第 1 章　概 述

1-1　程序设计

程序是计算机能直接或间接地接受处理的一系列指令的序列。它规定了计算机执行的动作序列，从而完成某一特定的任务。程序设计就是对所要处理的问题借助于某一程序设计语言设计出解决它的指令序列。近年来，计算机的软件技术有了飞速的发展，各种应用软件、系统软件、工具软件应运而生，程序设计作为一个重要内容和技能在软件开发过程中仍起着不可替代的作用。程序设计的原材料是数据和数据的存储容器，而处理这些原材料的工具是命令、函数和操作符。著名的计算机科学家沃思曾提出了一个有名的公式：

程序=数据结构＋算法

数据结构一般指数据的组织形式及其在计算机中的存储形式，而算法指对这些数据实施的具体的操作步骤。由此可见，程序设计除了对所涉及的数据组织外，关键是进行算法设计。程序设计的语言可分为三类：机器语言、汇编语言、高级语言。

1. 面向机器的机器语言和汇编语言

机器语言由 0，1 代码组成，由这种代码组成的指令序列，可由机器直接理解、执行，但用这种语言进行程序设计效率低，容易出错，很难掌握。

为了克服机器语言的弱点，人们开始用一些“助记符”来代替 0，1 编码，这种用助记符描述的指令系统称为汇编语言。用汇编语言编写的程序，机器不能直接识别，要先将汇编语言翻译成机器语言，才能被计算机理解和执行。这两种语言都是面向机器的，程序设计和机器设备有密切的关系。

2. 面向过程的高级语言

为了降低程序设计的复杂性，尽量减少机器设备对程序设计的影响，人们研制出了面向解题过程的一些高级语言，如 Basic，FORTRAN，Pascal 和 C 等。用这些语言进行程序设计时，语言描述更接近于自然，语法规则更趋向于合理，人们不必熟悉计算机的内部详细结构，只需集中精力解决问题的描述和求解上。其中 C 语言，因其具有丰富的数据类型，灵活方便的多种运算符，精练的语言表达式，高效的代码和可移植性，既能进行系统软件的开发，也可用于应用软件的编制，因而得到广泛的应用，并受到用户的普遍欢迎，成为目前进行程序设计的一种最常用的语言。

3. 面向对象的高级语言

进入 20 世纪 90 年代，提出了一种新的面向对象的程序设计方法，与之对应的是面向对象的程序设计语言，它更适合于对客观事件中实体及其状态和行为的描述，使得计算机求解问题更符合人的思维活动及其概念，所开发的程序可维护性、可重用性、可移植性更强，开发效率更高。如 Smalltalk，Eiffet，C++，Java 等，都是面向对象的程序设计语言。

进行程序设计，要采用适当的程序设计方法。由荷兰学者 E. W. Dijkstra 提出的结构化程序设计，是目前广泛采用的一种程序设计方法，它的要点包括“自顶向下，逐步求精和模块化”。即将一个大的复杂性问题分解成若干个较小的子问题，每个子问题就是一个模块，对每一子问题，若有必要可再行分解。对每一模块，首先定义所要实现的功能；然后设计为实现这些功能所必需执行的步骤，这些步骤也就是过程；接下来选择适当的程序结构来构建这些步骤；最后编写代码来实现这些结构，使每一步直接对应其程序代码。因此，面向过程的程序设计是一种以功能分析为基础，以算法设计为中心，以结构化为手段的程序设计方法。结构化程序设计常采用的结构有三种：顺序结构、选择结构和循环结构。

1-2 C 语言的发展和特点

C 语言是一种面向过程的结构化程序设计语言。作为高级语言，编程者不必熟悉具体的计算机内部结构和指令，就可以进行应用软件的开发；也可以借助于其对端口 I/O 操作、位操作、地址操作等，对计算机硬件进行直接操作，进而实现系统软件的编制。C 语言起初就是为了取代可读性和可移植性较差的汇编语言进行操作系统软件的开发而研制的。

C 语言的前身是 1963 年英国剑桥大学的 Matin Richards 推出的 BCPL（Basic Combined Programming Language）语言，它是一种开发系统软件时，作为描述语言使用的程序语言，既有高级语言的功能，又有机器语言的特征，离硬件较近，适合于系统软件的编写。1970 年美国贝尔实验室的 Ken Thompson 对 BCPL 作了进一步的简化和改进，提出了简单而更接近于计算机硬件的 B 语言，并用 B 语言完成了 Unix 操作系统软件初版的开发工作。1972 年至 1973 年间，贝尔实验室的 Dennis Ritchie 和 Ken Thompson 合作把 Unix 系统中的 90% 用 C 语言进行了改写，并在 PDP-11 小型机上调试、运行成功。随着 Unix 系统在各种机型上的移植和推广，C 语言也随之移植到大、中、小和微型机上，成为最广泛应用的程序设计语言之一。

为了规范不同机种、不同字长（8~32bit）计算机上所使用的各种 C 编译系统，便于软件的移植和推广，美国国家标准局（ANSI）语言标准化委员会于 1983 年公布了一个 C 语言标准草案（83ANSIC），随后又对该草案作了进一步的修改和完善，于 1987 年推出了 C 语言标准——87ANSIC，目前商业运营的各种 C 编译系统，基本上都是遵循这一标准。广泛使用的 Turbo C，Microsoft C 等也遵循这一标准，本书所介绍 C 语言内容，均以 87ANSIC 标准为基础。

