



21世纪高职高专规划教材
计算机专业基础系列

C语言程序设计 项目化教程

王彩霞 主 编
任 岚 副主编



清华大学出版社



21世纪高职高专规划教材
计算机专业基础系列

C语言程序设计

项目化教程

王彩霞 主 编
任 岚 副主编

清华大学出版社
北京

内 容 简 介

本书以“小学生四则运算测试系统”和“简易学生成绩管理系统”这两个项目为主线,既系统讲解了传统 C 语言程序设计的有关知识,又在讲授有关知识的同时,用学到的知识实现项目设计功能,随着知识讲解的深入,逐步完善有关功能。在此过程中我们可以体会到知识点的对比,使读者知道各类知识点的应用范围,并在项目开发中选择最适宜的算法和数据结构。

即使读者没有任何编程语言基础,都可以选择本书从零开始。本书以通俗的语言、深入浅出的讲解方式,以项目为牵引,带领读者一步一个台阶,逐步走向 C 语言深处,培养形成了强大的再学习能力,为今后很轻松地学习其他的程序设计语言打下基础。

本书是本科院校、高职高专院校以及各培训机构的首选教材,由于其通俗的叙述和到位的分析,本书更可以作为广大程序设计爱好者的自学教材。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

C 语言程序设计项目化教程/王彩霞主编. —北京:清华大学出版社,2012.12

(21 世纪高职高专规划教材. 计算机专业基础系列)

ISBN 978-7-302-29189-3

I. ①C… II. ①王… III. ①C 语言—程序设计—高等职业教育—教材 IV. ①TP312

中国版本图书馆 CIP 数据核字(2012)第 142866 号

责任编辑:孟毅新

封面设计:常雪影

责任校对:袁芳

责任印制:何芊

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社总机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质量反馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

课件下载: <http://www.tup.com.cn>, 010-62795764

印 刷 者:北京富博印刷有限公司

装 订 者:北京市密云县京文制本装订厂

经 销:全国新华书店

开 本:185mm×260mm

印 张:25.75

字 数:590 千字

版 次:2012 年 12 月第 1 版

印 次:2012 年 12 月第 1 次印刷

印 数:1~3000

定 价:48.00 元

产品编号:045937-01

前言

C 语言程序设计项目化教程

真而。同时用空词实其志善。中。金照册册丁。成虽只书书融图，余照重若材殊的册以不流内填固。学殊的言部。个。一。书。希翻册成随关册言袋成困善学册。大澳舞眼目取册突新百翻本。要需就册页册册，翻求善表随能登登掌同册音好伙，书民。目取册大澳些那鬼。材其的富富流。可册册的导册人决下善到甘，册是徐画又相同。目取册面合音又余照的册志愿册计本一。可石自丁求当善多，不帮背个益消

大多数计算机程序设计工作的从业者几乎都是从学习 C 语言开始的，而我国大多数理工院校几乎都把 C 语言作为计算机相关专业的入门课程。所以 C 语言教材在专业教学中就显得尤其重要。

笔者从事计算机语言教学二十余年，先后教授过十多门语言课程，在教学过程中笔者发现这样一个现象：某些学生 C 语言学得比较扎实，那么这部分学生在学习其他计算机编程课程时就显得非常轻松，理解和掌握得都比较快。相反，C 语言学得不好的那部分学生，在学习其他计算机课程时就非常吃力，从而导致他们对所学的课程产生恐惧和厌恶感，最终完全放弃有关课程的学习。

我曾经多次和学生们座谈，了解他们的想法，特别是那些学习吃力的学生，他们这样对我说：“老师讲课的时候听得很明白，可是一旦自己动手就不知道如何下手。”归根结底是学生对所学的知识不理解或者不会运用，并且在有疑难问题的时候又得不到及时的指导。

我也曾经让学生多看书，而学生反映某些书的表述太过专业和枯燥，有些内容看不懂，从而失去了阅读教材的兴趣。

作为一名老师，不可能总是和学生们在一起为他们解答疑难问题。再说一名老师要面对数十个甚至上百个学生，这样分配给每个学生的机会就非常少。此时唯一能做的是给学生们选择一本好的书来陪伴他们，使学生随时可以从书中找到问题的症结所在。

