



普通高等教育“十一五”国家级规划教材

高等学校计算机专业规划教材

数据结构(Java版)

(第2版)

■ 叶核亚 编著
■ 陈本林 主审



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

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

普通高等教育“十一五”国家级规划教材
高等学校计算机专业规划教材

数据结构 (Java 版)

(第 2 版)

叶核亚 编著
陈本林 主审



电子工业出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书为普通高等教育“十一五”国家级规划教材。

本书全面系统地介绍数据结构的基础理论和算法设计方法,包括线性表、树、图等数据结构以及查找和排序算法。本书采用 Java 语言以面向对象方法设计并实现了全部的数据结构及算法。本书精选基础理论内容,重点是数据结构设计和算法设计,通过降低理论难度和抽象性,加强实践环节等措施,力求增强学生的理解能力和应用能力。内容涉及的广度和深度符合本科培养目标的要求。

本书可作为普通高等学校计算机及相近专业本科生的数据结构课程教材,也可作为从事计算机软件开发和工程应用人员的参考书。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,侵权必究。

图书在版编目(CIP)数据

数据结构:Java 版/叶核亚编著.-2 版.-北京:电子工业出版社,2008.7

高等学校计算机专业规划教材

ISBN 978-7-121-06892-8

I. 数… II. 叶… III. ① 数据结构—高等学校—教材 ② JAVA 语言—程序设计—高等学校—教材
IV. TP311.12 TP312

中国版本图书馆 CIP 数据核字(2008)第 084812 号

策划编辑:章海涛

责任编辑:章海涛

印 刷:北京市海淀区四季青印刷厂

装 订:三河市万和装订厂

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:787×1092 1/16 印张:19.75 字数:500 千字

印 次:2008 年 7 月第 1 次印刷

印 数:4 000 册 定价:28.00 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。

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

服务热线:(010) 88258888。

第2版前言

本书是普通高等教育“十一五”国家级规划教材，可作为普通高等学校计算机及相近专业本科的数据结构课程教材。

数据结构是软件设计的重要理论和实践基础，数据结构设计和算法设计是软件系统设计的核心。数据结构课程讨论的知识内容，是软件设计的理论基础；数据结构课程介绍的技术方法，是软件设计中使用的基本方法。数据结构是理论与实践并重的课程，要求学生既要掌握数据结构的基础理论知识，又要掌握运行和调试程序的基本技能。因此，数据结构课程在计算机类各专业本科学生的培养过程中有着十分重要的地位，是计算机类各专业本科的一门核心课程，是培养程序设计能力的必不可少的重要环节。

数据结构课程内容多，概念抽象，理论深奥，递归算法难度较大，一直是计算机专业最难学的课程之一。本书精选基础理论内容，重点是数据结构设计和算法设计，通过降低理论难度和抽象性，加强实践环节等措施，进一步增强学生的理解能力和应用能力，力求取得较好的教学效果。

本书特色说明如下。

(1) 内容全面、注重基础

本书全面系统地介绍数据结构的基础理论和算法设计方法，阐明线性表、树、图等数据模型的逻辑结构，讨论它们在计算机中的存储结构，讨论每种数据结构所能进行的多种操作，以及这些操作的算法设计与实现；针对软件设计中应用频繁的查找和排序问题，根据不同数据结构对操作的实际需求，给出多种查找和排序算法，并分析算法的执行效率。

本书的内容选择适合工科院校，理论叙述精练，简明扼要，结构安排合理，由浅入深，层次分明，重点突出，算法分析透彻，程序结构严谨规范。内容涉及的广度和深度符合本科培养目标的要求。

(2) 采用 Java 语言描述数据结构和算法

数据结构和算法的设计不依赖于程序设计语言，但数据结构和算法的实现依赖于程序设计语言。描述数据结构所采用的软件方法和算法语言，需要随着软件方法及算法语言的不断发展而发展。面向对象程序设计方法是软件分析、设计和实现的新方法，是目前软件开发的主流方法。

Java 语言是目前功能最强、应用最广泛的一种完全面向对象程序设计的语言。Java 语言建立在成熟的算法语言和坚实的面向对象理论基础之上，具有强大的应用系统设计能力，其特有的面向对象、平台无关、内存自动管理、异常处理、多线程等机制，使其更健壮、更安全、更高效。

