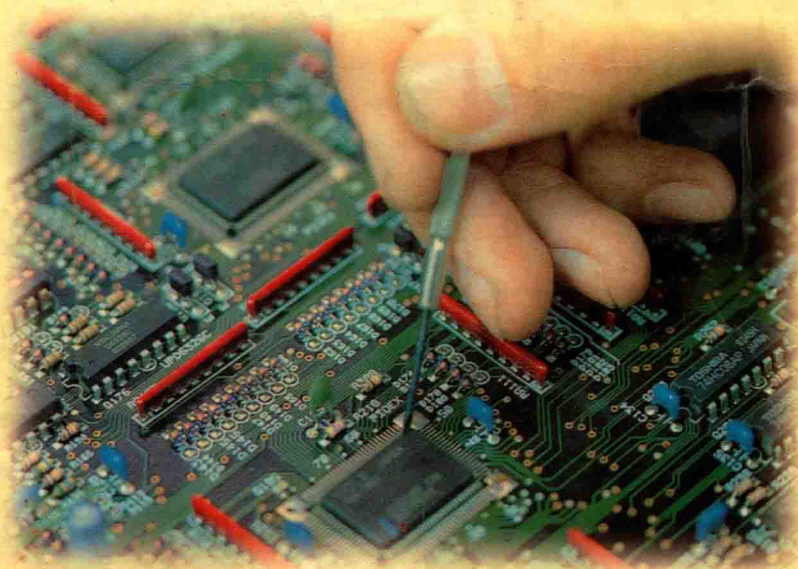


COMPUTER

C/C++语言与 软件开发技术



● 周绍梅 陈 炼 主编

江西高校出版社

C/C++ 语言与软件开发技术

周绍梅 陈 炼 主编



南昌大学计算中心

上机证

姓名 凌家良

班级 电子924

性别 男

江西高校出版社

C/C++ 语言与软件开发技术

周绍梅 陈 炼 主编

江西高校出版社

(江西省南昌市洪都北大道 96 号)

邮编:330046 电话:(0791)8512093,8519894

各地新华书店经销

南昌市印刷五厂印刷

1999 年 1 月第 1 版 1999 年 1 月第 1 次印刷

787mm×1092mm 1/16 17.625 印张 450 千字

印数:4100 册

定价:19.80 元

ISBN 7-81033-918-4/TP·17

(江西高校版图书如有印刷、装订错误,请随时向承印厂调换)

内 容 简 介

本书共分十二章,主要内容包括四部分:C语言、实用数据结构、软件开发技术和C++语言简介。其中C语言部分除介绍C语言的基本概念、语法规则和程序流程以外,还以一些常用算法为例,介绍程序设计的方法;实用数据结构部分以C语言作为工具,描述一些常用的数据结构及其实现,其中包括线性表、栈、队列、树与二叉树、排序、查找等;软件开发技术部分包括软件系统与需求分析、软件设计及程序编码、软件测试技术、软件维护与软件文档;C++语言简介部分将C++对C的扩展作了简单介绍,以使读者对C++有一个初步的了解。

本书由作者根据多年教学经验编写而成,力求深入浅出、通俗易懂。它既可作为高等院校非计算机专业理工科本科、专科学生的教材,也可供自学使用。

前 言

计算机技术的千变万化和应用领域的不断拓宽,使高校计算机基础教育这一关系到培养跨世纪人才的大事日益受到全社会的广泛关注。教育部高教司“八五”期间已把计算机基础教育纳入高校的重要基础课程中,并成立了工科计算机基础课程教学指导委员会(以下简称课委会)。去年,课委会对全国高等理工科院校计算机基础教学状况进行了调查,并在此基础上,制定了《工科非计算机专业计算机基础教学指南》(以下简称教学指南),将计算机基础教学分为三个层次:第一层次为计算机文化基础,主要教学内容是从DOS操作系统的应用界面过渡到WINDOWS图形窗口界面,并引入网络和多媒体的使用常识,第二层次为计算机技术基础,包括计算机软件技术基础和计算机硬件技术基础两方面的内容,计算机软件技术基础围绕程序设计,讲解程序设计的基本方法,阐述计算机软件中的一些重要概念,培养学生利用计算机处理问题的思维方式,计算机硬件技术包括微机原理、微机接口和微机应用;第三层次为计算机应用基础,主要包括计算机信息管理基础、计算机辅助设计基础以及多媒体应用基础。

