第 1 章　程序设计基础知识

1.1　计算思维与自动化计算

1.1.1　科学思维及其分类

思维是社会人所特有的反映形式，它的产生和发展都同社会实践和语言紧密地联系在一起。思维是人所特有的认识能力，是人的意识掌握客观事物的高级形式。

思维由生命进化而产生，生命在生存过程中进化出意识、思维。思维是人类高级的心理活动形式。思维由思维原料、思维主体和思维工具组成：客观世界提供思维的原料，人脑是思维的主体，认识的反应形式则是思维的工具，这三者结合才能产生思维活动。思维在社会实践的基础上，对感性材料进行分析和综合，通过概念、判断、推理的形式，形成合乎逻辑的理论体系，反映客观事物的本质属性和运动规律。思维过程是一个从具体到抽象，再从抽象到具体的过程，其目的是在思维中再现客观事物的本质，达到对客观事物的具体认识。思维规律由外部世界的规律决定，是外部世界规律在人的思维过程中的反映。

思维有多种分类方法。按照思维的凭借物和解决问题的方式不同，可以把思维分为直观动作思维、具体形象思维和抽象逻辑思维。按照解决问题时的思维方向不同，可以把思维分为收敛思维与发散思维、横向思维与纵向思维等。按照思维的形成和应用领域，可以把思维分为科学思维和日常思维。

科学思维通常是指形成并运用于科学认识活动、对感性认识材料进行加工处理的方式与途径的理论体系；是真理在被认识的统一过程中，对各种科学的思维方法的有机整合，是人类实践活动的产物。科学思维比日常思维更具严谨性和科学性。

在科学认识活动中，科学思维有 3 个基本原则：在逻辑上要求严密的逻辑性，达到归纳和演绎的统一；在方法上要求辩证地分析和综合两种思维方法；在体系上，实现逻辑与历史的一致，达到理论与实践的具体的历史的统一。

从人类认识世界、改造世界的思维方式出发，科学思维又可分为理论思维、实验思维和计算思维三种。

（1）理论思维，又称逻辑思维，是指通过抽象建立描述事务本质的感念，再应用科学的方法探寻概念之间联系的一种思维方法。科学思维既包括以归纳推理为主要内容的归纳逻辑，也包括以演绎推理为主要内容的演绎逻辑，它是一个从具体到抽象，再从抽象到具体的过程，其目的是在思维中再现客观事物的本质，达到对客观事物的具体认识。例如，在数学中，定义是理论思维的灵魂，公理化方法则是理论思维方法。

（2）实验思维，又称实证思维，是通过观察与实验获取规律的一种思维方法，以观察和归纳自然规律为特征。与理论思维不同，实验思维往往需要借助特定的设备获取数据来进行分析。例如，物理、化学、生物等学科中常常用到实验思维的方法。

（3）计算思维，又称构造思维，是从具体的算法设计规范入手，通过算法过程的构造与实施

解决问题的一种方法。计算思维是生活在信息时代的人们应具备的一种基础思维方式，它以设计和构造为特征，以计算机学科为代表，是思维过程的计算模拟方法论。计算机不仅为不同专业领域提供了解决专业问题的有效方法和手段，而且提供了一种独特的处理问题的思维方式。

1.1.2 计算思维的概念与特征

2006 年 3 月，美国卡内基·梅隆大学的周以真教授，在美国计算机权威期刊 Communications of the ACM 上给出了计算思维的定义：计算思维是运用计算机科学的基础概念进行问题求解、系统设计及人类行为理解等涵盖计算机科学之广度的一系列思维活动。

为了让人们更易于理解，周教授进一步将计算思维定义为：通过约简、嵌入、转化和仿真等方法，把一个看起来困难的问题重新阐释成一个我们知道问题怎样解决的方法；是一种递归思维，是一种并行处理，是一种把代码译成数据又能把数据译成代码，是一种多维分析推广的类型检查方法；是一种采用抽象和分解来控制庞杂的任务或进行巨大复杂系统设计的方法，是基于关注分离的方法（SoC 方法）；是一种选择合适的方式去陈述一个问题，或对一个问题的相关方面建模使其易于处理的思维方法；是按照预防、保护及通过冗余、容错、纠错的方式，并从最坏情况进行系统恢复的一种思维方法；是利用启发式推理寻求解答，即在不确定情况下的规划、学习和调度的思维方法；是利用海量数据来加快计算，在时间和空间之间、在处理能力和存储容量之间进行折中的思维方法。

计算思维吸取解决问题所采用的数学思维方法，结合现实世界中复杂系统设计的工程思维方法，是涉及复杂性、智能、心理、人类行为的理解等的一般科学思维方法。计算思维建立在计算过程中个人的能力和限制之上，计算方法和模型使得我们可以去处理一些原本无法由个人独立解决的问题。

计算思维具有以下特性。

（1）概念化，不是程序化。计算机科学不是计算机编程。像计算机科学家那样去思维意味着远远不止能为计算机编程，还要求能够在抽象的多个层次上思维。

（2）基础性，不是机械的技能。基础的技能是每个人为了在现代社会中发挥职能所必须掌握的。生搬硬套机械的技能意味着机械地重复。具有讽刺意味的是，只有当计算机科学解决了人工智能的宏伟挑战——使计算机像人类一样思考之后，思维才会变成机械的生搬硬套。

（3）计算思维是人的思维，不是计算机的思维。计算思维是人类求解问题的一条途径，但绝非试图使人类像计算机那样来思考。计算机枯燥且沉闷，人类聪颖且富有想象力。配置了计算设备，我们就能用自己的智慧去解决那些计算时代之前不敢尝试的问题，就能建造那些其功能仅仅受制于我们想象力的系统。

（4）数学和工程思维的互补与融合。计算机科学在本质上源自数学思维，因为像所有的科学一样，它的形式化解析基础建立于数学之上。计算机科学又从本质上源自工程思维，因为我们建造的是能够与实际世界互动的系统。基本计算设备的限制迫使计算机科学家必须计算性地思考，不能只是数学性地思考。构建虚拟世界的自由使我们能够超越物理世界去建造各种系统。

（5）计算思维是思想，不是人造品。不只是我们生产的软硬件人造品将以物理形式到处呈现，并时时刻刻触及我们的生活，更重要的是还将有我们用以接近和求解问题、管理日常生活、与他人交流和互动的计算性的概念。

