

程序

计算机网络技术系列教材

• 陈强 主编 • 孙建华 副主编 •

程序设计基础

程序设计基础

程序设计基础

程序设计基础

程序设计基础

程序设计基础

林建民 朱喜福 编著

程序设计基础

程序设计基础

程序

程序设计基础

程序设计基础

程序设计基础

设计基础

程序设计基础

程序设计基础

程序设计基础

程序设计基础

人民邮电出版社
www.pptph.com.cn

计算机网络技术系列教材

程序设计基础

林建民 朱喜福 编著

人民邮电出版社

图书在版编目 (C I P) 数据

程序设计基础 / 林建民, 朱喜福编著; ——北京: 人民邮电出版社, 2002.2
(计算机网络技术系列教材)

ISBN 7-115-09925-1

I. 程... II. ①林... ②朱... III. JAVA 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字 (2002) 第 002475 号

计算机网络技术系列教材

程序设计基础

◆ 编 著 林建民 朱喜福
责任编辑 潘春燕

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 315@pptph.com.cn
网址 <http://www.pptph.com.cn>
读者热线: 010-67180876
北京汉魂图文设计有限公司制作
北京朝阳展望印刷厂印刷
新华书店总店北京发行所经销

◆ 开本: 787×1092 1/16
印张: 16.75
字数: 398 千字 2002 年 2 月第 1 版
印数: 5 001-10 000 册 2002 年 3 月北京第 2 次印刷

ISBN 7-115-09925-1/TP·2656

定价: 22.00 元

本书如有印装质量问题, 请与本社联系 电话: (010)67129223

内 容 简 介

本书以 Java 语言为例讲述程序设计的基础知识。内容主要包括：Java 语言基础；运用 Java 语言阐述面向对象的程序设计基础理论、主要原则和思维方法，并用易于理解的程序实例帮助读者较全面和深入地建立起面向对象的程序设计概念；通过面向对象的 Java 语言的类对链表、堆栈、队列、二叉树以及图形结构等做了详尽的描述，并使用这些类解决一些实际问题；对算法的设计方法及如何提高算法的效率等进行了深入细致的分析，对排序和查找等实用的数据处理做了深入浅出的讲述。本书最后还对软件工程的一些知识进行了较全面的介绍。

本书可作为高职高专计算机网络专业程序设计语言课程的教材，也可作为高等院校计算机相关专业面向对象程序设计课程的教材和教学参考书。

丛书前言

计算机网络技术是近年来兴起的、发展相当迅速的计算机新技术。21 世纪的今天不会使用网络就等于不会使用计算机。IBM 公司早在 20 世纪 80 年代就提出：网络就是计算机。因此，必须十分重视计算机网络的普及与推广应用，使更多的人能够利用网上的资源，提高工作水平和效率。

为适应社会的需要和计算机网络技术的发展，全国高等院校的各个专业都开设了有关计算机网络技术课程，特别是近年来高等职业教育的发展，急需以计算机网络应用为主的实用教材，使学生在学计算机网络时，适当减少那些枯燥难懂的理论，取而代之的是建网、管网、上网的实操和网络的应用开发技术。根据这一发展趋势，我们组织了一批学术水平高、教学经验丰富、实践能力强的老师编写了这套计算机网络技术系列教材。

本套教材共 11 本。为了便于教学，本套教材每本均配有实训内容，每一章都有内容提要 and 习题。

本套教材在编写过程中力求做到：网络理论以必需、够用为度，注重网络实用技术及实际应用的介绍，并以实际需要的技术、操作和使用技巧为主体，使学生在学计算机网络理论的同时，掌握相关的实操和应用技巧。本套教材突出了内容新、讲述方法浅显、重应用和技术的特点，适合高等职业学校、成人高校、本科院校设立的二级职业技术学院学生和自学计算机网络应用和开发的人员使用。

