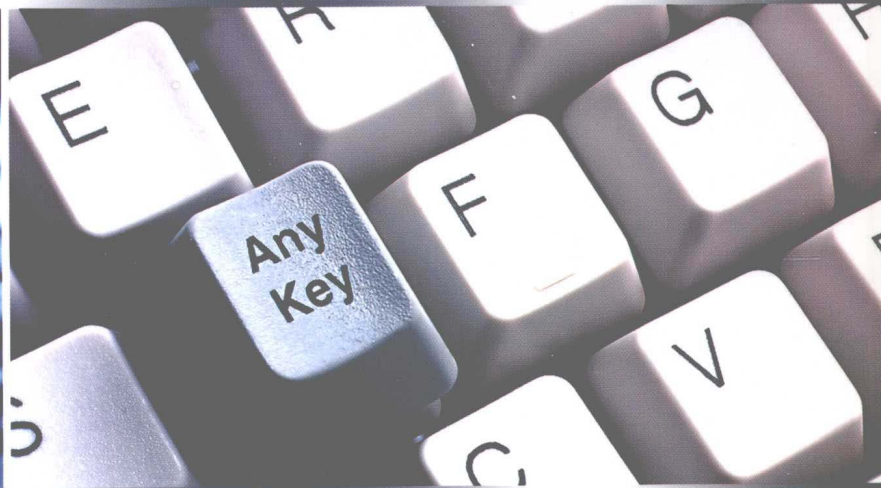




普通高等教育“十一五”规划教材

C 语言程序设计

C YUYANCHENGXUSHEJI



主 编 胡宏智
副主编 王小林 黄晓梅 王广正



中国水利水电出版社
www.waterpub.com.cn

普通高等教育“十一五”规划教材

C 语言程序设计

主 编 胡宏智

副主编 王小林 黄晓梅 王广正



中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

本书以程序案例为导向,全面介绍C语言的基本概念、数据类型、语句及其特点,深入浅出地讲解C语言程序设计的基本方法,注重读者程序设计思想和程序设计能力的培养。

全书共分8章,第1~4章介绍了C语言基础知识、基本的数据类型、运算符与基本的程序语句、C语言结构化程序的算法设计与程序的控制结构、函数及编译预处理,通过前4章的学习,读者可以掌握C语言简单程序设计的方法、步骤,C语言结构化、模块化程序设计的思想和方法。第5~8章的内容包括:数组、指针、构造数据类型和文件的概念与应用。读者通过5~8章的学习,能够利用C语言进行比较复杂的程序设计。

本书每章均设置适量的习题、常见问题的问和答,并附习题解答。全书语言通俗易懂,内容取舍合理、由浅入深,适合用于普通高等院校本、专科的教材,也可供工程技术人员参考。

本书配有电子教案及相关资料,包括Turbo C (V2.0)编译错误信息、Turbo C (V2.0)使用指南以及习题参考答案,读者可从中国水利水电出版社网站或万水书苑免费下载,网址:<http://www.waterpub.com.cn/softdown/>或<http://www.wsbookshow.com>。

图书在版编目(CIP)数据

C语言程序设计 / 胡宏智主编. — 北京:中国水利水电出版社,2010.2
普通高等教育“十一五”规划教材
ISBN 978-7-5084-6545-6

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

中国版本图书馆CIP数据核字(2010)第018778号

策划编辑:庞永江 责任编辑:杨元泓 加工编辑:陈洁 封面设计:李佳

书 名	普通高等教育“十一五”规划教材 C语言程序设计
作 者	主 编 胡宏智 副主编 王小林 黄晓梅 王广正
出版发行	中国水利水电出版社 (北京市海淀区玉渊潭南路1号D座 100038) 网址: www.waterpub.com.cn E-mail: mchannel@263.net (万水) sales@waterpub.com.cn 电话: (010) 68367658 (营销中心)、82562819 (万水)
经 售	全国各地新华书店和相关出版物销售网点
排 版	北京万水电子信息有限公司
印 刷	北京市天竺颖华印刷厂
规 格	184mm×260mm 16开本 17.25印张 442千字
版 次	2010年2月第1版 2010年2月第1次印刷
印 数	0001—4000册
定 价	29.00元

凡购买我社图书,如有缺页、倒页、脱页的,本社营销中心负责调换
版权所有·侵权必究

前 言

21 世纪的信息时代，承担培养信息技术人才任务的高校在众多的程序设计语言中，选用“C 语言程序设计”作为计算机类和多数理工类专业的重要基础课程之一，不仅仅是因为 C 语言从诞生之日起到现在一直流行，未来也会常葆青春，而且也是由于 C 语言自身固有的、其他程序设计语言难于媲美的优点所决定的。任何事物都有两面性，C 语言也不例外。C 语言介于高级语言与低级语言之间，具有晦涩难懂的指针概念及既严格又灵活的语法规则，对于初学者来说，确有其不易入门的缺点。为了帮助初学者尽快掌握 C 语言的基础知识、基本概念，开拓程序设计的分析思路、建立模块化程序设计思想，培养编程技能，我们综合各位老师多年的教学经验，编写此教材。

本书通过 188 个示例，全面介绍了 C 语言语句的功能、用法，重点讲解问题求解的算法思想、编程思路。全书共由 8 章和 4 个附录组成。

第 1 章 C 语言概述。通过本章的学习，使读者了解 C 语言的主要特点、C 语言源程序的结构组成、开发工具和程序设计步骤。

第 2 章主要介绍基本的数据类型、运算符、表达式与标识符、常量、变量的概念，基本的输入输出函数调用。通过本章的学习，读者可以进行简单的顺序程序设计。

第 3 章主要介绍算法的概念、程序的选择和循环结构。通过本章的学习训练，读者可以进行较复杂问题的求解算法描述，并转化为 C 语言描述的程序设计技能。

第 4 章全面介绍了函数的概念与编译预处理。通过本章的学习与练习，逐步学会把复杂问题分解成若干个相对简单、独立的功能模块，实施分而治之，形成模块化程序设计思想。学会如何把分散的多个程序文件组合成一个完整的 C 程序。

