

21世纪高等学校计算机科学与技术规划教材

C语言程序设计教程

C YUYAN CHENGXU SHEJI JIAOCHENG

主编 杨路明

主审 郭浩志



21 世纪高等学校计算机科学与技术规划教材

C 语言程序设计教程

主 编 杨路明

副主编 谭长庚 沙莎

主 审 郭浩志

北京邮电大学出版社

内 容 简 介

C语言是在当今软件开发领域有着十分广泛应用的一种语言。本书全面地介绍了C语言的基本概念,C语言的数据类型、语句及结构特点,系统地讲述了C语言程序设计的基本方法和技巧。对面向对象程序设计及C++的编程技术也作了初步阐述。

全书选材先进,内容丰富,讲述力求理论联系实际、深入浅出、循序渐进,注重培养读者的程序设计能力,培养良好的程序设计风格和习惯。

本书可作为高等院校计算机程序设计教学用书,也可作为从事计算机应用的科技人员的参考书、培训教材。本书配有CAI教学课件。

为了配合本书的学习,作者还编写了与本书配套的《C语言程序设计上机指导与习题选解》一书,可供学习时参考使用。

图书在版编目(CIP)数据

C语言程序设计教程/杨路明主编. —北京:北京邮电大学出版社,2003

ISBN 7-5635-0737-X

I. C... II. 杨... III. C语言—程序设计—教材 IV. TP312

中国版本图书馆CIP数据核字(2003)第102958号

书 名: C语言程序设计教程

编 著: 杨路明

策 划: 三文工作室

E-mail: sanwen99@mail.edu.cn

责任编辑: 陈露晓 付晓霞

出版发行: 北京邮电大学出版社

社 址: 北京市海淀区西土城路10号(100876)

电话传真: 010-62282185(发行部) 010-62283578(传真)

经 销: 各地新华书店

印 刷: 国防科技大学印刷厂印刷

开 本: 787mm×1092mm 1/16

印 张: 22.75

字 数: 560千字

版 次: 2003年12月第1版 2003年12月第1次印刷

ISBN 7-5635-0737-X/TP·100

定 价: 28.00元

如有质量问题请与发行部联系

版权所有 侵权必究

21 世纪高等学校计算机科学与技术规划教材

编 委 会

- | | | |
|-----|-----|---------------------|
| 主 任 | 陈火旺 | 中国工程院院士 国防科技大学教授 |
| 委 员 | 周立柱 | 清华大学计算机系主任 |
| | 杨放春 | 北京邮电大学计算机科学与技术学院院长 |
| | 杨学军 | 国防科技大学计算机学院院长 |
| | 徐晓飞 | 哈尔滨工业大学计算机科学与技术学院院长 |
| | 李仁发 | 湖南大学计算机与通信学院院长 |
| | 卢正鼎 | 华中科技大学计算机学院院长 |
| | 李志蜀 | 四川大学计算机学院院长 |
| | 戴居丰 | 天津大学信息学院、软件学院院长 |
| | 蒋昌俊 | 同济大学计算机科学与工程系主任 |
| | 何炎祥 | 武汉大学计算机学院院长 |
| | 周兴社 | 西北工业大学计算机系主任 |
| | 陈志刚 | 中南大学信息学院副院长 |
| | 姜云飞 | 中山大学软件学院院长 |
| | 周昌乐 | 厦门大学软件学院院长 |
| | 齐 勇 | 西安交通大学计算机科学与技术系主任 |
| | 赵书城 | 兰州大学计算机学院院长 |
| | 孟祥旭 | 山东大学计算机学院院长 |

序

自 20 世纪 80 年代以来,高等学校计算机教育发展迅速,计算机教育的内容不断扩展、程度不断提高;特别是近十年来,计算机向高度集成化、网络化和多媒体化发展的速度一日千里;社会信息化不断向纵深发展,各行各业的信息化进程不断加速;计算机应用技术与专业的教学、科研工作结合更加紧密;专业课与计算机技术为核心的信息技术的融合促进了学科的发展,专业对学生的计算机应用能力也有更高和更加具体的要求。