本套教材已经被选为北京市高等教育自学考试高职自考“网络技术应用与服务”专业的指定教材。

目 录

第 1 章 计算机基础	1
1.1 计算机的硬件和软件	1
1.1.1 计算机系统的硬件组成	1
1.1.2 计算机系统的软件组成	2
1.1.3 二进制数与编码	2
1.2 计算机语言	7
1.2.1 机器语言与汇编语言	7
1.2.2 高级语言	8
1.2.3 程序的编辑、编译与运行	8
练习题	8
第 2 章 Java 语言初步	10
2.1 Java 语言的产生和发展	10
2.2 Java 的特点和应用	11
2.2.1 Java 语言的特点	11
2.2.2 Java 程序的应用	13
2.3 简单的 Java 语言程序	14
2.3.1 Java 应用程序	14
2.3.2 Java 小应用程序	15
2.3.3 Java 程序的编辑、编译和运行	17
2.4 Java 语言的简单数据类型	18
2.4.1 数据类型综述	18
2.4.2 简单数据类型的实例化——变量与常量	19
2.4.3 整数类型数据	20
2.4.4 浮点型（实型）数据	22
2.4.5 字符类型数据	24
2.4.6 布尔类型数据	25
2.4.7 各类型数据间的相互转换	26
2.5 Java 语言的运算符和表达式	28
2.5.1 运算符综述	28
2.5.2 算术运算符	29
2.5.3 关系运算符	32
2.5.4 逻辑运算符	33

2.5.5	位运算符	34
2.5.6	赋值运算符	35
2.5.7	三目条件运算符	36
2.5.8	表达式	37
2.6	Java 语言的控制结构	41
2.6.1	顺序结构	41
2.6.2	分支结构	41
2.6.3	循环结构	50
2.6.4	各种结构之间的嵌套	59
2.6.5	循环体内的跳转:break 语句和 continue 语句	60
2.6.6	结构化程序设计	62
2.7	Java 语言的方法	62
2.7.1	方法定义	63
2.7.2	方法的调用	67
2.7.3	方法重载	68
2.8	Java 语言的数组	70
2.8.1	定义数组	70
2.8.2	数组元素的访问	71
	练习题	73
第 3 章	面向对象的程序设计基础	75
3.1	面向对象的程序设计基础	75
3.1.1	类 Time	76
3.1.2	构造函数	80
3.1.3	成员访问控制	85
3.1.4	对象的高级使用	90
3.1.5	对象的清除——finalizer 方法	96
3.1.6	static 类成员	97
3.1.7	数据抽象和封装	99
3.2	深入面向对象的程序设计	100
3.2.1	继承性	101
3.2.2	多态性	112
3.2.3	继承性和多态性综述	122
	练习题	122
第 4 章	算法分析与算法设计	124
4.1	算法的概念、性质和操作	124
4.1.1	算法的概念	124

4.1.2	算法的性质	125
4.1.3	算法的操作	125
4.1.4	算法的控制结构	126
4.1.5	算法与程序设计	127
4.2	算法的描述工具	129
4.2.1	流程图	129
4.2.2	框图	131
4.2.3	结构化程序设计语言	132
4.2.4	几种算法描述工具的比较	134
4.2.5	控制结构的应用实例	134
4.3	算法分析	142
4.3.1	算法的时间复杂度	142
4.3.2	算法的空间复杂度	144
4.4	算法设计的基本方法	144
4.4.1	穷举法	144
4.4.2	递推法	160
4.4.3	递归法	164
4.4.4	模拟法	170
	练习题	174
第 5 章	数据结构基础	176
5.1	数据结构的基本概念	176
5.2	线性结构	177
5.2.1	基本概念	177
5.2.2	线性表的顺序存储结构	178
5.2.3	线性表的链式存储结构	190
5.2.4	堆栈	197
5.2.5	队列	202
5.3	树型结构	204
5.3.1	树的基本概念	205
5.3.2	二叉树及其遍历	205
5.3.3	二叉排序树	208
5.4	图型结构	213
5.4.1	图的定义及基本术语	213
5.4.2	图的存储结构与图的遍历	214
5.4.3	图的应用	216
5.5	排序	218
5.5.1	选择排序	218

