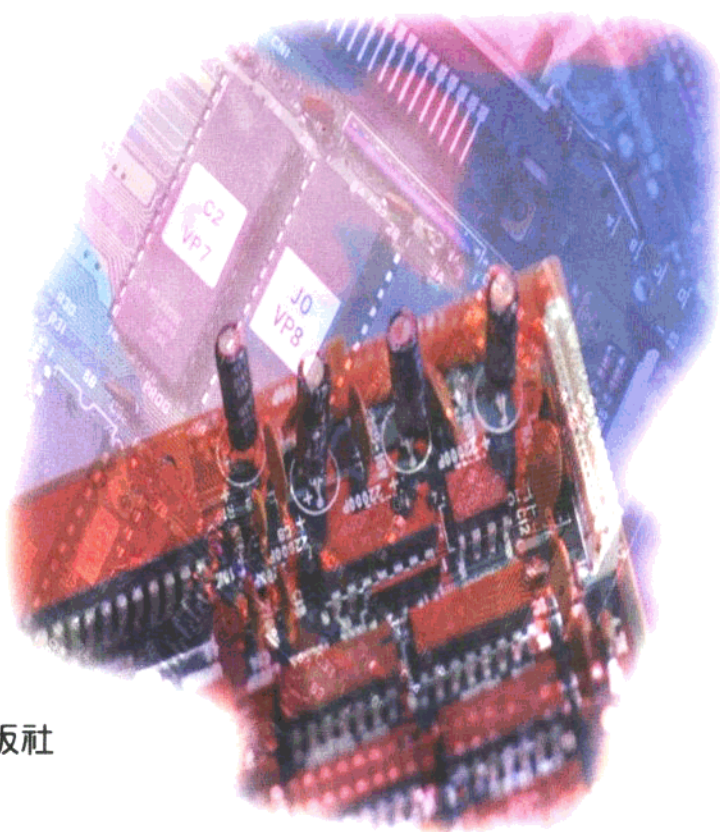


山东省高校统编教材

Java语言 基础教程

山东省教育厅组编



石油大学出版社



前 言

以计算机为核心的现代信息技术,正在对人类社会的发展产生难以估量的影响。计算机是人类创造思维的产物,反过来计算机又发挥了人脑思维的延伸与拓展的作用,成为人类思考、计算、运筹与谋划的有力工具。各个行业都要求其专业技术人员不仅要熟悉本专业领域知识,而且要能够利用计算机解决本专业领域的实际问题。面对 21 世纪信息社会的到来,计算机信息技术教育已是高校素质教育不可缺少的重要组成部分。人们已经意识到,在信息时代,计算机跟语言一样,已是人类社会每时每刻都不可缺少的;计算机基础教育已经与数学、英语等基础课程同等重要,甚至更具有实用性。计算机已经成为“人类通用智力工具”。计算机应用水平的高低已经成为衡量一个合格专门人才的指标之一。为此,学生学习计算机的热情空前高涨,学校也越来越重视,绝大部分院校除了开设《计算机文化基础》外,有些专业还相应开设了其他计算机基础课程,以提高学生应用计算机的能力,适应社会的需要。

为了促进计算机教学的开展,山东省教委于 1995 年发布了《关于加强计算机教学的意见》,陆续组织编写了《计算机文化基础》(有 DOS 和 Windows 两种版本)、《Foxpro 原理与程序设计》、《FORTRAN 语言程序设计》、《Pascal 语言程序设计》、《C 语言程序设计》、《Visual Foxpro 实用教程》、《Visual Basic 程序设计》、《C/C++语言程序设计》、《Internet 基础教程》等教材,开展了计算机文化基础和计算机应用基础教学考试,逐步将计算机基础教学引上了规范化的道路,有力地推进了我省高校计算机教学工作的开展。

计算机的发展日新月异。目前,Internet 席卷全球,得到迅猛发展;以网络为中心的网络计算模式也日益普及。Java 语言具有面向对象、平台无关性、安全性、鲁棒性和多线程等优良特性,为用户提供了良好的程序设计环境,特别适合于 Internet 的应用开发,成为网络时代最重要的语言之一。现在它已发展成为一门技术,包括 Java 的芯片技术、Java 的编译技术、Java 的数据库连接技术、基于 Java 的信息家电的连网技术(Jini 技术)、企业信息服务的综合求解方案技术(Enterprise JavaBeans, EJB)等等。它可以用来进行商业软件的开发,从事信息服务产业,甚至进行科学研究等。在西方发达国家,几乎所有著名的高等学校都先后开设了有关 Java 技术的课程。我国也正在掀起使用 Java 的热潮,从网上购物、网上银行、远程教学、虚拟课堂、远程医疗和诊断,直到“电子政府”,都在纷纷使用 Java 技术。为了满足不同层次、不同硬件水平、不同教学要求的学校的教学需要,保持教材的先进性、实用性和教学引导作用,我们组织有关人员编写了这本《Java 语言基础教程》,供我省各高校作为非计算机专业计算机文化基础后续课程的选用教材,以及计算机专业及其相关专业网络程序设计语言课程的选用教材。

