21世纪高等院校规划教材

计算机程序设计基础教程

上册

姬 涛 周启生 主编

中国传媒大学出版社

图书在版编目(CIP)数据

计算机程序设计基础(上册)教程/姬涛,周启生主编. —北京:中国传媒大学出版社,2010.12

ISBN 978-7-5657-0121-4

Ⅰ. ①计… Ⅱ. ①姬… ②周… Ⅲ. ①电子计算机-高等学校:技术学校-教材 Ⅳ. ①TP3

中国版本图书馆 CIP 数据核字(2010)第 095052 号

计算机程序设计基础(上册)教程

主　　编:姬　涛　周启生

策　　划:胡　云

责任编辑:黄松毅

装帧设计:亿辰时代

责任印制:曹　辉

出 版 人:蔡　翔

出版发行:中国传媒大学出版社　(原北京广播学院出版社)

社　　址:北京市朝阳区定福庄东街 1 号　　**邮编**:100024

电　　话:86-10-65450532 或 65450528　　**传真**:010-65779405

网　　址:http://www.cucp.com.cn

经　　销:全国新华书店

印　　刷:北京市彩虹印刷有限责任公司

开　　本:787mm×1092mm　1/16

印　　张:22.5

字　　数:360 千字

版　　次:2010 年 12 月第 1 版　2010 年 12 月第 1 次印刷

ISBN 978-7-5657-0121-4

前　言

“计算机程序设计基础”是大学计算机基础教学系列中的核心课程，主要介绍程序设计语言的基本知识和程序设计的方法，其内容以程序设计语言的语法知识和程序设计技术的基本方法为主，同时包括程序设计方法学、数据结构与算法基础等方面的初步内容。通过学习，使学生掌握计算机程序设计的基本思想和方法，初步掌握应用计算机的能力，并为后续课程的学习创造条件。

本套教材分为《计算机程序设计基础》和《计算机程序设计基础上机指导与习题》，力求详细介绍C语言结构化程序设计，并兼容了计算机等级考试的要求。《计算机程序设计基础》教材是以程序设计的基础理论为主，重点介绍程序设计的基本理论和方法及C语言的语法知识。《计算机程序设计基础上机指导与习题》教材与《计算机程序设计基础》配套，包括实验上机指导、实验思考题、练习题、二级考试等内容。

本书是以计算机基础课程教学指导委员会制定的，以“计算机程序设计基础课程教学基本要求”为依据，结合近年来C语言程序设计基础教学实践经验编写而成，其内容涵盖了C语言结构化程序设计的基本理论、基本概念、方法和规范。是以培养学生掌握程序设计的基本理论、方法及计算机应用编程能力为目标。教材中的例题都在Visual C++6.0集成开发环境下编译通过。

本书以具体案例——学生成绩管理系统为主线，将C语言程序设计的各个知识点分配到各个章节中详细介绍，并逐步实现该系统的全部功能。通过该案例，使学生对C语言程序设计的过程和所涉及的知识有一个清晰的理解，力求理论与实践相结合，有利于培养学生分析问题和解决问题的能力。

本书共分为8章，内容包括：

第一章程序设计概述。介绍程序、程序设计、算法等基本概念，程序设计的过程和程序设计的方法及C语言字符集、词法符号和C语言程序的基本结构。

第二章数据类型、运算符与表达式。介绍C语言的数据类型、各种类型的常量、变量的声明和使用方法，各种类型运算符及表达式求值的运算规则。

第三章程序控制结构。介绍结构化程序设计的基本概念和结构化程序设计的方法，C语言的顺序控制、选择控制、循环控制语句及应用问题的求解方法。

第四章函数与预处理。介绍用函数实现模块化程序设计的思想，函数的定义、函数的声明、函数的调用的基本概念和方法，函数的返回值及其类型，函数参数传递机制，递归函数，变量的作用域和存储类型及预处理等。

