

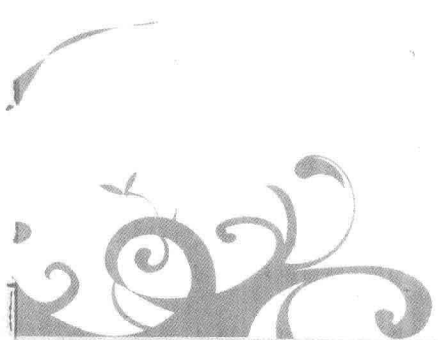
高职高专计算机任务驱动模式教材

C语言程序设计 与项目实践

陈建国 易永红 马宁 靳光明 黄明清 编著

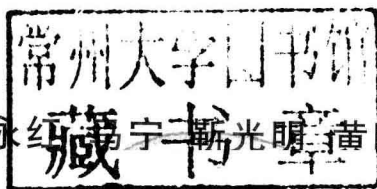


清华大学出版社



高职高专计算机任务驱动模式教材

C语言程序设计 与项目实践



陈建国 易永红 宁二新 光明 黄明清 编著

清华大学出版社
北京

内 容 简 介

本书作为学习编程的入门教材,强化项目实践,重在逐步提高读者的编程能力。本书按照 CDIO 的模式编写,即按照构思(分析)、设计、实现、运行的结构来构建项目模块,将“学生成绩管理系统”贯穿到全书的每个项目中,项目基于工作任务,工作任务基于教学案例,将基础知识融入项目任务中。本书共有 11 个项目、34 个工作任务、32 个拓展任务、150 多个案例程序,每个项目有关键词(中英文对照)、小结和习题(含全国等级考试的精选试题)。

本书内容包括 C 语言概述、基本数据类型及运算、顺序程序设计、分支程序设计、循环程序设计、模块化程序设计;简单构造类型、复杂构造类型、指针、文件组织与使用;位运算、C 语言与汇编语言的混合编程、C51 程序设计基础。

本书可以作为计算机及相关专业程序设计课程的入门教材,也可以作为等级考试和其他从事计算机编程人员的参考用书。

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 语言程序设计与项目实践/陈建国等编著. —北京:清华大学出版社,2013.1

(高职高专计算机任务驱动模式教材)

ISBN 978-7-302-29876-2

I. ①C… II. ①陈… 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 197318 号

责任编辑:刘 青

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 装 者:北京兆成印刷有限公司

经 销:全国新华书店

开 本:185mm×260mm 印 张:26.5 字 数:644 千字

版 次:2013 年 1 月第 1 版 印 次:2013 年 1 月第 1 次印刷

印 数:1~3000

定 价:49.00 元

前言

本书作为学习编程的入门教材,着重讲述 C 语言程序设计的基础知识、基本算法和结构化程序设计的基本方法。本书强化项目实践,重在提高学生的编程能力。本书具有如下特色。

(1) 知识体系层次分明。在内容上,从学生的思维方式、理解能力以及在后续课程中的应用等诸方面的因素出发,内容编排顺序与其他教材相比有较大的改变,其框架结构分为 3 个部分。

第一部分 C 语言基础

- 项目 1 分析学生成绩管理系统的结构——C 语言概述
- 项目 2 描述学生的特征信息——基本数据类型及运算
- 项目 3 系统的菜单程序设计——顺序程序设计
- 项目 4 学生成绩的分类处理——分支程序设计
- 项目 5 学生成绩的统计分析——循环程序设计
- 项目 6 模块化与团队协作——模块化程序设计

第二部分 C 语言应用编程

- 项目 7 学生成绩管理系统的设计(1)——简单构造类型
- 项目 8 学生成绩管理系统的设计(2)——复杂构造类型
- 项目 9 学生成绩管理系统的设计(3)——指针
- 项目 10 学生成绩信息的存储与管理——文件组织与使用

