

世界著名计算机教材精选

PEARSON
Prentice
Hall

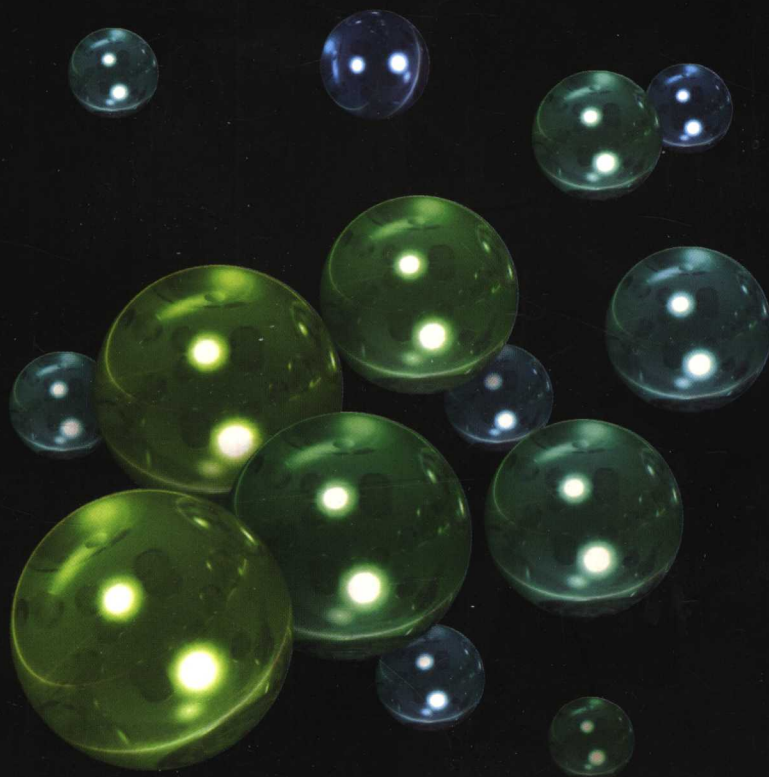
数据结构与抽象

Java语言版

Frank M. Carrano
Walter Savitch

著

严蔚敏 袁 昱 朱薇薇 译



DATA STRUCTURES AND ABSTRACTIONS WITH JAVA

清华大学出版社



世界著名计算机教材精选

数据结构与抽象 (Java 语言版)

Frank M. Carrano, Walter Savitch 著

严蔚敏 袁 昱 朱薇薇 译



清华大学出版社
北京

Simplified Chinese edition copyright © 2004 by PEARSON EDUCATION ASIA LIMITED and TSINGHUA UNIVERSITY PRESS.

Original English language title from Proprietor's edition of the Work.

Original English language title: Data Structures and Abstractions with Java by Frank M.Carrano and Walter Savitch, Copyright © 2003

EISBN: 0-13-07489-0

All Rights Reserved.

Published by arrangement with the original publisher, Pearson Education, Inc., publishing as prentice-Hall, Inc..

This edition is authorized for sale only in the People's Republic of China (excluding the Special Administrative Region of Hong Kong and Macao).

本书中文简体翻译版由 Prentice-Hall, Inc.授权给清华大学出版社在中国境内（不包括中国香港、澳门特别行政区）出版发行。

北京市版权局著作权合同登记号 图字：01-2003-1792 号

版权所有，翻印必究。举报电话：010-62782989 13901104297 13801310933

本书封面贴有 Pearson Education（培生教育出版集团）激光防伪标签，无标签者不得销售。

图书在版编目（CIP）数据

数据结构与抽象（Java 语言版）/ 卡兰罗（Carrano, F. M.），萨维切（Savitch, W.）著；严蔚敏等译。
—北京：清华大学出版社，2004.11