作为高级语言，C 语言有许多明显的特点，具体体现在以下几个方面：

（1）提供了丰富的数据类型和运算符。数据类型除整型、实型、字符型和数组外，还有指针、结构体、共用体等；运算符除了算术运算、关系运算、逻辑运算三大运算符外，还有条件运算、位运算、指针运算等 34 种运算符。

（2）含有结构化程序设计的基本语句。如 if ~else 语句、while 语句、for 语句、switch 语句等，以此实现常用的三种程序结构。

（3）函数是 C 语言程序设计的基本模块。C 语言提供了大量的标准函数，用户还可以根据解题需要自己设计函数，通过函数的组合，实现程序设计的模块化。

（4）语言描述简洁，设计自由度大，使用方便。如++i 就相当于 i=i+1，for(i=1;i<m;i++)

中就含有三条语句。

（5）能直接访问物理地址，可对硬件进行直接操作。

（6）有编译预处理功能，通过预处理可以进行宏定义、文件包含等处理，提高程序的可读性和可调试性。

（7）大小写字母有区别，除宏定义中参数名习惯上用大写字母外，其他情况一般用小写字母。

（8）目标代码质量高，可移植性好。

C 语言的其他特点，在随后的章节中都有详细的介绍。

1-3 简单的 C 语言程序

用 C 语言编写的程序，称为 C 语言源程序，简称 C 程序。以下通过例 1-1 来介绍 C 程序的基本组成。

例 1-1 任给两个整数 a 和 b，求表达式 $y=\sqrt{a^2+b^2}$ 的值。

```
#include <math.h>
main()
/*Calculating the   value of a expression*/
 {
    int a,b;
    float y;
    a=3;
    b=4;
    y=sqrt(a*a+b*b);
    printf("This is test program.\n");
    printf("a=%d,b=%d\n",a,b);
    printf("Y=%f\n",y);
  }
```

程序的执行结果为

```
This is test program.
a=3,b=4
y=5.000000
```

（1）一个 C 语言程序通常由带＃的编译预处理语句开始。如例 1-1 中要用到求平方根的运算，它可以通过 C 语言的标准库函数 sqrt(x)来实现,因而必须在程序的一开始，通过 # include <math.h> 对含有此函数的头文件进行包含说明。

（2）C 程序由一个或多个函数组成，函数是 C 程序的基本单位，可使用标准的库函数，也可以使用自定义函数。一个自定义函数由函数的说明部分和函数体两部分组成。如本例中的 main()说明该函数名为 main，介于 main()下面大括号{... }之间的为函数体。

（3）一个完整的 C 程序只能而且必须有一个主函数 main()，不论 main()函数在程序的任何位置，执行时总是从 main()函数开始。

（4）函数中用到的任何变量，使用前都必须加以说明。如例 1-1 中 int a,b;说明 a 和 b 是整型变量，在程序中 a 和 b 是用来存放整型数据的，float y;说明 y 是实型变量，用来存放实型数。

（5）函数体内可包含 0 到多行语句，每行语句可包含 0 到多条语句，每条语句和数据定义的最后必须用‘;’作为其结束的标志。如例 1-1 中每行中只含有一条语句，皆以“；”结束。

（6）C 程序中是通过标准的 I/O 库函数来实现输入输出的。如例 1-1 中通过标准函数 printf(进行字符串和变量值的输出。

（7）/* */为注释语句，用于说明变量、语句和程序段的功能，它不被编译执行，可放在程序中的任意位置，以加强程序的可读性。对本书中的注释语句，在调试程序时，为了节省输入时间，可以略去不输入。

（8）C 源程序不能直接执行，一般需经编译、连接生成可执行文件后，方可执行。

1-4 问题的描述与 C 编程

1-4-1 算法设计

算法设计是程序设计的中心，一个问题若能用适当的算法加以描述，设计其程序代码也就变得较为简单，容易实现。因此，进行程序设计，首先要进行算法设计。所谓算法，简单地讲就是求解问题的方法，是对解题过程的抽象和精确描述，进行算法设计一般应当遵循以下几条原则。

（1）有效性：即算法中的每一步必须能转化成一条或多条能执行的指令。

（2）确定性：算法中每一步的含义是确定的，各步的执行顺序是确定的。

（3）输入输出：一个算法中应有适当个数的输入和输出，以便进行信息交换。

（4）有穷性：一个算法应当包括有限个操作步骤，最后必须能结束。

对应于程序设计的三种结构，算法也是通过三种基本结构来描述解题过程的，它们是

（1）顺序结构：该结构中各个操作是按书写顺序执行的。

（2）选择结构：按指定条件进行判断，根据判断结果，选择两条或多条分支中的一条路径执行。

（3）循环结构：根据循环条件，反复多次地执行循环体中的语句。

一个算法可以通过自然语言、流程图、N-S 图、PAD（Problem Analysis Diagram）图、伪代码等多种方式加以描述。 N-S 图是由美国学者 I. Nassi 和 B. Schneiderman 1973 年提出的，N 和 S 是取他们二人名字的首字母。这一种算法描述图适合于在结构化程序设计中进行算法设计，因此得到广泛的应用，是目前较为流行的一种算法描述方式，它的具体形式如下：

