



工业和信息化普通高等教育“十三五”规划教材立项项目

21世纪高等学校计算机规划教材

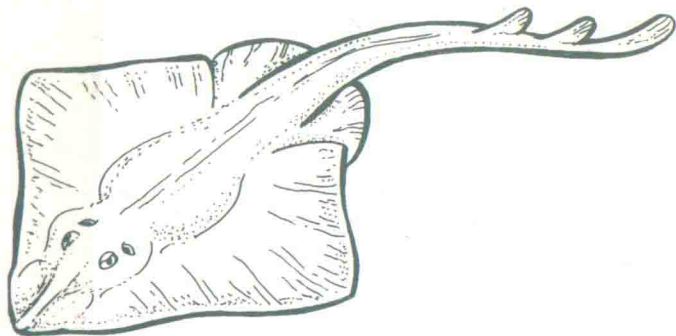
21st Century University Planned Textbooks of Computer Science

C语言 程序设计

The C Programming Language

陈维 曹惠雅 鲁丽 杨有安 编

- 以培养计算思维能力为指导
- 以知识讲授能力培养为原则
- 以培养自主开发能力为目标



高校系列

 中国工信出版集团

 人民邮电出版社
POSTS & TELECOM PRESS



工业和信息化普通高等教育“十三五”规划教材立项项目

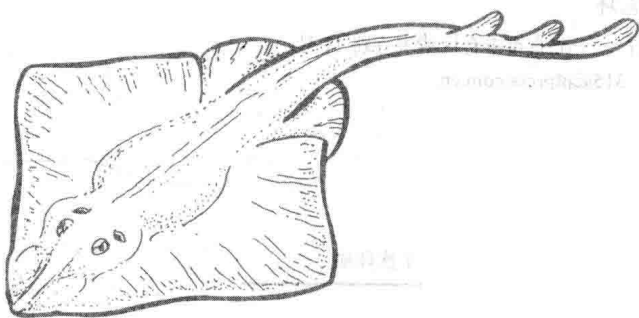
21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

C语言 程序设计

The C Programming Language

陈维 曹惠雅 鲁丽 杨有安 编



高校系列

人民邮电出版社

北京

图书在版编目 (C I P) 数据

C语言程序设计 / 陈维等编. — 北京 : 人民邮电出版社, 2018.1
21世纪高等学校计算机规划教材
ISBN 978-7-115-35863-9

I. ①C… II. ①陈… III. ①C语言—程序设计—高等学校—教材 IV. ①TP312.8

中国版本图书馆CIP数据核字(2017)第300356号

内 容 提 要

本书根据全国高等学校计算机基础教育研究会发布的“中国高等院校计算机基础教育课程体系2014”中有关“程序设计基础(C语言)”课程的教学要求及人才培养的新要求编写而成。全书共11章,主要内容包括C语言概述、基本数据类型和运算符、顺序和选择结构程序设计、循环结构程序设计、数组、函数和模块设计、指针、结构体与联合体、预处理和标准函数、文件、数据结构和数据抽象。

本书内容全面,由浅入深,详略得当,注重实践,实例丰富,面向应用。各章附有适量的习题,便于自学。另外,针对书中各章内容和上机实验,本书还配有辅导教材《C语言程序设计实训教程》,引导读者学习和掌握各章节的知识。

本书为高等学校非计算机专业“C语言程序设计”课程的教材,也可以作为初次学习C语言程序设计的读者、准备计算机二级考试的考生和计算机工程技术人员的参考书。

-
- ◆ 编 陈 维 曹惠雅 鲁 丽 杨有安
责任编辑 刘向荣
责任印制 沈 蓉 彭志环
 - ◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路11号
邮编 100164 电子邮件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京市艺辉印刷有限公司印刷
 - ◆ 开本: 787×1092 1/16
印张: 18.5 2018年1月第1版
字数: 477千字 2018年1月北京第1次印刷
-

定价: 54.00 元

读者服务热线: (010)81055256 印装质量热线: (010)81055316

反盗版热线: (010)81055315

前 言

计算机基础教育是培养大学生综合素质和创新能力不可或缺的重要环节,是培养复合型人才的重要组成部分。“C 语言程序设计”是高等院校计算机基础教育中的重要课程之一。该课程可以让学生了解程序设计的思想和方法,掌握高级语言程序设计的知识,提高问题求解和程序语言的应用能力。因此,本书旨在实现“以人为本、传授知识、培养能力、提高素质、协调发展”的教育理念,使学生的计算机知识、技能、能力和素质得以协调发展。

本书针对高等院校学生的特点和认知规律,全面、系统地介绍 C 语言程序设计及应用知识,包括 C 语言概述、基本数据类型和运算符、顺序和选择结构程序设计、循环结构程序设计、数组、函数和模块设计、指针、结构体与联合体、预处理和标准函数、文件、数据结构和数据抽象等内容。编者结合多年从事程序设计教学和研究的经验,参考了大量同类教材,并吸收其优点,在编写过程中以夯实学生程序设计的理论基础、培养学生程序设计的能力和养成良好的程序设计风格为宗旨,充分体现“教师易用,学生易学”的特点。

全书分为 11 章,其中第 1 章、第 2 章和第 4 章由曹惠雅编写,第 3 章、第 8 章和第 10 章由鲁丽编写,第 7 章、第 9 章、第 11 章由陈维编写,第 5 章、第 6 章由杨有安编写。陈维负责全书的统稿工作。本书在编写的过程中得到文华学院各级领导的大力支持,在此表示衷心的感谢。