第 5 章介绍了数组的概念与用法。通过本章的学习，读者将学会对数据类型相同的多个变量的简化定义、字符数组和字符串的引用的方法。

第 6 章着重介绍了指针的概念、指针与数组的关系、指针函数、函数指针、指针数组、数组指针、动态数组、图形处理函数等。通过本章的学习，读者可以学会高级、高效的程序设计。

第 7 章介绍了结构体、共用体、枚举等构造数据类型及其变量的定义、引用方法。

第 8 章介绍了文件的概念、分类和文件的使用。

书中前面加*号的章节，可根据各自的具体情况，作为选学或因课时有限不作为讲授内容。全书例题和习题中的程序一一经过上机验证通过，确保程序的正确性。

本书特色在于：

(1) 每一章的开篇都给出教学内容提要、教学目标、教师要求与对学生的要求。对教与学都具有一定的建议或指导性参考价值。

(2) 全书以实例为导引，避免空洞的抽象描述。结合问题的分析与程序设计讲解概念、语句的应用和注意事项，做到理论联系实际。

(3) 本书具有配套的全部实例与习题答案的电子资料、教学课件。本书的读者可以通过电子信箱 hhz163@126.com 与作者联系。

本书的编写工作主要由安徽工业大学的胡宏智、王小林、王广正、安徽建筑工业学院的黄晓梅和安徽冶金科技学院的胡恩勇承担。其中的第1、4章由胡宏智编写，第2~3章由王小林编写，第5章由黄晓梅编写，第6章由胡恩勇、胡宏智编写，第7、8章和附录A、B、C、D由王广正编写。全书由胡宏智统编定稿。

本书在编写过程中得到了安徽工业大学、安徽建筑工业学院有关领导的热情关怀和王喜凤、郭玉华、程改荣、朱艳萍、刘凯、王华、陈国钦等老师的真诚帮助，自始至终在中国水利水电出版社的雷顺加、徐海洋关心下，顺利完成书稿。在此作者对他们的无私支持与奉献，深表谢意！

由于作者水平有限，书中难免有不妥或错误，敬请读者批评指正。

编 者

2009年12月

目 录

前言

第1章 C语言概述	1	2.4.10 C语句概述	37
1.1 程序设计语言的发展	1	2.5 标准的输入/输出函数	38
1.2 C语言的发展过程	3	2.5.1 格式化输出函数 printf()	38
1.3 C语言的主要特点	4	2.5.2 格式化输入函数 scanf()	42
1.4 C语言程序的结构	5	2.5.3 字符输出函数 putchar()、putc()、putch()	44
1.5 C语言程序的执行	8	2.5.4 字符输入函数 getchar()、getc()、getch()	46
1.5.1 源程序翻译	9	2.6 程序范例	47
1.5.2 链接目标程序	10	2.7 本章小结	49
1.6 C语言程序的开发工具	11	2.8 问与答	51
1.7 本章小结	12	习题二	52
1.8 问与答	12	第3章 算法设计与程序的控制结构	56
习题一	12	3.1 算法的概念与特点	56
第2章 基本的数据类型、运算符与程序语句	13	3.1.1 什么是算法	56
2.1 数据在计算机内的存储形式	13	3.1.2 算法的特点	57
2.1.1 字符型数据	13	3.1.3 引例	57
2.1.2 整型数据	14	3.2 算法的描述方法	60
2.1.3 实型数据	14	3.2.1 用自然语言表示算法	60
2.2 标识符、变量和常量	15	3.2.2 用流程图表示算法	60
2.2.1 标识符	15	3.2.3 用伪代码表示算法	65
2.2.2 变量	15	3.2.4 用计算机语言表示算法	65
2.2.3 常量	16	3.3 顺序结构	66
2.3 基本数据类型	16	3.3.1 顺序结构	66
2.3.1 整型	17	3.3.2 应用程序范例	67
2.3.2 实型数据	18	3.4 选择结构	68
2.3.3 字符型	20	3.4.1 分支语句 if()~与 if()~else~	68
2.4 运算符和表达式	23	3.4.2 条件运算符和条件表达式	73
2.4.1 算术运算符和算术表达式	24	3.4.3 多分支语句 switch	74
2.4.2 关系运算符和关系表达式	26	3.4.4 应用程序范例	76
2.4.3 逻辑运算符和逻辑表达式	27	3.5 循环结构	80
2.4.4 赋值运算符和赋值表达式	28	3.5.1 while 语句、do~while 语句和 for 语句	80
2.4.5 逗号运算符和逗号表达式	30	3.5.2 单重循环应用程序范例	85
2.4.6 位运算符与位运算	31	3.5.3 嵌套循环及其应用程序范例	86
2.4.7 sizeof 运算符	35		
2.4.8 数据类型的转换	35		
2.4.9 表达式的计算次序	37		