（1）顺序结构的 N-S 图，如图 1-1 所示。它表示先执行 A 的内容，然后再执行 B 的内容。A 和 B 分别表示一个基本操作。

（2）选择结构的 N-S 图，如图 1-2 所示。它表示当条件 P 成立时，执行 A 的内容，而后执行分支结构下的语句；当条件 P 不成立时，执行 B 的内容，而后执行分支结构下的

语句。

（3）循环结构的 N-S 图，如图 1-3 所示。它表示先检查给定的循环条件 P 是否成立，若成立，则执行 A 的内容，A 的内容整个执行完后，再来判断条件 P 是否成立。若某一次判断条件 P 不成立，则不执行 A 的内容，退出循环，而转至循环结构下的 B 语句执行。

使用这三种基本结构，就可以构建各种满足问题需求的算法设计。

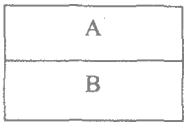


图 1-1 顺序结构 N-S 图

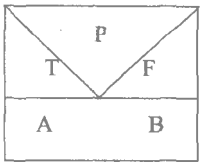


图 1-2 选择结构 N-S 图

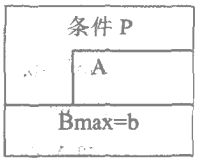


图 1-3 循环结构 N-S 图

1-4-2 问题的描述与 C 编程

对于给定的问题，首先分析该问题要求做什么，用自然语言或数学公式来进行算法的抽象描述；然后将这些抽象算法具体化，用 N-S 图等算法描述工具逐步解决如何去做的问题；最后把全部算法用程序语言描述成由计算机可实现的基本操作。这也就是通常所说的“逐步求精”程序设计方法。下面看一下问题的具体描述和如何用 C 语言进行编程。

例 1-2 任意给定三个数，找出其中最大的一个，并将其输出。

对这样一个问题，首先用自然语言分步加以描述：

（1）任意给定三个数。即要求提供输入信息，而输入信息在计算机中要求存放在某个位置，在此提供三个变量 *a,b,c*，用于存放三个数据。因此，这一步可具体化为：输入三个数，放入 *a,b,c* 中。

（2）找出其中最大的。要想找出三个数中最大的，可以通过两组比较来完成：

- 找出存放在 *a,b* 中的最大数，把其值放在另一变量 max 中。此时，max 中存放的是 *a,b* 中大的一个数。要实现这一步，必须判断 *a* 和 *b* 中的数哪个大，如果 *a* 大于 *b*，则 max=*a*，否则（说明 *a* 小于 *b*），max=*b*。
- 找出 *c* 和 max 中较大的一个，并将其放入 max 中。要实现这一步，必须判断 c 和 max 中的数哪个大，如果 *c* 大于 max，说明 *c* 是三个数中最大的一个（因为 max 中放的是 *a,b* 中的大数，*c* 比 *a,b* 中较大的数还大，说明 *c* 是最大的），则 max=c，否则（说明 *c* 比 max 中的数小，也就是 *c* 比 *a,b* 中较大的小，此时，max 中存放的就是三个数中最大的），max 的值保持不变。

（3）输出最大的数。此时保存在 max 中的数就是 *a,b,c* 三个数中最大的一个，直接输出即可，即输出 max。

为了看起来简洁、直观，便于将来用程序设计语言来进行编程，我们用 N-S 结构图对以上内容描述如图 1-4 所示。

按照 C 语言的编程要求，该问题的 C 语言程序可编制如下：

```
main()
/*find  the  max  in a,b,c*/
```

```
    int a,b,c,max;                    /*定义 a,b,c,max 为四个整型变量*/
    printf("Please input a,b,c:");    /*在屏幕上显示输入提示*/

    sacnf ("%d,%d,%d",&a,&b,&c);      /*输入 a,b,c 三个整型数*/
    if(a>b)                           /*若 a>b，则 max=a*/
      max=a;
    else                              /*若 a<b，则 max=b*/
      max=b;
    if(c>max)                         /*若 c>max，则 max=c*/
      max=c;
     printf("Max=%d",max);            /* 输出最大值 max*/
}
```