虽然 Java 语言放弃了 C++ 语言的全局函数、结构体、指针、友元类、运算符重载、多重继承等功能，但这些丝毫不影响 Java 语言的功能和性能。Java 通过引用模型实现了指针的功能，通过类实现了结构体类型，通过“单重继承+接口”方式实现了多重继承，通过内存自动管理机制自动释放动态分配的存储空间。Java 语言提供的各种机制均体现出简单性原则，

即对一个问题只给出一种解决方法,使程序简单、直接并且不造成歧义。

Java 语言不仅具备表达数据结构和算法的基本要素,而且能使算法更简明、更直接。因此,采用 Java 语言描述数据结构和算法是可行的,作为面向对象的程序设计方法训练是十分恰当的,也是数据结构课程内容改革的必然,完全符合本科培养目标的要求。

(3) 增强实际应用

数据结构是一门理论和实践紧密结合的课程,因此,需要在透彻理解理论知识的基础上,通过实践性环节,逐步锻炼学生的程序设计能力。

注重传授基础理论知识,注重在实践环节中培养程序设计的基本技能,是本书第 1 版的重要特色,也是本书的重要特色。本书精心选择并设计一系列与实际应用息息相关的例题、习题、实验题、课程设计题等,使原本枯燥难懂的理论变得生动有趣,并以此说明数据结构和算法的必要性,以及它们对实际应用程序设计的指导作用。同时,要求学生能在生活中发现问题并解决问题,提高学习兴趣,力求在潜移默化中使学生理解和体会理论知识的重要性,并掌握如何使用它们的方法。

除了每章的实验题给出详细的训练目标、设计内容和设计要求之外,针对课程设计实践性环节,本书给出多种算法设计与分析的综合应用程序设计实例,详细说明需求方案、设计思想、模块划分、功能实现、调试运行等环节的设计方法,贯彻理论讲授和案例教学相结合的教学方法。

程序设计有其自身的规律,不是一蹴而就的,也没有捷径。程序员必须具备基本素质,必须掌握程序设计语言的基本语法以及算法设计思想和方法,并且需要积累许多经验。这个逐步积累的过程需要一段时间,需要耐心,厚积而薄发。

本书第 0 章概述 Java 语言的基础知识和面向对象的程序设计思想;第 1~9 章是数据结构课程的主要内容,包括线性表、树、图等数据结构设计以及查找和排序算法设计;第 10 章综合应用设计,包括课程设计的范例和参考选题;第 11 章介绍 Java 开发运行环境,包括 JDK 和 JCreator,这章虽然放在最后,却是贯穿始终的,配合各章逐步进行,熟悉集成开发环境、掌握程序调试技术也需要经过一个逐步积累的过程。

本书所有程序均已在 JDK 1.6 中调试通过。

本书由叶核亚编著。在写作过程中得到了许多帮助和支持。感谢南京大学计算机科学与技术系陈本林教授,陈教授认真细致地审阅了全稿,提出了许多宝贵意见,感谢陈立、廖雷、阚建飞、陈建红、陈瑞、王青云等老师给予的帮助,感谢电子工业出版社的支持,感谢众多读者朋友对本书第 1 版提出的宝贵意见。

对书中存在的不妥与错漏之处,敬请读者朋友批评指正。

本书为任课教师提供配套的教学资源(包含电子教案和例题源代码),需要者可登录华信教育资源网站(<http://www.huaxin.edu.cn> 或 <http://www.hxedu.com.cn>),注册之后进行免费下载,或发邮件到 unicode@phei.com.cn 进行咨询。

作者

目 录

第 0 章	Java 程序设计基础	1
0.1	Java 的特点和优势	1
0.2	Java 语言基础	2
0.2.1	语言成分	2
0.2.2	流程控制语句	4
0.2.3	数组	5
0.2.4	字符串	7
0.3	面向对象程序设计	8
0.3.1	类和对象	8
0.3.2	类的封装性	10
0.3.3	类的继承性	14
0.3.4	类的多态性	17
0.3.5	抽象类和最终类	21
0.3.6	接口	22
0.3.7	内部类	23
0.3.8	包	24
0.4	异常处理	25
0.4.1	Java 的错误和异常	25
0.4.2	抛出和处理异常	27
0.5	Java 的标准输入/输出	28
	习题 0	29
	实验 0 Java 程序设计基础	30
第 1 章	绪论	31
1.1	数据结构的基本概念	31
1.1.1	为什么要学习数据结构	31
1.1.2	什么是数据结构	31
1.1.3	数据的逻辑结构	33
1.1.4	数据的存储结构	34
1.1.5	数据操作	35
1.1.6	用 Java 语言描述数据结构	36
1.2	算法	36
1.2.1	什么是算法	36
1.2.2	算法分析	38