3.6 程序流程的转移控制.....	87	应用程序范例	142
3.6.1 跳转语句 goto.....	87	5.3 多维数组的定义、引用和初始化	147
3.6.2 break 和 continue 语句.....	88	5.3.1 二维数组的概念.....	147
3.6.3 exit()函数.....	90	5.3.2 二维数组的定义.....	147
3.6.4 应用程序范例.....	91	5.3.3 二维数组及多维数组的初始化.....	148
3.7 结构化程序设计方法.....	94	5.3.4 二维数组元素的引用及其 应用程序范例	149
3.8 本章小结	95	5.3.5 二维数组作为函数的参数与 应用程序范例	152
3.9 问与答	96	5.4 字符数组.....	156
习题三.....	99	5.4.1 字符数组与字符串的关系.....	156
第4章 函数与编译预处理	103	5.4.2 字符数组的初始化.....	157
4.1 函数的概念及其分类.....	103	5.4.3 字符串的输入	158
4.2 函数的定义	104	5.4.4 字符串的输出	159
4.3 函数的声明与调用.....	107	5.4.5 字符串处理函数.....	160
4.3.1 函数能被使用的条件.....	107	5.4.6 字符数组的应用程序范例	162
4.3.2 函数声明的场合.....	107	5.5 本章小结.....	165
4.3.3 函数声明的格式.....	108	5.6 问与答.....	166
4.3.4 函数声明的位置.....	108	习题五	166
4.3.5 函数的调用、参数与返回值.....	109	第6章 指针	170
4.3.6 函数的传值方式.....	110	6.1 指针的概念.....	170
4.4 变量的作用域和存储类别	111	6.1.1 什么是指针	170
4.4.1 变量的作用域.....	111	6.1.2 为什么要引入指针概念.....	171
4.4.2 变量的存储类别.....	112	6.1.3 指针变量的定义.....	171
4.5 内部函数与外部函数.....	117	6.1.4 指针变量的使用.....	173
4.6 递归函数	118	6.2 指针的运算.....	176
4.7 编译预处理	123	6.2.1 指针的算术运算.....	176
4.7.1 宏定义与宏替换.....	123	6.2.2 指针的关系运算.....	177
4.7.2 文件包含预处理命令#include.....	126	6.3 指针与数组之间的关系.....	177
4.7.3 条件编译命令.....	127	6.3.1 指向一维数组元素的指针及其 应用程序范例	177
4.8 综合范例	129	6.3.2 指向二维数组的指针及其 应用程序范例	180
4.9 本章小结	130	6.3.3 字符指针及其应用程序范例	181
4.10 问与答.....	131	6.3.4 指针数组及其应用程序范例	183
习题四.....	132	6.4 指针作为函数的参数及其 应用程序范例	185
第5章 数组	136	6.5 指针作为函数的返回值及 其应用程序范例	187
5.1 数组的应用场合.....	136	6.6 指向函数的指针及其应用程序范例.....	189
5.2 一维数组的定义、引用和初始化.....	137		
5.2.1 一维数组的定义.....	137		
5.2.2 一维数组的初始化.....	137		
5.2.3 一维数组元素的引用与 应用程序范例.....	138		
5.2.4 一维数组作为函数的参数与			

6.7 带参数的 main 函数及其应用程序范例	192	数据类型的定义	225
6.8 指向指针的指针及其应用程序范例	194	7.6.2 共用体（联合体） 类型变量的定义与引用	225
*6.9 动态数组的实现	195	7.7 枚举数据类型	228
6.9.1 C 程序的内存映像	195	7.7.1 枚举数据类型的定义	228
6.9.2 动态内存分配与释放函数	196	7.7.2 枚举类型变量的定义与引用	228
6.9.3 一维动态数组的实现	199	*7.8 单向链表的概念	230
6.9.4 二维动态数组的实现	199	7.8.1 问题的提出	230
*6.10 图形处理函数的简单应用	200	7.8.2 单向链表的建立	231
6.11 本章小结	204	7.8.3 单向链表的节点的插入	231
习题六	205	7.8.4 单向链表节点的删除	232
第 7 章 构造数据类型	210	7.8.5 单向链表的销毁	233
7.1 问题的提出	210	*7.9 位段	235
7.2 结构体数据类型	211	7.10 本章小结	236
7.2.1 结构体数据类型的定义	211	7.11 问与答	236
7.2.2 关键字 typedef 的用法	212	习题七	237
7.2.3 结构体变量的定义	213	第 8 章 文件	242
7.2.4 结构体变量的初始化	215	8.1 问题的提出	242
7.2.5 结构体变量成员的引用	216	8.2 文件的概念	242
7.2.6 结构体变量成员的输入/输出	218	8.3 文件操作的常用函数	243
7.3 结构体数组	219	8.3.1 文件的打开与关闭	244
7.3.1 结构体数组的定义与初始化	219	8.3.2 文件的错误检测与处理	245
7.3.2 结构体数组的应用程序范例	219	8.3.3 文件的顺序读写及其范例	246
7.4 结构体指针	220	8.3.4 文件的随机读写及其范例	251
7.4.1 结构体指针变量的定义与 应用程序范例	220	8.4 本章小结	252
7.4.2 结构体数组与指针变量及 其应用程序范例	221	8.5 问与答	253
7.5 结构体变量与函数	222	习题八	253
7.5.1 结构体作为函数的形式参数 与实际参数	222	附录 A C 语言的保留字与关键字	256
7.5.2 函数的返回值类型为结构体	223	附录 B C 语言运算符优先级结合性表	257
7.6 共用体（联合体）数据类型	224	附录 C ASCII 字符表	258
7.6.1 共用体（联合体）		附录 D C 语言常用库函数	260
		参考文献	265

第 1 章

C 语言概述

教学内容

- C 语言概述
- C 语言程序的构成
- C 语言程序的开发工具介绍
- C 语言程序的设计步骤

教学目标

通过本章的学习，编写最简单的 C 程序，上机编辑、编译、连接、运行，熟悉 C 语言的特点、程序设计步骤和基本方法。

教师要求

重点解决学习 C 语言程序设计入门难点问题，教会学生掌握 C 语言程序设计的学习方法。学生通过阅读、模仿典型程序例题，编写简单的程序，并上机调试，直至正确。初步建立求解问题的程序设计思想。关键是要及时掌控学生的整体学习状况。建议采用灵活多样的教学方法和方式，如课堂程序调试、课后机房现场答疑、网上交流、常见的典型问题在网上公开等。

学生要求

做到“一预三勤”。课前预习：对于不理解或难理解的问题，既可以同学之间相互讨论，也可以带着问题认真听讲；勤思考：课后勤思考勤复习；勤问题：有问题必须问同学问老师，不能积累问题，极力避免积重难返；勤动手：勤动手做读书笔记、课堂笔记，一定要不折不扣地完成作业，勤上机验证自己编写的程序和练习。

1.1 程序设计语言的发展

随着计算机科学及其应用的迅猛发展，计算机已广泛地应用于人类的生产、生活的各个领域，特别是随着国际互联网（Internet）日益深入千家万户，我们已经无法摆脱对计算机的依赖，计算机已将人类带入了一个新的时代——信息时代。