第五章数组。介绍数组的概念，一维数组的定义、初始化及应用；二维数组的定义、初始化及其应用；字符串与字符数组的概念及其应用。

第六章指针。介绍指针的基本概念，指针变量的定义及其相关运算，指针与函数，指针与数组，动态内存分配等。

第七章结构体与共用体。介绍结构体、共用体等自定义数据类型的机制，结构体类型与结构体变量，结构体数组，结构体与函数，链表的概念和常用操作，共用体的概念及应用。

第八章文件。介绍文件的基本概念以及对文件的常用操作。

附录包括 ASCII 码表、C 语言运算符、常用 C 标准库函数。

本书第 1～2 章由田红梅编写，第 3 章由郭炜编写，第 4 章由刘菲编写，第 5 章由姬涛编写，第 6 章由江红编写，第 7～8 章由周启生编写。全书由姬涛和周启生统稿与审定。特别感谢学校一级本科教学团队责任教授刘宝忠教授对本书的指导与审阅。在本书编写过程中，还得到了王海晖教授、庄鹏老师的大力支持，在此也表示衷心感谢！

本书作者都是长期从事程序设计教学的教师，本书凝集了各位教师多年的教学实践经验，由于计算机科学技术发展迅速，程序设计的教学内容、方法和手段日新月异，加之水平有限，书中不足之处在所难免，敬请读者批评指正，以便在今后进一步完善。

编者

2010 年 10 月

目 录

第 1 章 程序设计概述 …… 1
1.1 概述 …… 1
1.1.1 程序与程序设计 …… 1
1.1.2 程序设计语言 …… 2
1.2 算法 …… 3
1.2.1 算法的概念 …… 3
1.2.2 算法的描述方式 …… 3
1.2.3 简单算法举例 …… 4
1.3 程序设计过程 …… 6
1.3.1 解决问题的基本步骤 …… 6
1.3.2 C 语言程序的设计过程 …… 6
1.4 C 语言简介 …… 7
1.4.1 C 语言的字符集与词法符号 …… 8
1.4.2 C 语言程序的基本结构 …… 10
1.4.3 C 语言程序的书写规则 …… 11
1.5 程序设计方法 …… 11
1.5.1 结构化程序设计 …… 12
1.5.2 面对象程序设计 …… 12
1.6 开发环境简介 …… 12
1.6.1 启动 vc++6.0 …… 13
1.6.2 创建工程与文件 …… 13
1.6.3 编辑程序 …… 15
1.6.4 编译与连接 …… 15
1.6.5 运行 …… 16
1.6.6 关闭 …… 17
1.7 案例应用 …… 17
第 2 章 数据类型、运算符与表达式 …… 19
2.1 数据类型 …… 19
2.1.1 基本数据类型 …… 20
2.1.2 构造数据类型 …… 22
2.1.3 其他数据类型 …… 23
2.2 常量与变量 …… 23
2.2.1 常量 …… 23

2.2.2 变量 …… 28
2.3 运算符与表达式 …… 32
2.3.1 运算符 …… 32
2.3.2 表达式 …… 50
2.3.3 类型转换 …… 50
2.4 标准输入/输出 …… 52
2.4.1 printf()函数 …… 52
2.4.2 scanf()函数 …… 55
2.4.3 其他输入输出函数 …… 57
2.5 数学函数 …… 58
2.6 随机数发生器函数 …… 59
2.7 案例应用 …… 60
第3章 程序控制结构 …… 63
3.1 程序的基本结构 …… 63
3.2 语句 …… 64
3.2.1 声明语句 …… 64
3.2.2 表达式语句 …… 65
3.2.3 复合语句 …… 65
3.2.4 控制语句 …… 65
3.2.5 空语句 …… 65
3.3 顺序结构 …… 66
3.4 选择结构 …… 68
3.4.1 if 语句 …… 68
3.4.2 switch 语句 …… 78
3.4.3 程序应用举例 …… 82
3.5 循环结构 …… 85
3.5.1 while 语句 …… 85
3.5.2 do—while 语句 …… 89
3.5.3 for 语句 …… 93
3.5.4 几种循环的比较 …… 95
3.5.5 循环的嵌套 …… 96
3.5.6 程序举例 …… 97
3.6 转向语句 …… 101
3.6.1 break 语句 …… 101
3.6.2 continue 语句 …… 102
3.7 结构化程序设计的方法 …… 103
3.8 案例应用 …… 104