基于近年来计算机学科的长足发展,以及国家教育部关于计算机基础教学改革和指导思路,我们确立了这套“21 世纪高等学校计算机科学与技术规划教材”的编写计划与编写思想。教材是教学过程中的“一剧之本”,是高校计算机教学的首要问题。该套系列教材编写计划的制定凝聚了编委会委员们及作者们的心血,是大家多年来在计算机学科教学与研究的成果的体现,得到了陈火旺院士的亲自指导与充分肯定。

这套系列教材由北京邮电大学出版社三文工作室精心策划和组织。在编写过程中,充分考虑了计算机学科的发展和《计算机学科教学计划》中内容和模块的调整,使得整套教材具有科学性和实用性。整套系列教材体系结构按课程设置进行划分。每册教材均涵盖了相应课程教学大纲所要求的内容,既具备学科设置的合理性,又符合计算机学科发展的需要。从结构上遵循教学认知规律,能够满足不同层次院校、不同教学计划的要求。

各册教材作者均为多年来从事教学和研究的专家和学者,他们都有丰富的教学实践经验,所编写的教材体系结构严谨、内容充实、层次清晰、概念准确、论理充分、理论联系实际、深入浅出、通俗易懂。

教材建设是一项长期艰巨的系统工程,尤其是计算机科学技术发展迅速、内容更新快,为使教材更新能跟上科学技术的发展,我们将密切注视计算机科学技术的发展新动向,以使我们的教材编写在内容上不断推陈出新、体系上不断完善成熟,以适应高校计算机教学的需要。

21 世纪高等学校计算机科学与技术规划教材编委会

2003 年 12 月

前 言

C语言概念简洁,数据类型丰富,表达能力强,运算符多而灵活,控制流和数据结构新颖,程序结构性和可读性好,能有利于培养良好的编程风格,易于体现结构化程序设计思想。它既具有高级语言程序设计的特点,又具有汇编语言的功能;既能有效地进行算法描述,又能对硬件直接进行操作;既适合于编写应用程序,又适合于开发系统软件,它是当今世界上应用广泛、最具影响的程序设计语言之一。C语言本身还具有整个语言紧凑整齐,设计精巧,编辑方便,编译与目标代码运行效率高,操作简便,使用灵活等许多鲜明的特点。特别是它的扩充了图形、彩色、窗口等功能以及高效集成开发环境,赢得了广大用户的喜爱,得到了广泛的应用。

本书全面介绍了C语言的概念、特性和结构化程序设计方法。全书共有13章,第1章介绍了C语言程序设计的基本知识。第2章介绍了C语言的基本数据类型、常量和变量以及表达式。第3、4、5章介绍了C语言进行结构化程序设计的基本方法,包括结构化程序的顺序结构、选择结构、循环结构及其设计方法。第6章介绍了函数与编译预处理。第7、8章对C语言的数组、指针作了充分阐述。第9、10章对结构体、共用体作了较详细的介绍。第11章对C语言文件操作作了详细的阐述。第12章介绍了C语言的综合编程应用,对C语言在图形处理以及应用程序与操作系统接口等方面的应用作了较详细的分析;对C语言与汇编语言混合编程、C语言在西文环境下完成中文显示等技术作了较详细的叙述。第13章对C++的基本特性、基本概念以及面向对象程序设计的基本方法进行了初步阐述。

本书是作者在多年从事C语言及计算机专业相关课程的教学实践,在多次编写讲义、教材的基础上编写而成的。内容充实,循序渐进,选材上注重系统性、先进性、实用性。全书精选了大量例题,且都用汉字注释,所有例题均已上机通过,可直接引用。

为配合读者学习本书,我们另编写了一本《C语言程序设计上机指导与习题选解》,作为本书的配套参考书,供读者复习和检查学习效果。

本书由中南大学杨路明主编,谭长庚、沙莎为副主编,由国防科技大学郭浩志教授主审。蒋外文编写第1、2章;沙莎编写第3、4、5、6章;谭长庚编写第7、8、13章;刘卫国编写第9、10章;杨路明编写第11、12章,并负责全书的统稿。

