21世纪高职高专规划教材 计算机系列

C语言实例教程

主　编　黄逵中
副主编　黄泽钧　周志敏

- 数名一线教师多年教学经验集萃
- 图文并茂、条理清晰、易教易学
- 采用任务驱动教学模式编写
- 精心设计习题与实训
- 免费提供PPT格式电子教案

中国电力出版社
www.infopower.com.cn

21世纪高职高专规划教材 计算机系列

C语言实例教程

主　编　黄逵中
副主编　黄泽钧　周志敏

中国电力出版社
www.infopower.com.cn

内容提要

本书按任务驱动教学法的要求编写，开始讲解了计算机程序设计原理、Turbo C 的运行环境及 C 语言的基本语法。然后通过计算器、成绩单、迷宫等七个大的实例（包含若干小例子）的讲解，将 C 语言的数据类型、运算符与表达式，顺序结构程序设计，选择结构程序设计，循环结构程序设计，函数，编译预处理，数组，指针，结构体、联合体与枚举类型及文件等知识点融于其中，运用案例法教学，使学生掌握相应的知识点，并且精心设计了实训和习题，真正做到了教、学、用相结合，充分体现高职高专教学的特点。本书由具有多年实践教学经验的一线老师编写，充分体现了高职高专教学的特色，理论上必需、够用，加强理论联系实际，突出实用性、操作性，语言上通俗易懂、趣味性强，做到了老师好教、学生易学。