第 4 章　函数与预处理 …… 109
4.1　概述 …… 109
4.1.1　函数的概念 …… 109
4.1.2　函数分类 …… 110
4.2　函数的定义与声明 …… 111
4.2.1　函数的定义 …… 111
4.2.2　函数声明与函数原型 …… 113
4.3　函数调用 …… 114
4.4　函数返回类型与返回值 …… 118
4.5　函数的参数 …… 120
4.5.1　形式参数与实际参数 …… 120
4.5.2　值传递与地址传递 …… 120
4.6　递　归 …… 124
4.6.1　递归的概念 …… 124
4.6.2　程序举例 …… 126
4.7　变量作用域 …… 129
4.7.1　内部变量 …… 129
4.7.2　外部变量 …… 131
4.7.3　作用域规则 …… 132
4.8　变量存储类别 …… 133
4.8.1　变量生存期 …… 133
4.8.2　auto 变量 …… 134
4.8.3　static 变量 …… 135
4.8.4　register 变量 …… 136
4.8.5　extern 变量 …… 137
4.9　内部函数与外部函数 …… 140
4.9.1　内部函数 …… 140
4.9.2　外部函数 …… 141
4.10　预处理 …… 144
4.10.1　文件包含 …… 144
4.10.2　宏定义 …… 146
4.10.3　条件编译 …… 149
4.11　案例应用 …… 152
第 5 章　数　组 …… 157
5.1　数组概述 …… 157
5.2　一维数组 …… 158
5.2.1　一维数组定义与初始化 …… 158
5.2.2　一维数组元素的引用 …… 161

5.2.3 一维数组作为函数参数 …… 167
5.2.4 一维数组应用举例 …… 170
5.3 二维数组 …… 179
5.3.1 二维数组定义与初始化 …… 179
5.3.2 二维数组数组元素的引用 …… 182
5.3.3 二维数组应用举例 …… 184
5.4 字符数组 …… 186
5.4.1 字符串与字符数组 …… 186
5.4.2 字符数组的定义与初始化 …… 187
5.4.3 字符数组的引用 …… 189
5.4.4 字符串输出和输入 …… 190
5.4.5 字符数组应用举例 …… 193
5.4.6 字符串处理函数 …… 195
5.4.7 字符串数组 …… 198
5.5 案例应用 …… 202
第6章 指 针 …… 206
6.1 指针的概念 …… 206
6.1.1 地址与指针 …… 206
6.1.2 指针的定义与初始化 …… 208
6.1.3 指针的运算 …… 210
6.2 指针与函数 …… 217
6.2.1 指针作函数的参数 …… 217
6.2.2 函数返回指针 …… 219
6.2.3 指向函数的指针 …… 221
6.3 指针与数组 …… 224
6.3.1 指针对数组元素的访问 …… 224
6.3.2 字符指针 …… 235
6.3.3 指向数组的指针 …… 241
6.3.4 指针数组 …… 244
6.3.5 指向指针的指针 …… 251
6.4 动态内存分配 …… 254
6.4.1 动态内存分配的含义 …… 254
6.4.2 动态内存分配的步骤 …… 255
6.4.3 常用的动态内存管理函数 …… 255
第7章 结构体与共用体 …… 263
7.1 结构体类型与结构体变量 …… 263
7.1.1 结构体类型的定义 …… 264
7.1.2 结构体变量的定义与初始化 …… 265