虽然目前市场上有关 C 语言的书籍和教材很多，但大多数书籍的语言表述太过专业，且复杂的例题和案例没有思路分析。这对于有编程语言基础的读者来说并不困难，但对于还没有入门的读者来说显得比较困难。所以特别需要一本通俗易懂、带思路分析和综合项目设计的书，这会对学生的学习起到事半功倍的作用。

目前，各大院校几乎都在鼓励和开展项目化教学。个人认为，项目化教学应当根据不同的课程、课程开设的不同时期以及课程面向的教学对象来区分进行，不能用同一个标准来要求所有的课程。

“C 语言程序设计”这门课程是理论与实践并重的课程，并且大多数院校都在一年级开设本课程。也就是说，“C 语言程序设计”这门课程是学生接触的第一门程序语言课程，学生们一点理论基础没有，如果此时盲目地开展项目化教学，势必会造成“头疼医头，脚疼医脚”的局面，同时由于学生们没有扎实的理论基础，以后遇到新的项目，他们就不知道如何下手。但是，我们也不能一味地讲解理论，而忽略项目应用，那我们的教学就成了纸上谈兵。

我们常说“授人以鱼不如授人以渔”，在项目化教学中，项目就是“鱼”，项目的种类很多，任何一本书不可能讲完所有的项目，而“渔”就是项目的开发方法以及相关的理论。如何把握“鱼”与“渔”的关系，选择什么样的项目，就成了项目化教学和相关教材的一个关键因素。

以往的教材注重理论，例题往往只是为了佐证理论，并不考虑其实际应用如何。而真实的项目规模庞大，初学者因为没有相关的知识储备，在一个 C 语言的教学周期内完不成那些庞大的项目。另外，对没有任何编程经验的读者来说，他们迫切地需要一本既有严谨的理论又有合适的项目，同时又通俗易懂，让读者在无人指导的情况下能读懂的教材。

在这个背景下，笔者产生了自己写一本书的想法。

本书设计了“小学生四则运算测试系统”和“简易学生成绩管理系统”这两个项目。考虑到知识的关联和学生的耐力，本教材在讲完循环结构(前 4 章)后，完成了“小学生四则运算测试系统”项目，这样既可以把前 4 章的知识串联起来，又可以让学生获取满足感和成就感。

学习完循环以后，学生的知识结构会上一个台阶，所以从数组开始，本书在讲授有关理论的同时，着手完成“简易学生成绩管理系统”这个项目。有“小学生四则运算测试系统”项目做铺垫，学生们自己很容易搭起“简易学生成绩管理系统”项目的框架。后面随着数组、函数、指针、链表以及文件的讲述，学生需要做的是用不同的数据结构去存储“简易学生成绩管理系统”中的原始数据，更改项目的细节部分，从而体会不同知识点的对比，直至讲完文件并实现学生信息的永久保存。当整个项目完成后，学生们从中不仅学习了理论，同时也知道这些理论都用在何处，例如如何开发一个项目等。

C 语言程序设计这门课程是一门实践忠于理论的课程，“看懂教材，听懂老师的讲课”并不是目的，大家一定要亲自动手，在计算机上实践所有的例题、习题和项目，尝试不同的实现方法。只有在计算机上反复使用，有关的理论才能变成自己的东西，成为大家活学活用的工具。