本书可作为高职高专相关专业学生学习 C 语言程序设计的教材，也可作为广大计算机编程爱好者学习 C 语言的自学教材和参考书。

图书在版编目（CIP）数据

C 语言实例教程 / 黄逵中主编. —北京：中国电力出版社，2004.8

21 世纪高职高专规划教材·计算机系列

ISBN 7-5083-2298-3

I.C... II.黄... III.C 语言—程序设计—高等学校：技术学校—教材 IV.TP312

中国版本图书馆 CIP 数据核字（2004）第 079197 号

丛 书 名：21世纪高职高专规划教材·计算机系列

书　　名：C语言实例教程

出版发行：中国电力出版社

地址：北京市三里河路6号　　邮政编码：100044

电话：（010）88515918　　传真：（010）88518169

本书如有印装质量问题，我社负责退换

印　　刷：汇鑫印务有限公司

开本尺寸：185×233　　**印　　张**：19　　**字　　数**：413千字

书　　号：ISBN 7-5083-2298-3

版　　次：2004年9月北京第1版

印　　次：2004年9月第1次印刷

印　　数：0001—4000册

定　　价：27.00 元

版权所有，翻印必究

21世纪高职高专规划教材·计算机系列

编 委 会

主任委员：

宗 健 岳国英

副主任委员：（以姓氏笔画为序）

丁亚明 马敬卫 王树勇 王晓光 冯玉东 刘广峰

朱世同 刘克兴 刘治安 齐现伟 孙奕学 孙春临

孙 辉 陈 东 李亚生 陈希球 陈 炜 寿建平

罗 众 林逢春 崔凤磊 黄华国 彭同明

委 员：（以姓氏笔画为序）

马冬生 万朝阳 王卫东 王建华 王展运 石文华

付晓波 朱卫红 安丰彩 吕 来 刘 阳 李大庆

何万敏 陈忠文 张国锋 李 娜 张海波 陈 智

罗亚东 胡文红 姚发洲 侯仰东 胡顺增 秦昌平

康玉忠 黄泽均 黄逵中 梁 曦 廖立军

21 世纪高职高专规划教材参编院校

（排名不分先后）

保定电力职业技术学院
山东电力高等专科学校
黄河水利职业技术学院
湖北水利水电职业技术学院
长江工程职业技术学院
郑州电力高等专科学校
武汉电力职业技术学院
江西电力职业技术学院
浙江水利水电高等专科学校
福建水利电力职业技术学院
广东水利电力职业技术学院
四川水利职业技术学院
兰州电力技术学院
兰州电力学校
南昌水利水电高等专科学校
贵州电力职业技术学院
福建电力职业技术学院
广西电力职业技术学院
内蒙古电力学校
浙江电力职业技术学院
四川电力职业技术学院
石家庄职业技术学院
秦皇岛职业技术学院
唐山工业职业技术学院
唐山科技职业技术学院
天津职业大学
天津大学职教学院
天津理工大学职业技术学院
北京科技大学（管庄校区）
天津渤海职业技术学院
天津轻工职业技术学院
天津中德职业技术学院
天津石油职业技术学院
北京联合大学
太原理工大学
长治职业技术学院
湖南工业职业技术学院
广西工学院职业技术学院
苏州职业大学
南通职业大学
常熟理工学院
常州工学院
徐州工程学院
常州纺织服装职业技术学院
常州轻工职业技术学院
常州信息职业技术学院
连云港职业技术学院
南京工程学院
武汉公交职业技术学院
湖北轻工职业技术学院
武汉职业技术学院
四川工程职业技术学院
四川托普信息技术职业学院
泸州职业技术学院

试读结束，需要全本PDF请购买 www.ertongbook.com

前　言

C 语言功能强大，表达能力强，使用灵活方便，用 C 语言编写的源程序代码紧凑，生成的目标代码质量高。这些不争的事实及优点，让诸多 PASCAL 语言爱好者不得不对 C 语言另眼看待，纷纷把多年来一直使用的 PASCAL 语言换成了 C 语言。

由于 C 语言的应用越来越广泛，全国计算机等级考试的三级和四级考试又都必考 C 语言，因此，各高校无论是计算机专业还是非计算机专业都开设了 C 语言课程。

然而，从我们所了解到的情况来看，有相当大的一部分的学校，尤其是高职高专学校，其 C 语言的教学效果并不理想。主要表现在：第一，绝大多数学生还没有入门，课程就结束了；第二，即使是“入了门”的学生也只会考试，不会编写程序。究其原因，我们认为主要是缺少真正具有职业教育特点的 C 语言教材。

高等职业技术教育虽然已开办了五年，市面上也出了很多“高职高专”教材，但在我们看来，有些教材与本科教材没有多大的区别，没有体现出职业教育的特点。首先，从教材的内容上看，有些教材只是本科教材在篇幅上的压缩版。书是变薄了，但内容没有减少，C 语言丰富的运算符和复杂的语法规则样样都有，“一个都不能少”。其次，从教材的编排和模式上看，有些教材通篇都只是为了验证知识点而去找例子讲，是为语言而讲语言，而不是从实际应用出发，根据需要讲解知识。第三，从应用能力的培养上看，有些教材根本就涉及不到这方面的教学与训练。21 世纪的高职高专教育是面向应用、面向职业的教育，因此，只懂一点理论知识，不懂实际应用，不会具体操作的“人才”将不会得到社会认可。

我们从高职高专教育的特点出发，针对目前高职高专计算机类教材的问题和不足，本着“不求全但求精”的原则，对 C 语言教材的教学内容和教学模式作了较大的改革，力求做到：第一，尽量舍去一些不常用的或者可用可不用的知识，又不失掉 C 语言的精华内容；第二，不为语言讲语言，而是从应用中讲语言；第三，讲 C 语言，更讲程序设计；第四，注重 C 语言基础知识的讲解，更注重 C 语言应用能力的培养。全书分为 9 章，只有第 1 章从语言规则讲起，其他 7 章（第 5 章为小结）则从具体的应用例子讲起，以实现任务为目标，讲解任务的实现方法和实现过程，连带讲解所需的 C 语言的其他知识。

第 1 章 基本知识。内容主要包括“存储程序控制原理”，程序设计的基本概念，高级程序设计语言的共性，C 语言的特点，计算机中数据的表示法，C 语言最基本的语法规则，Turbo C 2.0 集成开发环境，源程序编辑、编译、运行和调试方法等，不涉及复杂的语法和算法，目的是让初学者快速入门。

