的转换	88
4.3.3	树和森林的遍历	90
4.4	哈夫曼树和哈夫曼编码	91
	本章小结	103
	习题 4	103
第 5 章	图结构	107
5.1	图的基本概念	107
5.1.1	图的含义	107
5.1.2	图的抽象数据类型	111
5.2	图的存储结构	112
5.2.1	邻接矩阵	112
5.2.2	邻接表	114
5.2.3	图的实现	115
5.3	图的遍历及求图的连通分量	120
5.3.1	深度优先搜索	122
5.3.2	广度优先搜索	125

5.3.3 无向图的连通分量·····	128
5.4 有向无环图及拓扑排序·····	130
5.4.1 有向无环图·····	130
5.4.2 拓扑排序·····	131
5.5 生成树和最小(代价)生成树·····	136
5.5.1 生成树·····	136
5.5.2 最小(代价)生成树·····	137
5.6 单源最短路径·····	147
本章小结·····	151
习题 5·····	152
第 6 章 查找 ·····	155
6.1 查找的基本概念·····	155
6.2 顺序表的查找·····	156
6.2.1 顺序查找方法·····	157
6.2.2 折半查找方法·····	162
6.3 树形结构的查找·····	166
6.3.1 二叉排序树·····	166
6.3.2 B 树·····	175
6.4 哈希表及其查找·····	179
6.4.1 哈希的概念·····	179
6.4.2 哈希函数的构造方法·····	181
6.4.3 处理冲突的几种方法·····	183
6.4.4 哈希表的查找及其效率分析·····	185
本章小结·····	186
习题 6·····	187
第 7 章 内部排序 ·····	190
7.1 排序的基本概念·····	190
7.2 插入排序·····	193
7.2.1 直接插入排序·····	193
7.2.2 折半插入排序·····	196
7.2.3 希尔排序·····	197
7.3 交换排序·····	200
7.3.1 起泡排序·····	201
7.3.2 快速排序·····	203
7.4 选择排序·····	208
7.4.1 简单选择排序·····	208
7.4.2 堆排序·····	210



7.5 归并排序	216
7.5.1 两个有序序列的归并操作	216
7.5.2 归并排序简介	217
7.6 分配排序和基数排序	220
7.7 有关内部排序算法的比较	224
本章小结	225
习题 7	225
数据结构综合测试题	229
数据结构期末考试试卷一	232
数据结构期末考试试卷二	235
数据结构期末考试试卷三	238
参考文献	240

第1章

绪 论

数据结构是计算机专业的必修课程之一,在课程体系中占有非常重要的地位。该课程起着承上启下的作用,是学习计算机其他专业课程的基础。

自20世纪40年代计算机问世之后,计算机的处理能力越来越强,处理数据的类型也越来越复杂,这就需要有更强有力的数据表示及处理方法。

高级程序设计语言都提供了基本数据类型,有些还提供了让程序员构造更高级更复杂类型的机制。随着需求的深入及时间的推移,这些技术逐渐成熟,也逐渐从程序设计语言中脱离出来,形成了一个完整的独立体系。这就是最初的数据结构思想,数据结构课程也应运而生。从1968年起,高等院校中陆续开设了数据结构课程。

本章将介绍数据结构的基本概念和有关的预备知识,这些都是程序设计过程中必不可少的内容。在此基础上,还将介绍算法设计技术及时间复杂度的概念。

1.1 数据结构的基本概念和术语

最初的计算机只能处理二进制数,这种数据根本谈不上有什么结构。随着技术的发展,计算机可以处理十进制的整数和小数,例如,程序中可以定义一个整型变量用来保存数字3,定义一个实型变量保存圆周率,可以使用两个实型的量组合起来表示一个复数(一个用来表示复数的实部,另一个用来表示复数的虚部),甚至程序中还出现了数组和记录。不论是由两个实型量表示的复数,还是数组或记录,这些都可以看做是由基本数据组成的复杂数据,由此有了“结构化”的数据的概念,也就是出现了“结构”的雏形。