在本书的编写过程中，得到了众多领导和同事的帮助和指导，在此对他们表示深深的祝福和由衷的感谢。

由于水平有限，本书肯定有不足之处，请各位读者指正，在此一并表示感谢。

王彩霞

2012 年 11 月于青岛

| | | |
|---------------|--------------------------------|-----------|
| 2.2.4 | 赋值运算符和赋值表达式 | 42 |
| 2.2.5 | 自增(++)和自减(--)运算符 | 43 |
| 2.2.6 | 条件运算符和条件表达式 | 44 |
| 2.2.7 | 逗号运算符和逗号表达式 | 45 |
| 2.2.8 | 混合数据运算以及类型转换 | 45 |
| 2.3 | 整型数据在内存的表示形式以及运算溢出 | 46 |
| 2.3.1 | 整型数据的补码表示方法 | 46 |
| 2.3.2 | 数据的溢出以及舍入误差 | 47 |
| 2.4 | 附加——格式化输入输出 | 48 |
| 2.4.1 | 在 printf 语句中规定输出位数 | 48 |
| 2.4.2 | scanf()函数的格式化输入 | 50 |
| 2.5 | 四则运算测试系统——以不同的数据类型接受用户输入 | 53 |
| 习题与上机实验 | | 55 |
| 第 3 章 | 选择(分支)结构程序设计 | 57 |
| 3.1 | 顺序程序结构 | 57 |
| 3.2 | 用 if 语句进行选择 | 59 |
| 3.2.1 | if 语句的 3 种使用形式 | 59 |
| 3.2.2 | if 语句的嵌套使用 | 65 |
| 3.3 | 用 switch 语句进行选择 | 67 |
| 3.3.1 | switch 语句的使用形式 | 67 |
| 3.3.2 | switch 语句的使用举例 | 68 |
| 3.3.3 | 使用 switch 语句的注意事项 | 70 |
| 3.3.4 | 选择结构综合举例 | 70 |
| 3.4 | 小学生四则运算测试系统 | 73 |
| 3.4.1 | 对用户输入的判断 | 73 |
| 3.4.2 | 随机试题的生成 | 74 |
| 习题与上机实验 | | 76 |
| 第 4 章 | 循环结构程序设计 | 77 |
| 4.1 | 循环引例 | 77 |
| 4.1.1 | 问题分析 | 77 |
| 4.1.2 | 循环体以及循环条件 | 79 |
| 4.2 | 构成循环的语句 | 79 |
| 4.2.1 | 用 while 语句实现循环 | 79 |
| 4.2.2 | 用 do...while 语句实现循环 | 83 |
| 4.2.3 | 用 for 语句实现循环 | 86 |
| 4.2.4 | 3 种循环形式的相互转换 | 90 |

| | | |
|--------------|---------------------|------------|
| 4.3 | 循环的嵌套 | 93 |
| 4.3.1 | 嵌套的应用分析 | 94 |
| 4.3.2 | 循环以及循环嵌套注意事项 | 97 |
| 4.4 | 其他与循环有关的语句 | 99 |
| 4.4.1 | break 和 continue 语句 | 99 |
| 4.4.2 | 用 goto 构成循环 | 101 |
| 4.5 | 循环综合举例 | 101 |
| 4.6 | 完善小学生四则运算测试系统 | 104 |
| | 习题与上机实验 | 107 |
| 第 5 章 | 数 组 | 108 |
| 5.1 | 一维数组 | 108 |
| 5.1.1 | 问题提出与分析 | 108 |
| 5.1.2 | 一维数组的定义与初始化 | 109 |
| 5.1.3 | 一维数组的使用 | 110 |
| 5.1.4 | 一维数组元素的排序 | 114 |
| 5.2 | 二维数组 | 120 |
| 5.2.1 | 二维数组的定义与初始化 | 120 |
| 5.2.2 | 二维数组的应用 | 121 |
| 5.2.3 | 二维数组与一维数组的关系 | 127 |
| 5.3 | 字符数组与字符串 | 128 |
| 5.3.1 | 逐元素使用字符数组 | 128 |
| 5.3.2 | 以字符串的形式使用字符数组 | 130 |
| 5.3.3 | 常用的字符串处理函数 | 132 |
| 5.3.4 | 字符数组的应用 | 133 |
| 5.4 | 数组综合举例 | 136 |
| 5.4.1 | 问题提出 | 136 |
| 5.4.2 | 问题分析 | 136 |
| 5.4.3 | 代码实现 | 136 |
| 5.5 | 项目案例——简易学生成绩管理系统 | 138 |
| 5.5.1 | 简易学生成绩管理系统功能分析 | 139 |
| 5.5.2 | 具体功能的实现以及说明 | 142 |
| | 习题与上机实验 | 145 |
| 第 6 章 | 自定义函数 | 147 |
| 6.1 | 函数概述 | 147 |
| 6.1.1 | 什么是函数 | 148 |
| 6.1.2 | 函数的分类 | 150 |