第三部分 C 语言的高级编程技术

- 项目 11 基于 51 单片机竞赛抢答器设计——C 语言的高级应用
- 包括位运算、C 语言与汇编语言的混合编程、C51 程序设计基础。

(2) 零起点,循序渐进,系统学习。从结构上看,做到循序渐进、系统学习、广泛实践,便于学生接受。通过第一部分的学习,使学生能基本掌握利用 C 语言进行结构化程序的设计,在学习程序的 3 种基本结构后,紧接着学习模块化程序设计技术(函数),让学生接受结构化、模块化的编程思想。通过第二部分的学习,逐步引入工程项目的实际例子,以提高学生分析问题和解决问题的能力。在第二部分中先学习数组,利用经典的案例,让学生学会如何利用 C 语言实现数据的处理,如信息系统设计中增加、删除、修改、查找等基本操作,为后面的工程项目实践打下基础。通过结构、共用体、枚举的学习,让学生掌握如何运用计算机来描述现实生活的实体对象,如何解决实际问题。指针是 C 语言的精彩部分,在通过前面的学习,学生已经具备一定

的编程能力后,将指针与数组、函数、结构等结合起来,利用指针编写出具有更高效率的程序。最后利用 C 语言的文件处理,真正实现对数据的管理,结合前面的项目,完成一个完整的项目的设计。第三部分为选学部分,主要是为学生在工作中利用 C 语言实现实时控制的编程打下基础,达到学习引导的作用。

(3) 案例教学,任务驱动,项目实践。本书采用按照 CDIO (Conceive、Design、Implement、Operate) 的模式编写,即按照构思(分析)、设计、实现、运作的结构来构建项目模块,将“学生成绩管理系统”贯穿到教材的每个项目中,项目的完成基于相关的几个工作任务,工作任务的完成基于各个“相关知识”中的案例,通过案例讲解相关知识点,将基础知识融入项目任务中。当所有项目完成后,一个完整的“学生成绩管理系统”程序就完成了。本书有 11 个项目、34 个工作任务、32 个拓展任务、150 多个案例程序,各个项目既有一定的独立性,又相互关联,如项目 8、项目 9、项目 10,其项目的任务基本相同,但解决问题的思路和方法不同。并充分考虑程序的共享性、可维护性,大量采用模块化编程、多人协作编程、N-S 算法描述工具等工程技术。程序中很多代码是通用的,每个模块的代码量较小,学生很容易理解和实现。采用这种方式,通过完成所有项目的程序设计,逐步掌握 C 语言程序开发的基本方法,培养良好的编程习惯。

(4) 讲练结合,自主学习。为避免现在的教学中学生只能“抄程序”不能“写程序”的弊病,必须让学生在掌握一定 C 语言语法的基础上多写程序。为了完成每个工作任务,采用提出问题、分析问题,然后讲授解决问题所需要的基本知识,最后实施任务、拓展训练等几个教学阶段进行学习。每个知识点中结合大量的案例,这些案例是为解决工作任务而精心编写的,教学上采用“讲练结合”的方式教学,让学生在实践中学。学生通过相关案例的学习及拓展训练,在老师的指导下完成相关工作任务,最后完成“项目实践”中的项目。采用这种方法引导和帮助学生解决实际问题,提高学生解决具体问题及自我学习的能力。

(5) 重点突出,层次分明。在知识体系上按照高级语言程序设计本身教学特征的要求,分为 3 条线索:数据类型(基本数据类型、简单结构类型、复杂结构类型、指针)、程序设计的基本方法(顺序程序设计、分支程序设计、循环程序设计、模块化程序设计)、常用算法(算法及描述、数据处理、迭代、穷举、数据文件的存储与管理)。采用 VC++ 6.0 编程环境,并适当引入 C++ 的部分内容,如常量的声明、输入输出流、引用变量等。在“相关知识”中的重要部分用黑体表达,每个项目有关键词、小结,部分习题为全国等级考试的精选试题(注有考试的年份)。本书从多角度让学生掌握基础知识,通过知识体系的学习,让学生不但具有一定实践能力,还具有相应的专业理论基础,为提高学生的专业素质打下良好的基础。