1.2.3 算法设计实例	40
1.2.4 递归算法	45
习题 1	47
实验 1 算法设计	48
第 2 章 线性表	49
2.1 线性表抽象数据类型	49
2.2 线性表的顺序表示和实现	50
2.3 线性表的链式表示和实现	55
2.3.1 线性表的链式存储结构	55
2.3.2 单链表	56
2.3.3 双链表	67
*2.4 迭代器	70
2.4.1 迭代接口	70
2.4.2 基于迭代器的操作	71
2.4.3 提供迭代器对象	71
习题 2	73
实验 2 线性表的基本操作	73
第 3 章 栈和队列	75
3.1 栈	75
3.1.1 栈抽象数据类型	75
3.1.2 顺序栈	76
3.1.3 链式栈	77
3.1.4 栈的应用	78
3.2 队列	84
3.2.1 队列抽象数据类型	84
3.2.2 顺序队列	85
3.2.3 链式队列	87
3.2.4 队列的应用	88
习题 3	90
实验 3 栈和队列及其应用	90
第 4 章 串	92
4.1 串抽象数据类型	92
4.1.1 串的基本概念	92
4.1.2 串抽象数据类型	93
4.2 串的实现	93
4.2.1 串的存储结构	93
4.2.2 字符串类 String	94
4.2.3 字符串类 StringBuffer	98

4.3 串的模式匹配	101
4.3.1 朴素的模式匹配 (Brute-Force) 算法	101
4.3.2 无回溯的模式匹配 (KMP) 算法	106
习题 4	110
实验 4 串的基本操作及模式匹配应用	111
第 5 章 数组和广义表	112
5.1 数组	112
5.1.1 一维数组	112
5.1.2 多维数组	112
5.2 特殊矩阵的压缩存储	116
5.2.1 对称 (三角) 矩阵的存储	116
5.2.2 稀疏矩阵的压缩存储	117
5.3 广义表	123
5.3.1 广义表抽象数据类型	124
5.3.2 广义表的存储结构	125
习题 5	127
实验 5 特殊矩阵和广义表的存储和运算	127
第 6 章 树和二叉树	129
6.1 树及其抽象数据类型	129
6.1.1 树定义	129
6.1.2 树的术语	130
6.1.3 树的表示法	131
6.1.4 树抽象数据类型	131
6.2 二叉树及其抽象数据类型	132
6.2.1 二叉树定义	132
6.2.2 二叉树性质	132
6.2.3 二叉树抽象数据类型	134
6.3 二叉树的表示和实现	134
6.3.1 二叉树的存储结构	134
6.3.2 二叉树的二叉链表实现	136
6.3.3 二叉树的遍历	137
6.3.4 构造二叉树	142
6.3.5 二叉树的插入和删除操作	147
6.3.6 二叉树遍历的非递归算法	148
6.3.7 二叉树的层次遍历	149
6.4 线索二叉树	151
6.4.1 线索二叉树定义	151
6.4.2 中序线索二叉树	152

6.5	哈夫曼编码与哈夫曼树	157
6.5.1	哈夫曼编码	158
6.5.2	哈夫曼树	159
6.6	树的表示	165
6.6.1	树的存储结构	165
6.6.2	树的遍历	166
习题 6		167
实验 6	二叉树的基本操作	168
第 7 章 图		170
7.1	图及其抽象数据类型	170
7.1.1	图的基本概念	170
7.1.2	图抽象数据类型	173
7.2	图的表示和实现	174
7.2.1	图的邻接矩阵表示	174
7.2.2	图的邻接表表示	180
7.3	图的遍历	187
7.3.1	图的深度优先搜索遍历	187
7.3.2	图的广度优先搜索遍历	189
7.4	最小生成树	191
7.4.1	生成树	191
7.4.2	最小生成树的构造算法	193
7.5	最短路径	197
习题 7		201
实验 7	图的表示和操作	202
第 8 章 查找		203
8.1	查找的基本概念	203
8.2	基于线性表的查找	204
8.2.1	顺序查找	204
8.2.2	基于有序顺序表的折半查找	206
8.2.3	基于索引顺序表的分块查找	208
8.3	树结构的查找	213
8.3.1	二叉排序树及其查找	213
8.3.2	平衡二叉树	219
8.4	散列	220
8.4.1	散列表	220
8.4.2	散列函数	221
8.4.3	处理冲突	222
8.4.4	构造链地址法的散列表	223