本书同时出版了一本配套辅导教材《C 语言程序设计实训教程》,两者互为补充,相辅相成。辅导教材按本书的章节顺序,对各章重点及难点进行总结,对重、难点题型进行分析,并附有大量练习,对读者掌握程序设计的基本知识、提高程序设计的应用能力十分有益。

由于编者水平有限,加之时间仓促,书中难免有不足之处,敬请读者批评指正。

编 者

2017 年 12 月

目 录

第 1 章 C 语言概述 1

1.1 程序与程序设计语言.....1	
1.1.1 程序与指令.....1	
1.1.2 程序设计语言的功能.....2	
1.1.3 程序设计语言的语法.....2	
1.2 C 语言的发展和优缺点.....5	
1.2.1 C 语言的发展.....6	
1.2.2 C 语言的优缺点.....6	
1.3 C 程序的结构.....7	
1.3.1 简单 C 程序举例.....8	
1.3.2 C 语言程序的结构特点.....9	
1.3.3 书写程序时应遵循的规则.....10	
1.4 Visual C++ 6.0 上机简介.....10	
1.4.1 Visual C++ 6.0 集成开发环境简介.....10	
1.4.2 Visual C++ 6.0 集成环境上机步骤.....12	
小结.....16	
习题.....16	

第 2 章 基本数据类型和运算符 18

2.1 基本数据类型.....18	
2.1.1 常量和变量的概念.....18	
2.1.2 整型.....20	
2.1.3 实型.....22	
2.1.4 字符型.....23	
2.1.5 字符串.....27	
2.2 数据的存储.....28	
2.2.1 整型数据的存储.....28	
2.2.2 实型数据的存储.....28	
2.2.3 字符型数据的存储.....29	
2.3 运算符和表达式.....29	
2.3.1 算术运算符与算术表达式.....30	
2.3.2 赋值运算符与赋值表达式.....32	
2.3.3 位运算符及其表达式.....35	
2.3.4 增减量运算符与增减量表达式.....37	
2.3.5 关系运算符与关系表达式.....38	

2.3.6 逻辑运算符与逻辑表达式.....40	
2.3.7 条件运算符与条件表达式.....42	
2.3.8 逗号运算符与逗号表达式.....43	
2.3.9 其他运算符.....44	
2.3.10 运算符的优先级与结合性.....44	
2.4 数据类型的转换.....45	
2.4.1 自动转换.....45	
2.4.2 赋值转换.....46	
2.4.3 强制类型转换.....47	
小结.....48	
习题.....48	

第 3 章 顺序和选择结构程序设计 52

3.1 程序设计概述.....52	
3.1.1 程序设计基本步骤.....52	
3.1.2 C 语言编写风格.....53	
3.1.3 程序的语句.....54	
3.2 数据的输入/输出.....55	
3.2.1 数据的输入.....55	
3.2.2 scanf()函数的调用.....55	
3.2.3 scanf()函数使用中常见的问题.....57	
3.2.4 getchar()函数.....58	
3.2.5 putchar()函数.....59	
3.3 程序的 3 种基本结构.....60	
3.4 if 语句.....61	
3.4.1 if 语句的 3 种形式.....61	
3.4.2 if 语句的嵌套.....66	
3.5 switch 语句.....68	
3.6 程序设计举例.....71	
小结.....75	
习题.....75	

第 4 章 循环结构程序设计 80

4.1 for 循环.....80	
4.2 while 循环.....84	
4.3 do-while 循环.....87	

4.4 三种循环语句的比较	89	6.2.3 函数的返回值	139
4.5 跳转语句	89	6.2.4 函数参数及函数间的数据传递	142
4.5.1 break 语句	89	6.3 函数的嵌套调用和递归调用	147
4.5.2 continue 语句	90	6.3.1 函数的嵌套调用	147
4.5.3 goto 语句	92	6.3.2 函数的递归调用	150
4.6 循环语句的嵌套	93	6.4 作用域和存储类型	151
4.7 程序设计举例	96	6.5 内部函数和外部函数	158
小结	101	6.5.1 内部函数	159
习题	101	6.5.2 外部函数	159
第 5 章 数组	104	6.6 模块化程序设计	160
5.1 一维数组	104	6.6.1 模块化程序设计方法的指导思想	160
5.1.1 一维数组的定义	104	6.6.2 模块分解的原则	161
5.1.2 一维数组元素的引用	105	6.7 程序设计举例	161
5.1.3 一维数组元素的初始化	107	小结	166
5.2 二维数组	108	习题	167
5.2.1 二维数组的定义	108	第 7 章 指针	169
5.2.2 二维数组元素的引用	109	7.1 指针的概念	169
5.2.3 二维数组元素的初始化	110	7.1.1 地址与指针	169
5.3 字符数组	112	7.1.2 指针变量的定义和引用	170
5.3.1 字符数组的定义	112	7.1.3 指针变量的运算	173
5.3.2 字符数组的引用	113	7.2 指针变量作为函数参数	175
5.3.3 字符数组的初始化	114	7.3 指针与一维数组	178
5.3.4 字符串及其结束标志	115	7.3.1 一维数组的指针表示	178
5.3.5 字符数组的输入/输出	116	7.3.2 数组作为函数参数时的指针表示	180
5.3.6 常用的字符串处理函数	118	7.3.3 字符串的指针表示	182
5.4 使用数组的程序设计方法	121	7.4 指针与多维数组	185
5.4.1 排序	121	7.4.1 多维数组的处理	185
5.4.2 查找	124	7.4.2 指向多维数组的指针	186
5.5 程序设计举例	125	7.5 指针数组和多级指针	188
小结	128	7.5.1 指针数组的概念	188
习题	129	7.5.2 指针数组的应用	189
第 6 章 函数和模块设计	133	7.5.3 多级指针(指向指针的指针)	192
6.1 结构化程序设计	133	7.6 指针与函数	193
6.1.1 结构化程序设计的基本概念	133	7.6.1 指向函数的指针	193
6.1.2 结构化程序设计的基本特征	134	7.6.2 函数指针的应用	195
6.2 函数的定义和调用	135	7.6.3 返回指针的函数	198
6.2.1 函数的定义	135	7.7 命令行参数	201
6.2.2 函数的调用	136	小结	202
		习题	204