为使工科高等院校的计算机基础教学更能够适应计算机科学技术的发展与应用,我们在1997年编写了《计算机基础教程》一书,作为计算机基础知识的入门教材,该教材内容与课委会的教学指南中要求的第一层次教学内容基本吻合。考虑到C语言是近年来国内外得到迅速推广的一种计算机语言,参照教学指南中第二层次中软件技术基础部分的教学内容,我们组织有多年教学经验的教师编写本书,作为学完《计算机基础教程》课程的后继教材。

本书共十二章,分为四部分内容:第一部分(第一章到第九章)为C语言,第二部分(第十章)为实用数据结构,第三部分(第十一章)为软件开发技术,第四部分(第十二章)为C++语言简介。其中C语言部分以ANSI C为基础兼顾当前微型计算机上流行的Turbo C编译系统介绍C语言的基本概念、原理和方法,并通过实例帮助读者学会如何用C编写应用程序。对计算机软件的应用而言,掌握软件设计与开发的基本方法与技术是一个重要的课题,因此,本书以C为描述工具,介绍了一些实用数据结构和相应的运算,以及软件开发的基本方法,同时还介绍了软件测试与调试的基本概念及方法。面向对象的程序设计方法是一种全新的设计和构造软件的思维方法,C++语言是一个优秀的面向对象的程序设计语言,它保留了C语言许多重要特性,又融合了面向对象的能力,因此在本书最后对C++作了简单介绍,以使读者对C++有一个初步的了解。

全书由周绍梅主编,王命延教授主审,参加编写人员有喻国平(编写第一章、第二章、第九章)、陈炼(编写第三章、第四章、第十一章)、张云如(编写第五章、第六章、第七章)、周绍梅(编写第八章、第十章)、周兴斌(编写第十二章)。

本书在编写和出版过程中,得到了江西高校出版社和南昌大学计算中心的大力支持,在此表示衷心感谢。

由于编者水平有限,书中难免有缺点和错误,殷切希望读者批评指正。

编者

1999年1月于南昌

目 录

第一章 C语言概述	(1)
1.1 C语言发展历史和特点	(1)
1.1.1 C语言发展史	(1)
1.1.2 C语言的特点	(2)
1.2 C程序的基本结构	(3)
1.3 Turbo C 上机操作	(5)
习题一	(6)
第二章 数据类型与运算	(8)
2.1 C语言的数据类型	(8)
2.2 常量	(8)
2.2.1 符号常量	(9)
2.2.2 整型常量	(9)
2.2.3 实型常量	(10)
2.2.4 字符常量	(10)
2.2.5 字符串常量	(11)
2.3 变量	(12)
2.3.1 变量的定义	(12)
2.3.2 变量的初始化	(13)
2.4 运算符和表达式	(13)
2.4.1 算术运算和算术表达式	(14)
2.4.2 自增自减运算符	(14)
2.4.3 关系运算符和关系表达式	(15)
2.4.4 逻辑运算符和逻辑表达式	(16)
2.4.5 赋值运算符和赋值表达式	(17)
2.4.6 逗号运算符和逗号表达式	(17)
2.4.7 条件运算符和条件表达式	(18)
2.4.8 运算符的优先级和类型转换	(19)
2.4.9 表达式的应用举例	(20)
习题二	(22)
第三章 C程序设计	(24)
3.1 C语句简介	(24)
3.2 顺序结构程序设计	(25)
3.2.1 数据输入	(25)
3.2.2 数据输出	(27)
3.2.3 应用举例	(29)
3.3 选择结构程序设计	(30)
3.3.1 if 语句	(30)
3.3.2 switch 语句	(33)