5.5.2	冒泡排序	220
5.5.3	插入排序	222
5.5.4	快速排序	224
5.6	查找	227
	练习题	229
第 6 章	软件工程基础知识	232
6.1	软件工程概述	232
6.1.1	软件危机与软件工程	232
6.1.2	软件的生存周期	233
6.2	软件定义时期的任务	233
6.2.1	问题定义	234
6.2.2	可行性分析	234
6.2.3	需求分析	236
6.3	软件开发时期的任务	236
6.3.1	总体设计	236
6.3.2	详细设计	237
6.3.3	程序编码与调试	237
6.4	模块化原则	238
6.4.1	模块的划分	238
6.4.2	模块的联系与评价	238
6.5	软件的测试	239
6.5.1	测试的目的	239
6.5.2	测试的方法与步骤	239
	练习题	240
第 7 章	实际技能训练	242
7.1	实训 1 —— 熟悉 Java 程序设计环境	242
7.2	实训 2 —— 数据类型、运算符和表达式	244
7.3	实训 3 —— 运算符、表达式和分支语句	245
7.4	实训 4 —— 选择结构和循环结构	246
7.5	实训 5 —— 循环和跳转	247
7.6	实训 6 —— 方法和数组	248
7.7	实训 7 —— 定义类和使用类对象（一）	249
7.8	实训 8 —— 定义类和使用类对象（二）	249
7.9	实训 9 —— 算法的基本特性	251
7.10	实训 10 —— 算法设计的基本方法	251
7.11	实训 11 —— 算法的优化	251

7.12	实训 12 —— 递推和递归算法	252
7.13	实训 13 —— 顺序表的逻辑结构、物理结构和算法	253
7.14	实训 14 —— 链表的定义、作用和应用	253
7.15	实训 15 —— 堆栈和队列	254
7.16	实训 16 —— 二叉树与排序二叉树	254
7.17	实训 17 —— 排序与查找	255

第 1 章 计算机基础

本章提要

- 计算机的硬件和软件
- 计算机语言

1.1 计算机的硬件和软件

计算机系统由硬件和软件两大部分组成，硬件是构成计算机的设备实体，软件则是指为了运行、管理和维修计算机而编制的各种程序。

1.1.1 计算机系统的硬件组成

计算机的硬件系统由运算器、控制器、存储器、输入/输出（I/O）接口、输入/输出设备和电源系统等组成，如图 1-1 所示。

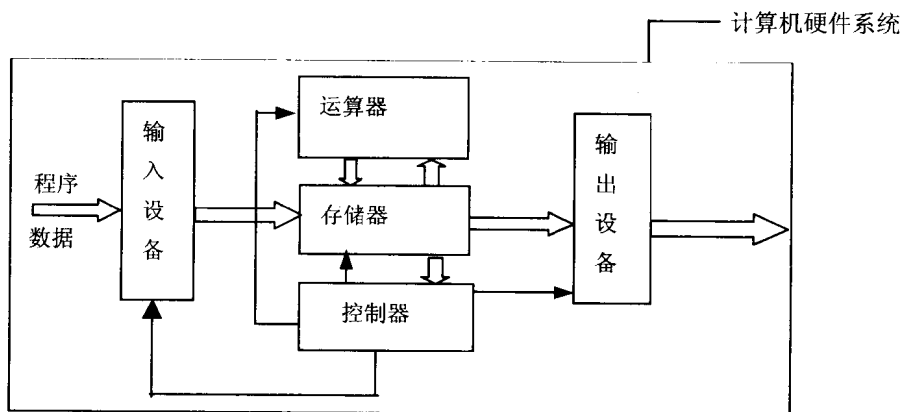


图 1-1 计算机系统的硬件组成

其中，运算器和控制器合称为中央处理器（Central Processing Unit, CPU）。CPU 与存储器、I/O 接口以及电源系统等组成了计算机系统的“主机”，输入/输出设备则被称为外部设备。