第 8 章 结构体与联合体 205

8.1 结构体	205
8.1.1 结构体类型的定义	206
8.1.2 结构体类型变量的定义与使用	206
8.1.3 结构体类型变量的赋值与初始化	208
8.1.4 结构体类型数组的定义与引用	210
8.1.5 结构体类型指针的定义与引用	213
8.1.6 结构体类型数据的动态存储分配	218
8.1.7 链表及其基本操作	219
8.2 联合体	224
8.3 其他自定义数据类型	226
8.3.1 枚举类型	226
8.3.2 类型定义符 typedef	228
小结	229
习题	230

第 9 章 预处理和标准函数 232

9.1 预处理命令	232
9.1.1 宏定义	232
9.1.2 文件包含	235
9.1.3 条件编译	235
9.2 输入/输出标准函数	237
9.2.1 格式输出函数	237
9.2.2 格式输入函数	239
9.3 自定义头文件设计的原则	241
小结	245
习题	245

第 10 章 文件 246

10.1 问题的引入	246
10.2 文件的基本概念	246
10.3 内存与外存的数据交流	247
10.4 程序针对文件的基本操作	248

10.4.1 打开文件	248
10.4.2 关闭文件	250
10.4.3 文件的读写	250
10.4.4 文件检测函数	259
10.5 程序调试与数据测试文件	259
10.6 程序设计举例	261
小结	264
习题	265

第 11 章 数据结构和数据抽象 266

11.1 数据抽象	266
11.1.1 数据结构和数据类型	266
11.1.2 抽象数据类型	267
11.2 线性表	267
11.2.1 线性表的定义	267
11.2.2 线性表的基本操作	268
11.2.3 线性表的顺序存储	268
11.2.4 线性表上基本运算的实现	270
11.3 堆栈	271
11.3.1 抽象栈的定义及基本操作	271
11.3.2 抽象栈的定义	272
11.3.3 顺序栈的基本运算的实现	272
11.4 队列	273
11.4.1 队列的定义	273
11.4.2 队列的存储结构及其相关算法	274
小结	276
习题	277

附录 A ASCII 代码对照表 278**附录 B C 库函数 279****附录 C Debugger 调试器使用简介 284**

第 1 章

C 语言概述

C 语言 (Combined Language) 是一门通用计算机编程语言, 应用广泛。C 语言的设计目标是提供一种能以简易的方式编译、处理低级存储器、产生少量的机器码以及不需要任何运行环境支持便能运行的编程语言。尽管 C 语言提供了许多低级处理的功能, 但仍然保持着良好跨平台的特性, 以一个标准规格写出的 C 语言程序可在许多计算机平台上进行编译, 甚至包含一些嵌入式处理器 (单片机或称 MCU) 以及超级计算机等作业平台。

对于将 C 语言作为第一门编程语言 (Programming Language) 的读者来说, 最关心的问题无疑是如何能尽快学会用 C 语言进行程序设计。要做到这一点, 首先需要对程序设计语言有所了解, 其次还要通过不断的编程实践, 去领会和掌握程序设计的基本思想和方法。在此, 建议读者先从看懂书中的程序做起, 而后再模仿书中的程序试着改写或编写程序, 循序渐进, 直到能独立地编写程序并解决一些较复杂的问题。

为了使读者能逐步地从简单的模仿中体会到程序设计的基本思想和方法, 本章将简要介绍程序设计语言的功能、语法要素, C 语言的特点、词汇以及 C 语言程序的结构和运行环境等知识。

1.1 程序与程序设计语言

计算机程序 (Program) 是人们为解决某种问题用计算机可以识别的代码编排的一系列加工步骤。计算机能够严格按照这些步骤去执行, 包括计算机对数据的处理。程序的执行过程实际上是对程序所表示的数据进行处理的过程。一方面, 程序设计语言提供了一种表示数据与处理数据的功能; 另一方面, 编程人员必须按照语言所要求的规范 (即语法规则) 进行编程。

1.1.1 程序与指令

程序最根本的功能是对数据的处理, 计算机最基本的处理数据的单元就是计算机指令。单独的一条指令本身只能完成计算机的一个最基本的功能, 那么, 计算机能实现的指令的集合则成为计算机的指令系统, 而一系列计算机指令的有序组合就构成了程序。

程序在计算机中是以 0、1 组成的指令代码来表示的, 即程序实际上是 0、1 的有序组合, 这个序列能够被计算机直接识别。程序和数据一样, 共同存放在存储器中。当程序要运行时, 当前准备运行的指令从内存被调入 CPU 中, 由 CPU 处理这条指令。

如果程序设计者直接用 0、1 的代码来编写计算机指令, 那将无疑是一件非常令人难以忍受的事情, 所以, 程序设计语言就应时而生。用程序设计语言来描述程序, 同时应用一种软件 (如编译系统) 将其描述的程序转换成计算机能够直接识别的指令序列。