| | | |
|------------|---------------------|------------|
| 6.2 | 函数的定义与调用 | 151 |
| 6.2.1 | 函数的定义及结构分析 | 151 |
| 6.2.2 | 函数的调用 | 155 |
| 6.2.3 | 函数的原型声明 | 156 |
| 6.3 | 函数的参数传递 | 160 |
| 6.3.1 | 一般变量做函数参数 | 160 |
| 6.3.2 | 数组名做函数参数 | 161 |
| 6.4 | 函数的嵌套调用和递归调用 | 168 |
| 6.4.1 | 函数的嵌套调用 | 168 |
| 6.4.2 | 函数的递归调用 | 169 |
| 6.5 | 函数变量以及变量共享 | 172 |
| 6.5.1 | 局部变量与全局变量 | 172 |
| 6.5.2 | 静态变量与动态变量 | 175 |
| 6.5.3 | 寄存器变量 | 177 |
| 6.6 | 内(外)部变量及内(外)部函数 | 178 |
| 6.6.1 | 内部变量与外部变量 | 178 |
| 6.6.2 | 内部函数与外部函数 | 180 |
| 6.7 | 多文件程序的编译和连接 | 181 |
| 6.8 | 用函数调用改写“简易学生成绩管理系统” | 183 |
| 6.8.1 | 改写后的主函数架构 | 183 |
| 6.8.2 | 有关函数的定义和部分代码 | 185 |
| | 习题与上机实验 | 197 |
| 第7章 | 结构体、共用体和枚举类型 | 199 |
| 7.1 | 结构体类型 | 200 |
| 7.1.1 | 结构体类型的定义 | 200 |
| 7.1.2 | 结构体变量的使用 | 202 |
| 7.2 | 结构体数组 | 207 |
| 7.2.1 | 结构体数组的定义以及使用 | 207 |
| 7.2.2 | 结构体变量的传递 | 209 |
| 7.2.3 | 结构体数组的综合应用 | 211 |
| 7.3 | 其他自定义类型的应用 | 213 |
| 7.3.1 | 共用体 | 213 |
| 7.3.2 | 枚举类型 | 218 |
| 7.3.3 | 用 typedef 定义类型符号 | 222 |
| 7.4 | 用结构体数组改写学生成绩管理系统 | 223 |
| | 习题与上机实验 | 225 |

| | |
|------------------------------------|-----|
| 第 8 章 指针 | 226 |
| 8.1 指针概述 | 226 |
| 8.1.1 变量的地址与指针..... | 227 |
| 8.1.2 指针变量的定义和初始化..... | 228 |
| 8.2 指针变量的使用 | 229 |
| 8.2.1 取地址运算与指向运算..... | 230 |
| 8.2.2 指向普通变量的指针变量..... | 231 |
| 8.2.3 指向结构体变量的指针变量..... | 232 |
| 8.2.4 指针在函数间传递..... | 235 |
| 8.3 数组与指针变量 | 238 |
| 8.3.1 通过指针变量来使用一维数组..... | 239 |
| 8.3.2 数组的指针变量与数组名的关系..... | 243 |
| 8.3.3 参数传递时的数组名及其指针变量..... | 245 |
| 8.4 字符串与指针变量 | 247 |
| 8.4.1 字符串的特点..... | 247 |
| 8.4.2 字符串的指针变量..... | 249 |
| 8.4.3 字符串指针变量做函数参数..... | 251 |
| 8.5 函数与指针 | 253 |
| 8.5.1 返回指针的函数..... | 253 |
| 8.5.2 指向函数的指针..... | 255 |
| 8.6 指针的其他使用 | 258 |
| 8.6.1 多维数组的指针..... | 258 |
| 8.6.2 指针数组..... | 263 |
| 8.6.3 指针变量的指针..... | 267 |
| 8.6.4 void 类型的指针变量 | 268 |
| 8.7 指针在“简易学生成绩管理系统”中的初步应用 | 269 |
| 习题与上机实验..... | 273 |
| 第 9 章 链表 | 276 |
| 9.1 链表的准备知识 | 276 |
| 9.1.1 与链表有关的概念..... | 277 |
| 9.1.2 与链表有关的函数(与动态内存分配有关的函数) | 279 |
| 9.2 单向链表的有关操作 | 281 |
| 9.2.1 无头结点链表的建立与输出..... | 281 |
| 9.2.2 带头结点链表的建立与输出..... | 287 |
| 9.2.3 链表结点的删除和插入..... | 290 |
| 9.3 应用举例 | 298 |