（1）存储器

存储器用来存放指令和数据。

计算机系统存储器是指内存储器（内存），也称主存储器（Main Memory）。通常所说的外存储器，也就是辅助存储器，是指硬盘、软盘等。

（2）运算器

运算器是进行算术运算和逻辑运算的部件（Arithmetic Logic Unit, ALU），它也是指令的

执行部件。

(3) 控制器

控制器是计算机的指挥中心。它负责对指令进行译码，产生出整个指令系统所需要的全部操作的控制信号，控制运算器、存储器以及输入/输出接口等部件完成指令规定的操作。

(4) 输入设备

输入设备通过输入接口电路将程序和数据输入内存，最常见的输入设备是键盘。

(5) 输出设备

CPU 通过输出接口电路将程序运行的结果及程序、数据送到输出设备上。最常见的输出设备有显示器和打印机等。

1.1.2 计算机系统的软件组成

不配有软件的计算机被称为“裸机”，裸机是不能干任何事情的。计算机软件又分为系统软件和应用软件两大类。

1. 系统软件

使用和管理计算机本身的软件称为系统软件，系统软件主要分下述几类。

- (1) 面向计算机管理的软件，如操作系统（Windows、Unix 等）；
- (2) 数据库管理系统，如 Sybase、FoxPro、Oracle、PowerBuilder 等；
- (3) 计算机网络管理软件。
- (4) 语言处理和服务性软件。

2. 应用软件

应用软件是计算机用户在各自的业务领域中开发和使用的各种软件，是为解决某一实际问题而编制的程序。例如，天气预报中的数据处理、企业的财务管理、工厂的仓库管理以及学校的辅助教学等。JDK（Java Developers Kits）、Jbuilder 和 Visual J++ 就是用于 Java 开发的几个应用软件。

1.1.3 二进制数与编码

数制是数的表示方法。在日常生活中，最常用的就是十进制数。由于用电子器件表示两种状态比较容易实现，也便于存储和计算，因此在计算机中一般采用二进制数。由于二进制数书写格式冗长，不便阅读，所以，在程序设计中往往又使用十六进制数和八进制数等。

1. 常用计数制

(1) 十进制数

在程序设计中，人们广泛使用十进制数。十进制数的特点是：每一位由 0~9 这 10 种数码组成，故基数为 10，高位权是低位权的 10 倍，加减运算的法则是“逢十进一，借一当十”。

(2) 二进制数

在计算机内部，所有信息都以二进制数形式出现。二进制数的特点是：只有两个不同的数字符号，即 0 和 1，因此基数为 2，高位权是低位权的 2 倍，加减运算的法则是“逢二进一，借一当二”。

(3) 十六进制数

十六进制数是二进制数的另一种书写格式。十六进制数的特点是：每 4 位二进制数用 1

位十六进制表示，它的每一位由 0~9 和 A~F 这 16 种数码组成，因此基数是 16，高位权是低位权的 16 倍，加减运算法则是“逢十六进一，借一当十六”。

(4) 八进制数

八进制数也是二进制数的另一种书写形式。把 3 位二进制数作为一组，每一组用等值的八进制数（实际上是十进制数的 0~7）来表示。八进制数的特点是：每一位由 0~7 这 8 种数码组成，基数是 8，高位权是低位权的 8 倍，加减运算的法则是“逢八进一，借一当八”。

(5) BCD 码 (Binary Coded Decimal)

BCD 码是计算机中十进制数的表示方法，BCD 码用 4 位二进制数码表示 1 位十进制数字。

用 4 位二进制数编码表示 1 位十进制数，有多种表示方法，常用的是 8421 BCD 码，它的表示规则以及与十进制数之间的等价关系如表 1-1 所示。

表 1-1 BCD 码与十进制数的转换

二进制数	十进制数	BCD 码	二进制数	十进制数	BCD 码
0000	0	0000	1000	8	1000
0001	1	0001	1001	9	1001
0010	2	0010	1010	10	非法 BCD 码
0011	3	0011	1011	11	非法 BCD 码
0100	4	0100	1100	12	非法 BCD 码
0101	5	0101	1101	13	非法 BCD 码
0110	6	0110	1110	14	非法 BCD 码
0111	7	0111	1111	15	非法 BCD 码