习题 8	225
实验 8 查找算法及其效率分析	226
第 9 章 排序	227
9.1 排序的基本概念	227
9.2 插入排序	228
9.2.1 直接插入排序和折半插入排序	228
9.2.2 希尔排序	230
9.3 交换排序	232
9.3.1 冒泡排序	232
9.3.2 快速排序	233
9.4 选择排序	236
9.4.1 直接选择排序	236
9.4.2 堆排序	237
9.5 归并排序	240
习题 9	242
实验 9 排序算法设计及分析	243
第 10 章 综合应用设计	244
10.1 Java 集合框架	244
10.1.1 数组	244
10.1.2 集合	248
10.2 使用复杂的 Swing 组件	251
10.2.1 列表框 JList	252
10.2.2 表格 JTable	252
10.2.3 树 JTree	259
10.3 课程设计的实例和选题	266
10.3.1 用预见算法解骑士游历问题	266
10.3.2 课程设计选题	271
第 11 章 Java 开发运行环境	275
11.1 JDK	275
11.1.1 JDK 的安装与设置	275
11.1.2 运行 Java Application 程序	276
11.1.3 查看 Java 类库与包等级	278
11.1.4 创建、声明和导入包	278
11.2 使用 JCreator	280
11.2.1 JCreator 集成开发环境	280
11.2.2 创建、编译并运行一个项目	282
11.2.3 一个工作区包含多个项目	285
11.2.4 配置 JDK	286

11.2.5 程序调试技术	288
附录 A ASCII 字符与 Unicode 值	293
附录 B Java 关键字	294
附录 C Java 基本数据类型	296
附录 D Java 运算符及其优先级	297
附录 E java.lang 语言包部分类库	298
附录 F JCreator 4.5 常用菜单命令	301
附录 G 本书数据结构包说明	304

例 题 索 引

【例 0.1】显示字符串的 Application 应用程序。	P3
【例 0.2】声明 Person 类及使用对象。	P9
【例 0.3】封装的 Person 类。	P12
【例 0.4】Student 类继承 Person 类。	P16
【例 0.5】类的多态性, Student 子类重定义父类成员。	P18
【例 0.6】标准输入/输出。	P28
【例 1.1】算法时间复杂度分析。	P39
【例 1.2】交换两个变量值。	P40
【例 1.3】数组的顺序查找和对对象比较相等方法。	P42
【例 1.4】已排序数组的顺序查找, 以及对对象比较大小方法。	P44
【例 1.5】求 $n!$ 。	P46
【例 1.6】求 Fibonacci 序列。	P47
【例 2.1】使用顺序表类求解约瑟夫 (Josephus) 环问题。	P54
【例 2.2】采用单链表求解约瑟夫环问题。	P62
【例 2.3】单链表逆转。	P63
【例 2.4】建立排序的单链表。	P66
【例 3.1】判断表达式中圆括号是否匹配。	P79
【例 3.2】使用栈计算表达式的值。	P80
【例 3.3】求解素数环问题。	P89
【例 4.1】String 类的模拟实现。	P94
【例 4.2】判断标识符。	P97
【例 4.3】使用 String 实现字符串的插入、删除操作。	P98
【例 4.4】StringBuffer 类的模拟实现。	P99
【例 4.5】串的逆转。	P101
【例 4.6】Brute-Force 模式匹配算法实现。	P102
【例 4.7】串的替换操作。	P104
【例 5.1】矩阵类。	P114
【例 5.2】稀疏矩阵的三元组顺序表实现。	P118