3.3.3 应用举例	(35)
3.4 循环结构程序设计	(36)
3.4.1 while 循环语句	(36)
3.4.2 do-while 循环语句	(38)
3.4.3 for 循环语句	(39)
3.4.4 多重循环	(40)
3.4.5 break、continue 和 goto 语句	(41)
3.4.6 应用举例	(43)
习题三	(45)
第四章 编译预处理	(47)
4.1 宏定义	(47)
4.1.1 符号常量	(47)
4.1.2 带参数的宏定义	(50)
4.2 文件包含	(52)
4.3 条件编译	(53)
4.3.1 #ifdef 与 #endif 命令	(53)
4.3.2 #ifndef 与 #endif 命令	(53)
4.3.3 #if 与 #endif 命令	(54)
4.3.4 条件编译的主要用途	(55)
4.4 应用举例	(55)
习题四	(57)
第五章 数组	(58)
5.1 一维数组	(58)
5.1.1 一维数组的定义与初始化	(58)
5.1.2 一维数组的使用	(59)
5.2 二维数组	(60)
5.2.1 二维数组的定义与初始化	(60)
5.2.2 二维数组的使用	(61)
5.3 字符数组	(62)
5.3.1 字符数组的定义与初始化	(62)
5.3.2 字符数组的使用	(63)
5.3.3 字符数组与字符串	(63)
5.3.4 字符串运算函数	(65)
5.4 应用举例	(68)
习题五	(70)
第六章 函数	(71)
6.1 函数的定义与调用	(71)
6.1.1 函数定义的一般形式	(71)
6.1.2 函数调用的一般形式	(72)
6.1.3 函数的参数	(73)
6.1.4 函数的返回值	(74)
6.1.5 对被调用函数的说明	(76)

6.2 函数的嵌套调用和递归调用	(77)
6.2.1 函数的嵌套调用	(77)
6.2.2 函数的递归调用	(78)
6.3 函数与数组	(80)
6.3.1 数组元素做函数实参	(80)
6.3.2 数组名做函数参数	(80)
6.4 变量的作用域	(83)
6.4.1 局部变量	(83)
6.4.2 全局变量	(84)
6.5 变量的存储类别	(87)
6.5.1 局部变量的存储方式	(87)
6.5.2 全局变量的存储方式	(89)
6.6 内部函数和外部函数	(91)
6.7 应用举例	(92)
习题六	(95)
第七章 指针	(96)
7.1 指针变量	(96)
7.1.1 指针的概念	(96)
7.1.2 指针变量的定义	(97)
7.1.3 指针变量的引用	(98)
7.2 指针与数组	(99)
7.2.1 指针与一维数组	(99)
7.2.2 指针与二维数组	(102)
7.2.3 指针与字符串	(106)
7.3 指针与函数	(107)
7.3.1 变量的指针做函数参数	(108)
7.3.2 数组指针做函数参数	(108)
7.3.3 指向函数的指针变量	(111)
7.3.4 返回指针值的函数	(114)
7.4 指针数组和指向指针的指针	(116)
7.4.1 指针数组	(116)
7.4.2 指向指针的指针	(117)
7.4.3 带参数的 main 函数	(118)
7.5 应用举例	(120)
习题七	(122)
第八章 结构体和共用体	(124)
8.1 结构体的基本概念	(124)
8.1.1 结构体类型的定义	(124)
8.1.2 结构体变量的定义	(125)
8.1.3 结构体变量的引用	(126)
8.1.4 结构体变量的初始化	(127)
8.1.5 结构体变量的输入输出	(128)