例如： $(3456)_{10} = (0011\ 0100\ 0101\ 0110)_{\text{BCD}}$

BCD 码是十六进制数的一个子集，1010~1111 是非法的 BCD 码。

2. 数制转换

(1) 任意进制数转换为十进制数

二进制数、十六进制数以及任意进制数转换为十进制数的方法较为简单，根据按权展开式把每个数位上的代码和该数位的权值相乘，再求累加和即可得到等值的十进制数。例如：

$$(101.11)_2 = 1 \times 2^2 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} = (5.75)_{10}$$

$$(713)_8 = 7 \times 8^2 + 1 \times 8^1 + 3 \times 8^0 = (459)_{10}$$

$$(\text{FA4})_{16} = 15 \times 16^2 + 10 \times 16^1 + 4 \times 16^0 = (4004)_{10}$$

(2) 十进制数转换为二进制数

当十进制数转换为二进制数时，根据该十进制数的类型来决定转换方法。

① 十进制整数转换为二进制数

方法为：除 2 取余。即十进制整数被 2 除，取其余数，商再被 2 除，取其余数，……，直到商为 0 时结束运算。然后把每次的余数按倒序规律排列，就得到等值的二进制数。例如：

$$N = (13)_{10} = (1101)_2$$

运算过程为：13 ÷ 2 = 6 余数为 1 D_0

$6 \div 2 = 3$	余数为 0	D_1
$3 \div 2 = 1$	余数为 1	D_2
$1 \div 2 = 0$	余数为 1	D_3

所以 $N = D_3 D_2 D_1 D_0 = (1101)_2$

② 十进制纯小数转换为二进制数

方法为：乘 2 取整。即把二进制纯小数乘以 2，取其整数（不参加后继运算），乘积的小数部分再乘以 2，取整……，直到乘积的小数部分为 0。然后把每次乘积的整数部分按正序规律排列，即为等值的二进制数。例如：

$$N = (0.8125)_{10} = (0.1101)_2$$

运算过程为： $0.8125 \times 2 = 1.625$	乘积的整数部分为 $D_{.1} = 1$
$0.625 \times 2 = 1.25$	乘积的整数部分为 $D_{.2} = 1$
$0.25 \times 2 = 0.5$	乘积的整数部分为 $D_{.3} = 0$
$0.5 \times 2 = 1$	乘积的整数部分为 $D_{.4} = 1$

所以 $N = (0.1101)_2$

③ 十进制带小数转换为二进制数

方法为：整数部分除 2 取余，小数部分乘 2 取整，然后再进行组合。例如：

$$(13.8125)_{10} = (1101.1101)_2$$

(3) 二进制数转换为十六进制数

以小数点为界，4 位二进制数为一组，不足 4 位用 0 补全，整数部分在前面补 0，小数部分在后面补 0，然后每组用等值的十六进制数表示。例如：

$$(1101110.11)_2 = (0110\ 1110.1100)_2 = (6E.C)_{16}$$

在计算机语言中，十六进制数用后缀“H”表示。所以， $(6E.C)_{16}$ 也可以用 6E.CH 来表示。

(4) 十六进制数转换为二进制数

把十六进制数的每一位用等值的 4 位二进制数来替换，例如：

$$(17E.58)_{16} = (0001\ 0111\ 1110.0101\ 1000)_2 = (101111110.01011)_2$$

3. 字符的编码

计算机处理的信息除了数字之外还有字母、符号等非数值数据，字母、符号统称为字符。在计算机系统中使用的字符最常用的是美国信息交换标准代码，即 ASCII 码（American Standard Code for Information Interchange）。标准 ASCII 码用 7 位二进制数作为字符的编码，但由于计算机通常用 8 位二进制数代表一个字节，故标准的 ASCII 码也写成 8 位二进制数，但最高位 D_7 恒为 0。 $D_6 \sim D_0$ 位代表字符的编码。

关于各个字符对应的 ASCII 码，读者可查看相关参考书。

4. 计算机中的码制

计算机中使用的数分为无符号数和有符号数两种。在计算机中表示一个有符号数的最常用的方法是：把二进制数的最高一位定义为符号位，符号位为 0 表示正数，符号位为 1 则代表负数，这样就把符号“数值化”了。在有符号数的运算中，其符号位上的 0 或 1 也被看作是数值的一部分来参加运算。