信息时代对于我们的基本要求之一是：自觉、主动地学习和掌握计算机的基本知识和基本能力，并把它作为自己应该具备的基本素质。缺乏计算机知识，就会成为信息时代的“文盲”。

对于当今的大学生而言，掌握一门高级程序设计语言及其基本的编程技能是必需的。大学学习，除了掌握本专业系统的基础知识外，科学精神的培养、思维方法的锻炼、严谨踏实的科研作风养成，以及分析问题、解决问题的能力训练，都是日后工作的基础。学习计算机语言，正是一种十分有益的训练方式，而语言本身又是与计算机进行交互的有力的工具。

一台计算机是由硬件系统和软件系统两大部分构成的，硬件是物质基础，而软件可以说是计算机的灵魂，没有软件，计算机就是一台什么也不能干的“白痴”，有了软件，才能灵动起来，成为一台真正的“电脑”。所有的软件都是用计算机语言编写的。

计算机程序设计语言的发展，经历了从低级到高级的历程。

1. 低级语言

(1) 机器语言。计算机只懂得由“0”和“1”组成的二进制数，二进制是计算机语言的基础。计算机发明之初，只是用计算机的语言去指挥计算机干这干那，一句话，就是写出一串串由“0”和“1”组成的指令序列交由计算机执行，这种语言就是机器语言。使用机器语言是十分痛苦的，特别是在程序有错需要修改时，更是如此。而且，由于每台计算机的指令系统往往各不相同，所以，在一台计算机上执行的程序，要想在另一台计算机上执行，必须另编程序，造成了重复工作。但由于使用的是针对特定型号计算机的语言，故而运算效率是所有语言中最高的。机器语言是第一代计算机语言。

(2) 汇编语言。为了减轻使用机器语言编程的痛苦，人们进行了一种有益的改进：用一些简洁的英文字母、符号串来替代一条特定指令的二进制串，比如，用“ADD A,50”代表加法，“SUB B,20”代表减法运算等，这样一来，人们很容易读懂并理解程序在干什么，纠错及维护都变得方便了，这种程序设计语言就称为汇编语言，即第二代计算机语言。然而计算机是不认识这些符号的，这就需要有一个专门的程序，专门负责将这些符号翻译成二进制数的机器语言，这种翻译程序被称为汇编程序。

汇编语言同样十分依赖于机器硬件，移植性不好，因而它仍然属于低级语言。针对计算机特定硬件而编制的汇编语言程序，能准确发挥计算机硬件的功能和特长，程序精炼而质量高，运行效率十分高，几乎等同于机器语言的效率，所以至今仍是一种常用而强有力的软件开发工具。

2. 高级语言

从最初与计算机交流的痛苦经历中，人们意识到，应该设计一种接近于数学语言或人的自然语言，同时又不依赖于计算机硬件，编出的程序能在所有机器上通用的这样一种语言。经过努力，1954年，第一个完全脱离机器硬件的高级语言——FORTRAN问世了，50多年来共有几百种高级语言出现，其中影响较大、使用较普遍的有FORTRAN、ALGOL、COBOL、BASIC、LISP、SNOBOL、PL/1、Pascal、C、PROLOG、Ada、C++、VC、Delphi、Java、C#等。

高级语言的发展也经历了从早期语言到结构化程序设计语言，从面向过程到非过程化程序语言的过程。相应地，软件的开发也由最初的个体手工作坊式的封闭式生产，发展为产业化、流水线式的工业化生产。

20世纪60年代中后期，软件越来越多，规模越来越大，而软件的生产基本上是人自为战，缺乏科学规范的系统规划与测试、评估标准，其恶果是大批耗费巨资建立起来的软件系统，由于含有错误而无法使用，甚至带来巨大损失，软件给人的感觉是越来越不可靠，以致几乎没有不出错的软件。这一切极大地震动了计算机界，历史上称之为“软件危机”。人们认识到：大型程序的编制不同于写小程序，它应该是一项新的技术，应该像处理工程一样处理软件研制的全过程。程序的设计应易于保证正确性，也便于验证正确性。1969年，提出了结构化程序设计方法，1970年，第一个结构化程序设计语言——Pascal语言出现，标志着结构化程序设计时期的开始。

20世纪80年代初开始，在软件设计思想上又产生了一次革命，其成果就是面向对象的程序设计。在此之前的高级语言，几乎都是面向过程的，程序的执行是流水线似的，在一个模块被执行完成前，人们不能干别的事，也无法动态地改变程序的执行方向。这和人们日常处理事务的方式是不一致的，对人而言是希望发生一件事就处理一件事，也就是说，不能面向过程，而应是面

向具体的应用功能，也就是对象 (object)。其方法就是软件的集成化，如同硬件的集成电路一样，生产一些通用的、封装紧密的功能模块，称之为软件集成块，它与具体应用无关，但能相互组合，完成具体的应用功能，同时又能重复使用。对使用者来说，只关心它的接口（输入量、输出量）和能实现的功能，至于如何实现的，那是它内部的事，使用者完全不用关心，C++、C#、Delphi、Java 就是典型代表。

高级语言的下一个发展目标是面向应用，也就是说：只需要告诉程序你要干什么，程序就能自动生成算法，自动进行处理，这就是非过程化的程序语言。

数据库操作语言就是这样的一种语言。例如，在某个数据库应用系统中，有了一个学生基本情况信息表 XQB，若要从其中查找 1992 年及其以后出生的所有男性同学的信息，采用 SQL 语言，可用如下的语句实现之：

```
Select 姓名,学号,出生地 from XQB where 出生日期>=19920101 and 性别="男"
```

1.2 C 语言的发展过程

计算机程序设计语言种类繁多，其中流行最广、影响最大之一就是 C 语言，程序设计语言此起彼伏，其中寿命最长的也是 C 语言。为什么呢？这是由 C 语言固有特点所决定的。因为它适合作为系统描述语言，既可用于编写系统软件，也可用于编写应用软件。它的发展过程如图 1.1 所示。