| | | |
|---------------|--------------------------|------------|
| 9.4 | 用链表实现“简易学生成绩管理系统”项目的所有功能 | 302 |
| 9.4.1 | 准备工作以及主函数架构 | 302 |
| 9.4.2 | “输出成绩”功能 | 304 |
| 9.4.3 | “录入成绩”功能 | 306 |
| 9.4.4 | “查询修改”功能 | 309 |
| | 习题与上机实验 | 313 |
| 第 10 章 | 文件 | 315 |
| 10.1 | 文件指针以及文件的打开和关闭 | 315 |
| 10.1.1 | 文件概述 | 315 |
| 10.1.2 | 文件类型的指针 | 316 |
| 10.1.3 | 文件的打开以及关闭 | 318 |
| 10.1.4 | 文件读写的准备知识 | 320 |
| 10.2 | 文件的顺序读写 | 321 |
| 10.2.1 | 以字符为单位读写文件 | 321 |
| 10.2.2 | 以字符串为单位读写文件 | 326 |
| 10.2.3 | 文件的格式化读写 | 329 |
| 10.3 | 二进制文件的顺序读写 | 333 |
| 10.3.1 | 写二进制文件 | 333 |
| 10.3.2 | 读二进制文件 | 335 |
| 10.4 | 文件的随机读写 | 337 |
| 10.4.1 | 文件读写位置的有关函数 | 338 |
| 10.4.2 | 随机读写举例 | 339 |
| 10.5 | 实现“简易学生成绩管理系统”的文件操作功能 | 341 |
| | 习题与上机实验 | 345 |
| 第 11 章 | 预处理命令 | 347 |
| 11.1 | 概述 | 347 |
| 11.2 | 宏定义 | 347 |
| 11.2.1 | 无参宏定义 | 348 |
| 11.2.2 | 带参宏定义 | 350 |
| 11.3 | 文件包含预处理 | 354 |
| 11.4 | 条件编译 | 356 |
| | 习题与上机实验 | 360 |
| 第 12 章 | 位运算 | 361 |
| 12.1 | 数值在计算机中的表示 | 361 |
| 12.1.1 | 数值在计算机中的存储方式 | 361 |

| | | |
|---------------|--------------------------|------------|
| 12.1.2 | 数据的表示方式 | 362 |
| 12.2 | 位运算符和位运算 | 363 |
| 12.2.1 | “按位与”运算 | 364 |
| 12.2.2 | “按位或”运算 | 366 |
| 12.2.3 | “按位异或”运算 | 366 |
| 12.2.4 | “求反”运算 | 367 |
| 12.2.5 | “左移”运算 | 368 |
| 12.2.6 | “右移”运算 | 368 |
| 12.3 | 位运算综合举例 | 369 |
| 12.4 | 位段 | 372 |
| | 习题与上机实验 | 374 |
| 第 13 章 | 算法简介 | 375 |
| 13.1 | 算法概述 | 375 |
| 13.1.1 | 什么是算法 | 376 |
| 13.1.2 | 算法的特点 | 376 |
| 13.1.3 | 衡量算法的标准 | 377 |
| 13.2 | 表示算法的方法 | 377 |
| 13.2.1 | 用自然语言表示算法 | 377 |
| 13.2.2 | 用传统流程图表示算法 | 377 |
| 13.2.3 | 用 N-S 流程图表示算法 | 378 |
| 13.2.4 | 用伪代码表示算法 | 379 |
| 13.2.5 | 用计算机语言表示算法 | 380 |
| 13.3 | 常用算法分析 | 381 |
| 13.3.1 | 常用基本算法 | 381 |
| 13.3.2 | 常用算法应用举例 | 382 |
| 附录 A | 常用字符与 ASCII 代码对照表 | 386 |
| 附录 B | 常用库函数 | 388 |
| | 参考文献 | 397 |

第1章

C语言概述

本章要点

- VC++ 6.0 开发平台的使用;
- C 程序文件的编辑、连接和运行;
- 程序的纠错;
- 简单的输入输出。

当今社会,小到家庭生活,大到国防科技卫星上天,几乎都离不开计算机的使用。无论计算机的功能多么强大,其实质都是计算机在执行人们事先预置的命令——人类是计算机的制造者和指挥者。反过来,如果计算机所执行的各种命令逻辑严密,功能广泛准确,那么计算机就可以更好地为人类服务。

要让计算机更好地为人类服务,就必须有一部分人能够透彻地了解计算机,掌握计算机的工作原理,熟悉计算机的语言,从而开发出许多满足行业需求,功能更加完善的计算机程序。如果有兴趣做其中的一员,那最好从学习 C 语言开始,而学习 C 语言也最好从本书开始。