【例 6.1】构造并遍历二叉树。	P139
【例 6.2】输出二叉树中指定结点的所有祖先结点。	P145
【例 6.3】建立二叉链表表示的完全二叉树。	P146
【例 6.4】构造中序线索二叉树并进行中根次序遍历。	P155
【例 6.5】构造哈夫曼树并获得哈夫曼编码。	P161
【例 7.1】带权有向图的构造、插入及删除操作。	P186
【例 7.2】构造带权无向图的最小生成树。	P195
【例 7.3】求带权图的单源最短路径。	P200
【例 8.1】判断一个字符串是否为 Java 关键字。	P210
【例 8.2】二叉排序树的插入和查找操作。	P214
【例 10.1】对象数组排序和查找。	P245
【例 10.2】电话簿。	P253
【例 10.3】以树结构显示中国城市。	P261
【例 10.4】用预见算法解骑士游历问题。	P267
【例 11.1】使用命令行参数作为输入数据。	P277
【例 11.2】创建 Hello 工作区和项目。	P283
【例 11.3】设置命令行参数。	P284
【例 11.4】一个工作区包含多个项目。	P286
【例 11.5】使用自定义的包。	P287
【例 11.6】使用调试技术发现程序逻辑错误。	P291

第 0 章 Java 程序设计基础

本章概述 Java 语言的基础知识和面向对象程序设计思想，以“够用必需为度”原则，介绍描述数据结构和算法所需要的面向对象技术，为后续章节做准备。有 Java 语言基础的读者可跳过本章。

本章介绍 Java 语言的基本语法，包括数据类型的特点、使用数组和字符串的方法等；介绍面向对象的基本概念和类的设计方法，包括类的封装、继承和多态原则，接口、内部类和包的作用及使用；介绍异常处理和标准输入/输出方法。

0.1 Java 的特点和优势

物竞天择，适者生存。同自然界的进化规律一样，程序设计语言、程序设计思想的变化和发展也是随着实际应用需要而变化 and 发展的。我们今天所看到、所使用的程序设计语言，经历了一系列竞争和淘汰之后仍然存在，说明它们有存在的道理，它们必定各有所长。这是一种自然选择的结果。

1995 年，作为一种完全面向对象的程序设计语言，Java 以一种全新的面貌问世。至今短短的十余年间，凭借其平台无关、健壮、安全、高效这些适应网络运行需要的特点，Java 快速成长，不仅在 Internet 上游刃有余，而且通过 J2ME、J2SE、J2EE 三大平台，其应用领域全面覆盖了嵌入式应用、桌面应用和企业级应用，所表现出的强大的应用系统设计能力，使 Java 无处不在。

Sun 公司于 1995 年发布 JDK (Java Development Kit, Java 开发工具包) 1.0，宣告 Java 语言诞生。1998 年，发布 JDK 1.2，自称 Java 2 平台。1999 年，发布 JDK 1.3，将 Java 平台划分为 J2ME (嵌入式)、J2SE (标准)、J2EE (企业级)，使 Java 技术获得了最广泛的应用。2006 年，发布 JDK 1.6。

Java 语言的主要特性如下。

1. 完全面向对象和简单性

Java 语言是完全面向对象的，所有设计都必须在类中实现，一个 Java 程序就是一个类；为 8 种基本数据类型提供相应的基本数据类型包装类，使基本数据类型与类相关联；将数组设计为引用类型，使用方法与对象相同，并且每个数组都有长度属性；不支持面向过程设计，不支持类似 C++ 的全局变量、goto 语句、宏定义、全局函数等语法规则，不支持结构体、联合和指针等数据类型；通过引用模型实现了指针的功能，通过类实现了结构体类型。

Java 语言提供单重继承机制，所有类，包括 Java 声明的类和程序员声明的类，形成具有树结构的类的层次体系，Object 类是这个树结构的根类，Java 在 Object 类中为所有对象定义基本状态和行为。Java 不支持多重继承、友元类和运算符重载，通过“单重继承+接口”方式实现了多重继承功能。

Java 语言提倡简单性原则，对一个问题只提供一种简单精练的表达方法，这样使程序简

单、直接并且不造成歧义。例如,使用下标形式对数组元素进行操作,则不需要使用指针;方法(函数)采用返回值或引用类型参数返回结果,也不需要使用指针;有了类,则不需要结构体类型;构造方法采用重载方式,则不需要采用参数默认值形式,避免产生歧义;通过方法实现类的操作,则不需要运算符重载等。