程序的执行结果如下：

```
Please input a,b,c:5,15,7↙
Max=15
```

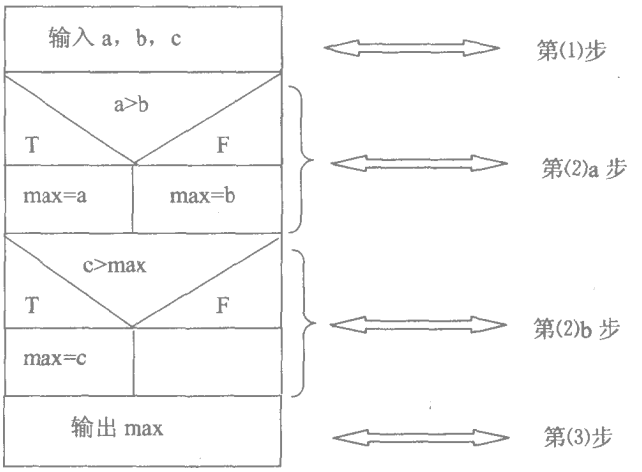


图 1-4　求 max 的N-S 结构图

1-5　C 语言程序的编译和执行

C 语言程序，不能直接运行，只有通过编辑、编译、连接后，才能成为一个由机器指令代码组成的可单独执行的程序。其过程可分为以下几步。

（1）编辑源程序：对编制好的源程序要输入到计算机内，通过一个编辑软件对输入的源程序进行适当的编辑，然后将其存储在磁盘上，一般该文件的后缀名为 .c。

（2）对源程序文件进行编译：第（1）步编辑好的扩展名为 .c 的源程序文件，用给定的 C 编译器进行编译，使其生成一个由机器指令组成的目标程序。

（3）连接目标程序：第（2）步生成的目标程序和程序中所用到的相关库函数，或其

他目标程序进行连接，使其生成一个可执行的文件，扩展名一般为.exe。

（4）执行程序：生成的扩展名为.exe 的可执行文件直接加以运行，就可以得到程序设计所要求的结果。

Turbo C 2.0 是由美国 Borland 公司 1989 年推出的一个 DOS 环境下微机上普遍使用的 C 语言程序开发软件，它是一个集编辑、编译、连接、调试为一体的集成开发环境，本书所介绍的 C 程序设计，基本上以此开发环境为基础。以下通过例 1-3 来说明在 Turbo C 下完成前述步骤的具体过程。

例 1-3　试对例 1-1 中的程序进行编辑、编译和运行。

步骤如下：

（1）进入 Turbo C 的集成环境。在 DOS 环境下进入装有 Turbo C 的子目录，然后键入 tc↙，即可进入 Turbo C 的集成环境，如图 1-5 所示。

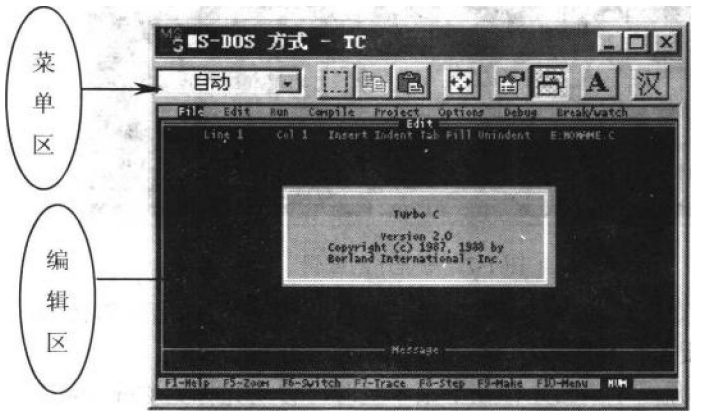
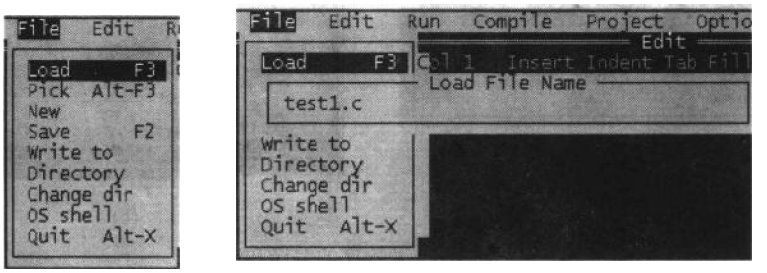


图 1-5　Turbo C 的集成环境

（2）建立一新文件，对其进行编辑。在 Turbo C 的集成环境下，此时光标处于集成环境的最上一行的 File 菜单上，按两次 Enter 键，显示 File 下的子菜单，如图 1-6 中(a)所示，表示在 File 菜单中有 9 个可供选择的菜单项；用↑↓键把光标移动到 Load 项上，按 Enter 键，表示要调用某一个 C 程序文件，如图 1-6 中(b)所示；输入源程序文件名 test1.c（此文件名由用户起），按 Enter 键，进入源程序的编辑状态。



(a)　　(b)

图 1-6　选择 Load 菜单项

（3）输入编辑内容。此时光标处于图中的编辑区，可以按前面所编程序逐行从键盘输

入，如图 1-7 所示。在编辑过程中，若需要修改程序内容。可通过键盘上←↑→↓等移动键来移动光标，使其到达要修改处，然后直接进行修改。可通过 Insert 键设置插入(Insert)和重写（Overwrite）状态；用 Delete 键删除当前光标所在的字符，用 Ctrl+y 键删除当前光标所在的一整行字符。在确认程序内容正确后，按 F2 键把所编辑的程序 test1.c 存放到磁盘指定的位置。

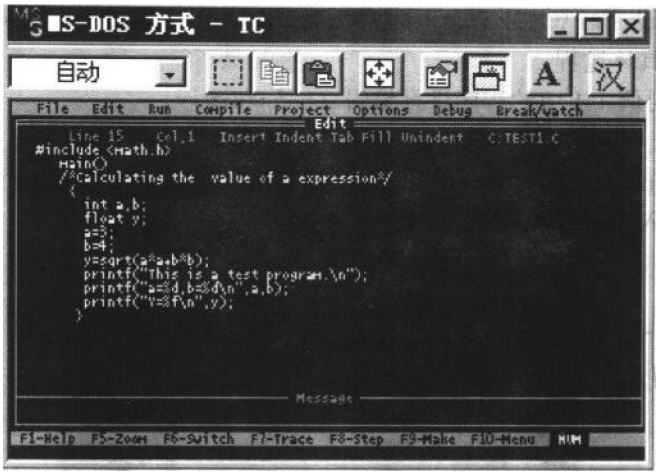
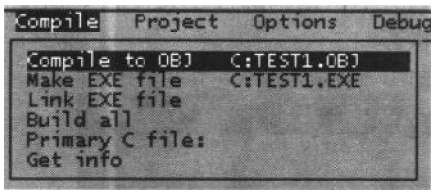