第 2 章 简易计算器。以实现简易计算器为例，讲解结构化程序设计方法，所涉及到的知

识包括函数的概念、函数的定义、函数的声明与调用、变量的作用域和生存期、逻辑运算与逻辑表达式。

第 3 章 成绩排名。首先讲成绩单的表示方法，然后讲解如何实现排序。所涉及到的知识包括数组的概念，一维数组的定义与引用，字符数组与字符串，指针的概念，指针的应用，指针与一维数组的关系，for 语句及循环嵌套。

第 4 章 迷宫。通过编写走迷宫游戏程序，让学生在编写一个复杂完整的程序的过程中得到能力的培养与训练，除掌握相关的知识之外，进一步掌握结构化程序设计的方法。所涉及到的知识包括二维数组的概念，二维数组的定义与引用，指针与二维数组的关系，switch 语句，文件的概念和文件操作等。

第 5 章 C 语言基本语法小结。本章的目的是让学生把前 4 章所涉及到的 C 语言各知识点相互联系在一起，构成一个相对完整的知识结构。

第 6 章到第 8 章主要从程序设计的方法来讲 C 语言的应用，对常用的“穷举法”、“迭代法”和“递归调用”作了较详细的讲解。目的是让学生进一步了解计算机算法的特点，从而提高程序设计的能力。

第 9 章 通信录。本章是 C 语言及程序设计综合应用能力的培养。通过设计、实现一个完整的通信录管理程序，使学生了解 C 语言应用程序的开发过程，包括从建立合理的数据结构开始到最终代码的实现的全过程。所涉及到的知识主要包括结构体的概念、结构体的相关知识和实际应用、枚举数据类型的应用等。

本书由武汉电力职业技术学院的黄逵中老师担任主编，由湖北水利水电职业技术学院的黄泽钧老师和浙江水利水电高等专科学校的周志敏老师担任副主编，参加编写的还有武汉电力职业技术学院的肖继文老师。黄泽钧老师编写第 3 章、第 5 章和附录 A 至附录 D，周志敏老师编写第 6 章到第 8 章及附录 E 和附录 F，肖继文老师编写第 9 章。

南昌工程学院的孙辉教授对本书的大纲提出了很多宝贵的意见和建议，在此，我们表示衷心的感谢。

尽管我们做了大量的工作，但本书肯定会有很多不足之处，敬请广大师生批评指正。可发电子邮件与我们联系，邮件地址如下：

E-mail：kuizhong@sina.com

作 者

2004 年 8 月

目　录

第1章 基本知识

1.1 计算机程序与程序设计

1.1.1 计算机程序控制原理

1. 计算机的诞生

电子数字计算机（electronic digital computer），简称计算机（computer），最初是为了计算炮弹弹道而设计的一种计算工具。在第二次世界大战期间，美国为了研制和开发新型的大炮和导弹，在马里兰州的阿伯丁设立了“弹道研究实验室”。军方要求该实验室每天为陆军炮弹部队提供6张弹道表（ballistic firing tables），以便对导弹的研制进行技术鉴定。别小看这区区的6张弹道表，它们所需的计算工作量大得惊人。在当时，要完成一张弹道表的计算就需要200多名计算员加班加点地工作两个多月！1942年8月，当时任职于宾夕法尼亚大学莫尔电机工程学院的莫奇利（John Mauchly）在他的文章“The Use of High Speed Vacuum Tube Devices for Calculating（高速电子管计算装置的使用）”中提出了试制第一台电子计算机的初始设想，期望用电子管代替继电器以提高机器的计算速度。美国军方得知这一设想之后，马上拨款大力支持，成立了一个以莫奇利和埃克特（J. Presper Eckert）为首的研制小组开始了研制工作。