实际上,最初的结构称为信息结构,后来才改用数据结构这个术语。1968年,由美国计算机协会(ACM)颁发了建议性的计算机教学计划,计划中规定把数据结构作为一门独立的课程。这在世界上还是第一次。同年,世界著名的计算机科学家、美国的D. E. Knuth教授的巨著《计算机程序设计艺术》第一卷《基本算法》出版,该书全面系统地论述了数据的逻辑结构和存储结构,并且给出了各种典型的算法,为数据结构奠定了理论基础。而著名的计算机科学家N. Wirth编写的《算法+数据结构=程序》一书则明确指出在程序设计中,数据结构与算法同等重要。

20世纪80年代以后出现了抽象数据类型概念,将数据和对数据进行的操作合为一体,而将操作的具体实现和它的定义分离开来,特别是在面向对象程序设计方法论中,提出类的

概念,从而将数据结构的理论和实践提高到一个新的水平。

那么,什么是数据呢?一般来说,凡是能够输入到计算机中并能被计算机处理的一切对象都可以称为数据。所谓数据绝不能仅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,一个用某种程序设计语言编写的源程序、文稿、地图、照片、歌曲、视频等等,都可以视为“数据”。今后随着计算机的发展,计算机能够处理的对象和应用领域不断扩充,还将不断扩大数据的范畴。

既然输入到计算机内部的一切对象都是数据,那么数据就可能是各种各样的,有复杂的,也有简单的,复杂的数据往往是由简单的数据构成的。构成数据的基本单位称为数据元素。当然,数据元素可大可小,大到可以是一幅地图、一本书、一部电影,小到可以是一个字符,甚至是计算机中的一位(bit)。

数据元素还可以细分为数据项,这是最基本的单位。假定计算机中保存了100名学生的信息,每位学生的信息包括学号、姓名、所学各门课程的成绩等,使用一条记录来保存。这条记录就可以看做是数据元素,而记录中的学号、姓名、各科成绩等都可以看做是数据项。全部学生的所有信息构成数据。

再比如,图书馆管理系统中保存了该图书馆中所有图书的信息和借阅情况,构成图书信息书名、作者名、出版社名、书号和出版日期等信息都可以看做是数据项,由一本书的信息组成的一条记录即是数据元素。

有了这些基本概念后,再来看看什么是数据结构。

数据结构是一门课程的名字。此外,还使用数据结构表明一个数据集合及对该数据集合进行的操作总和。

比如,现有整数集合 I ,对 I 可进行的操作包括加法、减法、乘法、整除、取模共五种,则集合 I 加上这五种操作共同组成一种数据结构,可以命名为DS1。再比如,学生记录集 R ,对该集合可以进行的操作有添加一个学生的记录、删除一个学生的记录、查找一个学生的记录等。集合 R 加上相应的操作构成另一个数据结构,比如称为DS2。

数据集合构成一种类型,类型即是一组值的集合。例如整数集合构成整型,浮点数集合构成实型。对数据的操作与数据本身有很大的关联性,数据集合确定下来后,在集合上能够进行的操作类型也就基本确定下来了。例如,浮点数集合 F 中,可以进行加法、减法、乘法、除法等操作,但不能进行两个浮点数的精确相等比较。因为受计算机字长的限制,计算机内保存的浮点数可能与实际值之间存在舍入误差,从而导致比较操作的不准确性。一般地,大多数语言不会对浮点数定义取模操作,也不会定义移位操作。再比如,阶乘只对自然数有定义,对小数没有定义。

这样,就有了数据类型的概念。所谓数据类型是指一个类型以及定义在这个类型上的一组操作。数据类型加上操作的具体实现组成数据结构。常见的数据结构包括四大类,分别是集合、线性结构、树结构和图形结构。