（6）计算思维面向所有人和所有地方。计算思维日渐渗透到其他学科中，并渗透到我们的

生活中。例如，在统计学中，采用机器学习理论中的统计学习方法，用于统计各类问题的规模。在生物学中，利用计算机科学在海量序列数据中搜索寻找模式规律的本领，如用合适的数据结构与算法，描述蛋白质的结构等。还有，计算博弈理论正改变着经济学家的思考方式，纳米计算改变着化学家的思考方式，量子计算改变着物理学家的思考方式。

1.1.3 计算思维的举例——自动化计算

学习程序设计是理解计算机工作特点的最好途径，程序设计课程的内容最能够体现语言级的问题求解方法，是计算思维能力培养的重要内容。以下举例可以体现采用计算思维解决特定问题的方式。

1．求最大值问题

首先看求 4 个整数中的最大值。假设这 4 个整数是 *a*、*b*、*c*、*d*，采用直观思维的方法：

（1）先求 *a*、*b* 的最大值，假设用 x 表示；

（2）再求 *c*、*d* 的最大值，假设用 *y* 表示；

（3）比较 *x*、*y* 的大小，大的一个即为所求最大值。

当数据量比较小时，这种方法还可以便捷地解决问题。但若数据量比较大，如求 10 个数、100 个数、100000 个数的最大值，这样的方法显然就不适用了。

仍然求 *a*、*b*、*c*、*d* 中的最大值，假设有一个变量 MAX，用来存放最大值。用以下思路进行求解。

（1）令 MAX 的值为 *a* 的值。

（2）比较 *b* 和 MAX 的大小，若 *b* >MAX，令 MAX 的值为 *b* 的值。

（3）比较 *c* 和 MAX 的大小，若 *c* >MAX，令 MAX 的值为 *c* 的值。

（4）比较 *d* 和 MAX 的大小，若 *d* >MAX，令 MAX 的值为 *d* 的值。

这样，MAX 中即可获得 4 个数中的最大值。

这种方法下，当数据更多时，求最大值的方法是不变的，只是多了若干次比较。

另外，若采用第一种方法，求 *n* 个数中的最大值需要另外 *n*-1 个变量来表示中间数据。而第二种方法，无论数据量多大，在比较过程中，只需用到一个变量 MAX。显然第二种方法更好，更适用一般情况。

2．排序问题

排序（Sorting）是计算机程序设计中经常使用的一种重要操作，其功能是将一个数据元素集合或序列重新排列成一个按数据元素某个数据项值进行有序排列的序列。在日常生活中，排序的例子很常见，如电话号码簿、词典、仓库清单等，都被整理得井井有条，整理的过程就是排序。

排序的方法很多，下面介绍一种思路比较简单的排序方法。

简单选择排序，其过程为：第一趟，从 *n* 个记录中找出关键码最小的记录与第一个记录交换；第二趟，从第二个记录开始的 *n*-1 个记录中再选出关键码最小的记录与第二个记录交换；以此类推，第 *i* 趟，则从第 *i* 个记录开始的 *n*-*i*+1 个记录中选出关键码最小的记录与第 *i* 个记录交换，直到整个序列按关键码有序排列。

例如，有以下序列：25　36　30　36　10　56　12，请按简单选择排序方法进行排序。排序过程如图 1.1 所示。

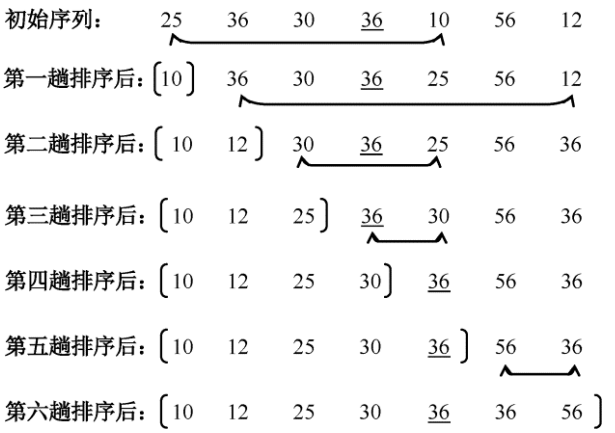


图 1.1　简单选择排序示例

1.2　计算机内的数据表示

随着计算机应用领域的日益广泛，计算机除了用于进行复杂的科学计算，还能进行文字的处理、图像的识别、声音的加工等。数字、汉字、图像和声音的表象千差万别，但对于计算机而言，它们都被称为数据或信息。在计算机科学中，数据是指所有能被计算机识别、存储和处理的符号的总称。

1.2.1　数制及其转换

1．进位计数制

进位计数制是常用的计数方法。进位计数制是采用有限个数码来表示数据，数据中各个数字所处的位置决定它的权值，每个数字所表示的数值就等于该数字本身乘以它的位置所代表的权值。

各数位只允许选用有限个数码，每一数位所能表示的最大值等于可选用的最大数码乘以其权值，超过这个值就要向高位进位。允许选用的数码的个数就是计数制的基数。

一般，基数为 r 的 r 进制数：

$$(N)_r=K_nK_{n-1}\cdots K_1K_0 \quad K_{-1}K_{-2}\cdots K_{-m}$$

的数值可表示为：

$$(N)_r=K_n\times r^n+K_{n-1}\times r^{n-1}+\cdots+K_1\times r^1+K_0\times r^0+K_{-1}\times r^{-1}+K_{-2}\times r^{-2}+\cdots+K_{-\mathrm{m}}\times r^{-\mathrm{m}}$$

其中，数位 K_i（$-m\leq i\leq n$）的数值可选择 r 个数码中的某一个，其权值是 r^i。

1）十进制计数法

日常生活中用的最多的是十进制计数法。例如，十进制数 $(1286.75)_{10}=1\times10^3+2\times10^2+8\times10^1+6\times10^0+7\times10^{-1}+5\times10^{-2}$。其各位的权值分别为 10^3、10^2、10^1、10^0、10^{-1}、10^{-2}。

十进制计数法“逢十进一”，即基数为 10；十进制的数码为 0、1、2、3、4、5、6、7、8、9。

例如，个位数的权值是 $1(10^0)$，那么个位数所能表示的最大数值为 $9\times10^0=9$，超过 9 就要向十位进位；十位数的权值是 $10(10^1)$，那么十位数所能表示的最大数值为 $9\times10^1=90$，超过 90 就要向百位进位。

2）二进制计数法

在计算机内部，数据是用二进制计数法表示的。计算机内用电子器件的两种不同状态来表

示数字信息，如用高电平来表示 1，用低电平表示 0，因此对于计算机的物理实现而言，用二进制表示数据更方便。例如，二进制数$(1001101.101)_2=1\times2^6+0\times2^5+0\times2^4+1\times2^3+1\times2^2+0\times2^1+1\times2^0+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}$。

