
重点大学 计算机基础课程教材
计算机程序设计基础
辅导与实验教程
赵 宏 主编
吴文虎 主审
清华大学出版社 · 北京交通大学出版社

重点大学计算机基础课程教材

计算机程序设计基础辅导与实验教程

赵 宏 主编

吴文虎 主审

清华大学出版社

北京交通大学出版社

·北京·

内 容 简 介

根据教育部非计算机专业计算机基础课程教学指导分委员会提出的高等学校计算机基础课程的教学基本要求组织编写了本书，用于实验课程和辅助学习。

全书分为 3 部分：第 1 部分为学习要点，包括重点、难点、内容总结、补充阅读内容及例题解析；第 2 部分为编译环境介绍，主要介绍 Visual C++ 6.0 的基本使用方法、调试程序的方法和常见错误分析；第 3 部分为实验指导，其中共编写了 10 个实验，每个实验由 3～6 个题目组成，实验题目有验证型、编程实验和设计型，难度由浅入深，循序渐进，可供不同层次的读者使用。

本书可作为高等学校非计算机专业的计算机基础课程教材，也可作为培训教材和读者自学参考书。

图书在版编目（CIP）数据

计算机程序设计基础辅导与实验教程 / 赵宏主编. —北京：清华大学出版社；北京交通大学出版社，2005.2

（重点大学计算机基础课程教材）

ISBN 7-81082-342-6

Ⅰ. 计… Ⅱ. 赵… Ⅲ. 程序设计-高等学校-教材 Ⅳ. TP311.1

中国版本图书馆 CIP 数据核字（2005）第 007555 号

责任编辑：陈 芳　　特邀编辑：李 莉

出 版 者：清 华 大 学 出 版 社　邮编：100084　电话：010-62776969　http://www.tup.com.cn

北京交通大学出版社　邮编：100044　电话：010-51686414　http://press.bjtu.edu.cn

印 刷 者：北京东光印刷厂

发 行 者：新华书店总店北京发行所

开　本：185×260　印张：9.25　字数：226 千字

版　次：2005 年 2 月第 1 版　2005 年 2 月第 1 次印刷

书　号：ISBN 7-81082-342-6 / TP・135

印　数：1～5000 册　定价：14.00 元

前　言

本书是根据教育部非计算机专业计算机基础课程教学指导分委员会提出的高等学校计算机基础课程教学基本要求编写的《计算机程序设计基础》一书的配套学习辅导教材。编写本书的目的是帮助读者更好地掌握程序设计的基本方法，掌握调试程序的工具、方法和技巧，提高实际操作的能力。

全书内容分为学习要点、编译环境和实验指导 3 部分。

第 1 部分（第 1～12 章）为学习要点，包括重点、难点、内容总结、补充阅读内容及例题解析。本部分用简明易懂的方式对各章知识点内容进行了概括和总结，指出学习的重点、难点；增加了补充阅读材料，针对每一章的知识点补充了许多课外阅读材料，扩大知识面；对每章知识点补充了一些例题，对例题的解题思路和方法进行了详细分析，帮助读者进一步巩固所学内容。

第 2 部分（第 13～15 章）为编译环境介绍，包括实验环境和调试程序的方法。在第 13 章中主要介绍了 Visual C++ 6.0 的基本使用方法；在第 14 章中通过实例介绍了 VC++ 6.0 环境下如何调试程序；在第 15 章中主要介绍了调试程序的方法，并对常见错误进行了分析。

第 3 部分（第 16～17 章）为实验指导，包括实验安排、实验要求和实验内容。在第 16 章中简要介绍了实验安排，提出了程序设计和调试方面应该进行的训练及要达到的要求，介绍了如何撰写实验报告；第 17 章是具体实验内容，编写了 10 个实验，每个实验有多个实验内容，对每个实验内容都明确提出了实验目的，对实验中的每个题目都提出了具体的设计和调试要求，对一些较难的题目则给出了解决问题的思路和算法说明。实验题目难度由浅入深，循序渐进，可供不同层次的读者使用。

全书由北京交通大学的 5 位教师集体编写完成。第 1，2，4，9，10，15，16 章及第 17 章的实验 1～实验 7、实验 10 由赵宏编写；第 3，8 章由李会霞编写；第 5，6 章由靳小燕编写；第 7，13，14 章由鲍志斌编写；第 11，12 章及第 17 章的实验 8 和实验 9 由翟高寿编写。全书由赵宏统稿。