8.1.6 结构体的嵌套	(129)
8.2 结构体数组	(130)
8.2.1 结构体数组的定义	(130)
8.2.2 结构体数组的初始化	(130)
8.2.3 结构体数组的引用	(131)
8.3 结构体指针	(132)
8.3.1 指向结构体变量的指针	(132)
8.3.2 指向结构体数组的指针	(134)
8.4 结构体与函数	(135)
8.4.1 结构体变量的成员作函数的参数	(135)
8.4.2 结构体变量作为函数的参数	(135)
8.4.3 结构体指针作为函数的参数	(137)
8.4.4 返回结构体类型的函数	(139)
8.5 共用体	(140)
8.5.1 共用体类型及变量的定义	(140)
8.5.2 共用体变量的引用	(141)
8.5.3 共用体的特点	(142)
8.6 枚举类型	(142)
8.6.1 枚举类型及其变量的定义	(143)
8.6.2 枚举类型的特点	(143)
8.7 类型定义 typedef	(145)
8.8 应用举例	(146)
习题八	(148)
第九章 文件	(149)
9.1 C 语言文件概述	(149)
9.2 文件的打开与关闭	(150)
9.2.1 文件的打开	(150)
9.2.2 文件的关闭	(151)
9.3 文件的读写	(151)
9.3.1 字符的输入输出函数	(151)
9.3.2 字符串的输入输出函数	(152)
9.3.3 记录块数据的输入输出	(154)
9.3.4 格式化输入输出函数	(156)
9.4 文件的定位和随机读写	(157)
习题九	(159)
第十章 实用数据结构	(161)
10.1 数据结构与算法	(161)
10.1.1 数据结构的概念	(161)
10.1.2 数据结构的存储	(162)
10.1.3 算法	(163)
10.1.4 算法与数据结构	(163)
10.2 线性表	(164)

10.2.1 线性表的顺序存储结构	(164)
10.2.2 线性表的链式存储结构	(166)
10.3 栈和队列	(168)
10.3.1 栈	(168)
10.3.2 队列	(170)
10.4 树与二叉树	(172)
10.4.1 树	(172)
10.4.2 二叉树	(173)
10.4.3 二叉树的存储结构	(174)
10.4.4 二叉树的遍历	(175)
10.5 查找	(176)
10.5.1 顺序查找	(176)
10.5.2 二分查找	(177)
10.5.3 散列表	(178)
10.6 排序	(181)
10.6.1 插入排序	(181)
10.6.2 交换排序	(184)
10.6.3 选择排序	(186)
10.6.4 二叉排序树	(186)
10.7 图	(189)
10.7.1 图的基本概念	(190)
10.7.2 图的存储结构	(190)
10.7.3 图的遍历	(192)
10.7.4 从一个源点到其余各顶点的最短路径	(196)
习题十	(198)
第十一章 软件开发技术	(200)
11.1 程序设计概述	(200)
11.1.1 问题分析	(200)
11.1.2 程序的总计设计和详细设计	(201)
11.1.3 编程	(202)
11.1.4 程序的测试	(206)
11.1.5 程序文档	(207)
11.2 实例分析	(207)
11.2.1 问题的描述	(207)
11.2.2 输入和输出的确定	(207)
11.2.3 问题的表示	(207)
11.2.4 问题的分解和解题步骤的确立	(208)
11.3 算法	(208)
11.3.1 常用算法介绍	(209)
11.3.2 算法的选择和表示	(213)
11.3.3 数据结构和控制结构的设计	(215)
11.3.4 算法和框图的细化	(215)
11.4 程序的测试	(218)

11.4.1 程序测试的目的、原则和步骤	(218)
11.4.2 测试用例的设计方法	(221)
11.4.3 错误信息的提供(错误陷阱)	(224)
11.4.4 程序测试实例	(224)
11.5 程序文档	(227)
11.5.1 用户文档和技术文档	(227)
11.5.2 程序文档的生成	(227)
习题十一	(237)
第十二章 C++语言简介	(238)
12.1 C++语言对C语言的扩展	(238)
12.1.1 C++语言对C语言在结构体方法方面的扩展	(238)
12.1.2 C++语言的面向对象的特征	(239)
12.2 常量、类型和说明	(240)
12.2.1 声明	(240)
12.2.2 变量的作用域	(241)
12.2.3 类型	(242)
12.2.4 常量和类型	(243)
12.3 指针	(244)
12.4 函数	(245)
12.4.1 函数原型	(245)
12.4.2 内联函数	(245)
12.4.3 缺省参数	(246)
12.4.4 函数名重载	(246)
12.4.5 不确定参数个数的函数	(247)
12.5 对象类	(247)
12.5.1 类和对象	(247)
12.5.2 构造函数和析构函数	(250)
12.5.3 友员	(253)
12.6 导出类和继承性	(253)
12.6.1 单继承的导出类	(253)
12.6.2 多继承	(255)
12.7 多形性与虚函数	(257)
12.7.1 多形性的概念	(257)
12.7.2 虚函数	(258)
附录 I	(260)
附录 II	(261)
附录 III	(261)
附录 IV	(263)