在本书的写作过程中,得到郭浩志教授的热情支持与指导,在此表示衷心感谢。

由于作者水平有限,加之时间仓促,书中错误和不当之处在所难免,敬请读者批评指正。

编 者

2003年12月

目 录

第 1 章 C 语言程序设计概述	(1)
1.1 程序与程序设计语言	(1)
1.1.1 程序的基本概念	(1)
1.1.2 程序设计语言	(1)
1.2 算法及其描述	(3)
1.2.1 算法的概念	(3)
1.2.2 算法的描述方法	(4)
1.3 C 语言的发展及特点	(5)
1.3.1 C 语言的发展概况	(5)
1.3.2 C 语言的特点	(6)
1.4 C 语言程序的基本结构	(7)
1.5 C 语言字符集、标识符与关键字	(8)
1.5.1 C 语言字符集	(8)
1.5.2 标识符与关键字	(9)
1.6 C 语言程序的开发环境	(9)
1.6.1 Turbo C 2.0 介绍	(9)
1.6.2 源程序输入	(11)
1.6.3 编译、连接与运行	(11)
第 2 章 C 语言的基本数据类型与表达式	(12)
2.1 C 语言的基本数据类型	(12)
2.1.1 数据类型概述	(12)
2.1.2 整数类型	(13)
2.1.3 实数类型	(13)
2.1.4 字符类型	(14)
2.2 常量与变量	(15)
2.2.1 常量与符号常量	(15)
2.2.2 变量与变量定义	(16)
2.3 运算符与表达式	(18)
2.3.1 算术运算符与算术表达式	(18)
2.3.2 赋值运算符与赋值表达式	(21)
2.3.3 逗号运算符和求字节运算符	(22)
2.4 数据类型转换	(23)
2.4.1 类型自动转换	(23)

2.4.2 赋值转换·····	(24)
2.4.3 强制类型转换·····	(25)
第3章 顺序程序设计 ·····	(26)
3.1 C语言的基本语句·····	(26)
3.1.1 简单语句·····	(27)
3.1.2 复合语句·····	(28)
3.2 数据输入与输出·····	(29)
3.2.1 字符输入输出函数·····	(29)
3.2.2 格式输出函数 printf ·····	(30)
3.2.3 格式输入函数 scanf ·····	(35)
3.3 程序举例·····	(36)
第4章 选择结构程序设计 ·····	(40)
4.1 关系运算符与关系表达式·····	(40)
4.1.1 关系运算符·····	(40)
4.1.2 关系表达式·····	(40)
4.2 逻辑运算符与逻辑表达式·····	(41)
4.2.1 逻辑运算符·····	(41)
4.2.2 逻辑表达式·····	(42)
4.3 if语句·····	(43)
4.3.1 单分支 if语句 ·····	(43)
4.3.2 双分支 if语句 ·····	(43)
4.3.3 多分支选择结构·····	(44)
4.3.4 if语句的嵌套 ·····	(45)
4.4 switch语句 ·····	(47)
4.5 结构嵌套程序举例·····	(49)
第5章 循环结构程序设计 ·····	(52)
5.1 while语句 ·····	(52)
5.2 do-while语句 ·····	(53)
5.3 for语句 ·····	(54)
5.3.1 for语句的一般形式·····	(54)
5.3.2 for语句中的各表达式含义·····	(54)
5.3.3 for语句与 while语句的比较·····	(55)
5.3.4 for语句的变形·····	(56)
5.4 break、continue 和 goto语句 ·····	(57)
5.4.1 break语句 ·····	(57)
5.4.2 continue语句·····	(58)
5.4.3 goto语句 ·····	(59)
5.5 循环的嵌套·····	(59)
5.6 复合结构程序举例·····	(61)