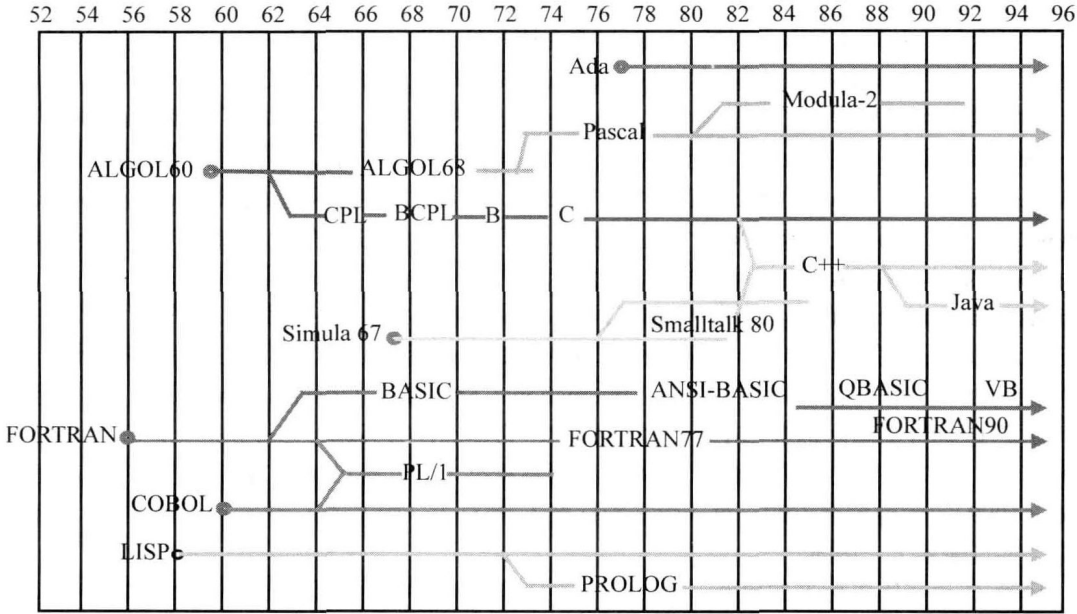


图 1.1 常见程序设计语言的发展过程

早期的系统软件主要是采用汇编语言编写的。从前述可知，汇编语言完全依赖于计算机硬件，用汇编语言编写的程序不仅可读性差，难于理解，不便交流与维护，更主要的是不可移植性。因此，为了提高程序的可读性、可维护性，实现程序的可移植性，自然会考虑到采用与计算机硬件无关的高级程序设计语言来编写程序。但是，在 C 语言出现之前，一般的高级程序设计语言难以实现汇编语言的某些功能（如对计算机硬件、内存地址的直接操作等）。软件开发开发者期望找到一种既

具有汇编语言和高级语言优点的程序设计语言。于是，C 语言就在这样的背景下诞生了。

C 语言的起源可以追溯到计算机的“古生代”时期，即 20 世纪 60 年代。C 语言是贝尔实验室由 Ken Thompson、Dennis Ritchie 及其同事在开发 UNIX 操作系统过程中的一个副产品。Thompson 独自编写了 UNIX 操作系统的最初版本，该操作系统运行于 DEC DPP-11 计算机上。与同时代的其他操作系统一样，UNIX 系统最初也是采用汇编语言编写的。众所周知，用汇编语言编写的程序往往难以调试和改进。当然，UNIX 也不例外。Thompson 意识到要更加有效地完成未来的 UNIX 操作系统的开发，于是设计了一种小型的 B 语言。Thompson 的 B 语言是在 BCPL (Basic Combined Programming Language) 语言的基础上开发出来的。BCPL 语言是 1967 年英国剑桥大学的 Martin Richard 推出来的，而 BCPL 语言的起源又可追溯到最早的程序设计语言 Algol60 (20 世纪 60 年代推出的一种高级程序设计语言)。

后来 Ritchie 也加入到 UNIX 项目组，并且着手采用 B 语言编写程序。1970 年，贝尔实验室为 UNIX 项目争取到一台 PDP-11 计算机。当 B 语言经过改进并且运行在 PDP-11 计算机上时，Thompson 就用 B 语言重新编写了部分 UNIX 程序代码。1971 年，Ritchie 开发了 B 语言的改进版，并命名为 NB (New B)。1972~1973 年间，Ritchie 对 B 语言的多次改进，越来越脱离了 B 语言，于是他决定将它更名为 C 语言。到了 1973 年，C 语言已经足够完善。Thompson 和 Ritchie 在开发中，出于考虑把 UNIX 移植到 PDP 系列之外的其他类型的计算机中使用，决定用 C 语言重新编写 UNIX 系统。C 语言的可移植性，在此得到了充分的体现。机器语言和汇编语言编写的程序都不具有可移植性，为 X86 开发的程序，不可能在 Vax、Alpha、SPARK、ARM 和 APPLE 计算机中运行，而 C 语言程序则可以运行于任何计算机中，只要该计算机中有 C 编译程序和相应的库函数。

时至今日，各种 UNIX (包括 Linux) 的内核和外设驱动程序仍然采用 C 语言作为最主要的编程工具，其中甚至还有不少继承自 Thompson 和 Ritchie 之手的源代码。

C 语言继续发展到 1978 年，由贝尔实验室正式发布的 C 语言版本不仅保留了 B 语言的简洁、精炼、接近硬件的优点，而且克服了 B 语言功能过于简单、有限的缺点。同时出版了由 B.W.Kerninhan 和 D.M.Ritchie (简称为 K&R) 合著的影响深远的《The C Programming Language》一书。该书虽未定义一个完整的 C 语言标准，但它介绍的 C 语言成为后来广泛采用的 C 语言版本的基础，该书介绍的 C 称为传统的 C。1982 年，美国国家标准协会 (American National Standards Institute, 简称 ANSI) 根据 C 语言问世以来的各种版本，进行了修改与扩充，制定了新的标准，并于 1983 年发布，称之为 ANSI C 或标准 C (1983)。由于程序设计是一项全球性的活动，1987 年国际标准化组织 (International Standards Organization, 简称 ISO)，建立了专门制定 C 标准的国际标准化小组，以 ANSI C 为基础，对其进行修订和扩充，并于 1989 年、1995 年和 1999 分别发布了标准 C (1989)，简称标准 C89；标准 C (1995)，简称标准 C95；标准 C (1999)，简称标准 C99。每一版本的标准 C 都对上一版本的 C 语法或函数库进行了修改或扩充，并且兼容上一版本。