本书将带领大家一步一步地走向 C 语言的中心地带。只要有足够的毅力和耐心,跟随本书踏踏实实地走完全程,就会发现,你已经拿到了开启计算机编程语言这个大门的钥匙,再学习其他的编程语言就变得非常简单。就好比一个人,他系统地学习了“二胡”的演奏方法,那么他就可以非常容易地学习演奏“小提琴”、“柳琴”、“琵琶”、“古筝”等乐器。

1.1 计算机语言的发展简介

1.1.1 计算机语言的种类以及发展阶段

自从第一台计算机在 1946 年诞生以来,计算机语言的发展大体可归纳为以下 3 个阶段。

1. 机器语言阶段

由于计算机硬件的工作原理决定了计算机只能认识 0 和 1 这两种符号,所以无论多么复杂的数据最终必须转化成 0 和 1 组成的代码才能被计算机识别。

计算机的硬件当中有一个名叫“中央处理器”的部件,也就是人们平常所说的 CPU,它是计算机的指挥中心,类似于人的大脑。不同系列的 CPU 对“0”和“1”的编码规则有不同的规定,这些有规则的二进制编码组合在一起,就是该系列 CPU 的指令系统——“机器语言”。不同系列的 CPU 所对应的“机器语言”也不尽相同,所以说机器语言是面向机器的。

计算机诞生之初,科学家们只能用 0 和 1 按逻辑编写程序供计算机识别和运行,这种程序就是机器语言程序。最可怕的是,如果换成一台 CPU 系列不同的计算机,同一个任务需要重新修改程序,所以机器语言的可移植性很差。可以想象,这种工作是多么可怕,那个时候的计算机以及有关工作就是处在高山之巅,普通人只能仰而望之!

2. 汇编语言阶段

用机器语言编写程序绝对不是一件幸福的事情,为了减少编程的痛苦,计算机科学家们发明了一些助记符号——帮助记忆的符号,即用英文单词或英文单词的缩写这些比较容易理解的符号来代替那些枯燥的二进制操作码。比如用 ADD 代表“加”运算,用 SUB 代表“减”运算等。这样一来,人们很容易读懂并理解程序在干什么,纠错及维护都变得比以前方便多了。

这种用助记符进行程序设计的语言就称为“汇编语言”。但这些用汇编语言编写的程序,计算机硬件是无法识别的,必须有一些特殊的程序,这些特殊程序的作用就是把用汇编语言编写的程序转换成机器语言,这样计算机硬件系统才能识别并运行它们,这个阶段叫汇编语言阶段。汇编语言和机器语言一样也是面向机器的,其可移植性也是很差的。

3. 高级语言阶段

尽管汇编语言比机器语言有了很大的进步,但汇编语言和机器语言是一一对一的,所以其编程效率依然很低。

“有需求就会有发展!”,这句话是非常经典的。为了解决汇编语言的不足,科学家们一直在不停地探索和研究,试图找到一种语言,它既能接近于数学语言和人的自然语言,同时又不依赖于计算机硬件,编出的程序能在所有机器上运行。科学家们的努力没有白费,从 1954 年开始至今,陆陆续续诞生了几百种满足上述条件的语言,使用比较广泛且有影响力的有 FORTRAN、ALGOL、COBOL、BASIC、LISP、SNOBOL、PL/1、PASCAL、C、PROLOG、Ada、C++、VB、Delphi、Java、C# 等。这些语言的特点是面向编程者,使人们易于接受和理解,且一条语句就可以对应多条机器语言语句,大大提高了编程效率,这个阶段就是高级语言阶段。

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这几个发展过程。相应地,软件的开发也从最初的个体手工作坊式的封闭生产,发展为产业化、流水线式的工业化生产——这就决定了软件开发产业需要大批的从业者,这也是人们学习计算机编程语言的动力之一。

前面讲过,不管多么先进的计算机,其硬件系统只能识别由 0 和 1 有机组成的指令系统,而用高级语言编写的程序显然计算机是无法识别的。所以必须把高级语言程序翻译成机器语言程序,计算机才能识别和运行。而担任这项翻译工作的是一个庞大的“程序团