7.1.3 结构体变量的引用 …… 268
7.1.4 指向结构体变量的指针 …… 272
7.1.5 关键字 typedef 的用法 …… 276
7.2 结构体数组 …… 277
7.2.1 结构体数组的定义 …… 277
7.2.2 结构体数组的初始化 …… 279
7.2.3 结构体数组的应用 …… 280
7.3 结构体与函数 …… 283
7.4 动态数据结构 …… 285
7.4.1 链表的定义 …… 286
7.4.2 链表的的特点及操作原理 …… 288
7.4.3 动态链表的建立 …… 289
7.4.4 链表的删除操作 …… 291
7.4.5 链表的插入操作 …… 293
7.5 共用体 …… 295
7.5.1 共用体类型的定义和引用 …… 295
7.5.2 共用体类型的初始化 …… 297
7.5.3 共用体类型举例 …… 297
7.6 案例应用 …… 301
第 8 章 文 件 …… 304
8.1 文件的基本概念 …… 304
8.1.1 文件概述 …… 304
8.1.2 文件的类别 …… 305
8.1.3 文件的操作流程 …… 307
8.1.4 文件的定义 …… 307
8.2 常用文件操作的标准函数 …… 308
8.2.1 文件的打开与关闭 …… 308
8.2.2 文本文件的读写 …… 311
8.2.3 二进制文件的读写 …… 317
8.2.4 文件的随机访问与定位 …… 319
8.2.5 文件使用举例 …… 321
8.3 案例应用 …… 323

附录 A ASCII 码表 …… 339
附录 B 运算符 …… 342
附录 C 常用 C 标准库函数 …… 344

第 1 章　程序设计概述

计算机所能完成的工作以及怎样完成工作都是由人们事先编好的程序来控制的。程序设计是人们为了通过计算机实现某一功能或解决问题，而使用某种程序设计语言编写程序并实现结果的过程。设计计算机程序常常借助于程序设计工具，如 Visual C++ 6.0 集成开发环境。

本章主要介绍程序和程序设计的概念、程序设计语言的发展历程、算法的基本概念、程序设计的过程、C 语言程序的组成和基本结构，以及简要介绍了程序设计基本方法、Visual C++6.0 开发环境以及使用结构化程序设计方法设计一个小型学生成绩管理系统的案例。

通过本章的学习，读者应掌握程序设计的基本概念，能用 C 语言设计一个简单的程序，熟悉 Visual C++6.0 开发工具的使用。

1.1　概述

1.1.1　程序与程序设计

程序通常指完成某项事务的执行过程，是一系列有序的工作步骤，它有方式、步骤等含义。在日常生活中，往往强调按程序办事，如“业务流程”、“司法程序”等。这些事务中，每个步骤的顺序一般是不能颠倒的，否则将不能顺利地完成这项事务。例如，到自动取款机上取款，可以描述为以下步骤：

1. 插入银行卡
2. 输入密码
3. 选择服务种类
4. 输入取款金额
5. 取钞
6. 退卡

这就是一个简单的程序，按顺序执行这些步骤，就可以完成取款。

计算机程序是指为实现特定目标或解决特定问题而用计算机语言编写的命令序列的集合。计算机本身是不会做任何工作的，它是按照程序中的有序指令来完成相应任务的。只有用事先编好的程序去控制它，才能让它为人服务。人们为了完成某项具体的任务而编写的一系列指令，并将这一系列指令交给计算机去执行，这个过程称为程序设计。

目前，计算机对人们的生活产生了深刻的影响，计算机能完成各种复杂的任务。在 Windows 环境下，只要用鼠标单击某一菜单项，就能使计算机执行某一项操作（例如，复

制、粘贴、删除)。有些人觉到计算机很神秘,不可思议。其实,计算机执行每一个操作都是按照人们事先指定的内容和步骤进行的。人们事先规定出计算机完成某项工作的操作步骤,每个步骤的具体内容由计算机能够理解的指令或语句来描述,这些指令或语句将告诉计算机"做什么"和"怎样做"。

在用计算机处理问题前要考虑很多的细节,写程序时把每个细节的执行过程都要描述清楚。例如,用计算机处理上述取款步骤中,若输入密码不正确时计算机该如何执行,若输入的取款金额多于存款余额又如何处理等。所有问题在编写程序前都要考虑清楚,明确告诉计算机"怎样做"。计算机的功能虽然强大,但更重要的是程序设计人员开发出的各种简单或复杂的程序。正是这些程序,使计算机更好地为人们服务。

1.1.2 程序设计语言

程序是由指令序列组成的,告诉计算机如何完成一个具体的任务。由于现在的计算机还不能理解人类的自然语言,所以还不能用自然语言编写计算机程序。人们要使用计算机,使计算机按人们的意志进行工作,就必须让计算机能理解并执行人们给它的指令。这就需要找到一种人和计算机都能识别的语言——程序设计语言。程序设计语言是用计算机能够理解的语言来表达所设计程序的含义,是人与计算机之间进行交流和通信的工具。