（世界著名计算机教材精选）

ISBN 7-302-09375-X

I. 数… II. ①卡… ②萨… ③严… III. ①数据结构—教材 ②JAVA 语言—程序设计—教材
IV. TP311.12 ②TP312

中国版本图书馆 CIP 数据核字（2004）第 089258 号

出版者：清华大学出版社

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

社总机：010-62770175

责任编辑：龙啟铭

印刷者：清华大学印刷厂

装订者：北京鑫海金澳胶印有限公司

发行者：新华书店总店北京发行所

开本：185×260 印张：50.75 字数：1262 千字

版次：2004 年 11 月第 1 版 2004 年 11 月第 1 次印刷

书号：ISBN 7-302-09375-X/TP·6546

印数：1~3000

定价：89.00 元

地址：北京清华大学学研大厦

邮编：100084

客户服务：010-62776969

本书如存在文字不清、漏印以及缺页、倒页、脱页等印装质量问题，请与清华大学出版社出版部联系调换。联系电话：（010）62770175-3103 或（010）62795704

译者序

“数据结构”是计算机专业的基础与核心课程之一，也是从事软件开发必不可少的入门和常用知识。程序编写得好不好，很大程度上取决于编程者对数据结构是否熟练地掌握和恰当地运用。由于它不仅重要，而且易学难精，“数据结构”一直被列入相关专业的研究生入学考试和相关行业的公司招聘考试的重点考查范围。

由于“数据结构”这门课程本身的特点，它必须依托于一种程序设计语言才能讲授，否则就成了空中楼阁、纸上谈兵。因此，尽管从抽象和逻辑的角度看来都大同小异，按照所依托的程序设计语言可以把“数据结构”的教材分为不同的版本——诸如 Pascal 版、C 版、C++ 版以及 Java 版。除了由于程序设计语言的不同特性而导致的程序实现上的差异，不同版本的“数据结构”教材所讲述的主要内容并无本质区别。因此，初学者可以根据自己已经掌握的或者将作为主要使用的程序设计语言选择相应版本的“数据结构”教材来学习。将来如果换用另一种程序设计语言，也不需要重新学习另一个版本的“数据结构”教材，只需将其作为参考，查阅同样的数据结构是如何用另一种语言实现的即可。这也是为什么不同版本的“数据结构”教材都有其存在的意义。

近年来，Java 越来越成为一种“热门”的语言。它不但是目前业界进行软件开发的主要语言之一，而且被国外不少大学选用为向计算机专业的学生传授的第一种程序设计语言。这固然除了 Java 语言本身的优点之外，也有商业利益和教育观念上的考虑，在此不作赘述。但本书并不是国内第一本基于 Java 的“数据结构”教材。之所以选择本书译介给国内读者，主要是因为本书有以下几个独特的优点：

- 总的来说，数组和链表是数据结构的两种基本的实现方式。基于这两种方式，可以实现线性表、栈、队列、二叉树、树、图等等从简单到复杂的数据结构及其各种变体。本书的内容涵盖了以上方面，以及对算法和算法复杂性的一些初步的介绍，从而构成了一本完整的数据结构教材。但本书不限于此，还在前三章和附录 A 到 D 中全面地介绍了 Java 语言，在某些方面其详细程度甚至超过专门讲述 Java 语言的书籍。因此，即使未学过 Java 语言的读者，也可以将本书作为起点，在学习 Java 语言的同时学习数据结构。
- 本书在编写过程中特别考虑到了面向对象程序设计（OOP）的思想与 Java 语言的特性。它不是从基于另一种程序设计语言的数据结构教材简单地“改编”而来的，因此在数据结构的实现上更加“地道”地运用了 Java 语言，并且自始至终强调以面向对象的方式来思考、分析和解决问题。
- 本书在内容编排上，将每一种数据结构的说明与实现分开来，全书被划分为 30 个相对较短的章，从而为读者提供了阅读顺序的机动性。读者可以在阅读了一个数据结构的说明之后紧接着就学习它是如何用 Java 语言实现的，也可以首先通读各种数据结构的说明，然后才考虑如何实现它们。

- 本书论述详细，深入浅出，含有大量的图示、应用示例和源程序，每章还配有自测题以及答案（在附录 E 中），不但可作为大专院校相关专业本科和研究生的教材，也非常适宜自学，以及作为开发软件时案头常备的参考书。

由于一些术语在国内存在不止一种译法，本书在翻译时只能尽量选用沿袭较广的译法，如果读者已经熟悉的是另一种译法，阅读时可能会感到不便，在此表示抱歉。书中的术语在第一次出现时均列出了相应的英文，供读者对照辨别。