队”，而这个程序团队就是人们常说的“编译系统”。可以想象：每推出一种高级语言，就伴随着一个编译系统的诞生。

1.1.2 C语言的诞生以及发展过程

从上面的叙述中大家可以想到，同时也可以看到，将要学习并且为之付出努力的C语言肯定也是高级语言的一种。各位读者肯定也迫切地想知道，C语言是如何来到这个世界上的。

2011年10月5日，苹果之父乔布斯离世，人们正处在深深的悲痛和怀念之中，另一个不幸的消息又接踵而至，C语言发明人、被誉为“C语言之父”，同时也是操作系统UNIX发明人之一的丹尼斯·里奇(D. M. Ritchie)于2011年10月12日去世，享年70岁。

在悲痛之余，我就在想，难道上帝也急需人才，不顾人类的反对，把人们尊敬并崇拜的丹尼斯·里奇先生给叫走了？不管什么原因，书写下面的文字有两个目的：一是为了向各位读者介绍C语言的历史；二是为了纪念这位伟大的科学家——丹尼斯·里奇先生。

著名的UNIX操作系统是1969年由贝尔实验室的肯·汤普逊(Ken Thompson)和丹尼斯·里奇在一起用汇编语言开发设计的。

要讲清楚C语言的诞生，必须从ALGOL 60语言开始讲起。ALGOL 60语言出现于1960年，它是面向问题的高级语言，不适合用来编写操作系统程序(不擅长直接操作硬件)。为此，1963年剑桥大学在ALGOL 60语言的基础上推出CPL语言，这种语言擅长直接操作硬件，但规模较大，难以实现。1967年剑桥大学又在CPL语言的基础上推出了BCPL语言。

1970年贝尔实验室的Ken Thompson以BCPL语言为基础，进一步简化，设计出B语言，并用B语言重新设计了UNIX操作系统。由于B语言过于简单，功能有限，1972—1973年之间，贝尔实验室的D. M. Ritchie，也就是伟大的丹尼斯·里奇先生在B语言的基础上设计出了C语言，并在1973年和Ken Thompson一起把UNIX操作系统用C语言改写。

C语言是使用最广泛的计算机程序设计语言之一。可以说，C语言的诞生是现代程序语言革命的起点，是程序设计语言发展史上的一座里程碑。

自C语言出现后，它的版本也进行了多次更新和修改。1983年美国国家标准协会(ANSI)根据C语言的各种版本对C的发展和扩充制定了新的标准——ANSI C。1987年又公布了新的标准——87 ANSI C。1990年，国际化标准组织ISO接受87 ANSI C为ISO C的标准，目前流行的C编译系统都是以它为基础的，但相互之间略有不同。

另外，由于C语言的诞生，以C语言为根基的C++、Java和C#等面向对象语言相继诞生，并在各自领域大获成功。但今天C语言依旧在系统编程、嵌入式编程等领域占据着统治地位。

而需要记住的是：C语言，这种最有效、最通用的编程语言，就是丹尼斯·里奇先生开发的，唯有努力地学习它，并且尽快地掌握C语言，这才是对丹尼斯·里奇先生最好的纪念。下面开始真实地触摸C语言。

1.2 简单C程序的编辑、调试以及运行

为了更好地纪念丹尼斯·里奇先生,第一个程序从丹尼斯·里奇先生开始。现在用C编写一个程序,程序运行后能在屏幕上显示:

```
Thank you——丹尼斯·里奇先生!
```

那么该如何动手呢?

1.2.1 源程序、目标程序以及可执行程序

要完成上面的要求,让计算机输出“Thank you——丹尼斯·里奇先生!”这句话,必须按照下面的步骤进行。

1. 建立一个C的源程序文件

需要把要做的工作,用C的语句书写出来,以文件的形式保存起来,这个文件就是C的源程序文件。

要建立一个C的源程序文件,可以用记事本建立,但用记事本建立的源程序文件,还必须再运行C的编译系统进行编译等,这些建立和编译工作不能在一个界面下进行,转换起来比较麻烦。

现在有许多集成开发平台,它能把源文件的建立、编译、运行等一系列的工作整合在一个环境下,使C程序的开发调试变得相当的简单。而VC++ 6.0就是这样的一个开发平台。

VC++ 6.0集成开发环境除了支持C的开发,同时也支持C++的开发,如果建立的源文件扩展名为.c时,则集成环境就会把它当成一个C的源文件。也就是说,当用VC++ 6.0开发C程序时,源文件的扩展名必须是.c。