经过三年紧张的工作，这台计算机终于在1946年2月14日问世了，如图1-1所示。它

图1-1 第一台电子计算机ENIAC外观（1946）

由 17468 个电子管、60000 个电阻器、10000 个电容器和 6000 个开关组成，重达 30t，占地 160m^2，耗电 174kW，耗资 45 万美元。这台计算机每秒只能运行 5000 次加法运算。

这台计算机是一台电子数字积分计算机，英文全称为 The Electronic Numerical Integrator And Computer，其缩写为 ENIAC，中文称为“埃尼阿克”。

2. 冯·诺伊曼原理

从 ENIAC 诞生至今已过去了 58 年，在这期间，尤其是近 20 年来，计算机以惊人的速度发展着，其运算速度越来越高、处理能力越来越强。目前，美国 IBM 公司正在研制的一台名为“Blue Gene/L”（蓝色基因/L）的超级计算机，其运算速度是每秒 360 万亿次浮点运算。它比 2002 年 4 月日本 NEC 公司研制的“地球模拟器”快 10 倍，比 ENIAC 快几百万亿倍。“蓝色基因/L”超级计算机每秒 360 万亿次浮点运算的性能预计将超过目前世界 500 强超级计算机计算能力的总和。“每秒 360 万亿次浮点运算”有多快？如果一个人一秒钟能完成一次浮点运算，那么，他做 360 万亿次浮点运算就需要昼夜分秒不停地运算 42 亿年！

从 ENIAC 到即将出现的 Blue Gene/L，尽管它们的运算速度、运算能力和结构都有很大的差别，但从原理上讲，它们都是一样的，都是基于“存储程序控制”原理的。

“存储程序控制”原理即“存储程序和程序控制”原理，是 1946 年由美籍匈牙利数学家冯·诺伊曼（John von Neumann）首先提出的，故又称为“冯·诺伊曼原理”。这一原理在计算机的发展过程中，始终发挥着重要作用，它确立了现代计算机的基本组成和工作方式，直到现在，各类计算机的工作原理还是采用冯·诺伊曼原理的思想。冯·诺伊曼原理的核心是“存储程序控制”，其结构框图如图 1-2 所示。

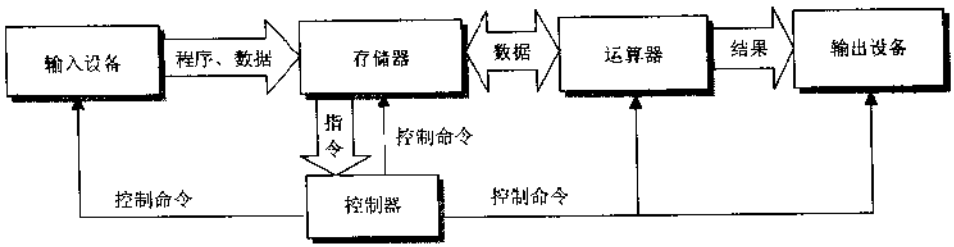

图 1-2 冯·诺伊曼计算机结构框图

“存储程序控制”原理的基本内容是：

（1）采用二进制形式表示数据和指令；

（2）将程序（数据和指令序列）预先存放在主存储器中，使计算机在工作时能够自动高速地从存储器中取出指令，并加以执行；

（3）由运算器、存储器、控制器、输入设备、输出设备五大基本部件组成计算机系统，并规定了这五大部件的基本功能。

冯·诺伊曼思想实际上是电子计算机设计的基本思想，奠定了现代电子计算机的基本结构，开创了程序设计的时代。

试读结束，需要全本PDF请购买 www.ertongbook.com

1.1.2 计算机程序

1. 计算机程序

根据冯·诺伊曼原理，计算机是在程序的控制之下——无须人的干预，自动地完成人们交给它的任务的。那么什么是计算机程序呢？

计算机程序（computer program）是指描述解决特定问题的方法和步骤的计算机指令序列。换言之，用计算机能识别的符号（计算机程序设计语言）把特定问题的解决方法和步骤描述出来的就是一个计算机程序。下面三例是用三种不同的计算机程序设计语言描述的解决“求任意圆的面积”的程序。