显然，其各位的权值分别为 2^6、2^5、2^4、2^3、2^2、2^1、2^0、2^{-1}、2^{-2}、2^{-3}。

二进制计数法“逢二进一”，即基数为 2；二进制的数码为 0、1。也就是说，二进制数的每一位或者为 0，或者为 1；超过 1 就要向高位进位。

3）八进制计数法与十六进制计数法

采用二进制计数法表示数据的一个缺点是所需位数很多，容易出错，因此为了便于阅读和书写，常采用八进制数与十六进制数来代替二进制数。

例如，八进制数

$$(621)_8=6\times8^2+2\times8^1+6\times8^0$$

其各位的权值分别为 8^2、8^1、8^0。

八进制计数法“逢八进一”，即基数为 8，八进制计数法的数码为 0、1、2、3、4、5、6、7。

这 8 个数码相当于十进制数的 0～7，接下来下一位数字就向高位进位，即$(10)_8$，相当于十进制数 8，那么$(11)_8$相当于十进制数 9。

又如，十六进制数

$$(8A1F)_{16}=8\times16^3+10\times16^2+1\times16^1+15\times16^0$$

其各位的权值分别为 16^3、16^2、16^1、16^0。

十六进制计数法“逢十六进一”，即基数为 16。十六进制计数法的数码为 0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F。

十六进制中有 16 个数码，但从 10 开始到 15 就没有阿拉伯数字可用了，因此规定用字母 A(a)、B(b)、C(c)、D(d)、E(e)、F(f)来表示 10、11、12、13、14、15。

2．数制转换

1）任意进制转换为十进制

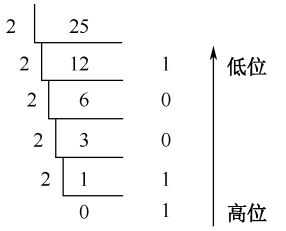
由 r 制数转换为十进制数可按照如下公式进行多项式展开求和即可。

$$K_nK_{n-1}\cdots K_1K_0 \quad K_{-1}K_{-2}\cdots K_{-m}=$$
$$K_n\times r^n+K_{n-1}\times r^{n-1}+\cdots+K_1\times r^1+K_0\times r^0+K_{-1}\times \mathrm{r}^{-1}+K_{-2}\times \mathrm{r}^{-2}+\cdots+K_{-\mathrm{m}}\times r^{-\mathrm{m}}$$

2）十进制转换为任意进制

可以采用除基取余法将十进制整数转换为 r 进制整数：将十进制整数除以 r，得到商和余数，余数对应为 r 进制数低位的值；继续让商再除以 r，得到商和余数，重复此操作，直至商为 0，如此得到的一系列余数就是所求的 r 进制数的各位数字，先得到的是低位，后得到的是高位。

例如，将$(25)_{10}$转换为二进制整数。



因此，$(25)_{10}=(11001)_2$。

可采用乘基取整法将十进制小数转换为 r 进制小数：将十进制小数乘以 r，去掉乘积的整数部分，再将余下的纯小数乘以 r，重复此操作，直至乘积等于 0 或达到所需的精度为止，如此得到的一系列整数就是 r 进制小数的各位数字，先得到的是高位，后得到的是低位。

例如，将$(0.625)_{10}$转换为二进制小数。

0.625×2=1.25　　1　高位

0.25×2=0.5　　0

0.5×2=1　　1　低位

因此，$(0.625)_{10}=(0.101)_2$。

由于整数和小数的转换方法截然不同，将十进制数转换为 r 进制数时，整数部分和小数部分要分开来进行转换。

r 进制小数能精确地转换为十进制小数，但十进制小数往往不能精确地转换为 r 进制小数。例如，$(0.1)_{10}=(0.0001100110011)_2$。

3）二进制与八进制、十六进制之间的转换

对于一个二进制数，只要依次（整数部分由低位到高位，小数部分由高位到低位）将其每 3 位或 4 位分成一组，就可以直接转换为八进制数或十六进制数。

例如，二进制数$(1100101111010011.01101)_2$

若按 3 位一组划分，就可以直接写出其对应的八进制数：

1　100　101　111　010　011 .　011　010

1　4　5　7　2　3 .　3　2

即$(1100101111010011.01101)_2=(145723.32)_8$。

若按 4 位一组划分，也可以直接写出其对应的十六进制数：

1100　1011　1101　0011 .　0110　1000

C　B　D　3 .　6　8

即$(1100101111010011.01101)_2=(CBD3.68)_{16}$。

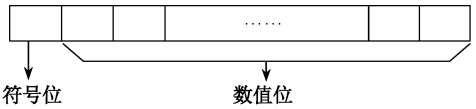
反之，将八进制数中的每一位用 3 位二进制数表示，将十六进制数中的每一位用 4 位二进制数表示，也能转换为对应的二进制数。

1.2.2　原码、反码及补码

计算机中，数据的最小单位是“位”，一般将 8 位二进制位定义为 1 字节，计算机中的存储量就是以字节来计算的。

在计算机内，不仅数值是用二进制数表示的，符号也是用二进制数表示的。一般规定：用 0 表示“＋”，用 1 表示“-”；符号位放在数值位之前。一个数连同其符号在机器中的二进制数表示形式称为机器数，它所代表的数值称为机器数的真值。

机器数的一般格式如下。



为简便起见，下面讨论中用 1 字节来表示带符号的数据。

在计算机中，带符号的数的表示方法有 3 种：原码、反码和补码。

1．原码

原码表示法是符号位用 0 表示正数，用 1 表示负数，数值位表示数值本身。例如，$[+27]_{原}=$

0 0 0 1 1 0 1 1，$[-27]_{原}$= 1 0 0 1 1 0 1 1。

在原码表示法中，0 的表示方法不唯一，即$[+0]_{原}$= 0 0 0 0 0 0 0 0，$[-0]_{原}$= 1 0 0 0 0 0 0 0。

2．反码

反码表示法中正数与负数的表示方法不同，正数的反码与原码同形。例如：

$[+27]_{反}$= 0 0 0 1 1 0 1 1。

负数的反码为：符号位仍为 1，数值位是对原码取反。例如：

$[-27]_{反}$= 1 1 1 0 0 1 0 0。

在反码表示法中，0 的表示也不唯一，即$[+0]_{反}$= 0 0 0 0 0 0 0 0，$[-0]_{反}$= 1 1 1 1 1 1 1 1。

3．补码

用原码表示数据，简单明了，但用原码进行加减运算却很不方便。例如，当两个数的符号相同时，可以数值位相加，结果符号不变。但当两个数的符号不同时，加法运算实际上要转换为减法进行；为了进行减法，应先判断两数的绝对值，让绝对值大的数减去绝对值小的数，运算结果的符号与绝对值大的数的符号相同。计算机实现上述过程是相当烦琐的。