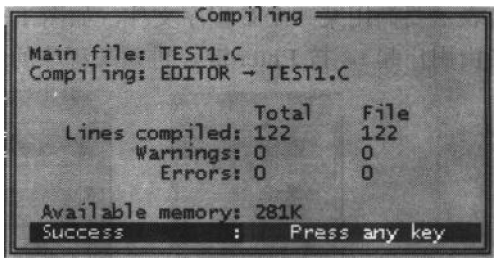
图 1-7　源程序的编辑

（4）退出编辑状态。当光标处于编辑区时，按 F10 键，退出编辑状态，使光标回到主菜单区；再按 F10 键，可使光标又回到编辑状态。

（5）编译、连接源程序。当光标在主菜单区时，移动光标到 Compile 项上，按 Enter 键，显示 Compile 的下拉子菜单，如图 1-8(a)所示，向下移动光标到 Compile to OBJ 上，按 Enter 键，对源程序文件 test1.c 进行编译，生成 test1.obj 目标文件，编译完成后的提示信息如图 1-8(b)所示。按任一键进入编辑区，再按 F10 键，使光标回到主菜单区，移动光标到 Compile 项上，按 Enter 键，显示 Compile 的下拉子菜单，如图 1-8(a)所示，向下移动光标到 Link EXE file 上，按 Enter 键，对程序进行连接，生成 test1.exe 可执行文件。



(a)



(b)

图 1-8　源程序的编译和连接

（6）执行程序。按 F10 键，使光标回到主菜单区，移动光标到 Run 菜单项上，按 Enter 键，显示 Run 的下拉子菜单，如图 1-9 所示，向下移动光标到 Run 上，按 Enter 键，对 test1.exe 可执行文件进行运行。按 Alt+F5 键（或选择 Run 菜单下的 User screen 子菜单项），就可

以看到显示在屏幕上的执行结果。

（7）退出 Turbo C 状态。按 F10 键，使光标回到主菜单区，移动光标到 File 菜单项上，按 Enter 键，显示 File 的下拉子菜单，如图 1-6(a)所示，向下移动光标到 Quit 项上，按 Enter 键，退出 Turbo C 状态。也可在编辑区，直接按 Alt+x 键，退出 Turbo C 状态。

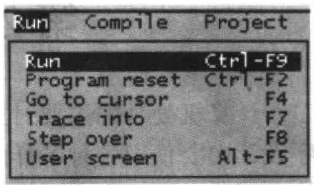


图 1-9 执行程序

有关 Turbo C 的更多功能和详细内容，可参看相应的参考书目。

习题一

1.1 程序设计的主要工作是什么？

1.2 在结构化程序设计中主要采用哪些结构？

1.3 C 语言的主要特点是什么？

1.4 进行算法设计需要遵循哪些原则？

1.5 N-S 图中的三种结构体有什么具体内涵？

1.6 上机调试例 1-2 的程序。

1.7 给定任意两个整数 a 和 b，求其差值且其值不能为负数。要求先用自然语言写出求解该题的解题步骤，然后用 N-S 图进行算法描述，最后参照本章的例题，编制 C 语言程序，并上机调试。

第 2 章　数据类型和基本运算

计算机处理的主要对象是数据，数据通过各种运算和操作，其状况可以发生变化。程序设计中要使用各种各样的数据，使用何种类型的数据，这些数据参加哪些运算，都应该给出明确的定义和表示。在 C 语言中，提供了丰富的数据类型和运算符，通过对这些数据和运算符的适当组合，可以构建各种各样的运算，满足不同用户的需求。本章主要介绍常用的数据类型和基本运算。

2-1　基本符号与标识符

2-1-1　基本符号

符号是组成源程序语句的基本单位。 C 语言中常用的一些基本符号有

（1）10 个阿拉伯数字 0，1，2，3，4，5，6，7，8，9。

（2）26 个小写英文字母 a,b,c…z 和 26 个大写英文字母 A,B,C…Z。

（3）常用的一些运算符,如算术运算符＋，-，*，/，%,关系运算符＞,＜，>=，<=，!=，初等运算符（），[]等。

（4）其他图形字符，如下划线_，以及一些非图形字符，如空格符（space），回车符（CR）和换行符（LF）等。

2-1-2　标识符

标识符是用来标识符号常量、变量、数据类型、函数、宏及文件等的名字。在 C 语言中，标识符是由字母、数字或下划线组成的符号序列，且要求符号序列的第一个字符是字母或者下划线，其长度随系统而有所不同，一般情况下不超过 8 个。作为变量名、函数名和文件名使用的标识符一般用小写字母，作为符号常量名和宏名使用的标识符一般用大写字母。