清华大学计算机系吴文虎教授以广博的知识，耐心细致地审阅了全稿，提出了许多宝贵的修改意见，在此表示感谢。

本书是作者在几年来的教学实践基础上编写的，希望为程序设计实践环节的教学起到积极的促进作用。但由于时间仓促、作者水平有限，本教材编写中难免有不足和疏漏，欢迎读者提出宝贵意见和建议，以供再版时改进。

作　者

2005 年 1 月

目　录

第1部分　学习要点

第 2 部分 VC++编译环境

第 3 部分 实 验 指 导

第1部分　学习要点

第1章　概　述

1.1　本章重点、难点及内容要点

本章重点

- 结构化程序设计的基本思想。
- 用自顶向下、逐步细化的方法构造算法。
- 结构化程序设计的三种基本控制结构。
- 高级语言上机如何实现。

本章难点

- 用自顶向下、逐步细化的方法构造算法。

内容要点

1．计算机语言和计算机程序

人和计算机之间通信使用的语言为计算机语言。当今使用的计算机语言大致可以分为 3 类，即机器语言、汇编语言和高级语言。汇编语言与机器语言一样是面向机器的，属于低级语言，通用性较差。但汇编语言一直被人们所使用，主要是由于其具有执行速度快、占用存储空间小、对硬件操作灵活等特性。高级语言不再是面向机器的语言，而是面向过程的语言，接近于人类自然语言和数学语言。

将计算机要完成的操作用一条条指令按序排好，计算机一步一步执行了这个指令序列，也就完成了我们希望它做的工作，而且整个指令序列执行过程中不需要人来干预。为了解决某一特定问题，用某一种计算机语言编写的指令序列称为程序。

2．结构化程序设计方法

结构化程序设计（面向过程程序设计）支持自顶向下、逐步细化和模块化的结构化分析方法。结构化程序设计基本原理是用顺序、选择和循环 3 种基本结构，通过组合和嵌套实现任何单入口、单出口的程序。例如，图 1-1 所示整体上是一个顺序结构，它由选择结构和循环结构组合而成。图 1-2 所示是一个循环结构，循环体中包含了一个选择结构。

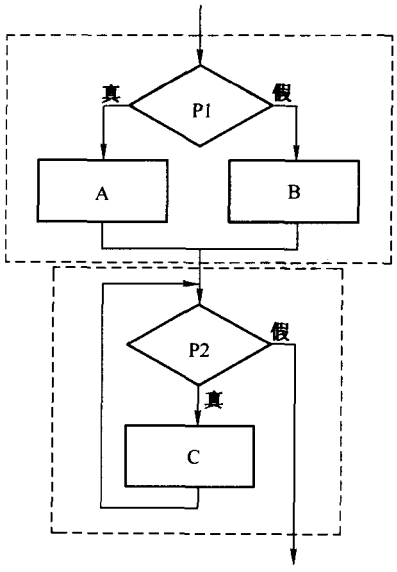

图 1-1 循环结构和选择结构的组合

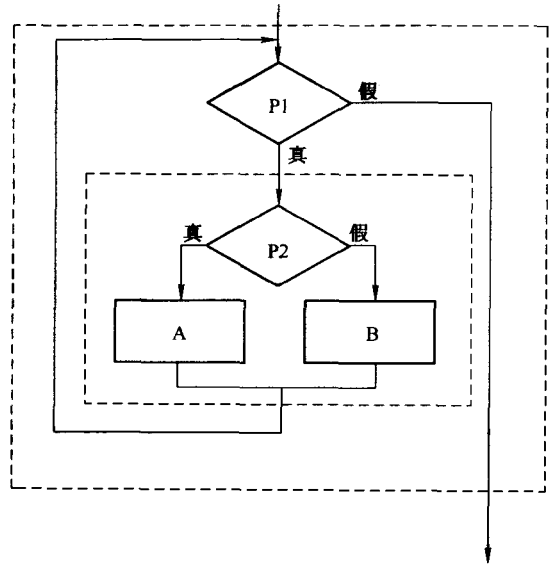

图 1-2 循环结构中嵌入选择结构

用结构化程序设计方法进行程序设计一般要经过以下几个步骤：

- 建立数学模型；
- 选定算法，用适当工具描述算法；
- 编程；
- 测试及调试。

3. 高级语言程序的建立、编译、连接及执行

开发一个高级语言程序需要经过编辑器生成源程序，用编译器生成目标程序，用连接器生成可执行文件以及运行已经生成的可执行文件 4 个过程。4 个过程中需要的编辑器、编译器、连接器等软件功能通常被集成到一个程序中，形成集成的开发环境。在本辅导教程的第 2 部分详细介绍了 VC++ 6.0 集成的编译环境。