通常，把用“+”、“-”表示的数称为真值数，把用符号位上的 0、1 来表示正、负数的数称为机器数。机器数可以用不同的方法来表示，常用的有原码、反码和补码

表示法。

(1) 机器数的原码、反码和补码

数 X 的原码记作 $[X]_{\text{原}}$ ，反码记作 $[X]_{\text{反}}$ ，补码记作 $[X]_{\text{补}}$ 。

例如，当机器字长 $n=8$ 时有

符号 ↓	符号位 ↓
设真值数 $X=+7=+0000111$	原码机器数写成 $[X]_{\text{原}}=00000111$
$X=-7=-0000111$	$[X]_{\text{原}}=10000111$
$X=+0=+0000000$	$[X]_{\text{原}}=00000000$
$X=-0=-0000000$	$[X]_{\text{原}}=10000000$
设真值数 $X=+7=+0000111$	反码机器数写成 $[X]_{\text{反}}=00000111$
$X=-7=-0000111$	$[X]_{\text{反}}=11111000$
$X=+0=+0000000$	$[X]_{\text{反}}=00000000$
$X=-0=-0000000$	$[X]_{\text{反}}=11111111$
设真值数 $X=+7=+0000111$	补码机器数写成 $[X]_{\text{补}}=00000111$
$X=-7=-0000111$	$[X]_{\text{补}}=11111001$
$X=+0=+0000000$	$[X]_{\text{补}}=00000000$

由上述例子可以得出以下结论。

- ① 机器数比真值数多一个符号位。
- ② 正数的原、反、补码与真值数相同。
- ③ 负数原码的数值部分与真值相同；负数反码的数值部分为真值数按位取反；负数补码的数值部分为真值数按位取反末位加 1。
- ④ 没有负零的补码，或者说负零的补码和正零的补码相同。
- ⑤ 由于补码表示的机器数更适合运算，为此，计算机系统中负数一律用补码表示。
- ⑥ 机器字长为 n 位的原码数，其真值范围是 $-(2^{n-1}-1) \sim +(2^{n-1}-1)$ ；
机器字长为 n 位的反码数，其真值范围是 $-(2^{n-1}-1) \sim +(2^{n-1}-1)$ ；
机器字长为 n 位的补码数，其真值范围是 $-(2^{n-1}) \sim +(2^{n-1}-1)$ 。

(2) 整数补码的运算

了解补码数是怎样进行加减运算的，我们先介绍以下几个概念。

① 模

模是计量器的最大容器。一个 4 位寄存器能够存放 0000~1111 共计 16 个数，因此它的模为 2^4 。一个 8 位寄存器能够存放 00000000~11111111，共计 256 个数，因此它的模为 2^8 ，依次类推，32 位寄存器的模是 2^{32} 。

② 有模的运算

凡是用器件进行的运算都是有模运算。例如，利用 32 位的运算器，当运算结果大于等于 2^{32} 时，超出的部分被运算器自动“丢弃”（保存在进位标志寄存器中）。

③ 求补运算

下面是一个由真值求补码的例子，机器字长 $n=8$ 。

假设真值 $X=+75$ ，则 $[X]_{\text{补}}=01001011$ ； $X=-75$ ，则 $[X]_{\text{补}}=10110101$ 。

即：对 $[+X]_{\text{补}}$ 按位取反末位加 1，就得到 $[-X]_{\text{补}}$ 。

对 $[-X]_{\text{补}}$ 按位取反末位加 1，就得到 $[+X]_{\text{补}}$ 。

因此，“求补运算”就是指对一补码机器数进行“按位取反，末位加 1”的操作。通过求补可以得到该数负真值的补码。

鉴于补码数具有这样的特征，用补码表示有符号数，则减法运算就可以用加法运算来替代，这样，在计算机中只需设置加法运算器就可以了。

④ 整数补码的运算

补码加法的规则是：

$$[X+Y]_{\text{补}}=[X]_{\text{补}}+[Y]_{\text{补}}$$

其中， X 、 Y 为正负数皆可，符号位参加运算。

补码减法的规则是：

$$[X-Y]_{\text{补}}=[X]_{\text{补}}+[-Y]_{\text{补}}$$

其中， X 、 Y 为正负数皆可，符号位参加运算。

例 1.1 设 $X=66$ ， $Y=51$ ，以 2^8 为模，补码运算求 $X+Y$ ， $X-Y$ 。

解：因为 $[X]_{\text{补}}=01000010$ ， $[Y]_{\text{补}}=00110011$ ， $[-Y]_{\text{补}}=11001101$