第 6 章 函数与编译预处理	(64)
6.1 模块化程序设计与函数	(64)
6.1.1 模块与函数	(64)
6.1.2 模块设计原则	(66)
6.1.3 算法描述简介	(66)
6.2 函数的定义与调用	(68)
6.2.1 标准库函数	(68)
6.2.2 函数的定义	(68)
6.2.3 函数的调用	(70)
6.2.4 参数的传递	(72)
6.3 函数的递归调用	(74)
6.4 变量作用域与存储方式	(79)
6.4.1 变量的作用域	(79)
6.4.2 变量的存储方式	(82)
6.5 编译预处理	(86)
6.5.1 宏定义	(87)
6.5.2 文件包含	(90)
6.5.3 条件编译	(90)
6.6 函数设计举例	(92)
第 7 章 数组	(95)
7.1 一维数组	(95)
7.1.1 一维数组的定义	(96)
7.1.2 一维数组元素的引用	(96)
7.1.3 一维数组的存储结构与初始化	(98)
7.2 二维数组	(100)
7.2.1 二维数组的定义	(100)
7.2.2 二维数组元素的引用	(102)
7.2.3 二维数组的存储	(103)
7.2.4 二维数组的初始化	(104)
7.3 数组的应用	(107)
7.4 字符数组与字符串	(111)
7.4.1 字符数组的定义与初始化	(111)
7.4.2 字符串的概念及存储	(111)
7.4.3 字符串的输入与输出	(112)
7.4.4 字符串处理函数	(115)
7.5 数组作为函数的参数	(118)
7.6 程序举例	(122)
第 8 章 指针	(127)
8.1 指针与指针变量	(127)

8.1.1 指针的概念	(127)
8.1.2 指针变量的定义与初始化	(128)
8.1.3 指针运算	(130)
8.1.4 多级指针	(133)
8.2 指针与函数	(135)
8.2.1 指针作为函数参数	(135)
8.2.2 指针函数	(137)
8.2.3 指向函数的指针	(138)
8.3 指针与数组	(141)
8.3.1 指向一维数组的指针	(142)
8.3.2 二维数组与多维数组的指针表示法	(144)
8.4 指针与字符串	(149)
8.4.1 字符串的指针表示法	(149)
8.4.2 字符串数组	(151)
8.5 指针数组与命令行参数	(151)
8.5.1 指针数组	(151)
8.5.2 指针数组与命令行参数	(153)
8.6 程序举例	(155)
第9章 结构体数据类型与链表	(160)
9.1 结构体类型的定义	(160)
9.2 结构体类型变量	(161)
9.2.1 结构体类型变量的定义	(161)
9.2.2 结构体变量的使用	(163)
9.2.3 结构体变量的初始化	(164)
9.2.4 结构体变量的输入与输出	(165)
9.3 结构体类型数组	(166)
9.3.1 结构体类型数组的定义	(166)
9.3.2 结构体类型数组的初始化	(166)
9.3.3 结构体数组的使用	(167)
9.4 结构体类型指针	(168)
9.4.1 指向结构体变量的指针	(168)
9.4.2 指向结构体数组元素的指针	(169)
9.5 结构体与函数	(170)
9.5.1 结构体变量作函数参数	(170)
9.5.2 指向结构体变量的指针作为函数参数	(171)
9.5.3 函数的返回值为结构体类型	(173)
9.6 链表	(173)
9.6.1 链表概述	(173)
9.6.2 内存动态管理函数	(174)
9.6.3 链表的基本操作	(176)
9.7 程序举例	(184)

第 10 章 共用体与枚举类型	(189)
10.1 共用体类型与共用体变量	(189)
10.2 共用体变量的引用	(190)
10.3 共用体变量的应用	(192)
10.4 位运算与位段结构	(193)
10.4.1 位运算	(193)
10.4.2 位段结构	(194)
10.5 枚举类型	(196)
10.6 类型定义	(198)
第 11 章 文件	(200)
11.1 文件概述	(200)
11.1.1 文件	(200)
11.1.2 数据文件的存储形式	(201)
11.1.3 标准文件与非标准文件	(201)
11.1.4 文件存取方式	(202)
11.2 标准文件操作	(203)
11.2.1 标准文件 FILE 结构指针	(203)
11.2.2 标准文件的打开操作	(204)
11.2.3 关闭标准文件	(206)
11.2.4 标准文件读写操作	(206)
11.2.5 标准文件的随机读写	(212)
11.3 非标准文件操作	(214)
11.3.1 建立非标准文件	(214)
11.3.2 非标准文件打开和关闭	(215)
11.3.3 非标准文件的读写	(216)
第 12 章 C 语言综合应用程序示例	(218)
12.1 多模块编程方法	(218)
12.1.1 程序模块组织	(218)
12.1.2 模块之间的通信	(219)
12.1.3 包含文件 .h 的应用	(220)
12.2 图形处理应用	(220)
12.2.1 基本图形处理库函数	(220)
12.2.2 图形处理库函数应用示例	(231)
12.3 操作系统接口应用	(234)
12.3.1 DOS 操作系统简介	(234)
12.3.2 操作系统接口库函数	(235)
12.3.3 鼠标器及其应用	(237)
12.4 C 语言与汇编语言混合编程	(243)
12.4.1 汇编语言子程序的一般格式	(243)