【例 1-1】用 C 语言描述的程序

```
main( )                                          * 程序开始 */
{
   float r,s;                                   /* 定义变量 */
   printf("Please enter radius of a circle: ");/* 提示输入半径 */
   scanf("%f", &r);                             /* 从键盘输入半径 r */
   s=3.14*r*r;                                  /* 计算圆的面积 s */
   printf("The area of the circle is: %f", s); /* 显示求出的面积 s */
}
```

【例 1-2】用 QBASIC 语言描述的程序

```
INPUT "Please enter radius of a circle: ", r    ' 从键盘输入半径 r
s=3.14*r^2                                      ' 计算圆的面积 s
PRINT "The area of the circle is: " ; s         ' 显示求出的面积 s
```

【例 1-3】用 PASCAL 语言描述的程序

```
program crclarea                                { 程序名 }
var r, s : Extended;                            { 定义变量 }
begin                                           { 程序开始 }
  Write('Please enter radius of a circle: ');   { 提示输入半径 }
  Readln(r);                                    { 从键盘输入半径 r }
  s := 3.14*r*r ;                               { 计算圆的面积 s }
  Writeln('The area of the circle is: ',s);     { 显示求出的面积 s }
end.                                            { 定义变量 }
```

以上的例子是同一个问题的同一种解决方法的三种描述——三个程序。一般情况下，同一个问题通常有多种不同的解决方法，有的简单，有的复杂，而同一种解决方法又可以用多种不同的语言进行描述，因此，解决任何一个问题的计算机程序都不是惟一的。无论是简单的还是复杂的，也无论是用什么语言来描述，一个计算机程序主要描述两部分内容。一是描述问题的每个对象和对象之间的关系——数据和数据结构，例如，以上程序中的半径 r、面积 s 和圆周率 3.14 及其之间的关系；二是描述对这些对象作处理的处理方法和处理步骤——求解的算法，例如，以上程序求解算法是：先从键盘输入半径的值，然后按面积与半径的关系公式计算

出面积，最后显示该圆的面积。

换言之，数据结构和算法是程序最主要的两个方面，数据结构描述处理的对象及其相互之间的关系，算法则描述对需要处理的对象“做什么处理”和“如何处理”。

著名的计算机科学家沃思（Nikiklaus Wirth）用一个简单的公式描述了程序、数据结构和算法之间的关系：

程序 = 数据结构（data structure） + 算法（algorithm）

2. 程序的一般结构

程序是数据和对数据进行加工处理的描述，因此，一个程序从结构上通常可划分为以下四个组成部分（参阅例 1-1～例 1-3）。

（1）定义数据部分：定义程序中要处理的数据的数据结构，包括数据的类型和数据之间的相互关系等；

（2）获取数据部分：从输入设备或文件中获取要处理的数据；

（3）处理数据部分：对获取数据根据问题要求，按一定规则进行加工处理；

（4）输出结果部分：将经过处理后所获得的结果从输出设备输出出来或写入文件中去。

1.1.3 程序设计与程序设计语言

1. 程序设计

程序设计（programming）就是对要解决的问题进行分析，用合理的数据结构描述问题所涉及的对象，找出解决问题的算法，并将解决问题的算法用计算机语言记录下来——称为编码（coding）。也就是说，程序设计主要包括三个方面的任务，一是数据结构的设计，二是算法设计，三是根据算法进行编码。数据结构的设计和算法设计是程序设计中最重要的两个方面，也是初学者最难以掌握的两个方面。由于算法和数据结构不是相互独立的，具体的算法通常依赖于特定的数据结构，设计合理的数据结构常可有效地简化算法，因此，数据结构设计比算法设计显得更为重要。当然，没有正确的算法和具体的编码就没有任何程序。