例如 sum，av_3，add，MULTI，a1，_div，B2 等是合法的标识符，而 1a，%32d，abc@，b^3，256# 等是不合法的标识符。

有关标识符的几点说明。

（1）大、小写字母表示的标识符不一样。如 const 和 CONST 不相同。

（2）不同的系统对标识符长度的规定不一样。大多数系统规定的标识符的有效长度为 8 个字符。如 schoolno1 和 schoolno2 是两个相同的标识符。

（3）用户定义的标识符不能和关键字以及标准库函数名同名。

（4）自定义标识符最好具有一定的含义，以便程序的阅读。如存放姓名的变量用 name，存放总和的变量用 sum 等。

2-1-3　关键字

关键字是 C 语言中具有特定意义的标识符，一般也称为保留字，有 32 个。如表示数

据类型的关键字有 char, int, short, float, double，long, unsigned, void, union, stru, enum 等，表示程序结构的关键字有 if, else, for, while, do, break, continue, return, goto, switch, case, default 等，表示存储方式的关键字有 auto,extern,static,register 等，还有表示数据类型定义的关键字 typedef 等。

例 2-1 编程计算任意两个整数的和。源程序 fil201.c 如下：

```
#include "stdio.h"
main()
    {
      int sum,a,b;
      scanf("%d,%d", &a,&b);
      sum=a+b;
      printf("%d\n",sum);
    }
```

程序的执行结果为

```
10,20↙
30
```

上述程序的语句是由一些基本符号组成， a,b,sum 是变量名，main 是函数名，int 是关键字，scanf 和 printf 是 I/O 库中输入和输出函数名。

2-2 数据类型

数据类型是描述数据所具有的数值的类型，常用的数据类型有三大类：基本类型，它有整型、实型（单精度实型，双精度实型）、字符型和枚举型；构造类型，它有数组类型、结构体类型和共用体类型；指针类型。本节主要介绍整型，实型、字符型三种数据类型，其他类型在后续章节中详细介绍。

2-2-1 整型

整型是数据值为整数的数据类型。整型类型有 int（整型）、short 或 short int（短整形），long 或 long int（长整型）、unsigned 或 unsigned int （无符号整型），unsigned short（无符号短整型）和 unsigned long （无符号长整型）几种。

在计算机中，数据是由二进制位组成，每一个二进制位称为位，每八个二进制位为一个字节，每两个字节为一个字，每两个字为一个双字。对一个放在内存单元中的二进制数来说，如果它的最高位为符号位，则该二进制数为带符号数，如果它的最高位为数值位，则该二进制数为无符号数。因此，同一个二进制数，如果最高为位 1，无符号数和带符号数表示的数值不一样。如 1111111111111111 从无符号数的角度来理解，是 65535，如果从带符号数的角度来理解，是-1。

1. 带符号的整型 int, short 和 long

int, short 和 long 表示的整型数为带符号的整型数。在 IBM PC 机器系统下：

int 和 short 表示的整型数和短整型数，在内存中占两个字节， 16 个二进制位，它所能表示的整数范围为 -32768 ~32767（即 -2^{15}~$2^{15}-1$）。如 100. -12. -3267, 7809 等，为 int 类型或 short 类型数据。

long int 表示的长整型数，在内存中占四个字节， 32 个二进制位，它所能表示的整数范围为 -2147483648~2147483647（即 -2^{31}~$2^{31}-1$）。如 7745789. -8654321, 3581 等，为 long int 类型数据。

2. 无符号整型 unsigned, unsigned short 和 unsigned long

unsigned, unsigned short 和 unsigned long 表示的整型数为无符号的整型数。在 IBM PC 机器系统下：

- unsigned 和 unsigned short 表示无符号整型数和无符号短整型数，在内存中占两个字节，16 个二进制位，范围为 0~65535（即 0~$2^{16}-1$）。如 63267, 32768 为 unsigned 类型或 unsigned short 类型数据。
- unsigned long 表示无符号长整型数，在内存中占四个字节， 32 个二进制位，范围为 0~4294967295 （即 0~$2^{32}-1$）。如 7375661. 32345880 等，为 unsigned long 类型数据。

2-2-2 实型

实型也称为浮点型，是表示实数的数据类型。根据实数所表示的范围，实型又分为单精度实型（ float 型）和双精度实型（ double 型）。

float 型数和 double 型数的有效数字随机器系统而定。在大多数机器系统下：

- float 型数在内存中占用 4 个字节，32 个二进制位来存储，它提供 7 位十进制数的有效数字，所能表示的实数范围为 10^{-38}~10^{38}。如 7572.3321. -32.453678 等，为 float 类型数据，但只有前面 7 位有效数字，后面的数字不再起作用。
- double 型数在内存中占用 8 个字节，64 个二进制位来存储，它提供 16 位十进制数的有效数字，所能表示的实数范围为 10^{-308}~10^{308}。如 12354678.774123, -0.987654321 等，为 double 类型数据，只要数字的个数不超过 16 位，皆为有效数字。

2-2-3 字符型

字符型是表示单字符或字符串的数据类型。在计算机中，单字符类型为 char 型，单字符型数据占用 1 个字节，8 个二进制位来存储，且存储的是字符的 ASCII 码值。例如

```
char c1='a';
```

就是将 a 的 ASCII 码值 97 赋给了字符变量 c1。因此，单字符数据可与整型数据相运算。

字符串是由一组字符组成的序列，字符串在内存中除了字符占有内存单元外，还要在其后加一个“ \0”，作为串结束标志。例如 you are good! 在内存中的存放形式为

y	o	u		a	r	e		g	o	o	d	!	\0

提示 

不同类型数据之间可以进行运算，运算时，机器系统可以自动进行类型转换。如字符型可以转换为整型，单精度型可以转换为双精度型。其一般转换规则为

char(short,unsigned short) -> int -> unsigned int -> long -> unsigned long -> float -> double