本书内容翔实、层次分明、结构紧凑,叙述深入浅出、通俗易懂,适合作为计算机及相关专业程序设计课程的入门教材,也可作为等级考试和其他从事计算机编程人员的参考用书。

由于编者水平有限,书中难免存在疏漏,恳请专家和广大读者批评指正。

编 者

2012 年 5 月

目 录

项目 1 分析学生成绩管理系统的结构——C 语言概述	1
任务 1.1 熟悉编程环境——程序设计语言与 C 语言	1
1.1.1 程序设计语言	2
1.1.2 C 语言的发展与特点	3
1.1.3 C 语言的上机调试步骤和方法	4
1.1.4 VC++ 6.0 集成开发环境简介	12
1.1.5 任务分析与实施	16
任务 1.2 分析并描述算法——算法及其描述	17
1.2.1 算法与结构化程序设计	17
1.2.2 算法的描述	19
1.2.3 任务分析与实施	23
项目实践	25
小结	26
习题	26
项目 2 描述学生的特征信息——基本数据类型及运算	28
任务 2.1 理解 C 语言的结构——C 语言的结构特征	28
2.1.1 C 语言的结构	29
2.1.2 C 语言的字符集与词汇	29
2.1.3 任务分析与实施	31
任务 2.2 理解数据在计算机中的表示与存储—— C 语言的数据类型	32
2.2.1 C 语言的数据类型	32
2.2.2 整型数据	34
2.2.3 实型数据	37
2.2.4 字符型数据	39
2.2.5 变量赋初值	42
2.2.6 测试数据长度	43
2.2.7 各类数值型数据之间的混合运算	43
2.2.8 任务分析与实施	45
任务 2.3 理解 C 语言的算术运算——运算符与表达式	47

2.3.1 运算符的分类	47
2.3.2 算术运算符与表达式	48
2.3.3 赋值运算符和赋值表达式	50
2.3.4 自增、自减运算符	51
2.3.5 逗号运算符和逗号表达式	53
2.3.6 任务分析与实施	54
项目实践	54
小结	56
习题	57
项目3 系统的菜单程序设计——顺序程序设计	61
任务 3.1 考试成绩绩点的计算——数据输入输出	61
3.1.1 字符数据的输入输出	62
3.1.2 格式输入与输出	63
3.1.3 输入输出流	68
3.1.4 顺序程序设计	71
3.1.5 任务分析与实施	72
任务 3.2 提高程序的可读性与易修改性——宏与预定义	74
3.2.1 宏定义	74
3.2.2 文件包含	80
3.2.3 条件编译	80
3.2.4 任务分析与实施	82
项目实践	83
小结	84
习题	85
项目4 学生成绩的分类处理——分支程序设计	91
任务 4.1 描述条件逻辑——分支和条件逻辑	91
4.1.1 关系运算符和表达式	92
4.1.2 逻辑运算符和表达式	93
4.1.3 条件运算符和表达式	94
4.1.4 任务分析与实施	96
任务 4.2 判断考试成绩是否合格——简单分支程序设计	97
4.2.1 单分支结构的程序设计	97
4.2.2 双分支结构的程序设计	99
4.2.3 任务分析与实施	100
任务 4.3 求解分段函数——多分支程序设计	101
4.3.1 if 语句的多重选择程序设计	101
4.3.2 switch 语句的多重选择程序设计	104
4.3.3 任务分析与实施	106

项目实践	108
小结	111
习题	112
项目5 学生成绩的统计分析——循环程序设计	117
任务 5.1 统计某门课程的平均成绩——循环语句	117
5.1.1 用 while 语句处理循环	118
5.1.2 用 do-while 语句处理循环	120
5.1.3 用 for 语句处理循环	121
5.1.4 循环嵌套的使用	126
5.1.5 任务分析与实施	128
任务 5.2 强制中断循环——控制转移语句	129
5.2.1 break 语句	129
5.2.2 continue 语句	130
5.2.3 goto 语句	131
5.2.4 任务分析与实施	133
任务 5.3 穷举与迭代——循环程序设计的应用	134
5.3.1 穷举	135
5.3.2 迭代	136
5.3.3 任务分析与实施	139
项目实践	141
小结	142
习题	143
项目6 模块化与团队协作——模块化程序设计	148
任务 6.1 成绩统计的模块化编程——函数	148
6.1.1 函数的概念	149
6.1.2 函数的声明与定义	150
6.1.3 函数的调用	152
6.1.4 函数间的数据传递	154
6.1.5 任务分析与实施	158
任务 6.2 多人协作完成一个任务——变量的作用域与生存周期	160
6.2.1 变量的作用域与可见性	161
6.2.2 变量的生存周期	166
6.2.3 内部函数和外部函数	170
6.2.4 任务分析与实施	171
任务 6.3 工资发放时币额数量的计算——嵌套与递归	173
6.3.1 嵌套	173
6.3.2 递归	174
6.3.3 任务分析与实施	176