总的来说, 计算机程序是人们为了解决某种问题, 用计算机可以识别的代码编排的一系列数据处理步骤, 计算机将严格按照这些步骤去做。

1.1.2 程序设计语言的功能

程序设计语言, 通常简称为编程语言, 它是一种被标准化的交流技巧, 是一组用来定义计算机程序的语法规则, 用来向计算机发出指令, 它可以让程序员准确地定义计算机所需要使用的数据, 并精确地定义在不同情况下所应当采取的行动。

程序设计语言必须具有数据表达和数据处理 (也称为流程控制) 的能力。

1. 数据表达

数据种类多种多样, 而语言本身的描述能力总是有限的。为了使程序设计语言能充分、有效地表达各种各样的数据, 一般将数据抽象为若干种类型。数据类型 (Data Type) 就是对某些具有共同特点的数据集合的总称。比如大家常说的整数、实数等就是数据类型的例子。数据类型涉及两方面的内容: 该数据类型代表的的数据是什么 (即数据类型的定义域)? 能在这些数据上做什么 (即操作, 或称运算)? 比如, 整数类型所包含的数据有 2, -2, 0, 100, ..., 而 +, -, *, / 等就是作用在整数上的运算。

在程序设计语言中, 一般都需事先定义几种基本的数据类型, 以供程序员直接使用, 如整型、实型、字符型等。同时, 为了使程序员能更充分地表达各种复杂的数据, 程序设计语言还提供了构造新的数据类型的手段, 如数组 (Array)、结构 (Structure)、指针 (Pointer) 等。

程序设计语言提供的基本数据类型以及构造复杂类型的手段, 为有限能力的程序设计语言表达客观世界中的多种多样的数据提供了良好的基础。

2. 流程控制

程序设计语言除了能表达各种各样的数据外, 还必须提供一种手段来表达数据处理的过程, 即程序的控制过程。程序的控制过程通过程序中的一系列语句来实现。

当要解决的问题比较复杂时, 程序的控制过程同样会变得十分复杂。一种常用的程序设计方法是: 将复杂程序划分成若干个相对独立的模块 (Module), 使完成每个模块的工作变得单纯而明确, 在设计一个模块时不受其他模块的制约。同时, 通过现有模块积木式的扩展就可以形成复杂的、更大的程序模块或程序。这种程序设计方法就是结构化的程序设计方法 (Structured Programming)。C 语言就是支持这种设计方法的典型语言。

在结构化程序设计方法中, 一个模块可以是一条语句 (Statement)、一段程序或一个函数等。一般来说, 从程序流程的角度看, 模块只有一个入口和一个出口。这种单入单出的结构为程序的调试 (Debug, 又称查错) 提供了良好的条件。

按照结构化程序设计的特点, 任何程序都可以将模块通过三种基本的控制结构进行组合来实现。这三种基本的控制结构分别是: 顺序结构、分支结构和循环结构。

顺序结构 (Sequential Structure): 按照程序的书写先后顺序, 执行完一个程序模块后, 再顺序执行下一个模块。

分支结构 (Branch Structure): 又称选择结构。在程序执行过程中, 根据不同的条件来选择所要执行的模块, 即判断某种条件, 若条件满足就执行某个模块, 否则就执行另一个模块。

循环结构 (Loop Structure): 是指反复执行某个模块的过程。当然, 重复执行这些模块通常是有条件的, 只有条件满足时才会去重复执行相应的模块。

1.1.3 程序设计语言的语法

一般把用程序设计语言编写的未经编译的程序称为源程序 (Source Code, 又称源代码), 而

源程序的编写必须符合相应语言的语法 (Grammar)。那么,从语法的角度来说,源程序实际上是一个字符序列。这些字符序列按顺序分别组成了一系列的“单词”。这些“单词”是为了按照一定的语法规则构成语言的各种成分而规定的。下面分别介绍 C 语言中的一些常用词汇和语法单位。

1. C 语言的词汇

(1) C 语言字符集。组成 C 语言源程序代码的基本字符称为 C 语言字符集,它是构成 C 语言的基本元素。C 语言允许使用的基本字符有如下几种。

- ① 大小写英文字符: A~Z, a~z;
- ② 数字字符: 0~9;
- ③ 特殊字符: +=_(下划线)()*&^%#!,.;:?'”~\|/<>{}[];
- ④ 不可打印的字符: 空格、换行符、制表符、响铃符。

一般的 C 语言源程序仅仅包含以上字符集中的字符,在具体的 C 语言编译系统中可对上述字符集加以扩充。

(2) 关键字。关键字是具有特定含义的、专门用来说明 C 语言的特定成分的一类单词。例如,关键字 int 用来定义整型变量,而关键字 float 则用来定义浮点型变量。C 语言的关键字都用小写字母书写,不能用大写字母书写。例如,关键字 int 不能写成 Int。由于每个关键字都有特定的含义,所以不能作为用户程序中的变量名和函数名等,否则会产生编译错误。

在 C89 标准中共有 32 个关键字:

```
auto      break   case    char    const   continue default
do        double  else    enum    extern  float    for
goto     if       int     long   register return   short
signed   sizeof  static  struct switch  typedef  union
unsigned void   volatile while
```

在新的 C99 标准中,又增加了 5 个关键字:

```
_Bool      _Complex  _imaginary  inline  restrict
```

(3) 标识符。计算机程序处理的对象是数据,程序用来描述数据处理的过程。在程序中,通过名字建立对象定义与使用的关系。为了满足这种需要,每种程序语言都规定了在程序中名字描述的规则。在 C 语言中用于标识名字的有效字符序列称为标识符,对标识符做了如下规定。

① 标识符的第一个字符必须是英文字母或下划线()。

② 如果第一个字符后面还有字符序列,则它应是英文字母、下划线或数字组成的序列。标识符中的英文字母大小写是有区别的,如标识符 abc 与标识符 ABC 不相同。为了便于读者对标识符有进一步的认识,下面列举若干正确的标识符和不正确的标识符。