也可以用强制类型转换符“( )”转换成所需要的数据类型。其一般格式为：

（类型名）（表达式）

将表达式值的数据类型转换成类型名所定义的数据类型。

例 2-2 各种数据类型的使用。源程序 fil202.c 如下：

```
#include "stdio.h"
main()
    {
      int a;
      long m;
      float x;
      double y;
      char ch;
      a=1705;
      m=-774916055;
      x=-412537.28;
      y=2315468975.6841;
      ch='B';
     printf("%c,%d,%ld\n",ch,a,m);
      printf("%f,%f\n",x,y);
    }
```

程序的执行结果为

```
B,1705,-774916055
-412537.281250, 2315468975.684100
```

上列程序中，a 中存放的是整型数据，m 中存放的是长整型数据，x 中存放的是单精度实型数据，y 中存放的是双精度实型数据，ch 中存放的是字符型数据。

2-3 运算量

运算量是在程序运行过程中处理的量，在 C 语言中，根据运算量值是否可以改变而将运算量分为常量、变量。库函数是以库的形式存放在计算机系统中的函数。在 C 语言中，数据的输入和输出是由 I/O（输入/输出）库函数来实现的。本节主要介绍常量、变量及一些常用的库函数。

2-3-1 常量

常量是指在程序运行过程中，数据值始终保持不变的运算量。根据常量的数据类型，常量可分为整形常量、实型常量和字符型常量。

1. 整形常量

在 C 语言中，数据类型为整型的常量称为整形常量或整数，可以用三种形式表示：八进值、十进值、十六进值。

（1）八进制整数：以 0，1，2，3，4，5，6，7 八个数值表示的整数，可为正数，也可为负数。八进制整数以数字 0 开头。例如 06，-034，0721，-01000 等。

（2）十进制整数：以 0，1，2，3，4，5，6，7，8，9 十个数值表示的整数，可为正数，也可为负数。例如-10，589，-9999 等。

（3）十六进制整数：以 0，1，2，3，4，5，6，7，8，9，a，b，c，d，e，f（或 A~F）表示的整数，可为正数，也可为负数。十六进制整数以 0x 开头。例如 0x1，-0x34，0xa，-0x2F5 等。同一个整数在不同进制下的表示方式如表 2-1 所示。

表 2-1 整数在不同进制下的表示

八进制整数	十进制整数	十六进制整数
07	7	0x7
-'010	-8	-0x8
014	12	0xc(0xC)
-377	-255	-0xff

2. 实型常量

数据类型为实型的常量称为实型常量或实数（浮点数），实数用两种形式表示：十进制小数形式和指数形式。

（1）十进制小数形式：一般形式是由整数部分、小数点和小数部分组成。整数部分或小数部分可以没有，但小数点不能省略。例如 15.2，-89.0，0.0，345，-678，1.0 等都是合法的十进制小数形式实数。

（2）指数形式：表示 $a*10^b$ 这种常量的实数形式。该实数形式由整数部分或小数部分、e(E)和整数部分组成。例如 1.25E2，-9.13E5，3E-2 是合法的十进制指数形式实数，分别表示 $1.25*10^2$，$-9.13*10^5$ 和 $3*10^{-2}$。同一个实数在不同形式下的表示方式如表 2-2 所示。

表 2-2 实数在不同形式下的表示

十进制小数形式	指数形式
123.5	1.235E2
-913	-9.13E2
0.00459	4.59E-3

3. 字符型常量

字符型常量指常量为一个字符或者一个字符串。字符型常量分为字符常量和字符串常量。字符常量主要有两种类型：

一种字符常量是用一对单引号括起来的常量。如‘a’，‘z’，‘0’，‘1’，‘@’，‘Z’等，其中‘z’和‘Z’表示不同的字符常量。

另一种字符常量是以“＼”开头的字符或字符序列，其中，“＼”为转义字符，它的功

能是按其后字符或字符序列的要求输出。这种字符常量可以表示 ASCII 码表中的所有字符、也可以输出一些控制字符。

常用的一些转义字符（字符序列）及其功能说明如下：

- \abc　输出一个八进制数所代表的 ASCII 码字符。
- \r　输出位置回到当前行的最开头。
- \n　输出位置移到下一行的开头。
- \b　输出位置回退一个字符。
- \t　输出位置到下一个输出区（通常一个输出区占 8 列）。
- \\　输出“\”。
- \'　输出“'”。

字符串常量指用双引号括起来的常量。如“this”，“good”，“b1”，“a ”等。注意，'a' 和"a"是两个不相同的字符型常量，'a'是一个字符常量，在内存中占有一个字节单元，"a" 是一个字符串，在内存中存放的是 a 和 \0，占有两个字节。

4. 符号常量

在 C 语言中，还有一种常量，称为符号常量。符号常量是用一个标识符来表示的常量。在程序中，如果一个常量经常出现时，就可以用一个符号常量来表示，这样有助于程序的修改。例如

```
#define PI  3.14159
```

定义了一个符号常量 PI，它的值为 3.14159，当这个常量需要修改时，可以通过修改它的定义形式来修改它，这样就可以达到一次定义多次使用的目的。定义符号常量的一般形式为

```
#define 符号常量名  常量
```