项目实践	178
小结	181
习题	183
项目 7 学生成绩管理系统的设计(1)——简单构造类型	186
任务 7.1 统计分析考试成绩——一维数组	186
7.1.1 一维数组的定义与初始化	187
7.1.2 一维数组的元素的引用	189
7.1.3 一维数组作为函数参数	192
7.1.4 任务分析与实施	195
任务 7.2 创建学生成绩表——一维数组的应用	197
7.2.1 数据元素的复制与选择	197
7.2.2 数据元素的移动	201
7.2.3 数据元素的查找与统计	202
7.2.4 数据元素的修改、删除与插入	207
7.2.5 数据元素的排序	211
7.2.6 任务分析与实施	213
任务 7.3 统计分析多门课程成绩——二维数组及其应用	216
7.3.1 二维数组的定义与初始化	217
7.3.2 二维数组的元素的引用	218
7.3.3 二维数组作为函数参数	221
7.3.4 任务分析与实施	222
任务 7.4 查找考试成绩——字符数组与字符串	226
7.4.1 字符数组的定义、初始化及引用	226
7.4.2 字符串与字符数组	226
7.4.3 字符串的处理	227
7.4.4 任务分析与实施	231
项目实践	234
小结	237
习题	238
项目 8 学生成绩管理系统的设计(2)——复杂构造类型	244
任务 8.1 创建成绩表(1)——结构体	244
8.1.1 结构与结构变量的定义	245
8.1.2 结构成员的初始化	247
8.1.3 结构成员的访问	248
8.1.4 结构与函数	249
8.1.5 结构数组	250
8.1.6 任务分析与实施	251
任务 8.2 创建成绩表(2)——枚举	255

8.2.1 枚举类型的定义	255
8.2.2 枚举类型变量	256
8.2.3 任务分析与实施	257
任务 8.3 创建成绩表(3)——共用体	259
8.3.1 共用体	259
8.3.2 共用体变量的引用	260
8.3.3 任务分析与实施	262
任务 8.4 学生基本信息的抽象分层处理——自定义类型与 数据的抽象与分层	264
8.4.1 自定义类型	265
8.4.2 自定义类型的应用	266
8.4.3 任务分析与实施	267
项目实践	269
小结	274
习题	275
项目 9 学生成绩管理系统的设计(3)——指针	279
任务 9.1 利用指针访问数据——指针与指针变量	279
9.1.1 指针的概念	280
9.1.2 指针变量的定义	281
9.1.3 指针运算符	281
9.1.4 指针表达式	283
9.1.5 存储空间的动态分配	286
9.1.6 任务分析与实施	288
任务 9.2 创建静态成绩表——指针与函数	289
9.2.1 指针变量作为函数的参数	289
9.2.2 main()函数中的参数	291
9.2.3 函数的返回值是指针	292
9.2.4 指向函数的指针	293
9.2.5 任务分析与实施	296
任务 9.3 创建动态成绩表——指针与数组	297
9.3.1 指针与一维数组	298
9.3.2 指针与多维数组	302
9.3.3 数组指针作为函数参数	305
9.3.4 字符串和指针	310
9.3.5 任务分析与实施	312
任务 9.4 创建链表——指针与结构	315
9.4.1 指向结构体变量的指针	316
9.4.2 指向结构体数组的指针	318
9.4.3 结构指针变量作函数参数	319

9.4.4 结构体的自引用	320
9.4.5 任务分析与实施	323
任务 9.5 再谈排序——指针数组与指向指针的指针	325
9.5.1 指针数组	325
9.5.2 指向指针的指针	326
9.5.3 任务分析与实施	327
项目实践	329
小结	333
习题	334
项目 10 学生成绩信息的存储与管理——文件组织与使用	339
任务 10.1 顺序存取学生信息——文件的顺序读写	339
10.1.1 文件的相关概念	340
10.1.2 文件的打开与关闭	342
10.1.3 文件的读与写	343
10.1.4 文件检测函数	350
10.1.5 任务分析与实施	352
任务 10.2 随机存取学生信息——文件的随机读写	357
10.2.1 文件定位	357
10.2.2 随机读写	358
10.2.3 任务分析与实施	359
项目实践	361
小结	368
习题	369
项目 11 基于 51 单片机竞赛抢答器设计——C 语言的高级应用	372
任务 11.1 文件的加密与解密——位运算及应用	372
11.1.1 位运算	373
11.1.2 位域(位段)	378
11.1.3 任务分析与实施	380
任务 11.2 理解混合编程——高级语言与汇编语言的混合编程	382
11.2.1 Visual C++ 中嵌入汇编语句的方法	383
11.2.2 采用模块调用的方法	385
11.2.3 任务分析与实施	387
任务 11.3 89S51 单片机控制 LED——C51 程序设计基础	390
11.3.1 C51 编程基础	390
11.3.2 Keil C51 集成开发环境简介	394
11.3.3 任务分析与实施	397
项目实践	399