正确的标识符:

```
Abc      abc      _Abc      _4a5
```

不正确的标识符:

```
A?      (含有不合法字符“?”)
2abc     (第一个字符不允许为数字)
a b     (标识符中不允许有空格)
yes/no  (含有不合法字符“/”)
πr      (“π”为不合法字符)
```

标识符中有效字符个数(也称长度)视系统不同而不同。例如, Turbo C 规定前 32 个字符有效,超过的部分忽略。比如,对于 8 个字符有效的标识符而言, identifi 与 identifier 被视为同一标识符,因后者中的 er 已被忽略。

以后将会看到,标识符用来为变量、符号常量、数组、函数等取名。使用时,标识符的选择

由程序员自定,但是不能与关键字相同。另外,为了增加程序的可读性,选择标识符时应遵循“见名知义”的原则,即选择描述性的标识符,标识符应尽量与所要命名的对象间有一定的联系,以助于识别和记忆。例如:

```
length    (表示长度)
time      (表示时间)
pi        (表示圆周率π)
```

(4) 保留标识符。保留标识符是系统保留的一部分标识符,通常用于系统定义和标准库函数的名字。例如,以下划线开始的标识符通常用于定义系统变量。虽然它们也是合法的标识符,但用作一般标识符时可能会出现运行错误,因此不能使用这些标识符来定义自己的变量。

(5) 注释。在 C 语言程序中,注释部分的格式是:

```
/*注释内容*/ 或 //注释内容
```

注释不是程序代码,是对程序解释说明的标注,它可以是任何可显示的字符,不影响程序的编译和运行,程序编译时编译程序把注释作为空白符跳过而不予处理。另外,注释不允许嵌套。

例如:/*学生成绩管理程序*/、//My c program

在程序中插入适当的注释,可以使程序容易被人理解。

2. C 语言的主要语法单位

(1) 变量定义。不同的变量有数据类型之分,在声明变量的时候一定要对其类型加以说明。变量类型的不同,说明其在计算机内存中所占的存储空间大小也不同。声明变量的一般格式为:

```
类型说明符 变量名;
例如: int i; /*声明了一个整型变量 i*/
```

(2) 表达式。由运算符及其运算对象可以组成形形色色的表达式。如: $3.14*3*\sin(x)$ 。表达式中的运算符有运算优先级,如:表达式 $2+3*4-4/4$ 中,应先执行运算符*和/,再执行运算符+和-。

(3) 语句。语句是程序最基本的执行单位,程序的功能就是通过执行一系列的语句来实现的。C 语言提供了多种语句,大致可分为五类:表达式语句、函数调用语句、控制语句、复合语句、空语句。

① 表达式语句。表达式语句由表达式末尾加上分号“;”组成。其一般形式为:表达式; 执行表达式语句就是计算表达式的值。例如:

```
m=2;    赋值语句
m+n;    算术表达式语句
m>n;    关系表达式
i++;    增量表达式
```

② 函数调用语句。由函数名、实际参数加上分号“;”组成。其一般形式为:函数名(实际参数表); 执行函数调用语句就是调用函数并把实际参数赋予函数定义中的形式参数,然后执行被调函数体中的语句,求取函数值。

调用库函数,输出字符串。

例如:

```
printf("%d",m); /*调用名为 printf 的标准库函数*/
```

③ 控制语句。控制语句用于控制程序的流程,以实现程序的各种结构方式。它们由特定的语句定义符组成。C 语言有九种控制语句,分成以下三类。

□ 条件判断语句

条件判断语句包括: if 语句、switch 语句。

□ 循环执行语句

循环语句包括: for 语句、while 语句、do while 语句。

□ 跳转语句

跳转语句包括：break 语句、continue 语句、return 语句和 goto 语句。（其中，goto 语句应尽量少用，因为这不利于结构化程序设计，滥用它会使程序流程无规律、可读性差）。

④ 复合语句。把多个相关语句用一对花括号“{}”括起来，组成的一个语句就称为复合语句。在程序中，应把复合语句当作是单条语句，而不是多条语句。例如：

```
{
    temp=x;          /*将 x 的值赋予 temp*/
    x=y;             /*将 y 的值赋予 x*/
    y=temp;         /*将 temp 的值赋予 y*/
}
```

这就是一条复合语句。复合语句内的各条语句都必须以分号“;”结尾，但是在花括号“}”外却不能加分号。

⑤ 空语句。只有分号“;”组成的语句称为空语句。空语句是什么也不执行的语句，在程序中空语句可用来作空循环体。

例如：while(getchar()!='\n'); 本语句的功能是，只要从键盘输入的字符不是回车则重新输入。这里的循环体为空语句。

(4) 函数定义。函数是完成特定任务的独立模块，是 C 语言唯一的一种子程序形式。函数的目的通常是接收 0 个或多个数据（称为函数的参数），并返回 0 个或 1 个结果（称为函数的返回值）。函数的使用主要涉及函数的定义与调用。

函数定义的主要内容是通过编写一系列语句来规定其所完成的功能。完整的函数定义涉及函数头和函数体。其中，函数头包括函数的返回值类型、函数名、参数类型；而函数体是一个程序模块，规定了该函数所具有的功能。函数调用则通过传递函数的参数并执行函数定义所规定的程序过程，以实现相应的功能。以下是函数定义的一个简单例子。

```
int max(int m,int n) /*函数头：函数类型说明符 函数名(函数参数列表)*/
{                  /*函数体的开始*/
    int x;        /*声明一个整型变量 x*/
    if(m>n)
        x=m;
    else
        x=n;
    return x;    /*判断 m、n 的大小，将其中的较大者值赋予变量 x*/
}                /*结束函数调用，并返回变量 x 的值*/
/*函数体的结束*/
```