本书的翻译出版是一个专家团队共同努力的结果。参加本书翻译工作的有严蔚敏、袁昱、朱薇薇、苏辛一、韩丹。严蔚敏负责全书的审校。真诚地欢迎读者对书中可能有的错误和不妥之处批评指正，读者的意见是我们改进的动力。

译 者

2004 年 10 月于清华大学

前言

本书是一本全新的数据结构入门课程。读者可以把本书作为 Walter Savitch 所著《Java 语言：计算机科学与程序设计》(*Java: An Introduction to Computer Science & Programming*)^① 一书的续集来阅读。

本书的编写特别考虑了对象与 Java，它不是从已有的为另一种程序设计语言编写的著作派生而出。为了使学习更容易，我们采用的方式是：将读者的注意力一次集中在一个问题上，为读者提供阅读顺序的灵活性，以及明确区分抽象数据类型 (ADT) 的说明与实现。为了达到这些目的，我们将内容划分成 30 个相对较短的章，一章中或是论述 ADT 的说明及使用，或是论述 ADT 的实现，读者可以选择在阅读了一个 ADT 的说明之后紧跟着学习它的各种实现方法，也可以选择考虑任何实现问题之前先探讨各种 ADT 的说明及使用。从本书的编写结构上讲，读者可以方便地选择自己偏爱的阅读顺序来学习。

特点

- 30 个相对短的章可以按各种顺序阅读。
- 单独但相关的章将 ADT 的说明与实现分开。
- 用很多例子说明新的概念。
- 突出的“注”强调了关键的内容并提供补充注释。
- “编程提示”给出附加的编程建议。
- 大量的插图使讲解更形象，更易于理解。
- 贯穿全书的自测题及其答案均是根据本书内容精心制作的。
- 前几章的内容覆盖了 Java 类、继承、多态性及类的设计。
- Java 代码包含 javadoc 注释。
- 附录复习了 Java 基础、异常、文件及文档。
- Prentice Hall 出版社的配套网站 www.prenhall.com/carrano 提供本书中出现的 Java 代码以及任何更新与修正。
- 为教师提供的补充材料包括 PowerPoint 幻灯片、实验素材以及经过挑选的部分练习与项目设计的答案。

学生注意事项

在看完这个前言之后，读者应该先读引言。在那里你将迅速了解到本书要讲述的内容，以及在开始学习前需要对 Java 了解多少。从 A 到 D 的附录复习了 Java 的基础知识，并可

^① 该书已由清华大学出版社影印出版。

作为关于异常、文件及 javadoc 注释的初级读物。注意, 本书还给出了 Java 的保留字、Java 的运算符优先次序以及 Unicode 字符列表。

本书涵盖的主题是读者今后学习计算机科学的基础。即使现在没有学完所有主题, 以后也很可能会遇到它们。我们希望读者现在爱读本书, 并希望它成为你将来课程用得着的参考书。

本书中的所有 Java 代码均可从本书的配套网站 www.prenhall.com/carrano 下载。在那里, 你还将找到可用于键盘输入的 SavitchIn 类。附录 A 描述了这个类, 并提供一些它的使用例子。

在每一章中, 读者都能找到有助于理解内容的自测题, 这些问题的答案在本书后的附录 E 中。然而, 我们不能给出每一章末的练习与项目设计的答案, 采用本书的教师能够从出版社收到精选的答案。未选这门课的读者就不能得到该答案了。关于这些练习与项目设计的帮助, 读者必须与自己的教师联系。

本书概述

本书的读者应该已经学完一门程序设计的课程, 最好是关于 Java 程序设计的。附录 A 涵盖了假设读者已学过的 Java 基础, 读者可以用这个附录作为复习, 或者作为从另一种程序设计语言过渡到 Java 的基础。

本书自身是从引言开始的, 引言为以后的学习做准备, 第 1 章全面地复习 Java 中的类与方法, 紧接着在第 2 章中讨论合成、继承及多态性, 第 3 章引入面向对象设计, 介绍 Java 接口, 并初步介绍如 CRC 卡片与统一建模语言 (Unified Modeling Language) 这样的设计工具。