教材建设是一项系统工程,需要不断地改进。希望各高校广大师生,在使用本教材的过程中,积极提出修改意见,以使其不断得到提高和完善。

山东省教育厅
2000 年 12 月

编者的话

为了推动我省高校计算机基础教学上一个新台阶,1997年,省教育厅组织有关人员推出了一套计算机基础教育教材,为推动我省计算机教学并使之规范化作出了积极的贡献。计算机技术总是在不断地发展,原来的DOS环境和单机环境下的教材显然已经不能满足教学需要了。为此,省教育厅组织有关人员编写了Windows环境和网络环境下的新教材。这本《Java语言基础教程》就是这一系列替代教材中的一种。我们的编写原则是既要符合教学实际,又要力求实用性、先进性,并对教学建设起到促进作用。在这一原则指导下,教材选择了目前广泛使用的Java 2(JDK 1.2)为版本,深入浅出地介绍了它的基本功能和开发工具的使用。

面向21世纪,教材内容、教学方法、教学手段的改革是我们每位教育工作者的重要任务。在编写本教材的过程中,每位作者都力图在内容编排、叙述方法上留有两个空间:一个是教师发挥的空间;一个是学生自学的空间。培养学生自学的能力、获取知识的能力是非常重要的。在上述思想的指导下,本教材主要具有以下特点:

1. 内容丰富,既有深度又有广度。主要是为了适应不同专业、不同层次的学生以及课时不同的需要。有些内容书中加注了“*”号。建议非计算机专业的学生选用不加“*”的内容,计算机专业及其相关专业重点讲述加注“*”号的内容。授课教师也可以根据实际情况选用相关内容。本教材参考学时为:计算机专业36学时;非计算机专业48学时。

2. 每章的“本章小结”是对一章内容的简略总括,因各章内容重要性不同而详略不同,对读者掌握本章内容和教学基本要求会有帮助。

全书共分十五章,其中第一、二、九、十、十一章由刘其成编写,第三、四、五章由夏方道编写,第六、七章由肖明编写,第八章由李凯里编写,第十二、十三、十四、十五章由张民编写。

在本书的编写过程中,得到了山东省教育厅高教处领导同志的大力支持,在此表示衷心感谢。

由于编写时间仓促,水平有限,书中难免有错误、不当之处,恳请各位同仁、读者不吝赐教。

编者

2001年1月

目 录

第一章 程序设计基础	1
1.1 程序设计语言	1
1.2 算法及其描述	3
1.2.1 算法的特征	3
1.2.2 算法的描述	3
1.3 结构化程序设计方法	3
1.3.1 基本控制结构	3
1.3.2 自顶向下、逐步细化与模块化程序设计	4
1.4 面向对象程序设计方法	5
1.4.1 概述	5
1.4.2 面向对象的基本概念	6
1.4.3 面向对象程序设计方法的特点	8
1.4.4 面向对象的程序设计语言	12
本章小结	13
习 题 一	13
第二章 Java 语言概述	15
2.1 Java 语言的历史与现状	15
2.1.1 Java 语言的历史	15
2.1.2 Java 语言的现状	15
2.2 Java 的工作机制	16
2.3 Java 语言的特点	17
2.3.1 Java 语言	17
2.3.2 Java Applet	19
2.3.3 丰富的类库	19
*2.3.4 Java 和 C、C++	19
2.4 Java 的开发工具	20
2.4.1 常见的 Java 开发工具简介	20
2.4.2 JDK1.2 开发工具	21
2.5 Java 的程序结构	24
2.5.1 Java Application 结构	24
2.5.2 Java Applet 结构	27
2.5.3 Java 程序结构	29
2.6 基本输入/输出处理	30
2.6.1 标准输入输出	30
2.6.2 数据输入输出流	32