在上一小节里讲到，算法是描述对需要处理的对象“做什么处理”和“如何处理”。因此，算法设计，简单地说，就是要回答“做什么”和“怎么做”两个问题。“做什么”是指对问题的认识、分析和判断，“怎么做”则是要找出具体的、正确的和有效的解决问题的方法和步骤。当然，方法和步骤的正确性与有效性是相对的。可以说，世界上没有一个人能一次性设计出绝对正确的和高效的程序来。

因此，程序设计的主要步骤如下：

1）认识问题，并找出要处理的数据及其相互关系；

2）设计合理的数据结构；

3）设计解决问题的算法；

4）按算法编写程序；

5）调试（debug）和测试（test）程序——给出若干组数据，找出程序的错误。

在程序的实际开发和设计过程中，上述步骤经常是要反复进行的。如果发现程序有严重

错误，可能会要重新认识问题和重新设计算法等。

2. 程序设计语言

语言本来只属于人类，是人类用以表达情意、交流思想的一种重要交际工具。人类要想借助于计算机来解决问题，根据冯·诺伊曼原理，人类就必须能“告诉”计算机“做什么”、“怎么做”。因此，在人类与计算机之间必须有能够进行互相交流的“语言”，这就是计算机程序设计语言（programming language），它是为了程序设计而设计出来的，用于人与计算机“交流”的语言。

从 ENIAC 诞生至今，用于程序设计的语言有上千种之多，常用的也有几十种，如汇编语言、BASIC 语言、C 语言、PASCAL 语言、SQL 语言、COBOL 语言、Ada 语言、FORTRAN 语言、LISP 语言、PROLOG 语言等，五花八门。尽管如此，但按大类分也只分为两类，即低级语言和高级语言。

（1）低级语言。

低级语言包括机器语言和汇编语言。机器语言和汇编语言都与具体的 CPU 有关，不同的 CPU 有不同的机器语言和汇编语言，而且一点都不能通用，即便是同一个公司的不同产品也是如此。譬如，Intel 公司的 80x86 CPU 与 8031 系列单片机，它们的机器语言和汇编语言是完全不同的。因此称它们为面向机器的语言。

机器语言（Machine Language）：计算机能直接识别和执行的二进制代码，是最早的程序设计语言。以下是用 Intel 80x86 CPU 的机器语言写出的“求一个数的绝对值”的程序：

```
00111101 00000000 00000000
01111101 00000010
11110111 11011000
```

求解这么简单问题的三行程序即使是受过专门训练的人也难以读懂，可以想象，如果一个求解复杂问题的有成千上万行语句的程序出了错，要想找出错误并改正，会有多么困难！在机器语言中，无论是数据还是指令都是用二进制代码来书写的，可读性极差，正因为如此才出现了其他语言。

汇编语言（assemble language）：用有助于记忆的符号来代表二进制代码的语言。具体地说就是，汇编语言是一种用与指令功能有关的英文缩写来表示指令的操作码，用可以识别的符号来表示数据和数据所在内存的地址的符号语言。以下是用 Intel 80x86 CPU 的汇编语言编写的，与上面机器语言程序等价的程序。

汇编指令	英文含义	中文含义
CMP AX, 0 ;	CoMPare AX with 0	AX 与 0 比较
JGE OK ;	Jump to OK if AX Great than or Equal to 0	若 AX≥0 则转 OK
NEG AX ;	else NEGative AX	否则 AX← -AX
OK:		

虽然用汇编语言编写的程序其可读性强多了，但由于汇编语言的指令与机器语言指令几乎有一对一的关系，因此仍然是面向机器的，属于低级语言。

用汇编语言编写的程序称为“汇编源程序”（assemble language source code），汇编源程序比机器语言程序可读性好。但是对机器而言是不可读的，即不能在计算机上直接执行，需要用汇编程序将汇编源程序翻译成机器语言程序后才能执行。

由于低级语言是面向机器的，因此低级语言程序是不可移植的——不能将在一种机器（指CPU）上编写的程序在另一种机器上运行。这显然是一种人力和物力的浪费。为了解决移植问题，20 世纪 50 年代末出现了与机器无关的程序设计语言，即高级语言。