其中，符号常量名一般使用大写字母。

例 2-3　常量的使用。源程序 fil203.c 如下：

```
#include "stdio.h"
#define EC  2.17
main()
    {
      int a,b,c;
      float x,y,g;
      char ch1,ch2,ch3;
      a=78;b=073;c=0xaf;
      x=12.35;y=9.13e3;g= EC;
      ch1='M';ch2='\101';ch3='\n';
     printf("%d,%o,%x\n",a,b,c);
      printf("%f,%e,%f\n",x,y,g);
      printf("%c%c%c\n",ch1,ch3,ch2);
    }
```

程序的执行结果为

```
78,73,af
12.350000,9.130000e+03,2.170000
M
A
```

在例 2-3 中，78，073，0xaf 为整型常量，12.35，9.13e3 为实型常量，'M'，'\101'（代表字符'A'）、'\n'（代表换行符）为字符型常量，EC 为符号常量。

2-3-2 变量

变量是指在程序运行过程中，数据值可以改变的运算量。每个变量用变量名来区分，变量名的命名规则与标识符的规定相同。通常，在程序中，变量是先定义其数据类型，然后才可以使用它。变量一经定义，即可根据它所属的数据类型确定其在内存中所占的字节数。变量的定义格式：

数据类型名 变量表

其中，数据类型名可以是前面所讨论的数据类型中的某一个；变量表可以是一个变量名或多个变量名组成的序列。多个变量名之间以“，”分隔。例如

```
int x,y,z;
```

定义了三个变量 x,y,z，它们都为 int 型变量，在内存中占据三个独立的存储单元，每个单元的大小为两个字节。根据变量的数据类型，变量可分为整形变量、实型变量、字符型变量。

1. 整型变量

整型变量有带符号型整型变量和无符号型整型变量。带符号型整型变量有 int，short 和 long 变量三种。无符号型整型变量有 unsigned，unsigned short 和 unsigned long 变量三种。在给整型变量赋值时，其值不能超过变量的数据类型所确定的数据字节范围。即不能把多字节数据类型范围内的变量，常量赋值给少字节数据类型范围内的变量。例如

```
int  x;
long y;
y=32769;
x=y;                (×)
```

此处，y 为长整型变量，在内存中占 4 个字节，x 为短整型变量，在内存中占 2 个字节，y 的值超出了 x 所定义的数据类型的范围。

2. 实型变量

实型变量有 float 型变量和 double 型变量两种。一个实型常量可以赋值给 float 型变量和 double 型变量，double 型变量所获得的数据值的精度比 float 型变量所获得的数据值的精度高。例如

```
float a,b;
double x,y;
```

此处，a,b 为两个单精度实型变量，每个变量在内存中占 4 个字节；x,y 为两个双精度实型

变量，每个变量在内存中占 8 个字节。

3. 字符变量

字符变量只有一种 char 型。它的取值为一个字符。定义形式如下：

char 变量名[，变量名…，变量名]

例如，char Letter='A' ；此语句定义了一个字符变量 Letter，它在内存中占 1 个字节，其值为字符'A'，在内存中实际存放的是该字符对应的 ASCII 代码值 65。

对于字符串常量，只能通过定义一个字符数组或字符指针来存放， C 语言中没有提供专门的字符串变量。例如

```
static char str[]={"book"};
```

就是定义了一个静态的字符数组 str[]，用来存储字符串“ book” ，有关字符数组或字符指针的更详细内容，将在后续章节中介绍。

4. 变量赋初值

定义好一个变量后，往往需要对该变量赋一个初值。可以通过赋值语句为变量提供一个初值,如在例 2-2 中通过 a=78 把整型常量 78这样一个初值赋给整型变量 a,通过 ch1='M' 把字符常量 M 赋给字符变量 ch1 ；也可以通过输入函数为一个或多个变量赋初值，如在例 2-1 中通过输入函数 scanf("%d,%d", &a,&b)从键盘为两个整型变量 a，b 提供两个整型常量。为了减少程序代码，省去对变量赋初值所使用的语句， C 语言中规定，在定义变量的同时，可以对变量进行初始化，即对变量赋初值。

例 2-4 变量的使用。源程序 fil204.c 如下：

```
#include "stdio.h"
main()
    {
      int a=10,b=20,sum;
      float x=79.85,y=92.45,aver;
      char ch1='f',ch2;
      sum=a+b;              /*求 a,b 的总和*/
      aver=(x+y)/2;         /*求 x,y 的平均值*/
      ch2=ch1+5;            /*求 ch1 中所放字符后的第 5 个字符*/
     printf("%d,%f,%c\n",sum,aver,ch2);
    }
```

程序的执行结果为：

```
30,86.14994,k
```

在上述程序中，定义了三个整型变量 a,b,sum，它们用于存放整型数据，并通过对变量赋初值，对 a 赋予 10、b 赋予 20 ；定义了三个实型变量 x,y,aver ，它们用于存放实型数据，并通过对变量赋初值，对 x 赋予 79.85、b 赋予 92.45 ；定义了两个字符型变量 ch1,ch2，它们用于单个字符的存放，并通过对变量赋初值，对 ch1 赋予字符 f。

2-3-3 库函数

程序设计中经常要用到某些函数，如三角函数、对数函数、字符处理函数、图形绘制