2.6.3	BufferedReader.....	33
2.7	异常处理语句简介.....	33
	本章小结.....	34
	习题二.....	35
第三章	Java 语言基础.....	36
3.1	标识符与保留字.....	36
3.1.1	标识符.....	36
3.1.2	保留字.....	36
3.2	表达式.....	37
3.2.1	表达式的概念.....	37
3.2.2	表达式的组成.....	37
3.3	数据类型.....	38
3.3.1	数据类型的划分.....	38
3.3.2	简单数据类型.....	38
3.4	常量.....	40
3.4.1	布尔型常量.....	40
3.4.2	字符型常量.....	40
3.4.3	整型常量.....	41
3.4.4	浮点型常量.....	42
3.4.5	字符串常量.....	42
3.5	变量.....	42
3.5.1	变量的概念.....	42
3.5.2	变量的声明.....	43
3.5.3	变量的有效范围.....	44
3.5.4	变量的赋值.....	45
3.5.5	类型转换.....	45
3.6	运算符.....	46
3.6.1	算术运算符.....	47
3.6.2	关系运算符.....	48
3.6.3	逻辑运算符.....	48
3.6.4	位运算符.....	49
3.6.5	赋值运算符.....	52
3.6.6	条件运算符.....	52
3.6.7	运算符的优先级和结合性.....	52
3.7	注释语句.....	53
	本章小结.....	54
	习题三.....	54
第四章	结构化编程.....	56
4.1	分支语句.....	56

4.1.1 条件语句 if.....	56
4.1.2 多分支语句 switch.....	59
4.1.3 分支语句使用举例	61
4.2 循环语句	63
4.2.1 while 语句	63
4.2.2 do-while 语句	64
4.2.3 for 语句.....	65
4.2.4 循环嵌套	66
4.2.5 循环语句使用举例	67
4.3 与程序转移有关的其他语句	69
4.3.1 continue 语句	69
4.3.2 break 语句	70
4.3.3 return 语句.....	71
4.3.4 转移语句使用举例	71
4.4 数 组	72
4.4.1 数组的概念	72
4.4.2 数组的定义	73
4.4.3 静态数组初始化	74
4.4.4 数组元素的引用	74
4.4.5 多维数组	76
4.5 应用举例	78
本 章 小 结	82
习 题 四	83
第五章 面向对象编程	85
5.1 类	85
5.1.1 类声明	85
5.1.2 类体	87
5.1.3 构造函数	91
5.2 对 象	91
5.2.1 对象的生成	91
5.2.2 对象的使用	92
5.2.3 对象的清除	93
5.3 继 承	95
5.3.1 继承关系的定义	95
5.3.2 成员变量的继承和隐藏	96
5.3.3 方法的继承、重载和覆盖	97
5.3.4 this 和 super.....	103
5.3.5 构造函数的继承与重载	103
5.4 多 态	105

5.4.1	编译时多态	105
5.4.2	运行时多态	107
5.5	Object 类	108
5.6	接 口	110
5.6.1	接口的基本概念	110
5.6.2	接口的定义	110
5.6.3	接口的实现	111
5.6.4	接口的继承	112
5.7	包	113
5.7.1	package 语句	113
5.7.2	import 语句	114
5.7.3	编译和运行包	115
5.8	Java 的类库结构	115
5.8.1	Java 类库	115
5.8.2	HotJava 类库	116
5.8.3	其他类库	116
5.9	静态初始化和垃圾回收机制	117
5.9.1	静态初始化	117
5.9.2	垃圾回收机制	119
5.10	Inner Classes (内部类)	120
5.11	应用举例	121
本 章 小 结		124
习 题 五		125
第六章	异常处理	127
6.1	编程中的错误	127
6.1.1	编译错误	127
6.1.2	运行错误	127
6.2	异常与异常类	128
6.2.1	异常处理机制概述	128
6.2.2	异常类结构与组成	130
6.2.3	系统定义的运行异常	131
6.2.4	用户自定义的异常	132
6.3	异常的处理	134
6.3.1	try-catch-finally	134
6.3.2	多异常的处理	136
6.4	异常的抛出	140
6.4.1	系统自动抛出的异常	140
6.4.2	语句抛出的异常	140
6.5	应用举例	141