集合是数学中最基本的概念之一。集合中各元素没有次序关系,对集合进行的操作包括将某元素加入集合、从集合中删除某元素、判定某元素是否属于集合等。

线性结构是指数据元素之间存在着依次排列的先后次序关系的结构,也就是说每个元素都对应着一个唯一的次序,这个次序决定着元素的位置,元素之间具有全序的关系。

树结构是一种层次结构,其中的元素按层排列。每个元素可以对应最多一个的上层元

素,同时可以有数目不等的下一层元素。而图形结构是一种网状结构,树结构可以看做是图的特例。树结构和图形结构都是比线性结构更复杂的结构。

由于集合的操作没有超出线性结构操作的范围,一般地,不单独讨论集合。当需要使用集合来描述操作对象时,可以借用线性表来完成。

图 1-1 形象地表示了集合、线性结构、树结构及图形结构中数据元素之间的关系。

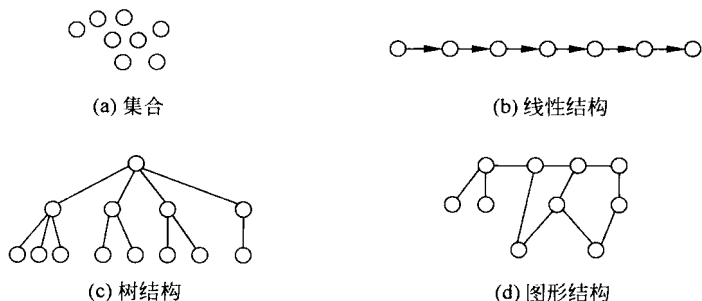


图 1-1 四类基本结构示意图

1.2 抽象数据类型及面向对象概念

抽象数据类型是指基于一个逻辑类型的数据类型以及这个类型上的一组操作,是不涉及具体实现的抽象概念。在面向对象的程序设计方法中,抽象数据类型与类的定义非常相近。本节介绍这些基本概念。

1.2.1 抽象数据类型

前面已经介绍数据结构的概念了。实际上,当命名一种数据结构时,都是先给出这种结构所涉及的数据集合,然后再讨论能够施加于这个数据集合的操作,最后是使用一种具体的程序设计语言来实现这些操作。

这个过程可以看做是两个阶段:一是定义或称为描述阶段,说明这个类型的名字是什么,描述数据元素的类型,同时指明能够施加的操作都有哪些,各个操作所带的参数是什么,返回什么类型,操作完成的功能是什么;二是实现阶段,即具体实现这些操作。

定义阶段只涉及数据集合和相关操作的描述,并没有涉及具体的实现细节,更没有使用某种程序设计语言写出相关的代码,所以这样定义的类型也称为抽象数据类型。可以把数据集合抽象为一个数学模型,所谓抽象数据类型(Abstract Data Type, ADT)就是数学模型及定义在其上的操作的整合。

总之,抽象数据类型是一组数据和允许施加于数据的具体操作。抽象数据类型中并不考虑对操作的具体实现细节,也不限定使用哪种程序设计语言来实现这些操作。

下面定义一个抽象数据类型。

考虑在银行某个柜台前排队等待办理业务的顾客。当有新顾客到来时,他必须排到队伍的最后。当前面某位顾客办理完业务后并从队伍中退出,下一位顾客排到队头并开始办理业务。这就是一个队列数学模型。由此可以定义一个抽象数据类型,称为队列。可以进行的操作包括:向队列中添加一个元素(新顾客排到队尾)、从队列中删除一个元素(办完业

务离开)等。

定义 1-1 元素类型为 T 的队列,由 T 的有限序列组成,对队列可以进行以下的操作:

- (1) 创建一个空队列;
- (2) 判定队列是否为空;
- (3) 入队列操作,即当队列不满时将一个新元素加入到队列的最后;
- (4) 出队列操作,即当队列不空时删除队列最前面的元素。

这里之所以称为抽象数据类型,是指定义中仅给出了数学模型的名称及操作的含义,并没有使用具体的程序设计语言来实现相关的操作。

再看另一个例子。定义一个有序的整数集合,集合中按整数的大小排好序,这个集合可以称为有序表。有序表也是一个抽象数据类型,它应包括下列操作:

- (1) 将新元素(整数)插入到有序表中,使之成为新的有序表;
- (2) 按有序表元素的大小依次输出各元素;
- (3) 查找有序表中的特定值,若找到则返回 TRUE,否则返回 FALSE;
- (4) 删除有序表中特定位置的元素;
- (5) 判定一个有序表是否为空。