因此,放弃结构体、指针、多重继承等,并没有影响 Java 语言的功能,而 Java 语言提供的机制具有更强的功能和更高的性能。

2. 平台无关性

平台无关性是指一个应用程序能够运行于不同的操作系统平台,也就是 Sun 公司设计 Java 的宗旨“Write once, run anywhere”。

Java 采用虚拟机技术支持平台无关性。Java 虚拟机 (Java Virtual Machine, JVM) 是一套支持 Java 语言运行的软件系统,运行于操作系统之上。Sun 为不同的操作系统配有相应的 Java 虚拟机。

高级语言通常将程序编译生成可执行文件 (*.exe),再由操作系统直接执行;与此方式不同,Java 源程序 (*.java) 经编译生成的是一种中间代码,称为字节码文件 (*.class),它由 Java 虚拟机解释执行,因而与操作系统平台无关。

Java 语言本身设计也体现出平台无关性。例如, int 型整数占 32 位,不依赖于操作系统是 16 位还是 32 位。

3. 安全性和可靠性

Java 语言提供严密的语法规则,在编译和运行时进行严格检查,降低了程序出错的可能性。例如,不同类型的数据不能进行运算,数组下标不能越界,不能使用指针对指定地址的内存区域进行操作等。

Java 语言的异常处理机制,能够及时发现所有运行错误,并将运行错误交由程序处理,保证 Java 程序运行的可靠性。

Java 语言的资源回收机制,能够自动跟踪程序使用的所有内存资源,并且自动收回不再使用的内存资源。因此,程序中不需要写释放内存语句,既减少了程序员的工作量,又提高了安全性。

0.2 Java 语言基础

本节介绍 Java 语言的基本语法,包括标识符与关键字、数据类型、运算符、变量与常量、流程控制语句、数组、字符串等。这些基本语法是 Java 程序设计的基础。

0.2.1 语言成分

1. 程序结构

Java 语言提供 Application 和 Applet 两种应用程序。Application 是能够独立运行的应用程序;Applet 是可以嵌入 Web 页面的最小应用,它不能独立运行,必须嵌入超文本 (.html) 中,由浏览器中的 Java 解释器解释执行。

一个 Java 程序就是一个类,类由关键字 class 声明,类中包含成员变量和成员方法,main() 方法是类首先执行的方法,语句必须写在类的方法中。

【例 0.1】显示字符串的 Application 应用程序。

Application 应用程序如下：

```
public class Hello {                                //声明类
    public static void main(String args[]) {         //类首先执行的方法
        System.out.println("Hello!");              //标准输出方法
    }
}
```

Application 应用程序的编辑、编译和运行方式详见第 11 章。

2. 字符集与标识符

Java 语言使用国际字符格式标准 (Unicode) 和浮点数 (IEEE 754)。Unicode 字符集采用 16 位编码, 其前 256 个字符与 ASCII 字符集完全一致, 此外还提供拉丁语、希腊语、汉字等其他语言文字。ASCII 字符集及其对应的 Unicode 值见附录 A。

Java 语言的关键字有 47 个, 关键字及说明见附录 B。标识符中允许出现大小写英文字母、下划线 (_)、美元符 (\$) 等。关键字、标识符之间采用空格作为分隔符。

Java 语言区分字母大小写, 约定关键字、变量名、对象名、方法名、包名中的字母全部小写; 若由多个单词构成标识符, 则首字母小写, 其后单词首字母大写, 如 toString; 类名首字母大写; 常量名全部字母均大写。