第一章 C 语言概述

C 语言自七十年代问世以来,就显示出了其强劲的发展态势。C 语言具有结构化控制语句,是一种结构化程序设计语言,它的数据类型与 PASCAL 一样丰富,能编写较大问题的程序。

本章主要介绍 C 语言发展历史及特点、C 语言的基本程序结构、及 Turbo C 上机的一些主要操作。

1.1 C 语言发展历史和特点

要弄清 C 语言的发展史必须先搞清楚下面几个基本概念:

1. 机器语言:一台未装任何软件的计算机叫裸机,它只能识别一组由“0”,“1”序列构成的代码,我们对计算机所作的任何操作都必须转化成“0”,“1”代码(包括指令和数据)。

2. 汇编语言:一种低级语言,它比机器语言进了一步,指令改成了一些助记符,如 add, mov 等,但所编写的程序必须经过编译以后才能为计算机所接受,编译以后产生目标代码程序。上述两种程序的编写,调试都比较难。

3. 高级语言:大多数高级语言是面向问题的程序设计语言,用高级语言编写程序,比较接近现实思维,它所编写的程序也必须经过翻译成目标代码,高级语言离硬件比较远,它所编写的程序可读性好,可移植性好,程序的编写,调试较容易。

1.1.1 C 语言发展史

在 C 语言出现之前,很多系统软件都是由汇编语言编写的,因为汇编语言是低级语言,它能直接控制计算机硬件,而且离硬件较近,所编写出的程序运行速度较快,但是汇编语言所编写出来的程序依赖于计算机硬件,所以程序的可移植性差,高级语言所编写的程序尽管可移植性好,但它不能直接控制计算机的硬件,是面向问题的程序设计语言,只能写应用软件,不适合写系统软件。人们开始寻找兼有汇编语言和高级语言两方面优点的,即适合于开发系统软件,又适合于写应用程序的语言工具,C 语言就是在这样一个背景下诞生的。

C 语言是在 B 语言的基础上发展起来的,它的根源可以追溯到 ALGOL 60。1960 年出现的 ALGOL 60 是一种面向问题的高级语言,它离硬件比较远,不宜用来编写系统程序。1963 年英国的剑桥大学推出了 CPL(Combined Programming Language)语言。CPL 语言在 ALGOL 60 的基础上更接近硬件一些,但规模比较大,难以实现。1967 年英国剑桥大学的 Martin Richards 对 CPL 语言作了简化,推出了 BCPL(Basic Combined Programming Language)语言。1970 年美国贝尔实验室的 Ken Thompson 以 BCPL 语言为基础,又作了进一步简化,设计出了很简单的而且很接近硬件的 B 语言(取 BCPL 的第一个字母),并用 B 语言写了第一个 UNIX 操作系统,在 PDP-7 上实现。1971 年在 PDP-11/20 上实现了 B 语言,并写了 UNIX 操作系统。但 B 语言过于简单,功能有限。1972 年至 1973 年间,贝尔实验室的 D.M. Ritchie 在 B 语言的基础上设计出了 C 语言(取 BCPL 的第二个字母)。C 语言既保持了 BCPL 和 B 语言的优

点(精练,接近硬件),又克服了它们的缺点(过于简单,数据无类型等)。最初的C语言只是为描述和实现UNIX操作系统提供一种工作语言而设计的。1973年,K. Thompson和D. M. Ritchie两人合作把UNIX的90%以上用C改写(即UNIX第5版。原来的UNIX操作系统是1969年由美国的贝尔实验室的K. Thompson和D. M. Ritchie开发成功的,是用汇编语言写的)。