本章小结	144
习 题 六	144
第七章 常用基础类库与工具类库	145
7.1 语言基础类库	145
7.1.1 数据类型类	145
7.1.2 Math 类	146
7.1.3 System 类	149
7.2 字符串	149
7.2.1 字符串常量的表示与 String 类	150
7.2.2 字符串变量表示与 StringBuffer 类	153
7.2.3 Java Application 命令行参数	155
7.3 向量类	156
7.4 枚举接口	160
7.5 日期类	161
7.6 随机类	163
本章小结	164
习 题 七	165
第八章 图形用户界面	166
8.1 Java 图形用户界面概述	166
8.2 一个最简单的窗口	167
8.3 绘制用户自定义成分	168
8.3.1 绘制图形	169
8.3.2 显示文字	170
8.3.3 控制颜色	172
8.3.4 显示图像	174
8.4 基于 AWT 的图形用户界面	175
8.4.1 AWT 简介	175
8.4.2 事件处理	177
8.4.3 标准组件及其事件	180
*8.5 基于 JFC/ Swing 的图形用户界面	227
8.5.1 JFC/Swing 简介	227
8.5.2 标准组件	228
8.5.3 事件处理	228
8.5.4 Swing 应用举例	228
本章小结	231
习 题 八	232
第九章 Applet	234
9.1 HTML 简介	234
9.2 Applet 基本工作原理	234

9.3	Applet 执行过程	235
9.3.1	Applet 的创建	235
9.3.2	Applet 的生命周期与绘图周期	236
9.3.3	Applet 和 HTML	240
9.3.4	同时实现 Applet 与 Application	242
9.4	Applet 的通信	244
9.4.1	同页 Applet 间的通信	244
9.4.2	Applet 和浏览器之间的通信	247
9.4.3	Applet 的网络通信	250
9.5	Applet 用户界面设计	250
9.5.1	标准组件	251
9.5.2	图形、文本与颜色	252
9.5.3	图像	254
9.5.4	声音	255
* 9.6	JFC Applet	257
9.7	应用举例	257
本章小结		261
习 题 九		262
第十章	输入/输出处理	263
10.1	输入/输出流概述	263
10.1.1	流的概念	263
10.1.2	输入/输出流层次	263
10.2	常用字节流处理	266
10.2.1	InputStream 和 OutputStream	266
10.2.2	过滤流	269
10.2.3	管道流	270
10.2.4	内存的读/写	270
10.2.5	顺序输入流	271
10.3	常用字符流处理	271
10.3.1	Reader 和 Writer	271
10.3.2	InputStreamReader 和 OutputStreamWriter	272
10.3.3	BufferedReader 和 BufferedWriter	273
10.4	文件处理	274
10.4.1	文件与目录管理	274
10.4.2	文件的顺序处理	278
10.4.3	随机访问文件	279
10.5	对象的串行化	282
10.5.1	对象串行化概述	282
10.5.2	对象串行化的实现	282

10.5.3 定制串行化	284
10.6 应用举例	285
本章小结	288
习题十	289
第十一章 多线程机制	291
11.1 线程	291
11.1.1 基本概念	291
11.1.2 线程的状态与生命周期	291
11.1.3 线程调度与优先级	292
11.2 Runnable 接口与 Thread 类	293
11.2.1 Runnable 接口	293
11.2.2 Thread 类	293
11.3 多线程的实现	295
11.3.1 创建 Thread 类的子类	295
11.3.2 实现 Runnable 接口	296
11.3.3 两种方法的比较	297
11.4 线程的同步与死锁	298
11.4.1 线程同步	298
11.4.2 线程死锁	301
11.5 动画制作	302
本章小结	304
习题十一	305
*第十二章 网络编程	306
12.1 机器的标识	306
12.1.1 InetAddress 类	306
12.1.2 服务器和客户机	308
12.1.3 端口	308
12.2 TCP 套接字 (Socket) 通信	309
12.2.1 TCP 套接字通信的一般结构	309
12.2.2 Socket 类	309
12.2.3 ServerSocket 类	310
12.2.4 一个简单的 Client/Server 程序	311
12.2.5 多用户机制	313
12.3 数据报通信	316
12.3.1 DatagramSocket 类	316
12.3.2 DatagramPacket 类	317
12.3.3 一个简单的 Client/Server 程序	317
12.4 URL (统一资源定位器)	320
12.4.1 URL 的格式	320