因此，在计算机中采用补码表示法来表示数据。采用补码表示数据，能够简化设计与运算，可以将减法运算转化为加法运算。

下面以时钟为例说明补码的原理，假设当前时刻是 3 点，若问两小时前是几点，那么可以将时针向前拨两小时，即 3－2＝1，得到 1 点；还可以将时针向后拨 10 小时，即 3＋10＝13＝12＋1，从时钟上看，还是 1 点。从而，3 减去 1，与 3 加上 10 能够达到同样的效果。这是因为时钟的刻度是以 12 为周期的，超过 12 就再从头开始计数，则称 2 和 10 是互补的，即减去 2 就相当于加上 10。这就是减法变加法的原理。

在计算机中，以定长的存储单元来表示数据，因此所能表示数据的范围是有限的，超过它的表示范围后，高位就会溢出。

正数的补码与原码、反码同形。例如：

$[+18]_{原}$＝$[+18]_{反}$＝$[+18]_{补}$＝ 0 0 0 1 0 0 1 0。

负数的补码：符号位为 1，数值位等于原码的数值位取反，再加 1，或者说$[x]_{补}$＝$[x]_{反}$＋1。

例如，

$[-18]_{原}$＝ 1 0 0 1 0 0 1 0，$[-18]_{反}$＝ 1 1 1 0 1 1 0 1，$[-18]_{补}$＝ 1 1 1 0 1 1 1 0。

采用补码表示数据，使得正、负的关系转换成了一种纯数值的关系。补码形式的数据进行运算时，符号位和数值位一样，均参与运算，不必考虑它特别的含义（正、负）。

例如，计算－36＋58 的值。

$[-36]_{补}$＝ 1 1 0 1 1 1 0 0，$[+58]_{补}$＝ 0 0 1 1 1 0 1 0。

$$
\begin{array}{rl}
 & 1\ 1\ 0\ 1\ 1\ 1\ 0\ 0 \\
+ & 0\ 0\ 1\ 1\ 1\ 0\ 1\ 0 \\
\hline
\text{溢出} \longleftarrow & \boxed{1}\ \ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 0
\end{array}
$$

显然，$(1\ 1\ 0\ 1\ 1\ 1\ 0\ 0)_2$＋ $(0\ 0\ 1\ 1\ 1\ 0\ 1\ 0)_2$＝ $(0\ 0\ 0\ 1\ 0\ 1\ 1\ 0)_2$ ＝$(22)_{10}$，从而验证了－36＋58＝22。

可见，采用补码表示的数据进行运算的规则是很简单的。而且，在补码表示法中，0 的表示是唯一的，若用 1 字节存放数据，则规定

$[0]_{补}$＝0 0 0 0 0 0 0 0，$[-128]_{补}$＝1 0 0 0 0 0 0 0。

1.2.3 定点数及浮点数

一个数据可能含有整数部分，又含有小数部分，这就涉及小数点的表示问题。对小数点的表示方法有两种：定点数和浮点数。

1．定点数

若对于所有的数据都限定小数点的位置固定不变，就是定点表示法。

1）定点小数

约定小数点的位置在符号位之后，数值部分之前，则定点小数的表示形式如下。

计算机中并不存放小数点，只是隐含约定小数的位置。

假设用 1 字节存放数据，

则符号位占 1 位，数值位占 7 位，那么，

0 1 0 0 0 1 0 1 表示二进制小数 0 . 1 0 0 0 1 0 1，1 0 1 1 0 0 0 1 表示二进制小数-0 . 0 1 1 0 0 0 1。

若计算机的字长为 n，其中一位是符号位，n-1 位是数值位，那么，当数值部分的最后一位 x_{n-1} 是 1，其余位是 0 时，数 x 的绝对值最小为 $|x|_{\min}=2^{-(n-1)}$；当数值部分全是 1 时，数 x 的绝对值最大为 $|x|_{\max}=1-2^{-(n-1)}$。

因此，计算机所能表示的定点小数的绝对值范围是 $2^{-(n-1)}$ ～ $1-2^{-(n-1)}$。

2）定点整数

约定小数点的位置在数值部分之后，即数据是纯整数。其表示形式如下：

字长为 n 的定点整数，考虑其绝对值，最小的数的数值部分全为 0，最大的数的数值部分全为 1，因此其所能表示的数据的绝对值范围是 $0\sim2^{n-1}-1$。

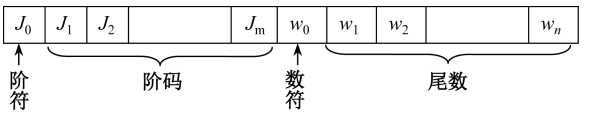
在实际处理数据时，一个二进制数 N 可能既有整数部分又有小数部分，那么就需要将其乘上一个倍数以转换成规定的定点整数或定点小数的形式：

$$N=X\times2^{i}$$

其中，X 为规定的定点整数或定点小数，i 的值一旦选取就固定不变，这是定点数的特点。i 的取值很重要：对于定点小数，i 若选取大了，数据可能溢出；i 若选取小了，可能会有损数据的精度。