后来,C语言多次作了改进,直到1975年UNIX第6版公布后,C语言的突出优点才引起人们普遍注意。C语言和UNIX可以说是一对孪生兄弟,在发展过程中相辅相成。1978年以后,C语言已先后移植到大、中、小、微型机上,已独立于UNIX和PDP了。现在C语言已风靡全世界,成为世界上应用最广泛的几种计算机语言之一。1983年,美国标准化协会(ANSI)根据C语言问世以来各种版本对C的发展和扩充,制定了新的标准,称为ANSI C。目前广泛流行的各种版本的C语言编译系统虽然基本部分是相同的,但也有一些不同。在微机上使用的有Microsoft C, Turbo C, Quick C等,它们的不同版本又略有差异,因此读者应了解所用的计算机系统的C编译的特点和规定(可以参阅有关手册)。

1.1.2 C语言的特点

C语言之所以能以如此快的速度迅猛发展,必定有自己的一些特点。这些特点大都是C的优点,下面就C语言的主要特点简述如下:

1.C语言既适合于写系统软件又适应于写应用软件。而且能直接控制计算机的硬件,如能直接控制数据的位等。

2.C语言是一种高级语言,具有9种结构化的控制语句,包含结构化程序的三种基本结构:顺序结构、选择结构、循环结构。并且由此三种结构能构造出任意复杂的程序结构。C函数是C程序的基本单位,由C函数可以象小孩子搭积木样构成C程序。因此,C程序设计语言是一种很好的结构化程序设计语言。

3.C的运算符很丰富,有34种之多,它把括号,赋值号(=),逗号(,)等都处理为一个运算符,使得C语言的表达式既灵活又多样化。C语言表达式的灵活多样性是初学者学习C的一难点,大家在学习C的开始建议要抓住C的基本内容,不要刻意追求那些灵活的东西,等到C语言的基本内容熟悉以后,再回过头来掌握C的提高部分(根据我们多年的教学经验,C有三大难点:一是数据输入输出到位,二是表达式的灵活多样性,三是指针)。

4.C语言数据结构很丰富,数据结构主要是通过类型来体现,C语言有整型、实整、字符型、结构体类型、共同体类型等,还可以由用户自己定义类型。C语言的数据类型几乎和PASCAL一样多,用C语言也能描述算法,所以新版的数据结构很多是用C来描述算法的。

5.C语言离硬件相比其它高级语言来说要更近一些,C语言编译产生的目标代码质量较高,只比汇编语言低10%~20%,程序执行的效率也较高。

6.C语言是一种高级语言,它所编写的程序可移植性好,所编写的程序不作修改或稍作修改就可以移植到其它不同型号的机器上。

上述简要介绍了一下C语言的主要特点,C语言除上述特点之外,还有一些特点,如C语言简洁、紧凑、使用方便、灵活,使用数组时C对下标超界不作检查等。C语言比其它高级语言来说要难学,编写C程序对程序设计者要求也要高。

1.2 C 程序的基本结构

初学者一般对程序比较感兴趣,下面我们就给出两个程序的例子,通过程序实例,使大家对程序有一个感性认识,进而介绍 C 程序的基本结构。

例 1.1

```
print_star()
{
    printf("*****\n");
}
print_message()
{
    printf("欢迎使用本系统!\n");
}
main()
{
    print_star();
    print_message();
    print_star();
}
```

main 总函数
print

本程序中一共有三个函数,一个是主函数 main(),还有二个用户所定义的 print_star() 函数和 print_message() 函数,这两个函数都是无参函数,在这两个函数中都只包含一个 printf() 函数,printf() 是 Turbo C 中的一个库函数,它的功能是实现输出操作(C 无输入输出语句,输入输出都是通过库函数来实现的)。计算机执行总是从 main 函数开始,因此,任何一个 C 程序都必须包含一个主函数且仅包含一个主函数 main()。在此 C 程序中,main() 函数通过 print_star() 和 print_message() 两语句调用了用户所定义的二个无参函数。本程序的输出结果为:

欢迎使用本系统!