小结	401
习题	402
附录 A C 语言中的运算符表	403
附录 B ASCII 对照表	404
附录 C C 语言常见库函数	405
参考文献	412

项目 1 分析学生成绩管理系统的结构

——C 语言概述

技能目标 掌握 C 语言编程环境与程序调试的基本方法。

知识目标 C 语言是什么,如何编写 C 语言的程序,C 语言程序又如何执行,通过本项目任务的学习来回答这些问题。本项目涉及如下的知识点。

- C 语言的编程环境;
- 算法及其描述。

完成该项目后,达到如下的目标。

- 了解 C 语言的特点;
- 熟悉 C 语言的开发环境;
- 掌握算法的基本概念及表示方法。

关键词 编辑(edit)、编码(coded)、连接(link)、编译(compile)、解释(interpretation)、目标程序(object program)、模块化(modular)、结构化(structured)、流程图(flow chart)、高级语言(high-level language)、汇编语言(assembly language)、机器语言(machine language)、调试(debug)

本项目是完成对学生成绩管理系统的需求分析,为完成本项目,必须熟悉算法的描述方法、描述算法的常用工具,逐步提高分析问题与解决问题的能力。在实施项目任务中要注意几个方面的问题:①了解成绩管理系统的目标任务;②选择合适的算法描述工具;③理解模块化、结构化的概念。

任务 1.1 熟悉编程环境

——程序设计语言与 C 语言



问题的提出

在开始学习程序设计时,初学者首先遇到的问题可能是:“什么是程序?”、“什么是程序设计语言?”本项目任务首先讨论这方面的问题,以期帮助读者在比较直观的基础上建立起对程序、程序设计、程序设计语言的基本认识。而后将简单介绍本书讨论的一种程序设计语言——C 语言,并通过一个简单实例介绍 C 语言程序的基本结构和有关概念。



相关知识

1.1.1 程序设计语言

程序一词也来自生活,通常指完成某些事务的一种既定方式和过程。从表述方面看,可以将程序看作对一系列动作的执行过程的描述。日常生活中也可以找到许多“程序”实例。例如,一个学生早上起床后的行为可以描述为:①起床;②刷牙;③洗脸;④吃饭;⑤早自习。

日常生活中程序性活动的情况与计算机里的程序执行很相似,为了与计算机交流,同样需要有与之交流的方式,需要一种意义清晰、计算机也能处理的描述方式。也就是说,需要有描述程序的合适语言。这种可供人编程序用的语言就是程序设计语言,它是一种人造语言,也常被称为编程语言。

语言是一套具有语法、词法规则的系统。计算机程序设计语言是计算机可以识别的语言,用于描述解决问题的方法,供计算机阅读和执行。

计算机完成各种复杂的工作,除了具有强大的硬件系统,还要有相应的软件系统,软件包含了计算机运行所需的各种程序及其文档。计算机的工作是通过程序来控制的,计算机按照规定的程序要求运行。程序是指令的集合。编写程序就是将解决问题的方法,编写成由一条条指令组成的程序,输入到计算机中,计算机执行这一指令序列,就完成了规定的任务。

所谓指令,就是计算机可以识别的命令。不同的计算机硬件系统有不同的指令集合,一台计算机硬件系统能识别的所有指令的集合,称为它的指令系统。

按级别计算机程序设计语言分为机器语言、汇编语言和高级语言。

1. 机器语言

由计算机硬件系统可以识别的二进制指令组成的语言称为机器语言。它是面向机器的语言,不同类型的机器其指令系统不同,机器语言也不同,程序具有不可移植性,代码可读性差,编写程序复杂,但其执行的速度快。