2．浮点数

为了协调数据的表示范围与精度的要求，提出数据的浮点表示方法，即小数点的位置不再固定不变，而是根据需要浮动。浮点数的格式如下：



浮点数的表示方法类似于数学中的科学计数法，它的真值为

$$\pm\text{尾数}\times 2^{\pm\text{阶码}}$$

其中，尾数的符号由数符表示，阶码的符号由阶符表示，1 表示正，0 表示负。浮点数的符号由尾数的符号决定，阶码的符号只决定小数点的位置。

浮点数的范围主要由阶码决定，有效数字的精度主要由尾数决定。为了充分利用有效数字，将尾数规格化，即限定尾数 W 的绝对值在一定的范围内，一般规定：

$$\frac{1}{2} \leq |W| \leq 1$$

也就是说，尾数部分的最高位是 1。

例如，有一采用浮点表示法的计算机，尾数有 7 位，阶码有 3 位，数符和阶符各一位。

J_0	J_1	J_2	J_3	w_0	w_1	w_2	w_3	w_4	w_5	w_6	w_7

那么，它能够表示的数据 X 的绝对值的范围如下。

（1）最小值的尾数最小，阶码最大，阶符为负，即

1	1	1	1	0	1	0	0	0	0	0	0

此时，尾数为 2^{-1}，阶码为 $2^3-1=7$，阶符为负，最小值为 $2^{-1}\times 2^{-7}$。

（2）最大值的尾数最大，阶码最大，阶符为正，即

0	1	1	1	0	1	1	1	1	1	1	1

此时，尾数为 $1-2^{-7}$，阶码为 $2^3-1=7$，阶符为正，最大值为$(1-2^{-7})\times 2^7$

因此，$2^{-1}\times 2^{-7} \leq |X| \leq (1-2^{-7})\times 2^7$

那么，尾数有 n 位，阶码有 m 位的浮点数的绝对值的表示范围为$2^{-1}\times 2^{-2^{m-1}} \sim (1-2^{-n})\times 2^{2^{m-1}}$。

1.3 程序与算法

1.3.1 程序及算法的概念

现实生活中，做任何事情都需要经过一定的步骤才能完成。例如，乐队演奏乐曲、教师按教学计划授课、工程师设计施工方案等都必须按照一定的步骤进行。

为解决一个问题而采取的方法和步骤，称为算法。算法就是被精确定义的一组规则，规定先做什么，再做什么，以及判断某种情况下做哪种操作；或者说算法是步进式的完成所需任务的过程。

为了让计算机来解决问题，除了需要明确解决问题的算法，还必须用特定的计算机指令来控制计算机按照我们的意图有步骤地工作，最终完成特定的任务。

计算机程序是指为让计算机完成特定的任务而设计的指令序列。程序设计是用来沟通算法与计算机的桥梁。程序是编程者写的、计算机能够理解并执行的一些命令的集合，是解决问题的具体算法在计算机中的实现。

1.3.2 算法的特点及评价标准

算法反映解决问题的步骤，不同的问题需要用不同的算法来解决，同一问题也可能有不同的解决方法，但是一个算法必须具有以下特性。

1）有穷性

一个算法必须总是在执行有限个操作步骤和可以接受的时间内完成其执行过程。也就是说，对于一个算法，要求其在时间和空间上均是有穷的。例如，一个采集气象数据并加以计算进行天气预报的应用程序，如果不能及时得到结果，起不到天气预报的作用，显然就超出了可以接受的时间。

2）确定性

算法中的每一步都必须有明确的含义，不允许存在二义性。例如，“ 将成绩优秀的同学名单打印输出 ”，在这一描述中“ 成绩优秀 ”很不明确，是每门功课均为 95 分以上？还是指总成绩在多少分以上？

3）有效性

算法中描述的每一步操作都应该能有效地执行，并最终得到确定的结果。例如，当 Y=0 时，X/Y 是不能有效执行的。

4）输入

一个算法应该有零个或多个输入数据。例如，计算 1～10 的累计和的算法，无须输入数据，而对 10 个数据进行排序的算法，却需要从键盘上输入这 10 个数据。

5）输出

一个算法应该有 1 个或多个输出数据。执行算法的目的是求解，而“ 解 ”就是输出，因此没有输出的算法是毫无意义的。

设计一个好的算法对于正确、高效地解决问题有着重要的意义。一个好的算法应达到以下目标。

（1）正确性：算法应该能够满足问题的要求。

（2）可读性：算法不仅仅是让计算机来执行的，更重要的是让人来阅读，可读性好的算法有助于调试程序、发现和修改错误，使得日后对软件功能的扩展维护易于实现。

（3）健壮性：指算法能够对非法输入做出合理的处理，而不是产生莫名其妙的结果。

（4）高效率与低存储空间需求：指解决特定问题的算法的执行时间应尽量短，算法执行过程中需要的存储空间应尽量小。

1.3.3 算法的表示

算法的表示方法很多，常见的有自然语言、传统流程图、N-S 结构图、伪码等。

1．用自然语言表示

自然语言就是人们日常使用的语言，可以是中文、英文等。例如，求 3 个数的最大值的问题，可以描述为先比较前两个数，找到大的那个数，再让其与第 3 个数进行比较，找到二者中大的数即为所求。

显然，用自然语言表示的算法通俗易懂，但文字冗长，表达上不易准确，容易出现歧义。所以，一般不用自然语言描述算法。

2．用传统流程图表示

传统流程图是用规定的一组图形符号、流程线和文字说明来表示各种操作的算法表示方

法。传统流程图常用的符号如表 1.1 所示。

表 1.1　传统流程图常用的符号

符　号	符号名称	含　义
	起止框	表示算法的开始和结束
	输入/输出框	表示输入/输出操作
	处理框	表示对框内的内容进行处理
	判断框	表示对框内的条件进行判断
	流程线	表示流程的方向
	连接点	表示两个具有同一标记的连接点应连接成一个点
	注释框	表示对流程图中某些框的操作进行必要的补充说明

用传统流程图描述求 3 个数中最大值的算法如图 1.2 所示。

用传统流程图表示算法直观形象，算法的逻辑流程一目了然，便于理解。但其占用篇幅较大，画起来比较麻烦，而且又由于允许使用流程线，使用者可以随心所欲，使流程可以任意转移，从而造成阅读和修改上的困难。

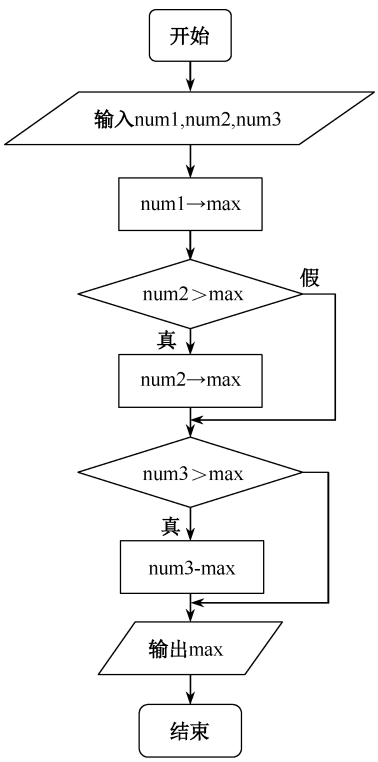


图 1.2　求最大值的流程图

3．用 N-S 结构图表示

针对传统流程图存在的问题，美国学者 I.Nassi 和 B.Shneiderman 于 1973 年提出一种新的

结构化流程图形式，简称为 N-S 结构图。Chapin 在 1974 年对其进行了进一步的扩展，因此，N-S 结构图又称为 Chapin 图或盒状图。

N-S 结构图的目的是开发一种不破坏结构化基本构成元素的过程设计表示。其主要特点是完全取消了流程线，不允许有随意的控制流，全部算法写在一个矩形框内，该矩形框以 3 种基本结构（顺序、选择、循环）描述符号为基础复合而成。