1.2 补充阅读内容

1. 什么是计算机算法

广义地说所谓算法就是解决问题的方法和步骤。计算机算法包括数值算法和非数值算法两大类，数值算法的主要目的是求数值解，例如，求方程的根，求函数的定积分，求函数的最大值、最小值等；非数值算法的应用也十分广泛，例如，计算机售票、办公自动化、图书情报的检索等。

并不是任何解决问题的方法、步骤都能称得上是计算机算法，一个真正的算法必须满足以下 5 个特征。

1）有穷性

一个算法必须在执行有穷步骤之后结束，而不能是无限的。例 1-1 满足了这个特征，它的执行导致一个递减的正整数序列必然要终止。

2）确定性

算法的每一个步骤必须是确定的，不应当是含糊的，模棱两可的。对于每种情况，有待执行的动作必须严格规定，不能出现二义性。例如，对例 1-1 而言，确定性意味着 m 和 n 都是正整数，两个正整数相除余数仍是正整数，否则例 1-1 就是不确定的。

3）有 0 个或多个输入

所谓输入是指在算法开始执行之前，对算法中的输入变量给出初始值。例如，在执行例 1-1 时有两个输入，即 m 和 n。也有的算法可以没有输入。

4）有一个或多个输出

一个算法的目的就是为了求解。没有输出，这个算法也就没有意义了。例 1-1 有一个输出，即 m 和 n 的最大公约数。

5）可行性

算法中的每一个步骤都能精确地进行，而且进行有穷次即可完成。例如，例 1-1 中用整除测试一个整数是否为 0，置一个变量的值等于另一个变量的值，这些运算都是可行的，结果也是确定的。

2. 算法的表示方法

除了流程图、N-S 图、伪码以外，还可以用 PAD 图、判定表、判定树等表示算法。

1）PAD 图

PAD 是问题分析图（Problem Analysis Diagram）的缩写，它用二维树型结构的图来表示程序的控制流，描述的程序结构非常清晰，将这种图翻译为程序比较容易。描述例 1-1 算法的 PAD 图如图 1-3 所示。

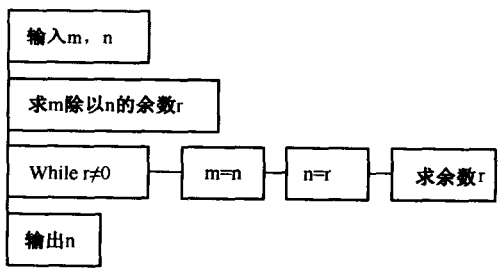

图 1-3 PAD 图

2）判定表

当算法中包含有多重嵌套的条件选择时，用流程图、N-S 图、PAD 图和伪码都不易清楚地描述，而判定表却可以清晰地描述复杂的条件组合与应做的操作之间的对应关系。

判定表分为 4 个区（见表 1-1）。一区列出所有条件类别，二区列出所有的条件组合，三区列出所有要执行的操作，四区内标出在相应的组合条件下，某个操作是否执行或执行情况。表 1-2 为使用判定表的一个实例。

表 1-1 判定表

一区条件类别	二区条件组合
三区操作	四区操作执行

表 1-2 使用判定表的一个实例

考试分数	≥620	≥620	<620	<620
单科成绩	有满分	有不及格	有满分	有不及格
发升级通知书	Y	Y	N	N
发免修单科通知书	N	N	Y	N
发留级通知书	N	N	Y	Y
发重修单科通知书	N	Y	N	N

1.3 例题解析

【例 1-1】 用自顶向下、逐步细化的方法写出求两个整数最大公约数的算法。

程序设计思路

例如 m=341，n=132，341 除以 132，商是 2，余数 r 为 77。因为 $r\neq0$，所以置 m=132，n=77，依此类推，最后 22 除以 11 时，r=0，算法终止，132 和 341 的最大公约数为 11。