2. 汇编语言

汇编语言是将机器指令用人们能读得懂的助记符来表示的语言,如:

```
MOV AX, 3
MOV BX, 5
SUB AX, BX
```

每条指令对应于一条机器语言指令,由于采用了助记的符号名,这样,每条指令的意义较机器语言都更容易理解和把握了。但是,汇编语言的程序属于非结构化程序,程序的结构性差。汇编语言也是面向机器的语言,程序的可移植性差。汇编语言必须编译成机器语言后才能执行。汇编语言的翻译软件称为汇编程序。

3. 高级语言

高级语言是面向问题的语言,其特点是在一定程度上与具体机器无关,程序中可以采用具有一定含义的数据名和容易理解的执行语句,这使得编写程序变得易学、易用、易维护。高级语言程序只有翻译成机器语言后才能执行。如何将高级语言翻译成机器语言呢?有以

下两种方式。

(1) 编译方式: 这种方式是将编写好的源程序经编译软件进行编译、链接后, 生成可执行文件, 然后再执行。这种方式交互性差, 但程序执行的效率高, 如 C 语言、FORTRAN、PASCAL 语言等。

(2) 解释方式: 程序执行时一边通过程序解释软件“翻译”, 一边执行。这种方式交互性好, 但执行效率低, 如早期的 BASIC 语言、FoxBase 等。

1.1.2 C 语言的发展与特点

C 语言的原型是 ALGOL 60 语言(也称为 A 语言)。

1963 年, 剑桥大学将 ALGOL 60 语言发展成为 CPL(Combined Programming Language) 语言。

1967 年, 剑桥大学的 Martin Richards 对 CPL 语言进行了简化, 于是产生了 BCPL 语言。

1970 年, 美国贝尔实验室的 Ken Thompson 将 BCPL 进行了修改, 并为它起了一个有趣的名字“B 语言”。他用 B 语言写了第一个 UNIX 操作系统。

而在 1973 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 的第二字母作为这种语言的名字, 这就是 C 语言。

为了使 UNIX 操作系统推广, 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》。

1978 年 Brian W. Kernighan 和 Dennis M. Ritchie 出版了名著《The C Programming Language》, 通常简称为《K&R》, 也有人称之为《K&R》标准。但是, 在《K&R》中并没有定义一个完整的标准 C 语言, 许多开发机构推出了自己的 C 语言版本, 这些版本之间的微小差别不时引起兼容性上的问题, 后来由美国国家标准学会 ANSI(American National Standard Institute)在各种 C 语言版本的基础上制定了一个 C 语言标准, 于 1983 年发表, 通常称之为 ANSI C。1987 年 ANSI 又公布了新标准——87 ANSI C, 目前广泛流行的各种 C 编译系统都是以它为基础的。

1983 年又由贝尔实验室的 Bjarne Stroustrup 推出了 C++。C++ 进一步扩充和完善了 C 语言, 成为一种面向对象的程序设计语言。

C 语言是一种通用、灵活、结构化、标准化、使用广泛的编程语言, 能完成用户的各种任务, 特别适合进行系统程序设计和对硬件进行操作的场合。它具有如下的特征。

(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, 程序书写自由, 主要用小写字母表示。它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 的运算符包含的范围很广泛, 共有 34 个运算符。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 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) 数据结构丰富。C 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。它能用来实现各种复杂的数据类型的运算, 并引入了指针概念, 使程序效率更高。另外 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。

(4) C 是结构式语言。结构式语言的显著特点是代码及数据的分隔化,即程序的各个部分除了必要的信息交流外彼此独立。这种结构化方式可使程序层次清晰,便于使用、维护以及调试。C 语言是以函数形式提供给用户的,这些函数可方便地调用,并具有多种循环、条件语句控制程序流向,从而使程序完全结构化。

(5) C 语法限制不太严格,程序设计自由度大。一般的高级语言语法检查比较严,能够检查出几乎所有的语法错误。而 C 语言允许程序编写者有较大的自由度。

(6) 允许直接访问物理地址,直接操作硬件。C 语言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、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,可以用来编写系统软件。

(7) 程序执行效率高。一般只比汇编程序生成的目标代码效率低 10%~20%。

(8) 可移植性好。C 语言有一个突出的优点就是适合于多种操作系统,如 DOS、UNIX,也适用于多种机型。

当然,C 语言也有自身的不足,比如 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,对数组的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。

1.1.3 C 语言的上机调试步骤和方法

1. 编制并运行程序的“四步曲”