（2）高级语言。

与机器无关的、类似于自然语言和数学语言的计算机程序设计语言就是高级语言。

几乎所有的高级语言所使用的词汇都是英语中最常见的词汇或其缩写，数据及其运算的表达方式与在数学中的表达方式几乎也是一样的，因此，可读性好。以下三句分别是用 C 语言、BASIC 语言和 PASCAL 语言描述的求绝对值的语句。

```
if (x<0) x = - x ;
if x<0 then x= - x
if x<0 then x := - x ;
```

高级语言目前也分为两大类：一类是以描述计算机的解题过程为主的语言；另一类则是以描述操作对象的特性和行为特征为主的语言。前者称为面向过程的语言（POP——Process-Oriented Programming language），如 BASIC、C、PASCAL 等。后者为面向对象的语言（OOP——Object-Oriented Programming language），如 VB、C++、Object PASCAL 等。

在高级语言中，也有人划分出一种称为“面向问题的语言”（这个划分实际上可划归于OOP）。面向问题语言是为了易于描述和求解某类特定领域的问题而专门设计的一种非过程语言。用面向问题语言解题时，不仅摆脱计算机的内部逻辑，也不必关心问题的求解算法和求解的过程，只需指出问题是做什么，数据的输入和输出形式，就能由相应的计算机系统得到所需结果。如报表语言、SQL（Structured Query Language）语言等。SQL 语言是数据库查询和操纵语言，能直接使用数据库管理系统。由于使用面向问题语言来解题只要告诉计算机做什么，不必告诉计算机如何做，能方便用户的使用和提高程序的开发速度。

3. 源程序的编译或解译

用某种高级语言编写的程序称为该语言的“源程序”（source code），如用 BASIC 语言编写的程序称为 BASIC 源程序，用 C++语言编写的程序称为 C++源程序。

与汇编语言源程序一样，无论是用 POP 源程序还是 OOP 源程序，计算机都不能直接执行源程序的语句，都必须翻译成机器语言程序才能执行。高级语言的翻译方式通常有两种：解释方式和编译方式。

解释方式（explain），即让计算机运行解释程序，解释程序逐句取出源程序中的语句，一边翻译成机器指令一边执行。BASIC、Java 等属于解释方式的语言。解释方式的主要优点是计算机与人的交互性好，调试程序时，能一边执行一边直接改错，能较快得到一个正确的程序。缺点是逐句解释执行，运行速度慢，而且源程序不能脱离解释程序单独运行。

编译方式（compile），即先运行编译程序，由编译程序将源程序一次性翻译成计算机能

识别的机器语言程序——目标程序（object program），再运行连接程序将多个目标程序和库程序（library）连接装配成一个可直接执行的机器语言程序——可执行程序（executable program），最后让计算机执行可执行程序。C、C++、PASCAL 等大多数高级语言属于编译方式的语言。编译方式的主要优点是计算机运行可执行程序的速度比较快，而且可执行程序不再需要编译程序和连接程序就能独立运行。缺点是修改源程序后必须重新编译和连接以产生新的可执行程序。

交给用户的程序一般都是可执行程序。

除了上述这两种纯解释和纯编译的方式外，目前也有将这两种方式结合起来的方式，即先编译源程序，产生计算机还是不能直接执行的中间代码，然后让解释程序解释执行中间代码。FoxPro 就属于这种方式。这样做的好处首先是比直接解释执行快；更大的好处是中间代码独立于计算机，只要有相应的解释程序，就可在任何计算机上运行。

1.1.4 程序设计语言的基本元素

不同的程序设计语言虽有千差万别，但它们都是由一些最基本的元素构成的，掌握了这些基本元素的相关规则和用法就等于掌握了这种程序设计语言的精髓。换言之，学习并掌握一种程序设计语言，首先要弄清楚它的基本元素。