12.4.2	全局变量的传递	(244)
12.4.3	参数和返回值的传递	(245)
12.4.4	汇编语言子程序的汇编及与 C 程序的连接	(246)
12.4.5	内嵌汇编指令的 C 程序结构	(246)
12.4.6	内嵌汇编指令的 C 程序的编译和连接	(247)
12.5	综合应用程序示例——汉字显示技术	(248)
12.5.1	汉字显示的基本原理	(248)
12.5.2	西文 DOS 环境下的汉字显示	(251)
第 13 章	C++ 基础	(266)
13.1	C++ 概述	(266)
13.2	C++ 对 C 的扩充	(268)
13.2.1	C++ 的输入输出	(268)
13.2.2	C++ 的行注释	(271)
13.2.3	const 常量定义与使用	(271)
13.2.4	局部变量的定义与全局变量作用域运算符	(273)
13.2.5	变量的引用	(274)
13.2.6	函数重载	(277)
13.2.7	带缺省参数的函数	(278)
13.2.8	内联函数	(279)
13.2.9	new 与 delete 算符	(280)
13.3	C++ 面向对象程序设计	(282)
13.3.1	类与对象	(282)
13.3.2	构造函数与析构函数	(287)
13.3.3	静态成员	(293)
13.3.4	友元	(295)
13.3.5	运算符重载	(297)
13.3.6	继承与派生	(300)
13.3.7	多态性与虚函数	(304)
13.3.8	I/O 流类库与文件输入输出	(309)
13.3.9	模板	(319)
13.3.10	异常处理	(324)
13.4	应用举例	(326)
附录 I	ASCII 字符编码一览表	(334)
附录 II	运算符的优先级别和结合方向一览表	(335)
附录 III	Turbo C 库函数	(336)
附录 IV	图形模式表	(347)
附录 V	鼠标器功能的编号及其作用	(348)
附录 VI	鼠标驱动程序常用功能及其参数	(349)
参考文献		(350)

第 1 章 C 语言程序设计概述

电子计算机自从 20 世纪 40 年代诞生以来,无论在硬件还是在软件方面,都有了极大的发展;在计算机应用的各个领域也都取得了丰硕的成果。

计算机本身是无生命的机器,要使计算机能够运行起来,为人类完成各种各样的工作,就必须让它执行相应的程序。这些程序都是依靠程序设计语言编制出来的。

在众多的程序设计语言中,C 语言有其独特之处。它作为一种高级程序设计语言,具备方便性、灵活性和通用性等特点;同时,它还向程序员提供了直接操作计算机硬件的功能,具备低级语言的特点,适合各种类型的软件开发。因此,C 语言是深受软件工作者欢迎的程序设计语言。

本章主要从程序设计的角度,介绍有关程序设计的基本概念,结合 C 语言的发展,描述 C 语言程序的基本结构、算法以及 C 语言程序的开发环境等内容。

1.1 程序与程序设计语言

1.1.1 程序的基本概念

为了让计算机按人的意图处理事务,人们必须预先设计好完成各种任务的程序,并预先将它们存放在存储器中。

所谓程序,实际上是用计算机语言描述的某一问题的解决步骤,是符合一定语法规则的符号序列。人们借助计算机能够处理的语言,告诉计算机要处理什么(即处理哪些数据)以及如何处理(即按什么步骤来处理),这便是程序设计。通过在计算机上运行程序,向计算机发出一系列指令,便可按人们的要求解决特定问题。