计算机能识别和直接运行的指令序列是二进制代码的形式,但二进制代码却令人感到晦涩难懂。为了避免直接面对机器指令,人们设计了众多的程序设计语言。可分为低级语言和高级语言,低级语言包括机器语言和汇编语言。

1. 机器语言

机器语言是以二进制代码的形式来表示基本的指令集合,每条指令均为 0 和 1 组成的二进制代码串。用机器语言编写的程序计算机能直接识别和执行,运算速度快,占用的存储空间小。但机器语言存在明显的局限性:

(1)程序不容易读写。指令中的二进制代码难以记忆,容易出错。为程序的编写、修改和调试带来了难度。

(2)机器语言程序对计算机硬件的依赖性很强,不容易移植。

(3)指令功能简单,没有按照数据类型分类,与数值计算中的运算符不能对应。

2. 汇编语言

针对机器语言中二进制指令不方便识别和记忆的缺点,汇编语言用指令英文名称的缩写作为助记符代替机器的操作指令,用标号和符号来表示地址、常量和变量。因为计算机只能识别机器指令,所以需要借助汇编语言翻译程序,将符号化的汇编语言转换成机器指令,才能被计算机执行。

汇编语言便于识别和记忆,执行效率也比较高,但没有解决机器语言的后两个局限性,仍然不能让人满意。于是,出现了高级程序设计语言。

3. 高级语言

高级语言中的语句一般采用类似人类自然语言中的自然词汇,使得程序更容易阅读和理解。高级语言提供了丰富的数据类型和运算符,语句功能强大,一条语句往往相当

于多条指令。高级语言还独立于具体的硬件系统，使得程序的通用性、可移植性和编写程序的效率大大提高。

针对各种应用领域，人们设计了很多种高级语言，其中 C 语言既具有其他高级语言的优点，又具有低级语言的许多特点。它功能丰富，移植性强，编译质量高，成为应用最广泛的高级语言之一。

同样，计算机也不能直接识别高级语言编写的程序，也需要把高级语言程序转换成机器语言指令，这种转换由编译器提供。不同的高级语言需要不同的编译程序，采用的翻译方法也有所不同，主要有两种方式：

(1)编译方式：首先针对具体高级语言(例如 C 语言)开发出一个编译软件，将该种语言程序翻译为所用计算机机器语言的等价程序，即目标程序(二进制代码的形式)。最后由计算机执行这个机器语言程序。

(2)解释方式：首先针对具体高级语言(例如 BASIC 语言)开发一个解释软件，该软件读入高级语言程序，并能一步步地按照程序要求工作，完成程序所描述的计算。这种解释软件边解释语句边执行，不生成目标程序。

1.2　算法

1.2.1　算法的概念

算法就是为解决一个具体问题而采取的方法和有限的步骤。计算机算法即计算机所能执行的算法。

计算机算法大致可分为如下两大类：

(1)数值运算算法：解决求数值的问题。例如，判断闰年、求最大公约数、求阶乘、求数列之和等。

(2)非数值运算算法：解决需要用分析推理、逻辑推理才能解决的问题。例如，博弈、查找和分类等。

算法描述了解决问题的方法和途径，采用高效的算法才可能设计出优质的程序。

1.2.2　算法的描述方式

算法的描述可使用自然语言方式、类似于高级程序设计语言的伪代码、程序流程图、N/S 盒图、PAD 图等方式。软件开发的不同阶段描述算法的目的不同，应根据需要选择适当的描述方法。