可以按下面的步骤建立一个C的源文件。

(1) 启动VC++ 6.0

运行VC++ 6.0应用程序,屏幕上会出现如图1-1所示的界面,单击“关闭”按钮,关闭“每日提示”提示框,或者直接取消选中“启动时显示提示”复选框。

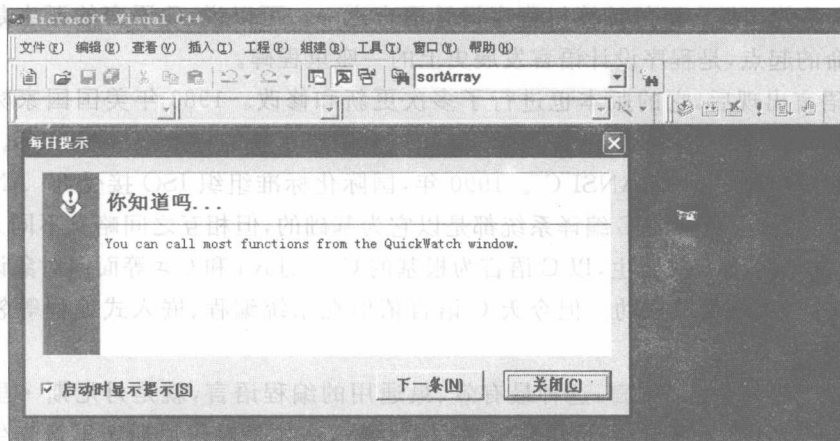


图 1-1 VC++ 6.0 的启动界面

(2) 新建一个扩展名为.c的源程序文件

在图 1-1 所示的窗口中选择“文件”→“新建”命令,会出现图 1-2 所示的“新建”对话框,在该对话框中选择文件类型、文件保存位置以及输入文件名字。

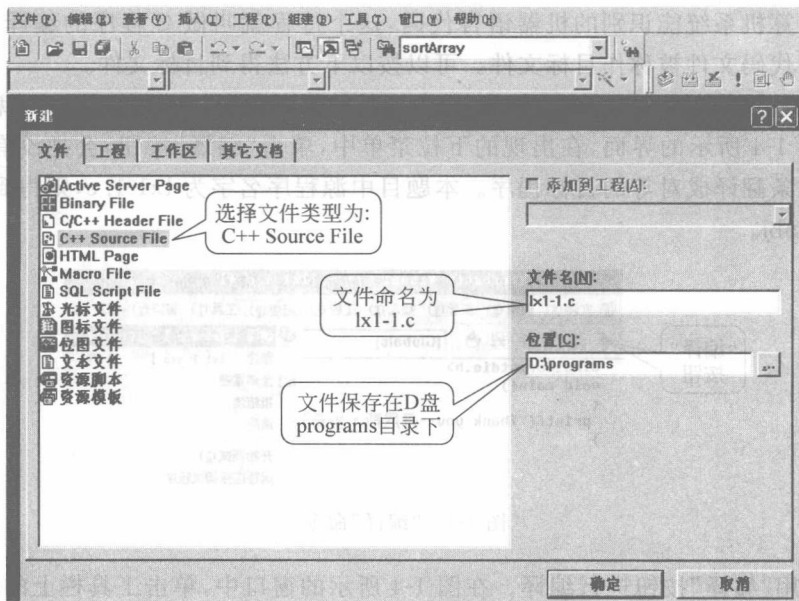


图 1-2 VC++ 6.0

(3) 编辑源程序

在图 1-2 所示的窗口中正确的选择并输入有关内容后,单击“确定”按钮,会进入图 1-3 所示的程序编辑窗口,在该窗口下,输入 C 的有关程序代码。

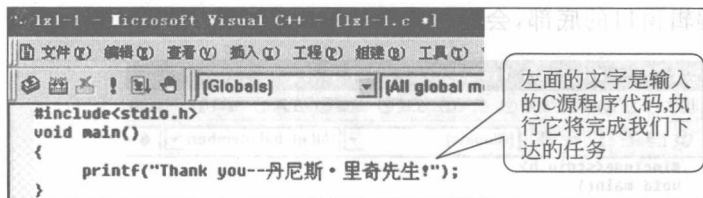


图 1-3 源程序编辑界面

【例 1-1】 建立源程序文件 lx1-1.c。

```
#include<stdio.h>
void main()
{
    printf("Thank you--丹尼斯·里奇先生!");
}
```

说明:除了要输出的“Thank you--丹尼斯·里奇先生!”这句话以外,里面用到的所有符号都必须是英文符号。