N-S 结构图表示的 3 种基本结构如下。

1）顺序结构

顺序结构是最简单的基本结构。在顺序结构中，要求顺序地执行且必须执行由先后顺序排列的每个最基本的处理单位。如图 1.3（a）所示为用传统流程图表示的顺序结构，图 1.3（b）所示为用 N-S 结构图表示的顺序结构，先执行语句 A，然后再顺序执行语句 B。

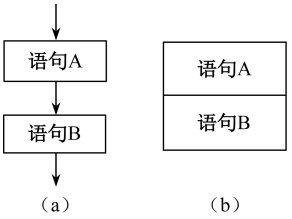


图 1.3　顺序结构

2）选择结构

在分支结构中，要根据逻辑条件的成立与否，分别选择执行不同的处理。如图 1.4 所示为当逻辑条件成立时，执行语句 A，否则执行语句 B。

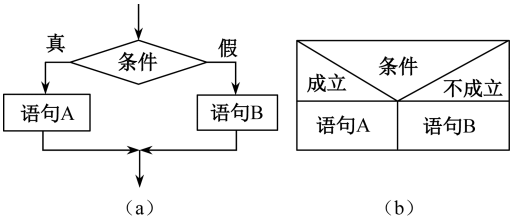


图 1.4　分支结构

3）循环结构

循环结构一般分为当型循环结构和直到型循环结构。

（1）当型循环结构。在当型循环结构中，当逻辑条件成立时，就反复执行语句 A（称为循环体），直到逻辑条件不成立时结束，如图 1.5 所示。

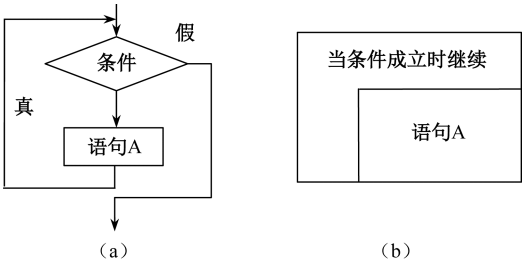


图 1.5　当型循环结构

（2）直到型循环结构。在直到型循环结构中，反复执行语句 A，直到逻辑条件成立时结束（即逻辑条件不成立时继续执行），如图 1.6 所示。

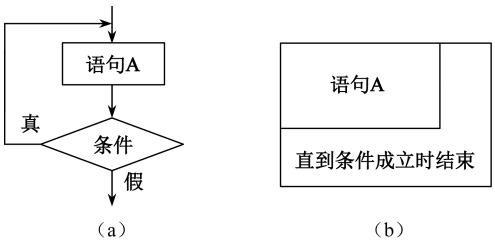


图 1.6　直到型循环结构

例如，用 N-S 结构图描述求 3 个数中的最大值的算法如图 1.7 所示。

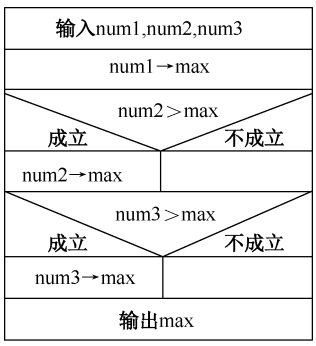


图 1.7　求最大值的 N-S 图

4．用伪码表示

伪码是用一种介于自然语言和计算机语言之间的文字和符号来描述算法。

例如，用伪码描述上述算法：

```
input num1,num2,num3
num1→max
if num2 > max then num2→max
if num3 > max then num3→max
print max
```

伪码不能在计算机上实际执行，但是用伪码表示算法既方便又友好，便于向计算机程序过渡。伪码的表现形式灵活自由、格式紧凑，没有严谨的语法格式。

1.4　C 语言简介

1.4.1　程序设计语言

计算机的语言有很多种，由于计算机硬件只能识别二进制代码，因此最早出现的计算机语言是机器语言，用机器语言描述的程序称为目标程序。

机器语言与计算机的硬件密切相关，特定的机器语言只能在特定的某一类计算机上使用；而且，机器语言是二进制代码序列，难于记忆和理解，用机器语言编写程序烦琐且易于出错。目前，除了编写计算机的核心程序，一般不使用机器语言。

汇编语言又称符号语言，它用助记符来表示指令的操作码，用符号来表示地址和变量。汇

编语言与机器语言是一一对应的。汇编语言使得指令好理解、易记忆，相对机器语言，用汇编语言编写程序要更简单，指令格式更清晰。

用汇编语言编写的源程序，必须翻译成目标程序，才能被计算机识别和运行。由源程序翻译成目标程序的过程称为汇编。完成这个翻译功能的软件称为汇编程序。

汇编语言和机器语言一样，依赖于具体的计算机指令系统，是“面向机器的语言”，因此称它们为低级语言。

高级语言的出现使得编写程序变得方便。高级语言更接近于人类的自然语言，易学易用，而且编程者不必了解计算机的硬件及指令系统就可以编写程序。高级语言编写的程序的通用型好、可移植性强，不依赖具体的硬件设备。

采用高级语言编写的源程序，不能直接运行，必须将其翻译成目标代码才能在计算机上运行。翻译过程有两种。

（1）解释方式：边解释边执行的方式，解释一句执行一句。完成解释功能的软件称为解释程序。例如，采用 BASIC 语言编写的程序的执行过程就是解释方式。

（2）编译方式：先将源程序全部翻译成目标程序，再执行目标程序。完成翻译功能的软件称为编译程序。例如，采用 C 语言编写的源程序要通过“C 语言编译程序”将其翻译成目标程序后，才能执行。

1.4.2 C 语言的历史

C 语言的前身是 ALGOL60 语言。ALGOL60 是一种面向问题的高级语言，它的描述算法很方便，但是它距硬件比较远，不适合用来编写系统程序。

1963 年，英国剑桥大学在 ALGOL60 语言的基础上添加了硬件处理的功能，推出 CPL 语言。但 CPL 语言规模比较大，难以实现。

1967 年英国剑桥大学的 Matin Richards 对 CPL 语言做了简化，推出了 BCPL 语言。

1970 年美国贝尔实验室以 BCPL 语言为基础，又做了进一步简化，设计出更简单且更接近硬件的 B 语言，并用 B 语言写第一个 UNIX 操作系统。

但 B 语言过于简单，功能有限。1972～1973 年间，贝尔实验室在 B 语言的基础上设计出了 C 语言。C 语言既保持了 BCPL 和 B 语言的优点（精练、接近硬件），又克服了它们的缺点（过于简单、数据无类型等）。