第 4 章~第 6 章引入一种抽象数据类型的线性表。我们把内容分成几章来讲, 就能清楚地在线性表的说明、使用及实现分开。例如, 第 4 章说明线性表, 并列几个使用示例, 第 5 章与第 6 章则涵盖了分别使用数组、向量及链表的实现。

在讨论其他各种 ADT 时, 本书自始至终以类似的方式将说明与实现分开。读者可以先阅读所有说明与使用 ADT 的章, 之后再阅读实现它们的各章, 或者按各章出现的顺序阅读, 每个 ADT 的实现都紧跟在学习它的说明与使用之后。

第 7 章与第 8 章以线性表为背景讨论了迭代器。在第 7 章中使用我们自己的接口以及实现它的各种方法提出迭代器的概念, 第 8 章继续这一讨论, 考察并实现 Java 的迭代器接口 Iterator 与 ListIterator。

第 9 章与第 10 章介绍算法的复杂性与递归, 这是两个在后续各章中要结合讨论的主题。第 11 章与第 12 章讨论各种排序方法与它们的相对复杂度, 并同时考察这些算法的迭代版本与递归版本。

接下来的第 13 章与第 14 章, 又回到了线性表的概念。第 13 章讨论了有序表, 考察了两种可能的实现及其效率。第 14 章介绍了如何使用线性表作为有序表的基类, 并讨论基类的一般设计要求。

第 15 章介绍了可变对象、不可变对象及克隆。如果客户可以维护 ADT 中数据的引用, 如果数据是可变的, 则它不使用 ADT 类的公有方法就能够修改此数据。我们考察了可用来

防止客户这样做的步骤。

第 16 章以线性表与有序表为背景，考察在数组与链表中查找的策略。这些讨论是第 17 章的良好基础，后者涵盖了 ADT 词典的说明与使用。第 18 章介绍了链表或数组的词典实现。第 19 章介绍了散列并用它实现词典。

第 20 章讨论栈，列举使用栈的例子，并考察栈与递归间的关系。第 21 章分别使用数组、向量及链表实现栈。

第 22 章介绍队列、双端队列及优先队列，第 23 章则考察它们的实现。在这一章中还讨论循环链表与双向链表。

第 24 章讨论树及其应用场合。在树的几个例子中，包括对二叉查找树与堆的初步介绍。第 25 章考察二叉树与通用树的实现，第 26 章则着重讨论二叉查找树的实现。第 27 章介绍如何用数组实现堆。第 28 章介绍平衡查找树，在这一章中包括 AVL 树、2-3 树、2-4 树及红黑树，还有 B 树。

第 29 章与第 30 章讨论图，考察图的几种应用及两种实现。

附录 A 到附录 D 提供了有关 Java 的补充材料。如前所述，附录 A 复习 Java 除类之外的基本内容，附录 B 涵盖了异常处理，附录 C 则讨论文件，附录 D 考察编程风格与注释，它介绍 javadoc 注释，并定义本书中使用的标记。

附录 E 包含全书各章的自测题答案。

教师使用的资源

我们为教师准备了若干补充材料，并将它们放在一张 CD 光盘上，其中包括 PowerPoint 幻灯片、实验素材、精选的练习与项目设计的答案、书中的 Java 代码以及附录 A 中描述的可用于键盘输入的 SavitchIn 类。注意最后两项在 Prentice Hall 出版社的配套网站 www.prenhall.com/carrano 上也可获得。该网站还包含本书的所有更新与修正。

要获得教辅资源光盘，应该与 Prentice Hall 的销售代表联系，关于销售代表的姓名与电话，请访问网站 www.prenhall.com/relocator 或者致电 Prentice Hall 高校服务 1-800-526-0485。有关本书及其他 Prentice Hall 产品的附加信息，也可在 Prentice Hall 的主页 www.prenhall.com 上找到。