虽然可使用自然语言描述算法，但是因为自然语言具有多义性、难以准确地表达语义，所以自然语言并不适合用来描述规模较大程序的算法。这里简单介绍几种常用的算法描述方式。

1. 流程图

流程图是一个描述算法的控制流程和指令执行情况的有向图。它是一种比较直观的算法描述方式。下面介绍几个常用的流程图符号，如图 1-1 所示。

●起止框:表示程序的开始或结束。

●处理框:表示算法的某个处理步骤。

●判断框:表示对给定条件进行判断,根据条件是否成立来决定如何执行。

●输入/输出框:表示输入或输出操作。

●流程线:带箭头的直线,表示程序执行的流向。

●连接符:流程图间断处的连接符号,圈中可以标注一个字母或数字。同一个编号的点是相互连接在一起的,流程图画不下时才会分开画,实际上同一编号的点表示同一个点。使用连接符还可以避免流程线的交叉或过长,使流程图更加清晰。

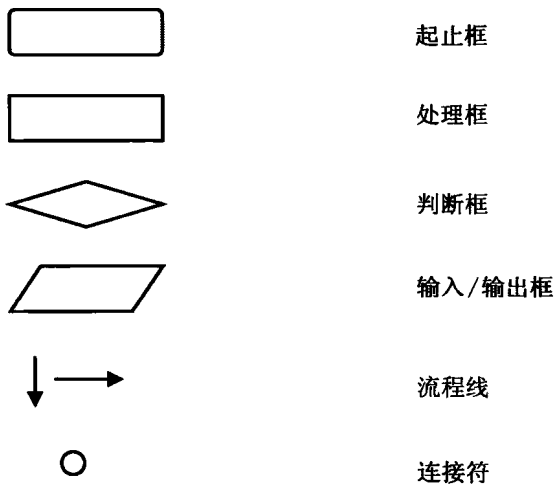

图 1-1 流程图中的常用符号

2. 伪码

伪码方式是用介于自然语言与计算机语言之间的文字及符号来描述算法。用伪码描述算法时书写格式比较自由,易于修改,便于向计算机语言过渡,但它不太直观,且容易出现逻辑上的错误。

3. 程序设计语言

用程序设计语言描述算法其实就是编写程序。计算机是无法识别流程图或伪码的,因此,用流程图或伪码描述的算法还需要将其转化为程序设计的语言程序(例如,C 语言程序)。用程序设计语言描述算法必须遵守相应的语法规则。

1.2.3 简单算法举例

【例 1.1】 用流程图描述计算 n! 的算法。

算法描述如图 1-2 所示。

【例 1.2】 用伪码描述计算 n! 的算法。

开始

输入大于 0 的值给 n

给变量 fac 赋初值 1

给变量 i 赋初值 1

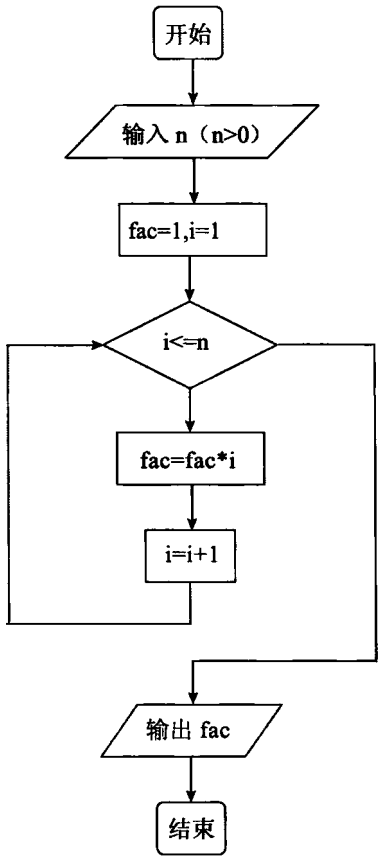

图 1-2　计算 n! 的流程图

当 i<＝n，执行下面的操作：
使 fac＝fac * i
使 i＝i＋1
（循环到此结束）
输出 fac 的值
结束

【例 1.3】 用 C 语言描述计算 n! 的算法。

```
#include<stdio.h>
voidmain()
{
double   fac=1;                    /*定义变量 fac,初值为 1,存放 n! */
int   i=1,n;                       /*定义循环控制变量 i,初值为 1,以及变量 n*/
printf("pleaseinputn(n>0):");      /*输出提示信息*/
scanf("%d",&n);                    /*输入一个整数*/
while(i<=n)                        /*在循环中计算 n! */
{
```

```
fac=fac*i;
i=i+1;
}
printf("%d! =%.0f\n",n,fac);        /*输出计算结果*/
}
```