最初，C 语言被用来编写 UNIX 操作系统，但由于C语言的强大功能和各方面的优点逐渐为人们认识，C 语言开始迅速传播，成为当代最优秀的程序设计语言之一。

多年来，C 语言在各种计算机上的迅速推广，发展为许多 C 语言版本。为了明确定义与机器无关的 C 语言，1983 年，美国国家标准化协会（ANSI）根据 C 语言问世以来的各种版本对 C 语言的发展和扩充，制定了新的标准，称为 ANSI C。1987 年，其又公布了新标准 87 ANSI C。1990 年，国际标准化组织 ISO 接受 87 ANSI C 为 ISO C 的标准。

1.4.3 C 语言的特点

C 语言是一种通用的、结构化的程序设计语言。无论是系统软件、应用软件，还是数据处理、数值计算等都可以很方便地使用 C 语言来开发。C 语言的主要特点如下。

1）简洁、紧凑、灵活

C 语言中的关键字较少，只有 32 个。语言组成精练、简洁，语法限制不太严格，而且使

用方便、灵活。C 语言提供了丰富的数据类型和运算符，数据结构描述能力及表达式能力强，且语法限制不太严格，程序设计自由度大，程序书写形式自由。

2）模块化、结构化

结构化程序设计方法是当前一种通用的程序设计方法。在 C 语言中，函数是组成程序的最小模块，每个函数实现特定的功能。C 语言中具备实现结构化程序设计的 3 种基本结构为顺序结构、分支结构和循环结构。用C语言编写的程序层次清晰，便于按模块组织程序，易于实现程序的结构化。

3）功能强大

C 语言除了能实现一般高级语言的功能外，还能够直接访问硬件的物理地址，进行二进制位操作，能实现汇编语言的大部分功能。C 语言常被称为“ 中级语言 ”，因为它兼有高级语言和低级语言的特点。C 编译程序生成的目标代码效率高，仅比汇编程序生成的目标代码执行效率低 10%～20%。

4）可移植性好

C 语言通过预处理命令等方法，允许程序尽可能地把与计算机硬件有关的部分从程序中分离出来，从而便于在不同的硬件环境和操作系统环境间实现程序的移植。一个 C 语言程序基本不做修改就可以在不同型号的计算机、不同操作系统上执行。

C 语言对简洁性与实用性、可移植性与高效性之间的矛盾处理得较好。C 语言因其限制少、自由度高、源代码紧凑、执行效率高，而得到广泛的应用。但限制少与自由度高、简单易学与功能强大，往往是不能兼得的。C 语言中为了突出某些优点，或者为了更加灵活，而使得 C 语言也存在一些不足，主要有以下几点。

（1）运算符多，难以掌握。C 语言中有 40 多个运算符，运算符中有 15 种优先级，两种结合性，而且有些运算符的含义和优先级与人类的日常习惯不一致，这使得运算和处理方便灵活的同时，又增加了学习和记忆的难度。

（2）C 语言中的类型转换相当灵活，如字符型数据与整型数据之间可以自动转换；数值型数据与逻辑型数据可以通用，使得对数据的处理变得很方便。C 程序中常常为了类型转换上的方便不要求类型检查，这就导致有些情况下即使要求类型一致，但实际类型不一致也不出错，而产生莫名其妙的运算结果。

（3）C 语言中对数组进行处理时，对数组元素的下标不做越界检查，若程序中引用数组元素时越界，容易造成数据混乱，甚至造成更严重的错误。

由于 C 语言中的弱类型转换，数组元素引用没有越界检测等处理方式带来了不安全因素，所以要求编程者在编写程序时要有慎重、严谨的态度，自己仔细检查程序，保证其正确，不可过分依赖 C 语言的编译程序去检查错误。

1.4.4 C 程序的结构

用 C 语言编写的程序，称为C语言源程序（以下简称“ C 程序 ”）。一个C程序由一个或多个函数组成。一个 C 程序可以包含一个或多个源文件。每个源文件可由一个或多个函数组成。例如，一个 C 程序可由 7 个函数组成，这些函数可以分布在 3 个源文件 file1.c、file2.c、file3.c 中。

最简单的 C 程序是只包含一个 main( )函数、由一个源文件构成的程序，举例如下。

【例 1.1】编写一个 C 程序，求一个圆的面积。

程序如下。

```
#include"stdio.h"
#define PI 3.14
main()                                          /*main()函数的首部*/
{                                               /*函数体的开始*/
    float   r, s;                               /*定义两个实型变量*/
    printf("\ninput the radius: ");             /*输出提示输入的信息*/
    scanf("%f",&r);                             /*从键盘输入圆的半径*/
    s=PI*r*r;                                   /*求面积*/
    printf("\nr=%f,s=%f",r,s);                  /*输出圆的半径和面积*/
}                                               /*函数体结束*/
```

说明：

（1）“ #include"stdio.h" ”是编译预处理命令，作用是将标准输入/输出头文件 stdio.h 的内容包含到本文件中来。程序中用到标准输入/输出库函数时，一般需要使用 #include 命令将 stdio.h 包含到源文件中。需要说明的是，C 语言特别规定对 scanf( )和 printf( )这两个函数可以省去对其头文件的包含命令。

（2）“ #define PI 3.14 ”是编译预处理命令，它的作用是定义 PI 表示圆周率 3.14。在程序中出现 PI 的地方都代表 3.14。

（3）main( )为主函数名，这一行称为函数首部。从第 5 行到第 9 行，花括号“ { } ”内的部分称为函数体。函数体内又分为声明部分和执行部分。其中，程序的第 5 行“ float r, s; ”的功能是声明变量，是声明部分；程序的第 6 行至第 9 行是执行部分。

（4）程序的第 7 行的功能是调用系统提供的标准输入/输出库函数 scanf( )读入变量 r 的值；第 9 行的功能是调用系统提供的标准输入/输出库函数 printf( )输出计算结果 s 的值。