例 1.2

```
#include "math.h"
main()
{
    float x, y, z;
    float area();
    scanf("%f%f", &x, &y);
    z = area(x, y);
    printf("z = %f", z);
}
```

```
float area(a, b)
float a, b;
{
    float c;
    c = sqrt(a * b);
    return(c);
}
```

这个 C 程序包含二个函数, 一个是主函数 `main()` 函数, `main()` 函数只实现数据的输入输出操作。 `main()` 函数调用了 `area()` 函数, `area()` 是有参函数, `a, b` 是它的两个形参, 它们分别接受由实参 `x, y` 传过来的二个数据(假定在运行程序时我们给 `x` 送了 1.1, `y` 送了 2.2)。则 `a, b` 的数据分别是 1.1, 2.2, `area` 函数的功能是: 计算二个传过来的数据的乘积的开方, 这个乘积的开方装在一个变量 `c` 中, 然后通过 `return(c)` 带回调用它的主函数中(`c` 作为函数的函数值带回赋给变量 `z`), 然后调用标准输出库函数 `printf()` 输出结果。

若输入为:

1.1 2.2<CR> (<CR>代表回车符)

则本程序的输出结果为:

`z = 1.555635`

我们通过二个实例直观地看到了 C 程序, 下面我们就几个方面来简述 C 程序的基本结构:

1. 任何一个 C 程序都必须包含且仅包含一个主函数 `main()`, 计算机执行 C 程序都是从 `main()` 函数开始。

2. 一个 C 程序除了必须有一个主函数 `main()` 外, 还可包含用户所定义的函数若干个和一些库函数。如果包含了库函数(除 `printf()`, `scanf()` 外), 必须用文件包含命令, 把它包含到 C 程序中来。如例 1.2, 我们用到了 `sqrt()` 数学库函数, 所以我们在程序开头就书写了如下一条文件包含命令:

```
#include "math.h"
```

3. 主函数 `main()` 可以通过函数调用语句调用用户所定义的函数。如 `print_star()`, `print_message()`, `z = area(x, y)` 等都是函数调用语句。用户所定义的函数不能调用 `main()` 函数, 用户所定义的函数可以相互调用。

4. 用户所定义的函数有有参函数和无参函数之分, 不论是有参函数还是无参函数, 函数名之后的一对圆括号不能少, 少了就不叫函数。函数名是用户所取的, 用一个合法标识符(关于标识符的概念在第二章)表示。

5. 用户定义函数按如下形式:

函数值的类型 函数名(形参表列)

形参说明;

{其它说明语句;

执行语句;}

以例 1.2 中的 `area` 函数为例说明如下:

```
float area(a, b) /* 函数值类型 float, 函数名是 area a, b 是它的两个形参 */
float a, b;      /* 说明两个形参的类型为 float 型 */
```

```

| float c;          /* 说明函数除形参之外的其它变量 */
c=sqrt(a * b);      /* 执行语句,计算两个数乘积的开方 */
return(c);|        /* 执行语句,把 c 作为函数值带回 main 函数 */

```

6. C 程序书写格式比较自由,不象 FORTRAN 规定那么多。大家在书写程序时注意下面几点:

- (1) C 程序中主要以小写字母为主,严格区分字母的大小写。关键字如 int 等都必须小写。
- (2) C 程序由 C 语句组成,一般一行只书写一个 C 语句,“;”是 C 语句的分隔符,是语句必须的组成部分(当然 C 并未规定一行只写一个 C 语句,这样做,主要为了便于程序的阅读)。
- (3) 编写程序尽量采用结构化方法,输入归输入,输出归输出,处理归处理。这样调试和阅读程序都很方便。

1.3 Turbo C 上机操作

一个 C 程序编写好以后,可以在三种编译系统环境下编译,一种是在 Microsoft C,另一种是 Quick C,还有一种是 Turbo C。

我们就 PC 机上用得比较多的 Turbo C,介绍其主要操作。

1. 源程序的编辑

一个手工编写好的 C 程序要送入计算机,可以在任何一个文字编辑器下编辑,例如可以在 WPS 下用“N”命令编辑。(注意被编辑的 C 程序取名时其扩展名为 .C)也可以在 Turbo C 集成开发环境下编辑。建议使用后一种,具体在计算机上操作可以按如下步骤进行:

启动 Turbo C

C>CD \ TC <CR>

C: \ TC>TC<CR>

进入 Turbo C 集成开发环境屏幕上出现 Turbo C 主菜单,编辑窗口等。主菜单为:

FILE EDIT RUN COMPILE PROJECT OPTIONS DEBUG

当光块覆盖 FILE 时(没有在此项上可以通过光标键,在主菜单上移动光标块)按下 Enter 键,此时又弹出一个子菜单如图 1.1 所示。

Load	F3
Pick	Alt - F3
New	
Save	F2
Write to	
Directory	
Change dir	
Os shell	
Quit	Alt - x

图 1.1

Load	File	Name
	a.	c

图 1.2

此时可以选择 Load 或 New, 当选择了 Load 时, 此时又出现一个子菜单如图 1.2 所示。

这时要求用户输入文件名, 例如输入 a.c 后按 Enter 键进入编辑窗口。如果 a.c 不存在, 则表示建立一个新文件, 这时我们可以把 C 程序送入计算机; 如果 a.c 已经存在, 则将文件显示在编辑窗口, 可以根据需要重新编辑修改。在 C 程序录入过程中经常有增、删、改等操作, 小规模增改只须将光标移到合适的位置后进行处理, 与 WPS 的文字编辑修改类似, 稍大的修改可以利用下面功能键。

<Ctrl> - kb	设置块首	<Ctrl> - kk	设置块尾
<Ctrl> - kv	块移动	<Ctrl> - kc	块复制
<Ctrl> - y	删除光标在行	<Ctrl> - n	插入一行
<Ctrl> - ky	删除块	<Ctrl> - kh	块取消

F10 功能键可以使光标在窗口与主菜单之间相互切换。

2. 编译 C 程序

编译是对 C 程序进行语法检查(用功能键 F9), 看所编写的 C 程序是否符合 C 语法的规定, 如果编译发现语法错误, 计算机会在屏幕下方给出错误提示信息。要改正错误, 可按以下方法: 当出现编辑窗口中的 press any key 提示时按空格键, 这时编辑窗口中出现一亮条标志着错误的相应位置, 再按回车键, 光标移到编辑窗口错误产生处, 可以对源程序作进一步的修改, 修改好以后再编译, 直到 succss 编译通过为止。此时产生一个与源程序同名但扩展名为 .obj 的目标文件和扩展名为 .exe 的可执行文件。

3. C 程序的运行

C 程序的运行可以使用下面二种方式:

在 C 集成开发环境下用 <ctrl> + <F9> 运行 C 程序。若程序需要输入数据, 在操作系统提示符下光标位置处输入程序需要的相关数据, 程序运行以后返回 C 编辑状态。用这种方法运行程序, 如果源程序没有被编译或编译后的源程序被修改过, 则在运行之前还会自动编译。

运行程序的另一种方法是: 在操作系统下运行 C 程序, 用这种方法运行程序必须先将程序编译好, 方能运行。用 Alt-x 退出到系统 DOS 提示符下输入 C 程序名运行那个可执行文件也就是运行 C 程序。例如产生了一个可执行文件 a.exe 时可以按如下方法运行:

c: >a<CR>

4. 看结果

C 程序运行结果不在 C 编辑状态下显示。当我们用 <ctrl> + <F9> 运行 C 程序以后, 它又回到了 C 编辑状态。此时应用 <Alt> + <F5> 看结果。如果此时发现结果不对, 这是程序的逻辑错误所致。修改逻辑错误, 要求用户对算法和 C 都比较清楚, 逻辑错误才易找到。排除逻辑错误比排除语法错误要难一些, 这些需要有程序调试的技巧, 程序调试的技巧要在程序调试过程中逐步积累。

上面我们从一个 C 程序的编辑、编译、运行及运行以后怎样看结果, 简述了其操作步骤, C 的上机操作远不至这些, 更为详细的操作请查看 C 语言上机指导的有关书籍。

习 题 一

1.1 C 语言有哪些主要特点? 你能领会到哪几个特点?