解决某一个问题所编写的程序并不是唯一的,不同的用户所开发的程序也都往往不相同。不同的程序有不同的效率,这涉及到程序的优化,涉及到程序所采用的数据结构以及算法等多方面因素的综合。

1.1.2 程序设计语言

完成程序设计,自然离不开程序设计语言。不同的问题,可以用不同的程序设计语言来解决,但解决问题的难易程度会各不相同。了解程序设计语言的发展过程,有助于我们加深对程序设计语言的认识,使我们能更好地利用程序设计语言来解决有关问题。

当今程序设计语言发展非常迅速,新的程序设计语言层出不穷,其功能越来越强大。但不管现代程序设计语言的功能如何增强,程序设计语言的种类怎样增多,从其发展历史以及功能情况看,大致可分成如下几个阶段:

1. 机器语言

由于物理器件和实现难易的影响,直到目前计算机还只能存放和识别由 0 和 1 组成的序列所表示的数据和指令。所谓机器语言,就是指该机器能够识别的指令的集合,即指令系统。在机器语言中,每条指令都用 0 和 1 组成的序列来表示。例如,以下是某计算机的两条机器指令:

加法指令:10000000

减法指令:10010000

不同类型的计算机,使用的机器语言不相同。

用机器语言编写的程序,计算机可以直接执行,执行效率高。但机器语言的指令不直观,难认、难记、难理解、且较烦琐,容易出错,写出的程序不能通用。编写机器语言程序时,要求程序员必须相当熟悉计算机结构,因而,目前很少直接用机器语言编程。

2. 汇编语言

20 世纪 50 年代中期,为了减轻人们使用机器语言编程的负担,开始采用一些“助记符号”来表示机器语言中的机器指令,这样便形成了汇编语言。助记符一般都是采用一个操作的英文字母的缩写,与机器语言相比,便于识别和记忆。例如,上例中的两条指令用汇编语言描述如下:

ADD A, B

SUB A, B

不过,计算机不能直接执行用汇编语言编写的程序,它必须经过一个叫汇编程序的系统软件翻译成机器语言程序后才能执行。我们称前者为源程序,后者为目标程序。

汇编语言指令和机器语言指令之间具有一一对应的关系。因而,不同的计算机其汇编语言也不尽相同,并且程序编写时仍需要对计算机内部结构比较熟悉,依然比较烦琐。但相对于机器语言而言,汇编语言要好多了。因而,在实际中,如果程序运行时间要求比较严格,程序与硬件操作联系紧密,人们还是常用汇编语言编写有关程序来解决这些问题。

3. 算法语言

汇编语言和机器语言是面向机器的,它们同属于低级语言的范畴。人们在使用它们设计程序时,要求对机器比较熟悉。为了克服低级语言的这一缺点,使人们将程序设计的精力集中在解决问题的算法上,便出现了面向算法过程的程序设计语言,称之为算法语言。比如:FORTRAN 语言、ALGOL 语言、PASCAL 语言、C 语言等。这类语言接近自然语言的形式,因而可以较大地降低编程的难度,提高编程的效率和质量,且设计的程序也更容易理解。

算法语言的一条语句相当于多条汇编语言指令或机器语言指令,表达能力强。在使用算法语言时,程序设计人员不需要熟悉计算机的指令系统,可以把精力集中在研究问题的求解方法步骤(过程)上。因此,算法语言也称为面向过程的程序设计语言。同时,算法语言不依赖于机器。为某种类型的计算机编写的算法语言程序,可以很方便地移植到其他类型的计算机上运行。相对于与机器有关的低级语言而言,算法语言也就称为高级语言。

当然计算机也不能直接执行算法语言程序。如同汇编语言程序一样,算法语言程序(源程序)也必须先经过编译程序(或解释程序)翻译成机器语言程序(目标程序)后,才能由计算机执行。