12.4.2	URL 类	320
12.4.3	URLConnection 类	322
12.5	应用举例	322
12.5.1	服务器端程序	323
12.5.2	客户机端程序	327
本章小结		332
习题 十二		332
*第十三章	JDBC 技术	333
13.1	关系数据库简介	333
13.1.1	数据处理技术的发展	333
13.1.2	数据库和数据库管理系统	333
13.1.3	关系数据库和关系数据库管理系统	334
13.1.4	结构化查询语句 (SQL)	334
13.2	JDBC 概述	334
13.2.1	JDBC 与 ODBC 的比较	335
13.2.2	JDBC 驱动程序	336
13.3	通过 JDBC 访问数据库的步骤	337
13.3.1	数据库的 URL	337
13.3.2	连接数据库	338
13.3.3	发送 SQL 语句	338
13.3.4	处理结果集 (ResultSet)	339
13.3.5	关闭连接	340
13.3.6	两层模型和三层模型	341
13.4	通过 JDBC 访问数据库的实现	341
13.4.1	表的结构操作	342
13.4.2	数据库记录操作	343
13.4.3	数据库查询、检索	344
13.5	应用举例	345
本章小结		347
习题 十三		348
*第十四章	Servlet 与 JSP	349
14.1	Servlet 简介	349
14.1.1	什么是 Servlet	349
14.1.2	Servlet 与 Applet 的区别	349
14.1.3	Servlet 运行环境	349
14.1.4	Servlet 的安装和调用	350
14.2	一个简单的 Servlet 实例	351
14.3	Servlet 实现方法	353
14.3.1	HTTP 协议	353

14.3.2 Servlet 的类与方法	354
14.4 JSP 简介	360
14.5 应用举例	363
本章小结	368
习题 十四	369
*第十五章 其他开发技术	370
15.1 反射机制	370
15.1.1 简介	370
15.1.2 Class 类	370
15.1.3 Member 接口及实现类	373
15.2 JavaBeans 组件	376
15.2.1 概述	376
15.2.2 JavaBeans 开发、测试工具	379
15.2.3 构造 JavaBeans 组件的一般过程	379
15.2.4 EJB	380
15.3 RMI	382
15.3.1 简介	382
15.3.2 RMI 系统的一般结构	382
15.3.3 一个简单的 RMI 实例	384
15.4 CORBA 与 Java	386
15.4.1 CORBA 概述	386
15.4.2 JavaIDL: Java 的 CORBA 接口	387
15.4.3 CORBA 与 RMI 比较	388
15.5 Jini 简介	388
15.5.1 Jini 的体系结构	389
15.5.2 JavaSpace	390
15.5.3 应用	390
15.5.4 结论	390
15.6 Java 安全性	391
15.6.1 Java 安全体系结构	391
15.6.2 安全管理器	391
15.6.3 JDK1.2 中的权限	392
15.6.4 Java 加密体系结构	393
15.6.5 Java 浏览器的安全性策略	393
本章小结	394
习题 十五	394

第一章 程序设计基础

1.1 程序设计语言

计算机系统由硬件系统和软件系统构成。计算机软件包括了使计算机运行所需的各种程序及其有关文档资料。它可分为系统软件和应用软件两类。系统软件是计算机设计制造者提供的使用和管理计算机的软件。通常包括操作系统、语言处理程序、数据库管理系统、网络系统和常用服务程序等。应用软件是专门为某一应用目的而用系统软件编制的软件系统。

计算机的工作是用程序来控制的，离开了程序，计算机将一事无成。程序是用程序设计语言按问题的要求事先编写的。程序设计语言是人与计算机交流信息的一种工具。随着计算机的发展，程序设计语言主要发展为机器语言、汇编语言和高级语言三代。

1. 机器语言

机器语言是一种面向计算机硬件的、计算机能直接识别和执行的语言，它是以二进制代码表示的指令集合。机器语言的优点是占用内存少、执行速度快。缺点是对于不同计算机系列的机器语言是不同的，而且机器语言程序的可读性和可移植性都很差。

【例 1-1】设有一元二次方程 $f(x) = ax^2 + bx + c$ ，若 $a=12$ ， $b=8$ ， $c=256$ ， $x=5$ ，求 $f(x)$ 的值。

将函数的表达式变换为： $(ax+b)x+c$ 。表 1-1 中的程序为 PC 机器语言程序。

表 1-1 PC 机器语言程序

地 址	指令代码	二进制指令代码	说 明
11B0:0100	B00C	1011 0000 0000 1100	将 00001100 送入 AL 寄存器
11B0:0102	B305	1011 0011 0000 0101	将 00000101 送入 BL 寄存器
11B0:0104	F6E3	1111 0110 1110 0011	$(AL)*(BL) \rightarrow AX$ 值为 3CH
11B0:0106	0408	0000 0100 0000 1000	$(AL)+8 \rightarrow AL$ 值为 44H
11B0:0108	F6E3	1111 0110 1110 0011	$(AL)*(BL) \rightarrow AX$ 值为 154H
11B0:010A	050001	0000 0101 0000 0000 0000 0001	$(AX)+0100H \rightarrow AX$ 值为 254H
11B0:010D	CD2	1100 1101 0010	中断返回，程序执行结束