当输入5时,程序运行结果如图1-3所示。

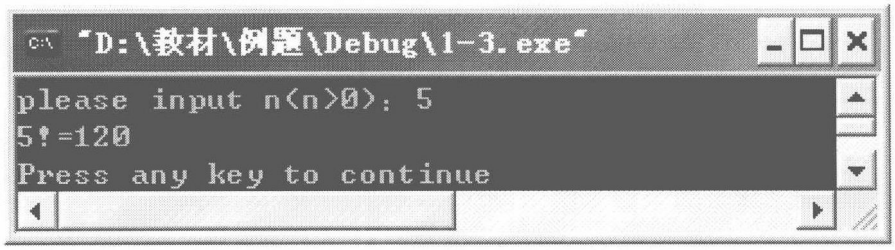

图1-3 例1.3运行结果

1.3 程序设计过程

1.3.1 解决问题的基本步骤

一般来说,用计算机解决问题的过程可分为三步:

第一步,是分析问题,设计一种解决方案;

第二步,是通过程序语言严格描述这个解决方案;

第三步,在计算机上调试这个程序,运行程序,看是否真能解决问题。如果在第三步发现错误,那么就需要仔细分析错误原因,弄清后退到前面步骤去纠正错误。

1.3.2 C语言程序的设计过程

进行C语言程序设计的过程可以分解为以下几个步骤:

(1)**分析问题,确定程序目标。**确定程序需要做什么,并设计解决问题的步骤。仔细考虑程序需要的信息、要进行的计算和操作、要反馈的结果信息等。

(2)**设计程序。**考虑如何组织程序、如何表示数据、数据的处理方法、用户界面信息的显示。

(3)**编辑程序。**通过C语言集成开发环境中的编辑器(如Visual C++6.0)或文本编辑器(如记事本、写字板等)编写程序代码。可考虑添加适当的注释信息,这会为后面的调试和维护带来极大的方便。

(4)**编译。**编译前系统会先执行程序中的预处理命令,生成中间文件。编译时则将该中间文件和C文件转换为机器可识别的二进制代码,生成相应的目标文件(.obj)。在编译过程中要进行词法和语法分析,发现不合法的代码,则以错误(error)或警告(warning)信息进行提示。如果程序编译正确,则进入下一步;如发现错误或警告,就需要设法确定出现问题的位置,回到第3步去修改程序,排除编译中发现的错误,处理警告信息

(通常可忽略)。

(5)**连接。**编译正确完成后,对程序进行连接。把不同的二进制代码片段(程序中包含的文件、多处定义的函数和数据等)连接成完整的可执行文件(.exe)。如果连接发现错误,就需返回前面步骤,修改程序后重新编译。

(6)**运行和调试。**程序正常连接后,会产生一个可执行文件,可以开始程序的运行和调试。此时需要用一些实际数据考查程序的执行效果。如果执行中出了问题,或发现结果不正确,那么就要设法确定错误原因,并回到前面步骤:修改程序,重新编译、连接等。

(7)**维护和修改。**程序被广泛应用后,可能会发现需要对其进行改进的地方,随着输入数据的增多(调试时所用的考查数据毕竟有限,这时会遇到其他可能情况),或许会出现之前没发现的小问题,也可能想到了更好的实现方式。遇到上述情况,可以重新改编程序。

上述设计过程并不是一条直线的过程,有时需要在不同步骤间来回反复,直到确定程序正确为止。良好的设计习惯会简化设计过程、节省时间。

1.4　C 语言简介

C 语言起源于美国贝尔实验室,是从 BCPL 语言和 B 语言演化而来的。BCPL 语言是 1967 年由 MartinRichards 为编写操作系统软件和编译器而开发的语言。1970 年,KenThompson 在模拟了 BCPL 语言的许多特点的基础上开发出了简单而且接近硬件的 B 语言,并用 B 语言编写了第一个 UNIX 操作系统。由于 B 语言依赖于机器,并且过于简单,功能有限。贝尔实验室的 D. M. Ritchie 在 B 语言的基础上设计了 C 语言,于 1972 年实现了最初的 C 语言。1973 年,KenThompson 和 D. M. Ritchie 合作用 C 语言改写了 UNIX。C 语言作为 UNIX 操作系统的开发语言而广为人们认识,当今许多操作系统都是用 C 语言或 C++语言编写的。C 语言在保持了 BCPL 和 B 语言优点的同时,增加了数据类型和其他功能。