(5) 输入与输出。C 语言没有输入输出语句，它通过调用系统库函数中的有关函数 [如：printf()、scanf() 函数等] 实现数据的输入和输出，这种处理方式对 C 语言在不同硬件平台上的可移植性提供了良好的基础。相关的输入、输出函数的使用及其功能将会在后面陆续做讲解。

1.2 C 语言的发展和优缺点

C 语言作为计算机编程语言，具有功能强、语句表达简练、控制和数据结构丰富灵活、程序时空开销小等特点。它既具有 Pascal、FORTRAN、COBOL 等通用程序设计语言的特点，又具有汇编语言（Assemble Language）中位（bit）、地址（Address）、寄存器（Register）等概念，拥有其他许多高级语言所没有的低层操作能力；既适合于编写系统软件，也可用来编写应用软件。C

语言的这些特点与其发展过程是密不可分的。

1.2.1 C 语言的发展

C 语言之所以命名为 C, 是因为 C 语言源自 Ken Thompson 发明的 B 语言, 而 B 语言则源自 BCPL。

1967 年, 英国剑桥大学的 Martin Richards 对 CPL 进行了简化, 于是产生了 BCPL (Basic Combined Programming Language)。

20 世纪 60 年代, 美国 AT&T 公司贝尔实验室(AT&T Bell Laboratory)的研究员 Ken Thompson 闲来无事, 手痒难耐, 想玩一个他自己编的、模拟在太阳系航行的电子游戏——Space Travel。他背着老板, 找到了台空闲的机器——PDP-7。但这台机器没有操作系统, 而游戏必须使用操作系统的一些功能, 于是他着手为 PDP-7 开发操作系统。后来, 这个操作系统被命名为——UNIX。

1970 年, 还是这位 Ken Thompson, 以 BCPL 为基础, 设计出很简单且很接近硬件的 B 语言(取 BCPL 的首字母)。并且他用 B 语言写了第一个 UNIX 操作系统。

1971 年, 同样酷爱 Space Travel 的 Dennis M. Ritchie 为了能早点儿玩上游戏, 加入了 Thompson 的开发项目, 合作开发 UNIX。他的主要工作是改造 B 语言, 使其更成熟。

1972 年, 美国贝尔实验室的 Dennis M. Ritchie 在 B 语言的基础上最终设计出了一种新的语言, 他取了 BCPL 的第二个字母作为这种语言的名字, 这就是 C 语言。

1973 年初, C 语言的主体完成。Thompson 和 Ritchie 迫不及待地开始用它完全重写了 UNIX。此时, 编程的乐趣使他们已经完全忘记了那个"Space Travel", 一门心思地投入到了 UNIX 和 C 语言的开发中。随着 UNIX 的发展, C 语言自身也在不断地完善。直到今天, 各种版本的 UNIX 内核和周边工具仍然使用 C 语言作为最主要的开发语言, 其中还有不少继承 Thompson 和 Ritchie 之手的代码。

在开发中, 他们还考虑把 UNIX 移植到其他类型的计算机上使用。C 语言强大的移植性(Portability)在此显现。机器语言和汇编语言都不具有移植性, 为 x86 开发的程序, 不可能在 Alpha、SPARC 和 ARM 等机器上运行。而 C 语言程序则可以使用在任意架构的处理器上, 只要那种架构的处理器具有对应的 C 语言编译器和库, 然后将 C 源代码编译、连接成目标二进制文件之后即可运行。

1977 年, Dennis M. Ritchie 发表了不依赖于具体机器系统的 C 语言编译文本《可移植的 C 语言编译程序》。

C 语言继续发展, 在 1982 年, 很多有识之士和美国国家标准协会为了使这个语言健康地发展下去, 决定成立 C 标准委员会, 建立 C 语言的标准。委员会由硬件厂商、编译器及其他软件工具生产商、软件设计师、顾问、学术界人士、C 语言作者和应用程序员组成。1989 年, ANSI 发布了第一个完整的 C 语言标准——ANSI X3.159—1989, 简称“C89”, 不过人们也习惯称其为“ANSI C”。C89 在 1990 年被国际标准组织 ISO (International Organization for Standardization) 一字不改地采纳, ISO 官方给予的名称为 ISO/IEC 9899, 所以 ISO/IEC 9899: 1990 也通常被简称为“C90”。1999 年, 在做了一些必要的修正和完善后, ISO 发布了新的 C 语言标准, 命名为 ISO/IEC 9899: 1999, 简称“C99”。在 2011 年 12 月 8 日, ISO 又正式发布了新的标准, 称为 ISO/IEC 9899: 2011, 简称为“C11”。

C 语言是 C++ 的基础, C++ 语言和 C 语言在很多方面是兼容的。因此, 掌握了 C 语言, 可为将来学习 C++ 打下坚实的基础。本教材使用 Visual C++6.0 作为 C 语言程序的运行环境。

1.2.2 C 语言的优缺点

1. C 语言的优点

C 语言既用来编写系统软件, 又用来开发应用软件, 已成为一种通用程序设计语言, 主

要具备以下优点。

(1) 简洁紧凑、灵活方便。C语言一共只有32个关键字、9种控制语句，程序书写形式自由，区分大小写。把高级语言的基本结构和语句与低级语言的实用性结合起来。C语言可以像汇编语言一样对位、字节和地址进行操作，而这三者是计算机最基本的工作单元。