【例 1.2】编写程序求任意 3 个整数中的最大值。

程序如下。

```
#include "stdio.h"
main( )                                         /*main( )函数的首部*/
{                                               /*main( )函数体的开始*/
    int num1,num2,num3;                         /*定义3个整型变量*/
    int maximum;                                /*定义变量maximum，用于存放最大值*/
    printf("\nEnter three integers: ");         /*输出提示信息*/
    scanf("%d,%d,%d",&num1,&num2,&num3);        /*读入变量的值*/
    maximum=max(num1,num2,num3);                /*调用自定义函数max，将所求的最大值赋给变量
maximum*/
    printf("\nMaximum is: %d ",maximum);        /*输出maximum的值*/
}
int max(int x,int y,int z)                      /*max( )函数的首部*/
{                                               /*max( )函数体的开始*/
    int m=x;                                    /*默认第一个数最大*/
    if (y>m)   m=y;                             /*如果第二个数更大，则修改m*/
    if (z>m)   m=z;                             /*如果第三个数更大，则修改m*/
    return m;                                   /*将m的值作为函数值返回*/
}                                               /*max( )函数体的结束*/
```

说明：

（1）该程序包含了两个函数：主函数 main( )和函数 max( )。这两个函数的定义是相互独立的。程序执行时 main( )函数调用 max( )函数。

（2）自定义函数 max( )的作用是选择 x、y、z 中的最大值赋值给变量 m，然后通过 return 语句将 m 的值返回给主调函数 main( )。

（3）程序的执行从 main( )函数开始，先输出提示输入的信息，用户输入 3 个整数后，调用

函数 max( ), 这时将实际参数（以下简称“ 实参 ”）num1、num2、num3 的值赋值给形式参数（以下简称“ 形参 ”）x、y、z, 执行 max( )函数后得到一个返回值, 将该值赋值给变量 maximum, 然后输出最大值。

通过以上两个简单的 C 程序，可以归纳出 C 程序的结构具有以下特点。

（1）C 程序是由函数构成的。一个函数由函数首部和函数体两部分组成。

函数的首部是定义一个函数的开始，包括函数的类型、函数名及函数参数表。

函数体是函数功能的实现，包括声明部分和执行部分。声明部分用来声明函数中需要的变量和调用的函数。执行部分用 C 语句构成，用来完成一定的操作任务。

一般函数的格式如下。

```
类型名    函数名（函数参数表）
{   声明部分
    执行部分
}
```

需要注意的是，函数体中也可以没有任何内容，即构成一个空函数。例如：

```
dummy( )
{ }
```

（2）一个源程序不论包含多少个函数、由多少个源文件组成, 都有一个且只能有一个 main( ) 函数（又称主函数）。不论 main( )函数在整个程序中的位置如何，C 程序的执行总是从 main( ) 函数开始，在调用其他函数后，最后回到 main( )函数中结束整个程序的执行。

（3）C 程序书写格式自由。既允许在一行内写多个语句, 也允许将一个语句分写在多行上, 但每个语句必须以分号结束。

（4）“ /*……*/ ”表示对函数或语句的功能做注释。在“ / ”和“ * ”之间不能有空格，且“ /* ”和 “ */ ” 必须配对使用。注释是供人阅读的，并不参与编译和运行。注释可以出现在程序的任何地方，添加必要的注释可以提高程序的可读性。

（5）源程序中可以使用预处理命令（如 include 命令、define 命令），预处理命令以“ # ”开头，一般预处理命令应放在源文件或源程序的最前面。

C 程序的一般形式如下。

```
预编译处理命令
外部变量定义
main( )
{
    声明部分
    执行部分
}
sub 1( )                        /*自定义函数*/
{
    ……
}
……
sub n( )                        /*自定义函数*/
{
    ……
}
```

1.4.5 C 程序的运行步骤

程序设计是针对给定问题进行设计、编写和调试计算机程序的过程。在纸上写好 C 源程序

代码后，下一步就要在计算机上来实现它。在计算机上实现一个 C 程序一般要经过编辑、编译和连接及运行 3 个步骤。

1．C 程序的编辑

编辑是指将写在纸上的 C 源程序代码输入到计算机中，经过一定的修改后，以源程序文件的形式存储到存储介质（如硬盘、软盘等）中。编辑是在编辑器中进行的，一般 C 语言的编译系统都自带编辑器。C 的源程序文件的扩展名规定为 .c，也就是说，编辑好的源程序要以 .c 为扩展名存放到存储介质中。

2．C 程序的编译和连接

1）编译

计算机只能直接识别和执行二进制代码，不能直接识别和执行用高级语言编写的源程序。高级语言编译系统的功能就是将源程序翻译成二进制代码。源程序文件经编译形成目标程序文件。

C 程序的编译过程分为两步：首先对源程序进行预处理，将源程序中出现的预处理命令进行处理；然后将预处理后的源程序进行编译，即进行词法分析和语法分析。

在编译过程中若发现词法错误或语法错误，就转由错误处理程序进行处理，将错误信息显示到屏幕上，通知编程者进行修改，编程者修改程序后再重复此过程，直至完全正确，形成扩展名为 .obj 的目标程序文件。

2）连接

编译是针对一个特定的源程序文件进行的。通常一个较大的 C 源程序可能包含多个源程序文件，这些源程序文件可能是用户编写的，也可能是编译系统提供的（如标准库文件），而编译系统只能单独编译每个源文件，为每个源文件生成一个目标程序文件。因此，还需要通过连接程序的处理，将一个 C 程序所包含的多个目标程序文件连接后形成一个可执行程序文件。可执行文件的扩展名规定为 .exe。

3．C 程序的运行

对于编译和连接后形成的可执行文件，可以直接调入内存执行。

一个 C 程序的运行步骤如图 1.8 所示。

常用的 C 编译系统有 Turbo C、Microsoft Visual C++ 6.0、Quick C 等，详细的操作步骤请参考有关的用户手册。

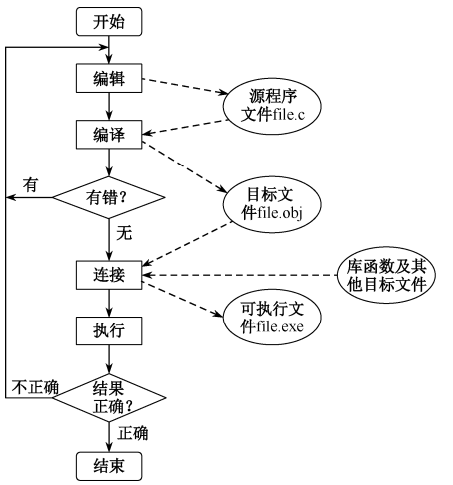


图 1.8　C 程序的运行步骤