使用辗转相除法求两个整数的最大公约数的过程如下。

（1）输入两个正整数 m，n。

（2）求 m 除以 n 的余数 r。如果 $r\neq0$，则重复执行（2.1），（2.2），（2.3）步，否则打印 n，算法终止。

(2.1) 把 n 值赋给 m

(2.2) 把 r 值赋给 n

(2.3) 求 m 除以 n 的余数 r

用“自顶向下、逐步细化”的方法求两个整数最大公约数。先用伪码表示一级算法。

一级算法

```
(1) Input two integer m and n
(2) Calculate the greatest common divisor
(3) Print the greatest common divisor
```

这里只用了顺序结构，即所列出的步骤是按顺序执行的。对一级算法中的每一步进行细分，就得到二级算法。由于一级算法中第 1 和第 3 步已经不需要再分解了，所以只对第（2）步进行细化。

二级算法

```
(2) Calculate the greatest common divisor
```

这条语句可以细化为：

```
(2.1) Set r to m%n
(2.2) While  r≠0
      Set m to n
      Set n to r
      set r to m%n
```

要用一个循环结构不断进行辗转相除。每次判断 r 不为 0 时，就用原来的被除数作为除数，用原来的余数作为被除数，再求余数。反复操作下去，直到余数 r 为 0，此时的被除数就是最大公约数。

用伪码描述算法如下：

```
start
input m, n
set r to m%n
while r≠0
    m=n
    n=r
    set r to m%n
print n
stop
```

【例 1-2】 有 200 名同学选修了程序设计这门课程，请编写程序，统计参加考试学生中不及格的人数、60～69 分的人数、70～79 分的人数、80～89 分的人数、90～99 分的人数、100 分的人数。用自顶向下、逐步细化的方法写出算法。

一级算法

```
(1) Initialize variables
(2) Input the 200 grades and count the counts by score
(3) Print the exam results
```

这里也只用了顺序结构。对一级算法中的每一步进行细分，就得到二级算法。这个算法中用计数器 failures 表示不及格人数、c60 表示 60～69 分的人数、c70 表示 70～79 分的人数、c80 表示 80～89 分的人数、c90 表示 90～99 分的人数、c100 表示 100 分的人数、counter 表示已经处理的数据个数，另外需要一个接收输入每个数据的变量 grade。

二级算法

```
(1) Initialize variables
```

可以细化为：

```
(1.1) Set failures to 0
(1.2) Set c60 , c70, c80 , c90, c100 to 0
(1.3) Set counter to 1
```

这里每个分数段人数通过累加求和得到，需要初始化为 0，计数器 counter 表示要处理第几个数据，需要初始化为 1，表示要处理第一个数据。

```
(2) Input the 200 grades and count the failures
```

这条语句可以细化为：

```
While counter<=200
    (2.1) Input the next grade
    (2.2) If the grade<60
```

```
            Add 1 to failures
        Else if grade<70
            Add 1 to c60
        Else if grade<80
            Add 1 to c70
        Else if grade<90
            Add 1 to c80
        Else if grade<100
            Add 1 to c90
        Else
            Add 1 to c100
    (2.3) Add 1 to counter
```

在每次循环时输入一个数据，在循环体内使用嵌套的选择结构确定该学生的成绩属于哪个分数段，则相应分数段的统计变量增加 1，之后计数器 counter 要增加 1。

```
(3) Print the exam results
```

可以细化为：

```
(3.1) Print the failures
(3.2) Print the c60, c70, c80, c90, c100
```

完整的细化过程如下。

```
Set failures to 0
Set c60 , c70, c80 , c90, c100 to 0
Set counter to 1
While counter<=200
    Input the next grade
    If the grade<60
        Add 1 to failures
    Else if grade<70
        Add 1 to c60
    Else if grade<80
        Add 1 to c70
    Else if grade<90
        Add 1 to c80
    Else if grade<100
        Add 1 to c90
    Else
        Add 1 to c100
     Add 1 to counter
Print the failures
Print the c60, c70, c80, c90, c100
```