联系方法

读者的评论、建议及更正永远受到欢迎。请发 E-mail 到：

carrano@acm.org

致谢

感谢下列审稿人认真地阅读我们的手稿，并且做出中肯的评论与建议，这极大地改善了我们的著作：

David Boyd—Valdosta State University

Dennis Brylow—Purdue University

Michael Croswell—Industry trainer/consultant

Matthew Dickerson—Middlebury College

Robert Holloway—University of Wisconsin, Madison

John Motil—California State University, Northridge

Bina Ramamurthy—SUNY, Buffalo

David Surma—Valparaiso University

感谢 Prentice Hall 出版社为本书的制作与出版做出贡献的所有人。他们全都不知疲倦与充满干劲地工作。特别要感谢我们的发行人 Alan Apt, 感谢他使这个项目成为可能, 并且感谢他在整个过程中宝贵的洞察与指导。感谢副主编 Toni Holm 对项目的监督及让我们保持脚踏实地按计划工作的定期交谈。感谢 Jake Warde 杰出地协调了手稿的审稿人及教师补充材料的制作。感谢我们的出版编辑 Chirag Thakkar 让本书如期出版。我们还要感谢 Heather Scott 的版式设计工作, 感谢 Xiaohong Zhu 整理艺术图稿。

特别感谢 Rebecca Pepper, 我们的文稿编辑, 她不但纠正了我们的语法错误, 还保证了我们的讲解清楚无误。感谢 Lianne Dunn, 她设计了封面图案, 并且绘制了本书中所有的插图。

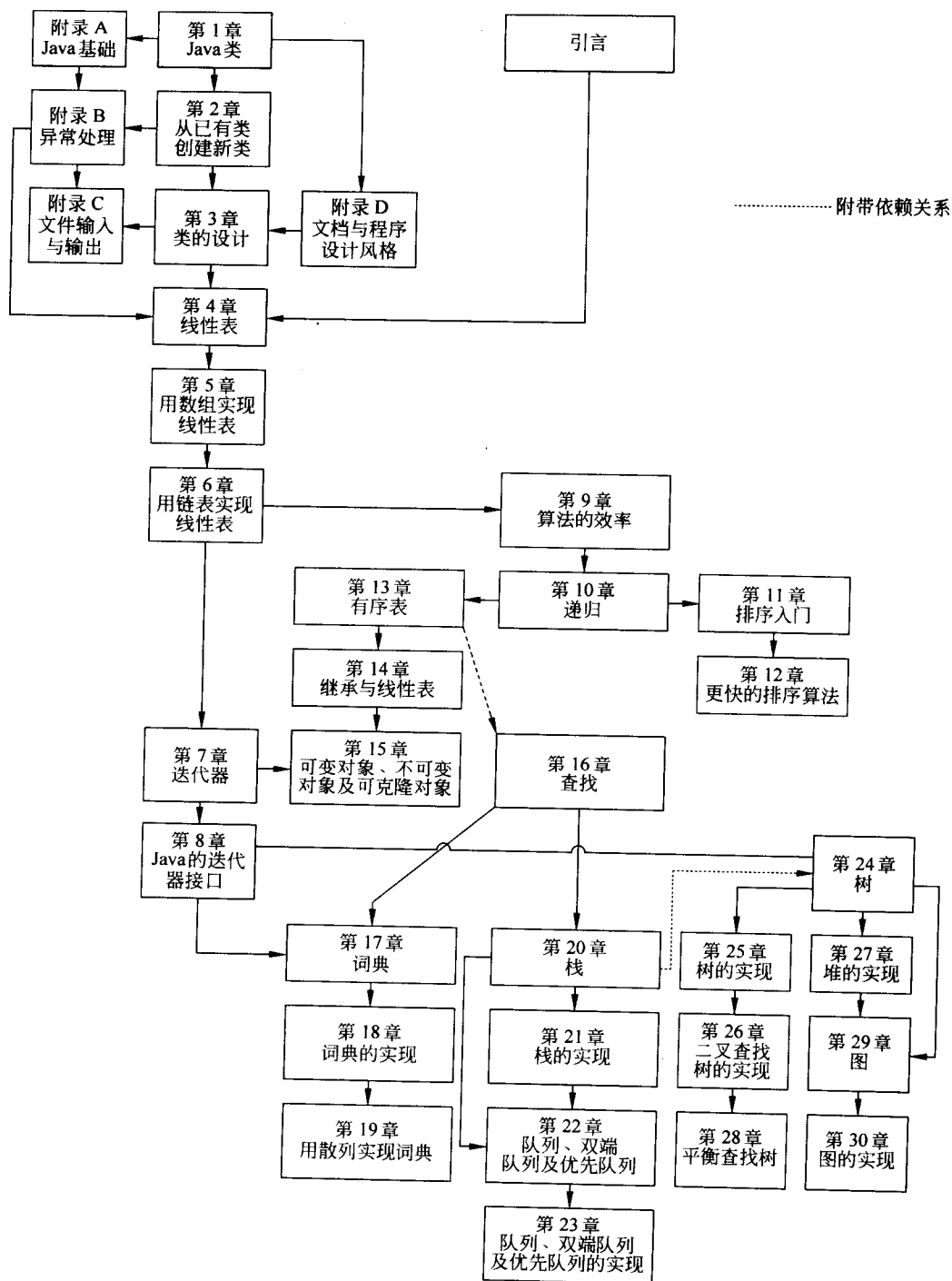
感谢下列人士为教师补充材料做出的贡献: Steve Armstrong (Le Tourneau University), Patty Roy (Manatee Community College), Charles Hoot (Oklahoma City University)。感谢 Nate Walker (the University of Rhode Island) 准备了用于分发的 Java 代码。