尽管 C 语言版本标准的更新，使 C 语言的功能进一步增强，但不同软件商开发的 C 语言编译器，都是在遵从标准 C 的基础上，扩展带有自己特色的功能。那么，究竟采用哪种 C 语言编译器比较好呢？为了保证源程序可在任意 C 编译器下均可成功编译，那么，只有遵守一个原则，就是在编写 C 程序时只使用遵从 ANSI C 的语法的语句。

1.3 C 语言的主要特点

C 语言通常称为中级计算机语言。中级语言并没有贬义，不意味着它功能差、难以使用，或

者比 BASIC、Pascal 那样的高级语言原始，也不意味着它与汇编语言相似，会给使用者带来麻烦。C 语言之所以被称为中级语言，是因为它把高级语言的成分与汇编语言的功能结合起来了。其主要特点如下：

1. 程序设计结构化

虽然从严格的学术观点上看，C 语言是块结构（Block Structured）语言，但是它还是常被称为结构化语言。因为它具有结构化的流程控制语句（如 if...else、switch、for、do...while 和 while 语句），允许缩进格式书写程序。C 语言的主要结构成分是函数。在 C 语言中，函数是一种构件（程序块），是完成程序功能的基本构件。函数允许把一个较大（或较复杂）的任务，分解成若干个较小（或较简单）的子任务，对于每个子任务分别定义函数和编码，实现程序模块化。

2. 语言简洁、紧凑、灵活

C 语言只有 32 个关键字，9 种控制语句；书写格式自由。一行可以书写多条语句，一条语句也可书写在多行上。

3. 丰富的运算符

C 语言有 34 种运算符，把赋值、括号、强制类型转换作为运算符。灵活地使用各种运算符，可以实现其他语言难以达到的运算。

4. 丰富的数据类型

C 语言具有整型、实型、字符型等基本的数据类型，还为程序员提供了灵活丰富的构造类型，如指针型、结构体类型、共用体类型、枚举类型等数据类型。并且允许用户自定义自己的数据类型，以满足程序设计的需要。

5. 生成的目标代码质量高、运行速度快、可移植性好

到目前为止，任何其他高级语言编写的程序，编译连接后生成的可执行程序，无论是代码量的大小，还是执行速度、效率都无法与 C 语言编写的程序相媲美。

6. C 语言程序可以直接访问内存地址、直接操作计算机硬件设备

C 语言的指针运算，可以实现汇编语言（或机器语言）的大部分功能，对于要求时效性高的应用，如操作系统，实时控制系统等应用中，其他高级语言难以胜任。

7. C 语言难学

C 语言程序的书写格式自由性和指针运用的灵活性，相对于其他高级语言来说，初学者往往感到难于理解和困惑。但是只要掌握 C 语言的语法规则，夯实基础、多思考、多写程序、多上机，善于归纳总结，这点困难是可以克服的。

1.4 C 语言程序的结构

为了了解 C 语言程序的结构，先来看一看以下 3 个简单的示例程序。这 3 个例子由简单到复杂，展现出 C 语言程序的组成结构。尽管这 3 个例子中涉及的有关知识尚不熟悉，但从中可以了解到一个 C 语言程序的基本结构成分与书写格式。

【例 1-1】 编写程序，在屏幕上显示“同学们好！”字样。

```
/*ex1_1.c 这是第一个 C 程序，它的功能是显示“同学们好！”*/
#include "stdio.h"
void main()
{
    printf("同学们好！\n");
}
```

程序运行结果:

同学们好!

程序说明:

(1) 第一行的“/*ex1_1.c”这是第一个 C 程序,它的功能是显示“同学们好!”*/”,其中符号“/*”与“*/”括起来内容是注释(即注释内容是:ex1_1.c,这是第一个 C 程序,它的功能是显示“同学们好!”)。“/*”与“*/”是成对出现的,“/*”表示注释的开始标志,“*/”表示注释的结束标志。注释可以出现在中的任何位置,注释的内容既可以书写在一行上,也可以分多行书写。

(2) 第二行的#include 叫做文件包含命令,其作用是把一对双引号(“”)内指定名称的文件加入到该程序中,成为该程序的一部分。这里被包含的文件是由系统提供的,其扩展名为.h,称作头(head)文件或首文件。C 语言的头文件是以扩展名为.h 的各种库函数。所谓“库函数”是指系统为我们事先编写好的、能够实现一定功能的一系列完整的函数(如求某个数 X 的 N 次幂 pow(X,N),某个弧度 X 的正弦值 sin(X)等)。系统提供的这些函数,我们可以直接利用包含命令,把它加入到自己的程序中,然后就可以直接使用它来计算求值了。stdio.h 是标准输入输出函数库文件,其中包括标准的输入输出函数,为了在我们的程序中实现屏幕显示文字“同学们好!”,可以直接采用标准函数 printf 来显示。此处的#号是一个特别的符号,称作编译预处理命令标志。也就是说,凡是在行首出现#号,紧跟其后的必须是编译预处理命令。关于编译预处理命令将在第 4 章予以详细讨论。