4. 面向任务的程序设计语言

利用算法语言求解一个复杂的问题,必须先要分析解决问题的过程,描述问题如何求解,然后才能用算法语言进行程序设计来实现。面向任务的程序设计语言是非过程化的语言,也就是说,不需要知道问题是如何求解的,只需要描述需求解的问题是什么,然后便可用程序设计语言来实现。

数据库操纵语言便是一种面向任务的程序设计语言。例如,设在某数据库应用系统中,有一个学生情况表 SS,若要从表中查找学生的信息,可以使用数据库查询语言(SQL)。采用 SELECT 语句便可完成这个任务,SELECT 语句描述如下:

```
SELECT SSNO,SSNAME,SSAGE,SSSEX FROM SS
```

该语句从 SS 表中查找到学生的 SSNO、SSNAME、SSAGE、SSSEX 等方面的信息。至于 SELECT 语句是如何进行查询的,用户就不必了解了。

面向任务的程序设计语言可以提高应用程序的开发速度和质量,也可使应用程序易于迅速地修改。同时,非计算机专业人员也能很方便地使用面向任务的程序设计语言开发自己的程序。

这类语言在应用系统的开发中使用较为广泛,尤其是一些管理信息系统应用程序的开发。

5. 面向对象的程序设计语言

面向对象程序设计语言在 20 世纪 90 年代开始流行。现在,面向对象的程序设计语言已成为程序设计的主流语言之一。由 C 语言发展出来的 C++ 就是一种非常优秀的面向对象的程序设计语言。

面向对象方法是一种分析方法、设计方法和思维方法的综合。面向对象方法学的出发点和所追求的基本目标是使人们分析、设计和实现一个系统的方法尽可能接近人们认识一个系统的方法。

在面向对象编程中,程序被看作是相互协作的对象集合,每个对象都是某个类的实例,所有的类构成一个通过继承关系相联系的层次结构。面向对象的程序设计语言具有对象生成功能、消息传递机制,以及类和继承机制。

对象是对客观的事物的抽象,面向对象的编程,就是针对客观的事物设计程序。因此,面向对象的编程是非常直观的。面向对象的程序设计方法比面向过程的程序设计方法更清晰,更适合于开发大型复杂的软件。

综上所述,每一种语言都有它的优势和劣势。对于不同的问题,要根据实际情况来选择程序设计语言,以便更高效更优质地解决相关的问题。

1.2 算法及其描述

在程序设计中,不可避免地需要涉及算法。有人这样说过:“计算机科学就是研究算法的科学”,足见算法在程序设计中的重要性了。下面从算法的概念和描述方法两方面,对算法的问题进行讨论。

1.2.1 算法的概念

著名的瑞士计算机科学家、PASCAL 语言发明者 N·沃思(Niklaus Wirth)教授提出了程序定义的著名公式:

$$\text{程序} = \text{算法} + \text{数据结构}$$

这个公式的重要性在于它说明了程序与算法的关系。同时,说明了数据结构的选择也是十分重要的。对于程序而言,算法与数据结构是统一的关系。

通常认为,算法是对特定问题求解步骤的一种描述。

算法应当具备以下几个方面特点：

- (1) 一个算法必须保证执行有穷步之后结束；
- (2) 算法的每一个步骤必须具有确切的定义；
- (3) 应对算法给出初始量；
- (4) 算法具有一个或多个输出；
- (5) 可行性——算法不能进行是不允许的，如“计算 $X/0$ ”。

算法必须能在有限时间内完成，且对相同的输入有相同的输出。在程序设计语言中，与算法密切相关的便是语句，包括与程序执行处理有关的“功能语句”（如：输入语句、输出语句、赋值语句、调用语句等）和与程序执行流程有关的语句（如：条件语句、循环语句等）。对程序设计而言，算法的确定也就是如何合理安排这些语句以完成人们要求的特定功能。

1.2.2 算法的描述方法

从上面的分析可知，算法是描述某一问题求解的有限步骤，而且必须有结果输出。设计一个算法，或者描述一个算法，最终是由程序设计语言来实现的。但算法与程序设计又是有区别的，主要是一个由粗到细的过程。算法是考虑实现某一个问题求解的方法和步骤，是解决问题的框架流程；而程序设计则是根据这一求解的框架流程进行语言细化，实现这一问题求解的具体过程。