其他热心的人士以各种方式做出了贡献。他们是: Doug McCreadie, Ted Emmott, Lorraine Berube, Marge White, Gérard Baudet, Joan Peckham, James Kowalski, Ed Lamagna, Bala Ravikumar, Victor Fay-Wolfe, Lisa DiPippo, Jean-Yves Hervé, James Heltshe, Brian Jepson, Ben Schomp, Patrick Lindner, James Blanding, Tim Henry。

感谢各位的贡献以及鼓励。

Frank M. Carrano

Walter Savitch



保 留 字

保留字（reserved word）也称为关键字（keyword）。不能重复定义这些保留字。它们的含义由 Java 语言确定，不能改变。特别地，不能把任何保留字用作变量名、方法名或类名。

abstract	goto	short
		static
boolean	if	strictfp
break	implements	super
byte	import	switch
		synchronized
case	instanceof	
catch	int	this
char	interface	throw
class		throws
const	long	transient
continue		true
	native	try
default	new	
do	null	void
double		volatile
	package	
else	private	while
extends	protected	
	public	
false		
final	return	
finally		
float		
for		

运算符优先次序

在下列列表中，同一行中的运算符的优先级相等。列表中越往下的行，优先级越低。如果运算的顺序未由括号标明，优先级较高的运算符在优先级较低的运算符之前执行。如果运算符的优先级相等，则二元运算符按从左到右的顺序执行，一元运算符按从右到左的顺序执行。

最高优先级

一元运算符 `+, -, ++, --, !, ~`

一元运算符 `new` 和 `(type)`

二元运算符 `*, /, %`

二元运算符 `+, -`

二元（移位）运算符 `<<, >>, >>>`

二元运算符 `<, >, <=, >=`

二元运算符 `==, !=`

二元运算符 `&`

二元运算符 `^`

二元运算符 `|`

二元运算符 `&&`

二元运算符 `||`

三元（条件）运算符 `?:`

赋值运算符 `=, *=, /=, %=, +=, -=, <<=, >>=, >>>=, &=, ^=, |=`

最低优先级

Unicode 字符列表

以下列出的可打印字符是 Unicode 字符集的子集，也称为 ASCII 字符集。这些字符无论是视作 Unicode 字符集的成员还是 ASCII 字符集的成员，其编号都是相同的。（编号 32 的字符是空格。）

32		56	8	80	P	104	h
33	!	57	9	81	Q	105	i
34	"	58	:	82	R	106	j
35	#	59	;	83	S	107	k
36	\$	60	<	84	T	108	l
37	%	61	=	85	U	109	m
38	&	62	>	86	V	110	n
39	'	63	?	87	W	111	o
40	(	64	@	88	X	112	p
41	)	65	A	89	Y	113	q
42	*	66	B	90	Z	114	r
43	+	67	C	91	[	115	s
44	,	68	D	92	\	116	t
45	-	69	E	93	]	117	u
46	.	70	F	94	^	118	v
47	/	71	G	95	_	119	w
48	0	72	H	96	`	120	x
49	1	73	I	97	a	121	y
50	2	74	J	98	b	122	z
51	3	75	K	99	c	123	{
52	4	76	L	100	d	124	
53	5	77	M	101	e	125	}
54	6	78	N	102	f	126	~
55	7	79	O	103	g		