当使用具体的方式实现了 ADT 后,就成为一种真正的数据类型。所以数据结构可以看做是 ADT 的物理实现。

每个数据类型都有逻辑形式和物理形式。所谓逻辑形式是指,每个数据类型都使用 ADT 来定义,也就是给出它的具体描述。而它的物理形式是说要使用数据结构来具体实现这个数据类型。涉及物理形式时,一要确定它使用什么样的存储方式,二要确定在这样的存储方式下各种操作是如何实现的。

存储方式一般有两类。一类是顺序存储方式,即使用数组来存储;另一类是链式存储方式,即借助指针来存储。根据具体的问题,有时需要将这两种方式结合起来,成为混合式的存储方式。

1.2.2 面向对象的概念

目前主流的程序设计方法是面向对象(Object-Oriented)程序设计方法,它特别适合于设计大型的软件系统。随着时间的推移,这项技术已经非常成熟。在此之前流行的设计方法是面向过程的设计方法。

程序设计中,离不开数据及对数据的操作。程序就是要解决现实世界中的问题,现实世界中的一个实体、一个概念或是一个过程在程序中是如何表示的呢?又是如何对它进行操作的?在传统的面向过程的设计方法中,数据类型及对它的操作是分离的。程序中往往定义了很多的变量来表示数据,同时设计了众多的方法对这些数据进行无限制的操作。这些方法只依靠名字来区分,在程序的任何地方都可以按名来调用,因此调用关系比较混乱,对数据的操作也没有办法控制。

面向对象的程序设计机制将数据及对数据的操作捆绑起来,形成一个统一体。这个统一体就是类。面向对象的程序设计过程就是类的设计过程。程序员可以定义数据和相关的方法,对特定数据的操作只通过相关的方法来实现。这样的处理方式解决了面向过程的设计方法中的很多问题。

可以很容易地将类的定义与抽象数据类型对应起来。实际上,它们可以看做是描述同一事物的两种不同术语。

1.3 有关的预备知识

学习数据结构课程之前,需要具备初步的程序设计能力,并掌握高等数学的基本知识。本小节介绍数据结构课程中用到的相关知识。

1.3.1 集合

集合是最基本的数学概念之一,也是四类基本数据结构之一。

1. 集合的概念

在数学中,并没有给集合下一个非常确切的定义,只形象地描述说,一群无重复的对象的全体称为集合。这些对象称为集合中的元素。数学中并没有规定集合中元素的类型,可以一致,也可以不完全一致。数据结构课程中,集合常常用线性表来表示,并要求元素的类型是统一的。元素类型不完全一致的集合,可以使用广义表来表示,广义表不在本书的讨论范围内。

描述集合的方法通常有两种。当集合中元素的个数较少时,可以一一列举出这些元素,并使用一对大括号括住全部元素,各元素间以逗号做分隔符。称这种描述集合的方法为穷举法或列举法。

【例 1-1】 使用穷举法表示集合。

$W = \{\text{星期一, 星期二, 星期三, 星期四, 星期五, 星期六, 星期日}\}$

$C = \{\text{红色, 绿色, 蓝色}\}$

在例 1-1 中,第 1 个集合 W 表示一个星期中的七天,第 2 个集合表示三原色。

如果集合中元素的个数比较多时,当然不方便使用穷举法一一列举各个元素。此时可以使用另外一种表示方法,即描述方法。

描述方法的一般格式是: $\{x | P(x)\}$,其中 x 表示该集合中的任一元素, $P(x)$ 是一个谓词,它对 x 进行限定。这个格式表示由满足 $P(x)$ 的一切 x 构成的集合。 $P(x)$ 可以使用某种描述语言来表示,如自然语言或是数学语言。

【例 1-2】 使用描述法表示集合。

偶数 $S1 = \{n | n \text{ 是偶数}\}$

偶数 $S2 = \{n | n \% 2 = 0\}$

例 1-2 中的两个示例表示的都是偶数集合,第 1 个是使用自然语言描述的,第 2 个是使用数学语言描述的。

2. 集合的操作

集合上的操作很简单,这些操作及记号都沿袭了数学上的表示方法。

(1) 属于。如果 x 是集合 S 中的一个元素,则称 x 属于 S ,记作 $x \in S$; 否则,称 x 不属于 S ,记作 $x \notin S$ 。