所谓基本元素是指编译器不能再将其分解成若干个组成成分的符号。例如 C 语言程序：

```
main( )
{
   int  x1 = 10, y1 = 20 ;
   printf("%d" , x1 + y1) ;
}
```

包含 16 个基本元素："main"、"( )" 、"{ }"、"int"、"x1"、"="、"10"、","、"y1"、"20"、";"、"printf"、""%d""、"+"、空格符和回车换行符。本小节只简单介绍程序设计语言的基本元素，其他基本元素将在后续章节详细讲解。

一般说来，程序设计语言的基本元素包括以下几个方面：

（1）关键词（keywords），又称保留字，是指那些对编译器而言具有特定意义的单词，通常为数据类型定义符和语句定义符。如：上例中的"int"。ANSI C 的关键词请参阅附录 C。

（2）标识符（identifiers），或称符号（symbols），是指程序员在程序中给变量、自定义数据类型、函数和标号取的名称，即用于标识不同的变量、类型、函数和标号的符号。如：上例中的"main"、"x1"、"y1" 和 "printf"。标识符在拼写和大小写上必须不同于关键词，也就是说，不能使用关键词作为标识符，因为关键词对编译器而言具有特定的含义。标识符的命名规则：以字母或下划线开头，后跟 0 个或多个字母、数字或下划线。例如：i，j，age，x0，SUM，_case，I_Like_C_Languagc，A0B1C2D3 等都是合法的标识符。除了 C 和 C 类语言之外，绝大多数程序设计语言不区分大小写。例如：ab、Ab、aB 和 AB，在 C 语言中，它们是四个不同的标识符，而在其他语言中，它们没有区别，是相同的一个标识符。

（3）常量（constants），是指在程序运行的全过程中不能改变的数、字符。如：上例中的“10”和“20”。

（4）字符串文字（Strings/Text），是用一对双引号（""）引起来的字符序列。如上例中的“"%d"”。

（5）运算符（operators）指定数据如何操作的符号。如：上例中的“=”和“+”。ANSI C的运算符请参阅附录 D。

（6）标点和特殊字符，起分隔或定界作用的符号。如：“[ ]”、“( )”、“{ }”、“,”、“;”、“#”、“/* */”、及空白符（空格符、制表符、回车符和换行符等）。有些标点符号也是运算符（参阅附录 D）。

1.1.5 C 语言程序结构

一个 C 语言源程序通常由三个部分组成：预处理部分、全局声明部分和函数定义部分。

【例 1-4】求圆的面积。

```
#include <stdio.h>
#delfine PI 3.1415926                          /* 定义圆周率符号 PI */
float area_of_circle(float r);                 /* 声明求圆面积的函数原型*/
/****** 定义主函数 *****************/
main( )
{
   float r,s;                                  /* 定义变量 */
   printf("Please enter radius of a circle: ");/* 提示输入半径 */
   scanf("%f", &r);                            /* 从键盘输入半径 r */
   s= area_of_circle(r);                       /* 计算圆的面积 s */
   printf("The area of the circle is: %f", s); /* 显示求出的面积 s */
}
/****** 定义求圆面积的函数 *******/
float area_of_circle(float r)
{
   return (PI*r*r);                            /* 返回半径为 r 的圆的面积 */
}
```

1. 预处理部分

预处理（preprocess）是指在对源程序进行编译的第一遍扫描（词法扫描和语法分析）之前所作的工作。预处理是C语言的一个重要功能，它由预处理程序负责完成。当对一个源文件进行编译时，系统将自动引用预处理程序对源程序中的预处理部分作处理，处理完后自动进入对源程序的编译。

C语言提供了多种预处理功能，如文件包含、宏定义、条件编译等。例 1-4 中的第一行是文件包含命令，第二行是宏定义命令。合理地使用预处理命令，将使程序更便于阅读、修改、移植和调试，也有利于模块化程序设计。

预处理及预处理命令详情请参阅附录 A。

试读结束，需要全本PDF请购买 www.ertongbook.com