引 言

环视四周，你将看到人们组织事物的方式。今天早上，你在商店里走到队伍的末尾等待收银员收款。这个队伍是人们按时间顺序排的一行，队伍中的第一个人最先得到服务并离开。最终，你到达队伍的前端，并带着装有已购物品的袋子离开商店。袋子里的东西没有特定的顺序，并且其中有一些物品是相同的。

在你的书桌上，可以看见待办事务列表。列表上的每一项所在位置，对你来说可能重要，也可能不重要，它可能是在你想到它们时顺手写下的，也可能是根据它们的重要性或者字母顺序写的，其顺序由你决定，列表只是简单地为这些项目提供位置。

看到你书桌上的一叠书或者一堆纸吗？很容易查看或移走一叠物品中最顶上的物品，或者将一个新物品加到一叠物品顶上，这叠物品也是按时间顺序排列的，最后放置的物品在顶上，最先放置的物品在底下。

你的词典是单词及其含义的字母顺序列表，你可查找一个单词并得到它的含义。如果你的词典是印刷的，字母顺序帮助你迅速找到单词，如果你的词典是计算机化的，它按字母顺序组织的方式被隐藏，但仍然能快速查找。

谈到你的计算机，你已经将文件放入文件夹或目录里，每个文件夹包含若干其他文件夹或文件，这种组织方式是层次化的。如果你为它画一张图，将得到类似于家谱树或者公司内部机构图示的东西。这些数据组织方式都是相似的，并称它为树（tree）。

最后，注意一下你用以规划周末出游的公路地图，公路与城镇的图解指示你如何从一个地方去另一个地方。常常有几条可能的路，一条路可能比较短，另一条可能比较快。公路图是一种称为图（graph）的组织方式。

计算机程序也需要组织它们的数据，它们做这些事情的方式与刚才列举的这些例子是类似的，也就是说，程序可以使用线性表、栈及词典，等等。这些数据组织的方式称为抽象数据类型。一个抽象数据类型（abstract data type），即 ADT 是描述一个数据集合以及在这个数据集上的操作的说明。每个 ADT 都说明了存放的是什么数据，以及这些数据上的操作做了什么。由于 ADT 并不指明如何存放数据或者如何实现操作，因此对 ADT 的谈论可以独立于任何程序设计语言。相反，数据结构（data structure）是 ADT 在程序设计语言中的一种实现。

集合（collection）是对含有一组对象的 ADT 的通称。有些集合允许包含相同的元素，有些集合则不允许，有些集合以某种顺序排列它们的元素，而另一些则不然。容器（container）是一个实现集合的类。有些人们交替使用“容器”与“集合”这两个术语。

可以创建一个 ADT 背包（bag），它由允许含相同元素的无序集合构成。就像是一个杂货袋、午餐包或者薯条袋。设想从一袋薯条中拿走一根，并不知道这根薯条是何时被放进袋里的，也不知道袋里是不是还有另一根其形状与刚拿走这根一模一样的薯条，但你并不在乎。如果你在乎的话，你就不会将薯条放在袋子里了。

背包不对其元素排序,但有时人们需要对事物排序。ADT 能够以各种方式对它们的元素排序。例如,ADT 线性表 (list) 简单地对它的元素编号,从而线性表有第一个元素、第二个元素,以此类推。元素不但可以被加入到线性表末尾,也可以插入到线性表的起始位置或者已有的两个元素中间,插入的同时对新元素之后的元素进行重新编号。此外,还可以删除位于线性表中指定位置的元素。因此,元素在线性表中的位置不一定表示它被插入的时间。注意线性表并不决定元素应该放在何处,这个位置是由你来定的。

相反,ADT 栈 (stack) 与队列 (queue) 按时间顺序排列它们的元素。从栈中删除一个元素时,删除的是最近插入的元素。从队列中删除一个元素时,删除的是最早插入的元素。因此,栈类似于—叠书,可以移走最顶上的书,或者将另一本书放到这叠书的顶上。队列则类似于人们排的队伍,人们从队的前端离开,从队尾插入。