C 程序在计算机上的实现与其他高级语言一样,一般要经过编辑、编译、连接、运行 4 个步骤,C 程序的实现流程如图 1-1 所示。为了说明 C 语言源程序结构的特点,用 VC++ 6.0 先来编制一个最简单的程序,并让它运行而得出结果,以此作为了解 VC++ 6.0 的开端。

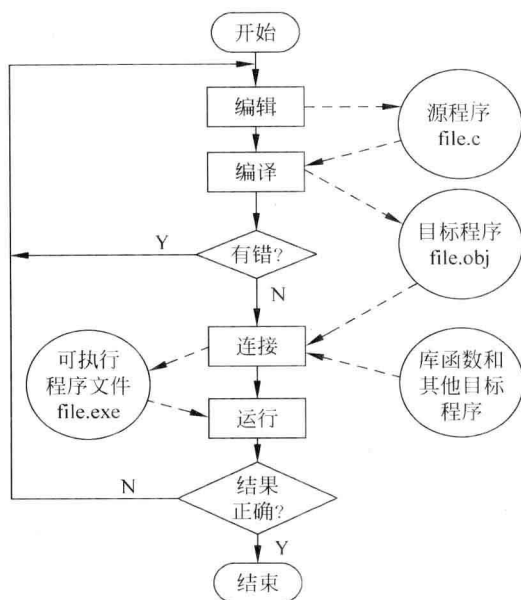


图 1-1 C 程序的实现流程

(1) 编辑

编辑就是建立、修改 C 语言源程序,并把它输入计算机的过程。C 语言的源文件以文本文件的形式保存,它的后缀名为 .c,如果是 C++ 则后缀名为 .cpp。

源文件的编辑可以用任何文字处理软件完成,一般用编译器本身集成的编辑器进行编辑。

(2) 编译(生成目标程序文件 .obj)

C 语言是以编译方式实现的高级语言,C 程序的实现必须经过编译程序对源文件进行编译,生成目标代码文件,它的后缀名为 .obj。编译前一般先要进行预处理,譬如进行宏代换、包含其他文件等。

编译程序把一个源程序翻译成目标程序的工作过程分为 5 个阶段:词法分析;语法分析;语义检查和中间代码生成;代码优化;目标代码生成。主要是进行词法分析和语法分析,又称为源程序分析,分析过程中若发现有语法错误,则给出提示信息。

(3) 连接(生成可执行程序文件 .exe)

编译形成的目标代码还不能在计算机上直接运行,必须将其与库文件进行连接处理,这个过程由连接程序自动进行,连接后生成可执行文件,它的后缀名为 .exe。如果连接出错同样需要返回到编辑步骤修改源程序,直至正确为止。

(4) 运行(可执行程序文件)

上述 4 个步骤中,第一步的编辑工作是最繁杂而又必须细致地由人工在计算机上来完成,其余几个步骤则相对简单,基本上由计算机来自动完成。

程序运行后,可以根据运行结果判断程序是否还存在其他方面的错误。编译时产生的错误属于语法错误,而运行时出现的错误一般是逻辑错误。出现逻辑错误时需要修改原有算法,重新进行编辑、编译和连接,再运行程序。

2. 上机运行 C 语言程序的步骤

(1) 启动并进入 VC++ 6.0 的集成开发环境

有两种方法启动并运行 VC++ 6.0,进入到它的集成开发环境窗口。

方法 1:若桌面上有 VC++ 6.0 图标,双击该图标。

方法 2:通过选择【开始】→【程序】→Microsoft Visual Studio 6.0→Microsoft Visual C++ 6.0 命令。

图 1-2 式样的窗口从大体上可分为 4 部分:上部为菜单栏和工具条;中左为工作区 (Workspace) 视图显示窗口,这里将显示处理过程中与项目相关的各种文件种类等信息;中右为文档内容区,是显示和编辑程序文件的操作区;下部为输出 (Output) 窗口区,程序调试过程中,进行编译、连接、运行时输出的相关信息将在此处显示。

 **注意:** 由于系统的初始设置或者环境的某些不同,可能启动的 VC++ 6.0 初始窗口式样与图 1-2 有所不同,也许会没出现 Workspace 窗口或 Output 窗口,这时可通过 View→Workspace 菜单选项的执行,使中左处的工作区窗口显现出来;而通过 View→Output 菜单选项的执行,又可以使下部的输出区窗口得以显现。当然,如果不想看到这两个窗口,可以单击相应窗口的  按钮来关闭窗口。