上述机器语言程序的地址和指令均为二进制代码。从这个程序可以看出，只有计算机的专业人员才可能去研究它。这对于推广、普及计算机是非常不利的。

2. 汇编语言

汇编语言是在机器语言的基础上，用符号去代替机器语言中指令的操作码、地址码等代码部分，以帮助记忆，所以称这种符号为助记符。用汇编语言编写的源程序必须翻译成机器语言程序才能执行，这个过程叫汇编。完成汇编任务的程序叫做汇编程序。源程序中的汇编语句一般和机器语言的指令一一对应。汇编语言的优点是比机器语言易学易记。缺点同机器语言，即通用性差，随机而异。例 1-1 的汇编语言源程序如下：

```
MOV AL,0C
MOV BL,05
MUL BL
```

```
ADD AL,08
MUL BL
ADD AX,0100
INT 20
```

3. 高级语言

第三代语言高级语言是 20 世纪 50 年代后开发的,它们比较接近于人们习惯用的自然语言和数学表达式。高级语言优点是通用性强、可移植性好,可以在不同的机器上运行,程序简短易读,便于维护,极大地提高了程序设计的效率和可靠性。常用的高级语言有:

(1) FORTRAN: 第一种高级语言,适用于科学和工程计算,目前应用面还较广。

(2) Pascal: 第一种结构化程序设计语言,适用于教学、科学计算、数据处理和系统软件开发等,目前逐渐被 C 语言所取代。

(3) C 语言: 程序简练、功能强,适用于系统软件、数值计算、数据处理等,目前成为高级语言中使用得最多的语言之一。

(4) BASIC: 简单易学。至今 BASIC 语言已有许多高级版本,尤其 Visual Basic 是面向对象的程序设计语言,给广大用户在 Windows 环境下开发软件带来了福音。

(5) Java 语言: 一种新型的跨平台分布式面向对象程序设计语言。通过全球的 Internet 网,Java 以它简单、安全、可移植、面向对象、多线程处理和具有动态等特性引起世界范围的广泛关注,被誉为“Internet 上的世界语”。实现例 1-1 的 Java 程序如下:

```
public class SimpleProgram{
    public static void main(String args[]){
        int a=12,b=8,c=256,x=5,f;
        f=(a*x+b)*x+c;
        System.out.println("("+"a+"*"+x+"+"+b+"")*"+x+"+"+c+"="+f);
    }
}
```

对于高级语言编写的程序,计算机是不能识别和执行的。要执行高级语言编写的程序,首先要将高级语言编写的程序通过语言处理程序翻译成计算机能识别和执行的二进制机器指令,然后供计算机执行。一般将用高级语言编写的程序称为“源程序”,而把已翻译成机器语言的程序称为“目标程序”,不同的高级语言编写的程序必须通过相应的语言处理程序进行翻译。

计算机将源程序翻译成机器指令时,通常分两种翻译方式:一种为“编译”方式,另一种为“解释”方式。编译方式是通过相应语言的编译程序将源程序一次全部翻译成目标程序,再经过连接程序的连接,最终处理成可直接执行的可执行程序。解释方式是通过相应的解释程序将源程序逐句解释翻译成一组机器指令,翻译一句执行一句,边翻译边执行。解释程序不产生将被执行的目标程序,而是借助解释程序直接执行源程序本身。执行过程中有错,机器显示出错信息,修改后继续执行。解释方式对初学者较有利,便于查找错误,但效率低。

注意,除 Java 语言外,前面所指的高级语言的可移植性是针对源程序的,一旦被编译成目标程序,就只能在相应类型的计算机上运行。但它的源程序还可通过其他类型计算机的相应编译程序翻译为它的目标程序,即达到源程序的移植。对于 Java 语言的可移植性,参

见第二章。

1.2 算法及其描述

算法就是计算机解题的步骤。一个程序的好坏，在很大程度上决定于它所采用的算法。为了设计出高质量的程序，先要对算法的基本概念、主要特征及算法的描述方法等方面的知识有所了解。

1.2.1 算法的特征

从概念上讲，算法是一组命令（或语句）序列，它用以解决某一类特定的问题，完成特定类型的任务。把解决问题或完成任务的步骤找出来，构成一个命令序列，这就是算法的基本组成部分。算法一般具有以下特征：