如果元素可以互相比,则有些 ADT 能维持它们的元素有序。例如,字符串可以按字母顺序排列。又如,在向 ADT 有序表 (sorted list) 插入元素时,由 ADT 决定元素应该放在何处,不需要像 ADT 线性表那样,由你为元素指定位置。

ADT 词典 (dictionary) 含有成对的元素,很像是含有单词及其含义的语言词典。在这个例子中,单词的作用是用来定位元素的键 (key)。一些词典让元素有序排列,另一些则不然。

ADT 树 (tree) 按照某些层次关系组织其元素。例如,在家谱树中,人们通过他们的孩子及其双亲联系起来。ADT 二叉查找树 (binary search tree) 有着层次化并有序相结合的组织方式,它使得查找指定元素更容易。

ADT 图 (graph) 是 ADT 树的推广,它关注的是元素间的关系,而不是任何层次化结构。例如,公路地图展示了城镇之间存在的公路及距离。

本书介绍了如何使用与实现这些数据组织方式。在开始学习之前,读者需要懂得 Java。附录 A 复习 Java 中的基本语句,第 1 章讨论了类与方法的基本构造。对这些内容,读者可以自己选择是粗略地浏览、仔细地阅读,还是需要时再回过头来看。第 2 章与第 3 章的重点仍然是 Java,但其中一些或全部内容对读者来说可能是新的。第 2 章涵盖了从已有的类创建新类的技术,包括合成与继承,第 3 章讨论了如何设计类,说明方法,以及如何编写 Java 接口。使用接口与编写说明方法的注释是本书介绍 ADT 的要点。

目 录

第 1 章	Java 类	1
1.1	对象与类	1
1.2	在 Java 类中使用方法	3
1.2.1	引用与别名	4
1.2.2	实参与形参	5
1.3	定义 Java 类	6
1.3.1	方法定义	7
1.3.2	传递实参	9
1.3.3	Name 类的定义	12
1.3.4	构造函数	13
1.3.5	toString 方法	15
1.3.6	静态的域与方法	16
1.4	包	17
第 2 章	从已有类创建新类	23
2.1	合成	23
2.2	继承	27
2.2.1	在构造函数中调用构造函数	30
2.2.2	基类的私有域与私有方法	31
2.2.3	方法的覆盖与重载	32
2.2.4	保护访问	35
2.2.5	多重继承	36
2.3	类型兼容性与基类	36
2.3.1	Object 类	37
2.3.2	抽象类与抽象方法	39
2.4	多态性	40
第 3 章	类的设计	50
3.1	封装	50
3.2	方法的说明	52
3.3	Java 接口	55
3.3.1	编写接口	55
3.3.2	实现接口	57

3.3.3 作为数据类型的接口	58
3.3.4 接口实现中的类型转换	58
3.3.5 扩展接口	59
3.3.6 接口中的符号常量	60
3.3.7 接口与抽象类的比较	61
3.4 类的选择	63
3.4.1 类的确定	64
3.4.2 CRC 卡片	64
3.5 类的复用	66
第 4 章 线性表	70
4.1 ADT 线性表说明	70
4.2 使用 ADT 线性表	78
4.3 Java 类库: List 接口	82
4.4 使用线性表如同使用自动售货机	82
第 5 章 用数组实现线性表	87
5.1 使用定长数组实现 ADT 线性表	87
5.1.1 类比	87
5.1.2 Java 实现	89
5.2 使用动态扩展数组实现 ADT 线性表	96
5.2.1 扩展数组	97
5.2.2 线性表新的实现	98
5.3 使用向量实现 ADT 线性表	100
5.4 用数组实现 ADT 线性表的优缺点	104
5.5 Java 类库	104
5.5.1 ArrayList 类	104
5.5.2 Serializable 接口	105
第 6 章 用链表实现线性表	108
6.1 链表	108
6.1.1 创建一个链表	109
6.1.2 创建另一个链表	111
6.1.3 仍创建一个链表	113
6.2 Node 类	116
6.3 使用链表实现 ADT 线性表	118
6.3.1 在线性表的末端插入元素	119
6.3.2 在线性表的指定位置插入元素	122