(2) 运算符丰富。C语言的运算符包含的范围很广泛，共有34种运算符。C语言把括号、赋值、强制类型转换等都作为运算符处理。从而使C语言的运算类型极其丰富，表达式类型多样化。灵活使用各种运算符可以实现在其他高级语言中难以实现的运算。

(3) 数据类型丰富。C语言的数据类型有整型、实型、字符型、数组类型、指针类型、结构体类型、共用体类型等，能用来实现各种复杂的数据结构的运算，并引入了指针概念，使程序效率更高。

(4) 表达方式灵活实用。C语言提供多种运算符和表达式求值的方法，对问题的表达可通过多种途径获得，其程序设计更主动、灵活。它语法限制不太严格，程序设计自由度大，如对整型量与字符型数据及逻辑型数据可以通用等。

(5) 允许直接访问物理地址，对硬件进行操作。由于C语言允许直接访问物理地址，可以直接对硬件进行操作，因此它既具有高级语言的功能，又具有低级语言的许多功能，能够像汇编语言一样对位(bit)、字节和地址进行操作，而这三者是计算机最基本的工作单元，可用来写系统软件。

(6) 生成目标代码质量高，程序执行效率高。C语言描述问题比汇编语言迅速，工作量小、可读性好，易于调试、修改和移植，而代码质量与汇编语言相当。C语言一般只比汇编程序生成的目标代码效率低10%~20%。

(7) 可移植性好。C语言在不同机器上的C编译程序，86%的代码是公共的，所以C语言的编译程序便于移植。在一个环境上用C语言编写的程序，不改动或稍加改动，就可移植到另一个完全不同的环境中运行。

(8) 表达力强。C语言有丰富的数据结构和运算符。包含了各种数据结构，如整型、数组类型、指针类型和联合类型等，用来实现各种数据结构的运算。C语言的运算符有34种，范围很宽，灵活使用各种运算符可以实现难度极大的运算。C语言能直接访问硬件的物理地址，能进行位(bit)操作，兼有高级语言和低级语言的许多优点。C语言具有强大的图形功能，支持多种显示器和驱动器，且计算功能、逻辑判断功能强大。

2. C语言的缺点

当然，C语言也有缺点，主要表现在如下几点。

(1) C语言的缺点主要表现在数据的封装性上，这一点使得C在数据的安全性上有很大缺陷，这也是C和C++的一大区别。

(2) C语言的语法限制不太严格，对变量的类型约束不严格，影响程序的安全性，对数组下标越界不做检查等。从应用的角度，C语言比其他高级语言较难掌握。也就是说，对用C语言的人，要求对程序设计更熟练一些。

1.3 C程序的结构

用C语言编写的程序称为C语言源程序，简称为C程序。为了说明C语言源程序的结构特点，先看以下几个程序。这几个程序由简单到复杂，虽然有关内容还未介绍，但可以从中了解到C语言源程序在基本组成结构上的特点及其书写风格。

1.3.1 简单C程序举例

【例 1-1】编写一个 C 语言程序，输出“Good Luck!”。

程序如下：

```
/*cl_1.c*/
#include <stdio.h> /*为文件包含，其扩展名为.h，称为头文件*/
void main()
{
    printf("Good Luck! \n"); /*通过显示器输出 Good Luck!*/
}
```

说明：

① C 语言程序中可以随时使用注释，但注释内容不参与编译。

② #include 称为文件包含命令或编译预处理命令，#include <stdio.h>是文件包含，其意义是把尖括号<>或引号""内指定的文件包含到本程序来，成为本程序的一部分。被包含的文件通常是由系统提供的，其扩展名为.h，称为头文件或首部文件。C 语言的头文件中包括了各个标准库函数的函数原型。因此，凡是在程序中调用一个库函数时，都必须包含该函数原型所在的头文件。需注意：编译预处理命令的末尾不加分号。详细内容将在后面章节介绍。

③ main 是主函数的函数名，表示这是一个主函数。每个完整的 C 语言源程序都必须有主函数，且只能有一个主函数(main 函数)，程序总是从 main 函数开始执行，并终止于 main 函数。函数体由一对大括弧“{}”括起来，其间一般包括程序的说明部分和执行部分。

④ printf 函数是一个由系统定义的标准函数，可在程序中直接调用。其功能是将输出的内容送到显示器显示。

该程序正确执行后，会在显示器上显示输出：

```
Good Luck!
```

【例 1-2】从键盘输入两个整数 yw 和 sx，将两数之平均值显示输出。

```
/*cl_2.c*/
#include<stdio.h>
void main()
{
    int yw,sx,sum; /*定义3个整型变量*/
    printf("Input two numbers:"); /*显示提示信息*/
    scanf("%d%d",&yw,&sx); /*输入 yw, sx 值*/
    sum=yw+sx; /*求出 yw 与 sx 之和，并把它赋予变量 sum*/
    printf("average=%d\n",sum/2); /*输出语文和数学的平均成绩*/
}
```

程序分析：

- ① 该程序中使用了 yw、sx 和 sum 3 个变量，所有变量在使用之前必须先定义。
- ② scanf 函数是一个由系统定义的标准函数，可在程序中直接调用。它的功能是输入变量 yw 和 sx 的值。&yw 和 &sx 中“&”的含义是“取变量地址”，表示将从键盘输入的两个值分别存放到地址标志为 yw 和 sx 的存储单元中。
- ③ “%d”是输入/输出数据的“格式说明”，用来指定输入/输出时的数据类型和格式，%d 表示“十进制整数类型”，在执行输出时，屏幕上显示一个十进制整数值。
- ④ sum=yw+sx 为赋值表达式，表示将 x+y 之和赋值给 sum 变量所标识的存储单元。