由于 C 语言与硬件无关,并且设计严谨,用 C 语言编写的程序能移植到大多数计算机上。C 语言在各种计算机上的快速推广导致了许多 C 语言版本,这些版本通常是不兼容的,人们需要一种标准的 C 语言版本。1983 年,美国国家标准化组织(ANSI)成立了 X3J11 技术委员会进行标准化工作,制定了 ANSIC 标准,该标准也被称为 C89(于 1989 年被 ANSI 批准)。国际标准化组织在 ANSIC 基础上进行了改进,采用了 ISOC 标准,通常也被称为 C90(于 1990 年被 OSI 批准)。因为 ANSI 版本是先出现的,人们习惯使用 ANSIC 这一术语。1994 年开始了标准的修订工作,结果产生了 C99 标准。因为有些编译器没有完全实现 C99 的修改,所以有些修改在一些系统上不可用。

作为一种简短、清楚、高效的程序设计语言,C 语言是应用最广泛的语言之一。它具有以下特点:

●简洁紧凑、方便灵活。C 语言仅有 32 个关键字,9 种控制语句。C 程序表达方式简洁,书写形式自由,主要由小写字母表示。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。

●运算符丰富。C语言包含34种运算符,范围广泛。它把括号、赋值、逗号等都作为运算符处理,使得运算类型丰富,表达式类型多样化,可以实现其他高级语言难以实现的运算。

●数据类型丰富。C语言的数据类型有整型、实型、字符型、枚举型和各种构造类型,还允许用户自定义类型。利用这些数据类型可以实现复杂的数据结构,完成各种类型的数据处理。

●C语言是结构化程序设计语言。它具有结构化的控制语句,包含顺序、选择、循环三种基本结构,结构清晰。函数是构成C语言程序的基本单位,用函数作为程序模块实现程序的模块化,使程序便于使用和维护。

●语法限制不太严格,程序设计自由度大。程序编写者有较大的自由度。

●可直接对硬件进行操作。C语言允许直接访问物理地址,能像汇编语言一样对计算机最基本的位、字节和地址进行操作。它既有高级语言的优点,又具有低级语言的功能。既可以编写系统软件,又可以编写各种应用软件。

●生成目标代码质量高,程序执行效率高。许多高级语言的效率比汇编语言低得多。对同一问题,C语言程序生成的目标代码比汇编程序低10～20%。

●可移植性好。与汇编语言相比,C语言适用范围更广泛。它可用于多种操作系统,也适用于多种机型。

在当今众多的计算机编程语言中,C语言被认为是最受欢迎的语言之一。学习好C语言可作为程序设计语言学习的良好开端。

1.4.1 C语言的字符集与词法符号

1.字符集

字符是组成语言的最基本的元素。C语言字符集是书写程序时允许出现的所有字符的集合,由字母、数字、空白符和特殊符号组成。

●字母:小写字母a～z,大写字母A～Z。

●数字:0～9共10个。

●空白符:空格符、制表符、换行符等统称为空白符。空白符只在字符常量和字符串常量中起作用。在其他地方出现时,只起间隔作用,编译程序对它们忽略不计。因此在程序中使用空白符与否,对程序的编译不发生影响,但在程序中适当的地方使用空白符将增加程序的清晰性和可读性。

●特殊符号:运算符、标点、括号和一些特殊字符。

不在字符集中的符号如汉字或其他可表示的图形符号,可以在两个双引号之间出现,也可以出现在注释行中。

2.词法符号

在C语言中使用的词汇分为:标识符、关键字、运算符、分隔符、常量、注释符等。

(1)标识符

标识符是用来标识程序中的变量、常量、数据类型、数组、函数等的名称,是合法的字符序列。