$[X]_{\text{补}}$	01000010	$[X]_{\text{补}}$	01000010
+) $[Y]_{\text{补}}$	00110011	$[-Y]_{\text{补}}$	11001101
-----	$[X+Y]_{\text{补}}$ 01110101	-----	$[X-Y]_{\text{补}}$ 100001111



被运算器丢失

所以

$$X+Y=+117$$

$$X-Y=+15$$

例 1.2 以 2^8 为模，补码运算求 $66+99$ ， $-66-99$ 。

解：因为 $[66]_{\text{补}}=01000010$ ， $[99]_{\text{补}}=01100011$

$$[-66]_{\text{补}}=10111110, [-99]_{\text{补}}=10011101$$

$[66]_{\text{补}}$	01000010	$[-66]_{\text{补}}$	10111110
+) $[99]_{\text{补}}$	01100011	$[-99]_{\text{补}}$	10011101
-----	$[66+99]_{\text{补}}$ 10100101	-----	$[-66-99]_{\text{补}}$ 101011011



被运算器丢失

所以

$$66+99=-91$$

$$-66-99=+91$$

由上两例可以看出，不论加数是正数还是负数，只要直接用它们的补码（包括符号位）相加，当结果不超出补码表示范围时，运算结果即为正确的补码。但当运算结果超出补码范围时，结果就不正确了。

(3) 无符号数

在处理某些问题时,若参与运算的数都是正数,如学生成绩、职工工资、字符编码以及内存地址等,则存放这些数时再保留符号位已没有实际意义。为了扩大寄存器所能表示的数的范围,取消了符号位。这样,一个数的最高位就不再是符号位而是数值的一部分了,这样的数被称为“无符号数”。因此,

8 位字长的无符号数,其数值范围是 $0 \sim 255$;

32 位字长的无符号数,其数值范围是 $0 \sim 2^{32}-1$;

计算机部件只知道寄存器的内容是一串 0、1 代码。也就是说,只有程序员才能决定一个数的物理意义。

假设一个 8 位寄存器的内容是 $(11111111)_2$ 。若它是无符号数,真值为 255;若他是补码数,则真值为 -1;若它是反码数,则真值为 -0。

(4) 进位和溢出

例 1.1 和例 1.2 引出了两个极为重要的概念:进位和溢出。在介绍了无符号数以后,我们可以进一步讨论溢出。

① 进位

运算之后,符号位向更高位的进位称为进位。进位值无论是 0 还是 1,都被运算器“丢弃”。如例 1.1 所示。对于有符号数的运算,进位值不能统计在运算结果之中。对于无符号数的运算,进位值是结果的一部分,不能随意丢弃,所以要妥善处理无符号数加法运算的溢出。

② 溢出

运算结果超出了计算机所能表示的数的范围,称为溢出。例 1.2 的运算结果就是因为超出了计算机所能表示的数的范围(8 位),所以结果是错误的。

关于这两个概念,如果读者有兴趣,可以参考硬件技术方面的相关书籍。

1.2 计算机语言

程序员用各种程序设计语言书写程序,有些可以直接被计算机所理解,而有一些则需要中间的解释和翻译步骤。目前正在使用的计算机语言有几百种,可以将它们分成 3 大类:机器语言、汇编语言和高级语言。

1.2.1 机器语言与汇编语言

机器语言是用二进制代码(0 或 1)书写的且能被机器识别的一种语言。它是由计算机的硬件设计定义的。机器语言是面向机器的,即某种机器语言仅可在某特定型号的计算机上使用。用机器语言编写程序既烦琐费时、容易出错又难以记忆、识别,人们用起来十分困难。所以用机器语言程序设计,无法推广应用。于是人们研制了一种利用助记符号来代表机器语言指令进行程序设计的语言,称为汇编语言。汇编语言程序必须通过“汇编程序”翻译,转换为机器语言表示的目标程序,再通过连接形成可执行程序,才能在计算机中执行。