第2章　程序设计初步

2.1　本章重点、难点及内容要点

本章重点

- C / C++结构化程序的基本结构。
- 基本数据类型整型、实型、字符型以及这些类型的常量和变量。
- 数据的算术运算和赋值运算。
- 数据的格式化输入 / 输出。
- 顺序结构程序设计。

本章难点

- 数据的格式化输入 / 输出中各种格式符的使用。

内容要点

1. C / C++结构化程序的基本结构

一般情况，一个 C / C++结构化程序的框架包含程序说明部分、预编译命令、主程序区、函数定义区几部分。一个程序可以包含多个函数，每个函数都要独立定义，其中有且只能有一个主函数。一个程序可以独立存在于一个文件中，也可以分成几个文件去存放。

程序说明部分主要是对程序进行简要说明，包括程序的名称、完成的功能、编写人及最后修改时间等内容，这部分为注释，虽然不是程序必需的，但却是非常重要的部分，可以方便阅读程序，在实际软件开发中应该养成良好的习惯。

以“#”开头的为预处理命令。要调用系统函数，就应该把描述函数所在的头文件包含进来，如使用输入 / 输出函数，要加上#include <stdio.h>，使用数学函数，应该加上#include <math.h>，等等。

主程序以 main()函数开始，是整个程序的入口，在 main 函数中主要包括如下内容。

1）声明部分

- 局部变量定义，例如：

```
int  i1, i2, sum;
```

- 对所调用函数的声明，例如：

```
int add(int x, int y);
```

2）执行部分

● 函数调用，例如系统函数调用：

```
scanf("%d, %d", &i1, &i2);
```

自定义函数的调用：

```
sum = add(i1, i2);
```

● 一般运算，例如：

```
sum = i1 + i2;
```

● 控制结构，例如：

```
if(x > y)  z = x;
```

函数定义区可以定义若干函数，每个函数由两部分组成，函数头和函数体。函数头即函数的第一行，包括函数类型、函数名、函数形式参数名、参数类型。

函数体与主函数相同，也包括本函数中用到的局部变量定义部分和执行部分。

注意

书写程序要养成良好的习惯。

（1）每个程序前面都要有注释，包括程序名、程序编写者、最后修改时间、简要描述该程序的功能。每个语句后可以根据情况决定是否加上注释。

（2）在 main 前加一行空行，如果 main 函数中没有 return 语句，要加 void。

（3）一般在二元运算符左右各加一个空格，更清晰。

（4）写长程序时，函数体应采用缩进格式书写，这种写法能够突出程序的功能结构，使程序便于阅读。

2. 基本数据类型

程序操作的对象是数据，数据及有关的概念和知识都是最基本的，应该深入理解和掌握。

1）数据

数据是程序处理的对象，它被划分为不同的类型。如 int（整型）、float（单精度浮点型）、double（双精度浮点型）、char（字符型）等。某种数据类型数据的取值范围是确定的，能对其进行哪些运算也是确定的。例如，对 int 类型的数据可以进行+（加）、–（减）、*（乘）、/（除）、%（求余）等运算。因此，在学习一种数据类型时，要把这种数据类型的数据书写规则和可以进行的运算联系起来学习。

C / C++语言除提供以上几种基本数据类型外，还允许用户定义其他数据类型，如枚举型、构造类型（数组类型、结构体类型、共用体类型）、指针类型和空类型。

2）常量和变量

数据有常量和变量之分，常量和变量是程序中使用的最基本的数据对象。

常量是程序执行过程中不变的量。有些类型的常量从字面形式就可以判别出来，如 123 为整型常量，0.65 为实型常量，'A' 为字符型常量，"CHINA"为字符串常量。

要仔细区分字符型常量和字符串常量。字符型常量是用单引号括起来的一个字符，实际存储的是机器字符集中字符的 ASCII 码值。字符串常量是由一对双引号括起来的字符序列，

是包含转义字符 '\0' 的一个字符数组。

C 语言规定也可以用标识符代表常量，称为符号常量。例如：

```
#define PI 3.14159    /*定义符号常量 PI 的值为 3.14159*/
```

变量是存放数据的存储单元，在程序执行过程中变量的值是可以改变的。使用变量定义，可以设置程序中使用的存储单元，每个存储单元有一个名字和存放数据值的类型。程序中用到的变量都要先定义后使用。变量定义的同时还可以进行初始化。C 语言中变量初始化的方式为：

```
int a = 5;
```