（1）结果性。一个算法通常总要有输入也要有输出，特殊情况下则可无输入但还要有输出。即一个算法必定要产生结果，它的输入信息可以是一个、多个或零个，但输出信息至少要有一个。一个不能产生结果的算法是没有用处的，也就不能称之为算法。

（2）有效性。一个算法应能有效地完成指定的任务。这就要求算法的每一步骤都是可以实现的，所包含的每个操作都是可执行的。例如，当 $B=0$ 时， A/B 就无法执行。

（3）确定性。一个算法的每一步都必须有明确定义，不允许有二义性。

（4）有穷性。一个算法必须在执行有穷步之后结束，在有限的时间内完成。如果一个算法在计算机上需要运行几个月甚至更长时间，就失去了实用价值。

1.2.2 算法的描述

把算法用某种方式表示出来，就是算法的描述。描述算法常用的有以下几种方法：

（1）用自然语言描述。一个算法可以用中文或英文等自然语言来描述。用自然语言描述算法，易于被人们接受和理解。但这种方法容易造成文字累赘和多义性。

（2）用流程图描述。流程图使用不同的几何图框表示相应的算法操作，在框内用简洁的字符来说明具体的操作内容，用流程线连接各个图框。基本流程图见图 1-1。

（3）用 N-S 图描述。N-S 图又称盒图，是由 Nassi 和 Shneiderman 两人提出的，它是一种对算法进行结构化描述的图形方法。用 N-S 图描述算法有很多优点：首先，它只提供了几种标准的图形符号，可以保证算法描述的质量；第二，N-S 图形象直观，具有良好的可见性，例如选择的范围、循环的范围都是一目了然的，易于理解；第三，N-S 图简单、易学易用，已被许多软件开发人员所接受。基本 N-S 图见图 1-2。

1.3 结构化程序设计方法

结构化程序设计方法的指导思想是以良好的结构来保证程序容易阅读和理解，它主要包括两个方面的内容：一是对编写程序时使用控制结构的要求，强调使用三种基本控制结构，避免使用可能降低程序清晰性的转向语句（goto 语句）；二是在软件开发的设计与实现过程中，提倡采用自顶而下和逐步细化的方法。

1.3.1 基本控制结构

结构化程序设计的三种基本控制结构是顺序结构、分支结构和循环结构。

(1) 顺序结构。是由严格地按顺序排列的一串语句（或语句块）所构成的。这串语句（块）的排列顺序，就是该结构的执行顺序。

(2) 分支结构。有两种形式：一种是条件判断，它是由一个分支条件，以及一个或两个语句块所构成的。该结构的执行方式由分支条件决定，当条件满足时，执行某一语句块，当条件不满足时，执行另一语句块。另一种是选择判断，它一般是由多个分支条件及对应的多个语句块构成的。该结构的执行方式是按顺序检查每个条件，当遇到成立的条件时，只执行其对应的语句块。

(3) 循环结构。由循环条件和语句块（循环体）构成。分为两种，即“当型”（WHILE 型）循环和“直到型”（UNTIL 型）循环。它们的差别是：“当型”循环把条件判断放在循环体前面，若不满足条件就一次循环也不执行；“直到型”循环把条件判断放在循环体后面，循环至少要被执行一次。

三种基本结构算法的流程图见图 1-1、N-S 图见图 1-2。

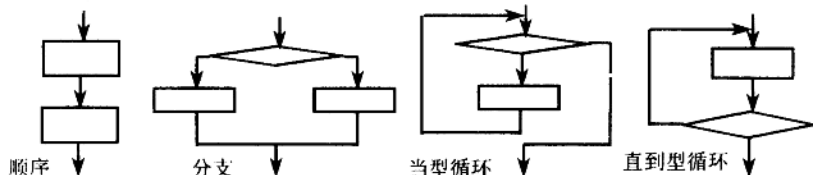


图 1-1 三种基本结构的流程图

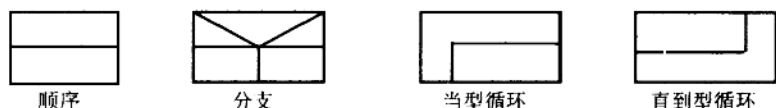


图 1-2 三种基本结构的 N-S 图

它们的共同特点是：①只有一个入口和一个出口；②每一部分都有一条从入口到出口的路径经过，所以各部分都有机会被执行；③不存在死“循环”（永无休止的循环）。

已经证明，任何复杂的问题，都可以用上述三种基本结构构成其结构化的程序，反过来说一个结构化的程序也一定能分解为上述基本结构。每一个基本结构只有单入口、单出口的性质，使得不同基本结构之间的关系简单，互相依赖性小。它们都是互相独立的单元，在修改程序时，可将各个基本结构孤立出来进行修改，既简单又不会影响其他基本结构中的语句。