(3) main()函数中的内容必须写在一对花括号内。

【例 1-2】接收键盘输入一个角度值,在屏幕上显示其正弦值。

```
/* ex1_2.c 这是第二个 C 程序,它的功能是显示角度为 X 的正弦值*/
#include "stdio.h" /*包含标准输入输出函数文件 */
#include "math.h" /*包含数学函数文件 */
#define PI 3.1415926 /*定义符号常量 PI 为 π 的近似值*/
/*定义函数 PRINTSinX 在屏幕上显示角度为 X 的正弦值 */
void PRINTSinX(int X)
{ double Y; /*定义变量 Y*/
  Y=X%360; /*除去周期值之后的剩余量存放到 Y 中*/
  Y=2*PI*Y/360; /*把 Y 的值转换为弧度值存放到 Y 中*/
  printf("Sin(%d)=%lf \n",X,sin(Y));
/*调用函数 printf 显示由函数 sin 求出的角度为 X 的正弦值*/
}

void main( )
{int X; /*定义整型变量 X*/
printf("Input value of X,please!\n"); /*屏幕提示,给变量 X 输入一个数值*/
scanf("%d",&X); /*接收键盘输入的数值存放到变量 X 中*/
PRINTSinX(X); /*调用前面自定义的函数 PRINTSinX 显示角度为 X 的正弦值*/
}
```

程序运行结果如图 1.2 所示。

程序说明:

- (1) 该程序包含了两个函数库的头文件 stdio.h 和 math.h。
- (2) 该程序由两个自定义函数组成的: main 函数和 PRINTSinX 函数。
- (3) 标准输出函数 printf 用于显示信息。
- (4) 标准输入函数 scanf 用于接收键盘输入数据。
- (5) 数学函数 sin(Y)用于求出弧度值为 Y 的正弦值。
- (6) 该程序中定义了一个符号常量 PI,它代表数值: 3.1415926。

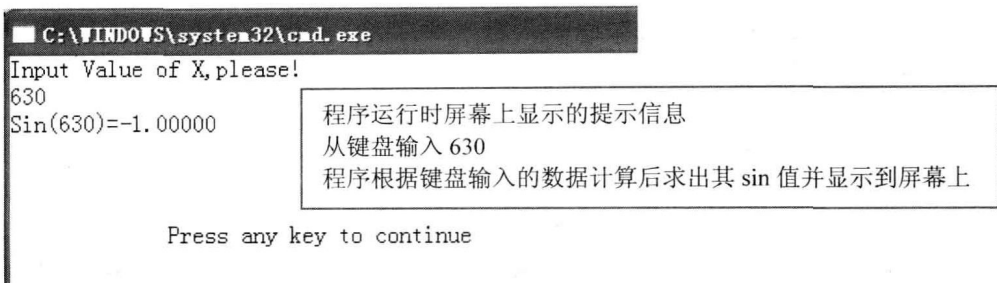


图 1.2 例 1-2 程序运行结果

本程序的例子中，用到了 4 个函数，其中 printf、scanf、sin 是由系统提供的，PRINTSinX 是编写程序者自己定义的函数，也称作自定义函数。

【例 1-3】 编写程序接收键盘输入三个整数，求出其中最大值和最小值，并输出在屏幕上。

```

/* ex1_3.c 这是第三个 C 程序，它的功能是显示键盘输入的三个整数中的最大值和最小值 */
main()
{int X,Y,Z,max,min;          /*定义整型变量 X,Y,Z, ,max, min */
 int Max(int A,int B);      /*声明自定义函数 Max */
 int Min(int,int);         /*声明自定义函数 Min */
 /*屏幕上显示 "Input Nubers into X,Y,Z,please!" */
 printf("Input Numbers into X,Y,Z,please!\n");
 scanf("%d,%d,%d",&X,&Y,&Z); /*从键盘输入数值给整型变量 X,Y,Z*/
 max=Max(X,Max(Y,Z));       /*求 X,Y,Z 中的最大值存放到变量 max 中*/
 min=Min(X,Y);              /*求 X,Y 中的最小值存放到变量 min 中*/
 min=Min(min,Z);           /*求 min,Z 中的最小值存放到变量 min 中*/
 printf("max=%d,min=%d\n",max,min); /*在屏幕上显示 X,Y,Z, 中的最大值和最小值*/
}

int Max(int A,int B)        /*求 A、B, 中的最大值的函数定义*/
{if(A>B) return A;        /*若 A>B,A 的值返回给调用者*/
 else return B;          /*否则 B 的值返回给调用者*/
}

int Min(int A,int B)       /*求 A、B, 中的最小值的函数定义*/
{if(A<B) return A;       /*若 A<B,A 的值返回给调用者*/
 else return B;          /*否则 B 的值返回给调用者*/
}

```

程序运行结果如图 1.3 所示。

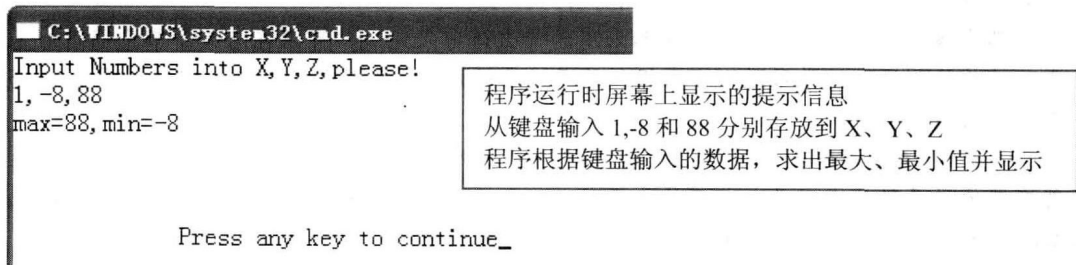


图 1.3 例 1-3 程序运行结果

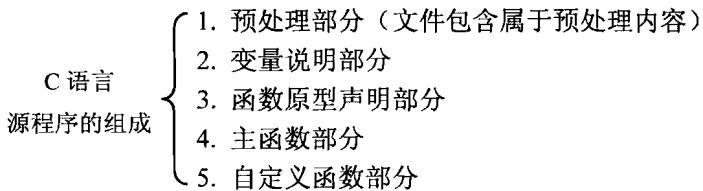
程序说明：

- (1) 该程序由 3 个函数组成：main 函数、Max 函数和 Min 函数。
- (2) 该程序调用了系统提供的：

- 标准输出函数 printf 用于显示信息。
- 标准输入函数 scanf 用于接收键盘输入数据。
- (3) 自定义的函数:
 - Max 用于求两个数的最大值。
 - Min 用于求两个数的最小值。