Java 提供三种注释方式: 单行注释、多行注释和文档注释, 格式如下:

```
// 单行注释
/* 多行注释 */
/** 多行文档注释 */
```

3. 数据类型

Java 语言的数据类型分为两大类: 基本数据类型和引用数据类型, 分类如图 0.1 所示。

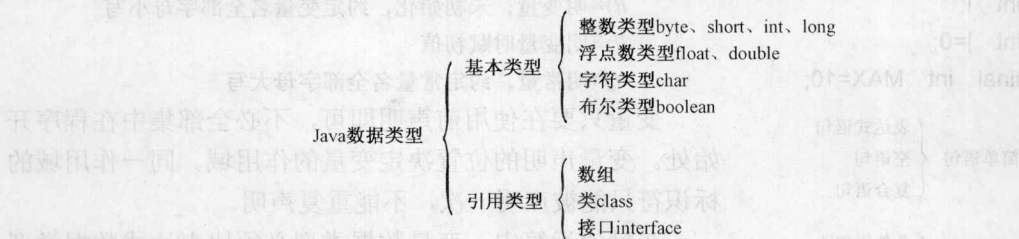


图 0.1 Java 数据类型

Java 语言定义了 8 种基本数据类型, 类型名是关键字, 说明如下。

- ◎ 每种整数类型都是带符号位的。整数默认类型为 int, 长整数后缀是 L 或 l。Java 提供十进制、八进制、十六进制等三种进制的整数表示形式。
- ◎ 浮点数默认类型为 double, float 浮点数后缀是 F 或 f。
- ◎ 布尔类型表示逻辑量, 只有真 (true) 和假 (false) 两个取值。注意, 布尔类型不是整数类型, true 和 false 不能转换成数值 0 或 1。
- ◎ 字符类型表示 Unicode 字符, 一个字符占用 2 字节。一个字符常量可用一对单引号括起来, 如 'A'、'a'、' '、'+' 等; 也可用 Unicode 值表示, 前缀是 "\u", 表示范围为 \u0000~

\uFFFF, 如\u0041 表示'A'。

不同整数类型、浮点数类型占用内存的大小不同, 数据取值范围也不同, 详见附录 C 中的表 C.1 和表 C.2。Java 转义字符及其对应的 Unicode 值见附录 C 中的表 C.3。

Java 语言不支持 C/C++ 中的结构体、联合和指针类型。结构体、指针类型可分别由 Java 语言的类和引用代替。

4. 运算符

Java 继承了 C++ 语言的多数运算符, 除了算术运算符、关系运算符、逻辑运算符、位运算符、强制类型转换符、条件运算符(?:)、括号运算符、点运算符(.)、逗号运算符(,)、赋值运算符(=)、复合赋值运算符之外, 增加字符串连接运算符(+)和对对象运算符(instanceof)。运算符及其优先级、结合性详见附录 D。

Java 是强类型语言, 对运算符与操作数类型的匹配要求比 C++ 语言严格, 说明如下。

① 参加算术运算的操作数只能是数值类型, 不同数值类型允许进行算术运算, 运算结果为范围大、精度高的那种数据类型。

② 参加关系运算的可以是数值、字符、字符串和布尔类型, 运算结果是布尔类型, 不是 0 或 1。整数与浮点数可以进行关系运算, 整数与字符不能进行关系运算。

③ 参加逻辑运算的操作数必须是布尔类型, 不能是 0 或 1, 运算结果仍是布尔类型。逻辑运算符&&(条件与)和|| (条件或)具有短路计算功能。

④ 位运算仅用于整数类型。

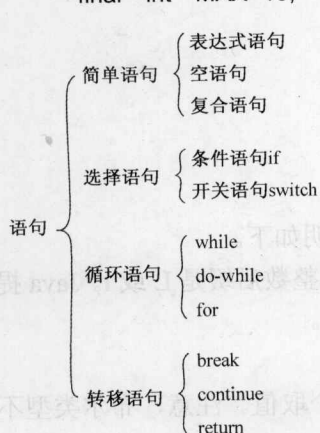
5. 变量与常量

变量有 4 个基本要素: 名字、类型、值和作用域。变量声明的语法格式如下:

[修饰符] 类型 变量 [=表达式] {, 变量 [=表达式]}

用关键字 final 声明的是最终变量, 只能进行一次赋值, 也称为常量。例如:

```
int i;           //声明变量, 未初始化, 约定变量名全部字母小写
int j=0;         //声明变量时赋初值
final int MAX=10; //声明常量, 约定常量名全部字母大写
```



变量只要在使用前声明即可, 不必全部集中在程序开始处。变量声明的位置决定变量的作用域。同一作用域的标识符只能被声明一次, 不能重复声明。

在赋值运算中, 变量数据类型必须比表达式数据类型占用的字节数长。如果表达式的数据类型较短, 则自动将其转化为较长的数据类型后再赋值; 反之, 需要采用强制类型转换, 例如:

```
int i =(int) 99L;           //数值的强制类型转换
```

并不是任意两个数据类型都可以进行强制类型转换的。

0.2.2 流程控制语句

Java 语言提供结构化语句实现程序的流程控制, 语句分类如图 0.2 所示。

图 0.2 Java 语句分类