1.3.2 自顶向下、逐步细化与模块化程序设计

为了编写一个程序有两种不同的思考方法，一种叫做“自顶向下”的方法，另一种叫做“自底向上”的方法。例如编著一部书时，往往先确定目标和主题，然后构思分成若干章，定出各章题目和大致内容，把每章分为若干节，再细分为若干段……，即由粗到细，逐步具体化。这种先有全局，先进行整体设计，然后再进行下一层的设计，逐步地实现精细化的方法称为“自顶向下，逐步细化”方法。日常生活中常使用这种方法。建筑师先设计鸟瞰图、平面图、立面图，在得到认可后再进行各部分的设计工作（由粗到细地逐步展开），就是自顶向下的设计方法。

自底向上方法则与此相反，是先从具体局部人手，由若干个局部构成整体。建筑工人盖

房子从一砖一瓦入手，先完成局部后完成整体，就是一种“自底向上”的方法。

显然，“自顶向下”方法能做到胸有全局、通盘考虑，不致顾此失彼、头重脚轻，结构化程序设计提倡这种方法。过去不少程序设计人员采用的往往是“自底向上”的方法，拿到题目未作充分考虑就动笔一个语句一个语句地写程序，这种程序通常漏洞百出，质量较差，可读性差。

“自顶向下，逐步细化”，就是把一个复杂的任务分解成较小的子任务，子任务还可能再分解为子子任务，整个复杂任务的完成就依赖于这些子任务、子子任务的逐个完成。这种思想体现在程序设计工作中就是模块化程序设计的思想。

所谓模块化，是将一个大任务分成若干个较小的部分，每一部分承担一定的功能，称为“功能模块”，各个模块都可以分别由不同的人编写程序和调试，便于组织人力完成较复杂的任务。

自顶向下、逐步细化和模块化技术三者是紧密结合的，是结构化程序设计方法不可分割的要点。

总之，自顶向下、逐步细化是结构化程序设计方法的核心，只用三种基本结构表示算法是保证设计出结构化程序的手段。

1.4 面向对象程序设计方法

面向对象的程序设计方法的出现和广泛应用是计算机软件技术发展中的一个重大变革和飞跃。它能够更好地适应当今软件开发在规模、复杂性、可靠性和质量、效率上的种种需求，因而被越来越多地推广和使用，其方法本身也在这诸多实践的检验和磨练中日趋成熟、标准化和体系化，逐渐成为目前公认的主流程序设计方法。本节将针对面向对象程序设计的一些基本概念和基本方法做一简单介绍，使读者对它的体系、原则、基本思想和特点有一定的了解。

1.4.1 概述

从 20 世纪 60 年代末开始，随着计算机软件系统规模的扩大和复杂性的增加，导致软件的开销也惊人地增加，但软件系统的可靠性和可维护性却明显地降低了，我们称之为软件危机。产生这种危机的根本原因是：我们用以求解问题的问题空间与所用的求解方法的方法空间的不一致性。传统的结构化程序设计方法以“过程”为中心来构造系统、设计程序，由于“过程”是不稳定和多变，以此设计程序，其思维成果的可重用性必然较差。为了解决这一危机，必须寻找一种新的程序设计方法，使得我们用以求解问题的方法空间与所要求解的问题空间尽可能地一致。面向对象的程序设计方法正体现了这一原则。采用面向对象的程序设计方法，以“对象”为中心来设计软件，符合人们对客观世界的一般认识过程，能够达到问题空间和方法空间尽可能地统一。

当我们遵照面向对象的思想进行程序设计时，首先进行面向对象的分析（Object Oriented Analysis, OOA），它的任务是了解问题域内该问题所涉及的对象、对象间的关系和作用（即操作），并构造该问题的对象模型，力争使这个模型能真实地反映出所要解决的实质问题。在这一过程中，抽象是最本质、最重要的方法。针对不同的问题性质选择不同的抽象层次，过简和过繁都会影响到对问题的本质属性的了解和解决。然后进行面向对象的设计（Object