注意: 该程序与前面两个例子不同之处, 在于它虽然也调用标准输入输出函数 scanf、printf, 但是程序的开头没有#include "stdio.h"编译预处理命令; main之前没有void关键字。程序为什么也可以正确运行呢? 这只是在 Turbo C 环境中可以这么写, 其他一些 C 开发环境还是要像例 1-2 那样的写法, 以保证程序的可移植性。

上述的 3 个 C 语言程序范例虽然功能简单, 但却说明了 C 语言源程序的基本组成部分, 概括地说, 一个 C 语言源程序可由如下 5 个部分组合而成:



注意:

(1) 并非所有的 C 语言源程序都必须包含上述的 5 个部分, 一个最简单的 C 语言程序可以只有包含文件的部分和主函数部分。

(2) 每个 C 语言源程序都必须有且只能有一个主函数, 主函数的组成形式如下所示:

```
main()
{
    变量说明部分
    程序语句部分
}
```

(3) 每个 C 语言源程序可以有 0 个或多个自定义的**非主函数**, 自定义非主函数的形式与主函数形式相同, 只是它的名称不能是 main。

```
函数名([参数列表])
{
    变量说明部分
    程序语句部分
}
```

(4) 每个 C 语言源程序的语句必须用分号“;”结束。

对于其中各部分的作用、使用方法和采用的什么语句来完成, 可以在后续章节中通过对基本表达式、结构控制语句的学习进一步掌握, 并通过了解模块化设计等方面的内容, 来掌握 C 语言程序设计的思想。

(5) 当 C 语言源程序由多个函数组成时, 主函数 (main) 可以定义在程序中的任何位置 (不能在其他的函数体内定义), 不论主函数 (main) 处于源程序的任何位置, 程序总是从主函数 (main) 开始执行, 且总是在主函数 (main) 中结束执行的。

1.5 C 语言程序的执行

用高级程序设计语言编写的程序称为源程序 (Source Program), 实际上计算机本身并不能直接理解这样的语言, 必须将源程序翻译成机器语言程序, 计算机才能理解。将源程序翻译成机器

语言程序的方法，目前以下有3种。

其一是解释型，如 Basic 语言。解释型系统中有个解释器，类似于人类自然语言的口头翻译员。它的职责是分析语法和执行语句的功能实现。解释型又可分为交互（对话）式和非交互式两种。交互式翻译过程是，每当向计算机输入完一条语句时，解释器就即时分析、翻译，如果语法正确，就立即执行；如果语句有错，立即提示错误，接着等待修改或输入下一条语句，当下一条语句输入完毕，重复上述过程，直到整个程序结束为止。非交互式的特点是，这种系统启用解释器，对事先编写的源程序从头到尾一条语句一条语句地逐条分析，如果有错误，即时报错，如果语句正确，立即执行，接着分析、执行下一条语句，直到整个程序结束为止。

其二是编译型，如 Pascal、C 等语言。编译型系统有一个编译器，类似于人类自然语言的书面翻译员。对事先编写的源程序从头到尾一条语句一条语句地逐条分析，如果发现错误，就记载下来，分析到程序结束为止；在整个程序分析完成后，如果有错误，就报错；如果未发现错误，就生成目标代码。将源程序翻译成机器语言程序的过程称为编译，编译的结果是得到源程序的目标代码（Object Code）；最后还要将目标代码与系统提供的函数和自定义的函数链接起来，就可得到一个完整的程序文件，这个程序文件称为可执行程序或可执行文件，也就是计算机可以直接运行的程序文件。

其三是解释编译型。这种系统既有编译器也有解释器。如 C#、Java 语言。这种系统的特点是：首先由编译器把源程序翻译成一种称之为中间语言代码（也称之为跨平台语言代码），以后由负责执行这种中间语言代码的解释器来解释并执行它。

1.5.1 源程序翻译

C 语言源程序的后缀名为.c。由前述可知，它是不能直接在计算机上运行的，必须通过翻译成目标代码，再将目标代码链接成可加载模块（可执行文件），才能在计算机上运行。这种把源程序翻译成目标代码的程序被称之为编译器或翻译器。适合 C 语言的编译器不止一种，不同系列的机器、不同的操作系统可能会有一种或多种不同的编译器。C 语言源程序的翻译过程如图 1.4 所示，它由词法分析器、语法分析器和代码生成器 3 部分组成。

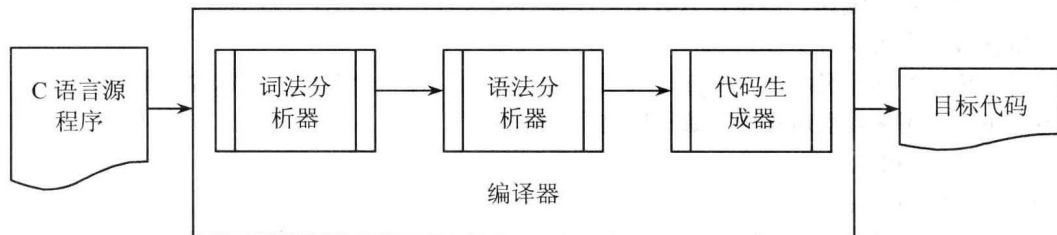


图 1.4 源程序的翻译过程

1. 词法分析器（Lexical Analyzer）

词法分析器主要是对源程序进行词法分析，它是按单个字符的方式阅读源程序，并且识别出哪些符号的组合可以代表独立的单元，并根据它们是数值、单词（标识符）或运算符等，将这些单元分类。词法分析器将词法分析结果保存在一个结构单元里，这个结构单元称为标记（Token），并将这个标记交给语法分析器，词法分析会忽略源程序中的所有注释。

2. 语法分析器（Parser）

在没有词法错误的情况下，语法分析器直接对标记进行分析，并识别每个单元成分所扮演的角色（也称为语义分析）。这些语法规则也就是程序设计语言的语法规则。