C++中还允许如下的初始化方法：

```
int a(5);
```

3. 运算符和表达式

C 语言提供了许多运算符，学习时，主要应该掌握这些运算符的运算规则、优先级和结合性。本章主要介绍了进行简单程序设计用到的算术运算符和赋值运算符。

算术运算符的运算规则与代数中的运算规则一致，优先级是先乘除、后加减，结合性为从左到右。

“=”就是赋值运算符，它的作用是将一个数据赋值给一个变量。如“a=5”的作用是执行一次赋值操作（或称赋值运算）。把常量 5 赋给变量 a。也可以将一个表达式的值赋给一个变量，如 x = 3 + 5 * 6，这属于简单的赋值运算。在赋值符“=”之前加上其他算术运算符，可以构成复合的运算符。例如，i + = 2 等价于 i = i + 2。赋值运算符按照自右向左的结合顺序。

在程序中，凡是表示一个值或要计算一个值的地方就要用到表达式，用运算符把运算量连接起来形成一个有意义的式子称为表达式。表达式中可以使用小括号，因此用括号括起来的表达式是有层次的，在求表达式的值时，要从内层往外层运算，即先计算内层括号，后计算外层括号，同一层的表达式，从左到右，按照优先级的高低依次计算，如果运算符两侧优先级相同，则按规定的结合方向进行。

4. 数据的输入 / 输出

输入和输出是以计算机为主体而言的。将数据从键盘或磁盘文件读到内存中叫输入，将内存中的数据输出到屏幕或磁盘文件叫输出。C / C++语言中数据的输入 / 输出可以使用标准输入 / 输出函数 scanf 和 printf。头文件 stdio.h 中声明了这两个函数，要使用输入 / 输出函数必须在程序开头把 stdio.h 头文件包含进来。例如：

```
#include<stdio.h>
```

注意

当输入一串数据时，用什么符号分隔每个数据呢？

（1）使用隐含的分隔符，即格式符之间不加任何分隔符。如：

```
scanf("%d%d%d",&a, &b, &c);
```

则输入数据时用空格或回车分隔数据。

（2）使用显示分隔符。在格式说明符中还可以有其他字符，在输入字符时应输入与这些字符相同的字符。如：

```
scanf("%d, %d, %d",&a, &b, &c);
```

则输入数据时，数据之间应该加上逗号。

（3）对于 float 和 double 类型数据的输出都使用%f 格式符，但输入时，前者使用%f，后者使用%lf。

2.2 补充阅读内容

1．标识符的命名规则

C / C++中标识符由字母、数字和下划线组成，且第一个字符不能是数字。

标识符通常具有描述性的名称，如看到 numberofStudent 就知道它代表学生人数，而像 xyz 就不是一个好名字，看不出它可能代表什么。定义标识符时，要做到见名知意。命名变量的方式决定了程序书写的风格。

C / C++中标识符区分大小写字母，比较流行的标识符命名规则有骆驼表示法和匈牙利表示法。

1）骆驼表示法

如果一个标识符有两个单词，一般以小写字母开头，以后的词则以大写字母开头写成，如 searchKey。

自定义类型名（如结构体名、类名）则以大写字母开头，一些函数名也可以用大写字母开头。

2）匈牙利表示法

该法在每个变量前加上表示类型的字符，如 iSearchKey 表示整型变量，ipSearchKey 表示整型指针变量，这种方法已经流行于现代软件环境中，如 Windows 中的类库和函数库。

2．数据的存储和舍入误差

1）整型数据在内存上的存放

整型数据在内存上以补码的形式表示。正数的补码与原码形式相同，负数的补码的求解方法是，将该数绝对值的二进制形式，按位取反再加 1。例如，求–10 的补码（见图 2-1），–10 在内存上存储的值为

11111111111111111111111111110110

在 32 位中，最高一位为符号位，该位为 0，表示正数，该位为 1，表示负数。

2）实型数据在内存上的存放

实型数据可以有小数和指数两种表示方法，但在内存上都是按照指数形式存储的。系统把一个实型数据分成小数部分和指数部分并分别存放。小数部分采用规范化的指数形式。例如，实数 3.14159 在内存中的存放形式如图 2-2 所示。

3）实型数据的舍入误差

实型数据的存储是有误差的，它只能提供一定的有效数字位数，有效位以外的数字将被