该程序正确执行后，会在显示器上显示输出：

```
Input two numbers:80 90
average=85
```

【例 1-3】从键盘输入两个整数 a 和 b，进行比较后将较大值输出。

```
/*cl_3.c*/
#include<stdio.h>
void main()
{
    int x,y,z;                /*定义三个整型变量*/
    int max(int a,int b);    /*函数类型说明*/
    printf("Input two number:"); /*显示提示信息*/
    scanf("%d%d",&x,&y);    /*输入 x, y 值*/
    z=max(x,y);             /*调用 max 函数*/
    printf("max=%d\n",z);   /*将较大数输出*/
}

int max(int a,int b)       /*定义 max 函数*/
{
    int c;                 /*定义一个整型变量*/
    c=a>b?a:b;            /*求出变量 c 的值*/
    return c;             /*将 c 的值返回到主调函数*/
}
```

程序分析：

① 本程序包括两个函数：主函数 main 和自定义函数 max。max 函数的作用是将 a 和 b 中较大的值赋予变量 c；return 语句将 c 的值返回主调函数。

② 在调用 max 函数时，将实际参数 x 和 y 的值分别对应传给 max 函数中的形式参数 a 和 b。

③ a>b?a:b 是一个条件表达式，当 a>b 成立时，a>b?a:b 表达式的值为 a 的值；反之则为 b 的值。详细内容将在第 2 章介绍。

该程序正确执行后，会在显示器上显示输出：

```
Input two numbers:10 20
max=20
```

本例中涉及函数调用、实际参数和形式参数等概念，如果读者对此不大理解，可先不予以深究，第 6 章中将会有详尽介绍。

1.3.2 C语言程序的结构特点

通过上面 3 个 C 语言源程序，可以看出其基本结构具有以下几个特点。

(1) C 语言源程序的基本组成单位是函数。所有的 C 语言程序都由一个或多个函数构成，其中 main 函数必须有且只能有一个。

(2) main() 函数可以出现在 C 源程序的任何位置，程序执行时总是从 main() 函数开始，又在 main() 函数结束。主函数可以调用标准库函数 [如 printf()、scanf() 等] 和用户自定义函数，但标准库函数和用户自定义函数却不能调用主函数。

(3) 源程序中的预处理命令通常放在源文件或源程序的最前面。

(4) 分号“;”是 C 语句的必要组成部分。每个语句或每个变量说明都必须以分号结尾。但预

处理命令、函数头和花括号“{”和“}”后面不能加分号。

(5) 标识符、关键字之间必须至少加一个空格以示分隔。

(6) 可以在程序的任何位置用“/*注释内容*/”或“//注释内容”的形式对程序或语句进行注释,以增加程序的可读性。

1.3.3 书写程序时应遵循的规则

C 语言程序的书写格式非常自由,但从书写清晰,便于阅读、理解、维护的角度出发,建议在书写 C 语言程序时遵循以下几个规则。

(1) 一个说明或一条语句占一行。

(2) 用{}括起来的部分,通常表示程序的某一层次结构(如函数体、循环体、复合语句等)。{}一般与该结构语句的第一个字母对齐,并单独占一行。

(3) 低一层次的语句或说明比高一层次的语句或说明向后缩进若干格后书写,同一层次的语句或说明左对齐,以增强程序编写的层次感,增加程序的可读性。

(4) 函数块与函数块之间加一空行分隔,以便清楚地分出程序中有几个函数。

编程时应力求遵循上述规则,以养成良好的编程习惯。

1.4 Visual C++ 6.0 上机简介

Visual C++(简称 VC++)是美国 Microsoft 公司开发的 Microsoft Visual Studio 的一部分,是一个基于 Windows 操作系统的可视化、面向对象且使用广泛的 C/C++集成开发环境(Integrated Development Environment, IDE)。它成功地将面向对象和事件驱动编程概念联系起来,并得到了很好的配合,使得编写 Windows 应用程序的过程变得简单、方便且代码量小。VC++ 6.0 集程序的代码编辑、编译、连接、调试于一体,给编程人员提供了一个完整、方便的开发界面和许多有效的辅助开发工具。

VC++ 6.0 的编辑环境包含了许多独立的组件,它们是:文本编辑器、资源编辑器、C/C++编译器、连接器、调试器、AppWizard、ClassWizard、源程序浏览器以及联机帮助。所有这些构件的功能都隐藏在 VC++ 6.0 的菜单和工具条中。通过该集成环境,程序员可以观察和控制整个开发过程。

1.4.1 Visual C++ 6.0 集成开发环境简介

在已安装 Visual C++的计算机上,可以直接从桌面双击 Microsoft Visual C++图标,进入 Visual C++ 集成开发环境,或者单击【开始】|【程序】菜单,选择 Microsoft Visual Studio 6.0 中的 Microsoft Visual C++ 6.0 菜单项,进入 Visual C++ 6.0 集成开发环境。

Visual C++集成开发环境主要由标题栏、菜单栏、工具栏、项目工作区、编辑区、输出区等组成,如图 1-1 所示。

1. 项目工作区

Visual C++集成开发环境以项目工作区来组织应用程序的工程,项目工作区文件扩展名为.dsw,这种类型的文件在 Visual C++中级别是最高的。项目工作区含有工作区的定义和工程中所包含文件的所有信息。所以,要打开一个工程,只需打开对应的项目工作区文件(*.dsw)即可。

项目工作区窗格位于屏幕左侧,包含 ClassView(类视图)、ResourceView(资源视图)和 FileView(文件视图)3种视图。