一般可以使用下面几种类型的工具描述算法：

1. 自然语言

自然语言即是人们日常进行交流的语言，如英语、汉语等。自然语言用来描述算法，分析算法，作为用户相互之间进行交流，是一种较好的工具。但是将自然语言描述的算法直接在计算机上进行处理，目前还存在许多困难，包括有诸如语音语义识别等方面的问题。

2. 专用工具

要对某一个算法进行描述，可以借助于有关图形工具或代码符号。常用的工具有流程图、N-S图等。

20 世纪 50~60 年代兴起的流程图几乎成为了程序设计及算法描述的必用工具。

人们已经提出了多种描述算法的流程图。这种方法的特点是用一些图框表示各种类型的操作，用线表示这些操作的执行顺序，图 1.1 为我国国家标准 GB1526—89 中推荐的一套流程图标准化符号的一部分。

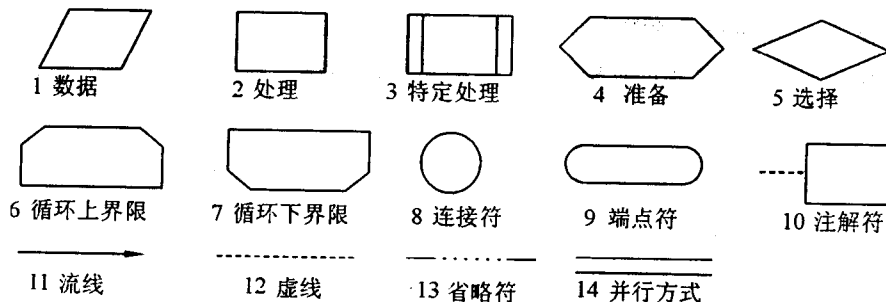


图 1.1 流程图标准化符号

- 数据 平行四边形表示数据，其中可注明数据名称、来源、用途或其他文字说明。
- 处理 矩形表示各种处理功能。矩形内可注明处理名称或其简要功能。

· 特定处理 带有双竖边线的矩形。矩形内可注明特定处理名称或简要功能,表示已命名的处理。该处理为在另外地方已得到详细说明的一个操作或一组操作。

· 判断 菱形表示判断。菱形内可注明判断的条件。它只有一个入口,但可以有若干个可供选择的出口。

· 循环界限 循环界限包含循环的上界和下界,中间是要循环执行的处理内容,称为循环体。循环界限由去上角的矩形(表示上界限)和去下角的矩形(表示下界限)构成。

· 端点 扁圆形表示转向外部环境或外部环境转入的端点符。例如,程序流程的起始点。

· 注解 注解是程序的编写者向阅读者提供的说明。它用虚线连接到被注解的符号或符号组上。

3. 部分常用的算法

在程序设计时,常常要通过一些特定的算法来求解。现在比较常用的算法有下列几种:

① 迭代法

一般的一元五次或更高次的方程,以及几乎所有的超越方程、微分方程问题都无法用解析方法通过求根公式来求解,人们只能用数值方法求其近似解。用事先估计的一个根的初始值 X_0 ,通过迭代算式 $X_{k+1} = G(X_k)$ 求出另一个近似值 X_1 ,再由 X_1 求出 X_2 ,从而获得一个求解的序列 $\{X_0, X_1, X_2, \dots, X_n, \dots\}$ 来逼近方程 $f(x)=0$ 的根。这种求解的过程称为迭代。

② 枚举法

枚举法的基本思想是首先依据题目部分条件确定答案的大致范围,然后逐一验证所有可能的情况,这种算法亦称为穷举法。

③ 递归法

递归是指一个过程直接或间接地调用它自身,递归过程必须有一个递归终止条件。

如:

$$n! = \begin{cases} 1 & n=0 \\ n(n-1)! & n>0 \end{cases}$$

④ 递推法

算法从递推初始条件出发,应用递推公式对问题进行求解。如 Fibonacci 数列存在递推关系:

$$F(1)=1, F(2)=1, F(n)=F(n-1)+F(n-2), (n>2)$$

若需求第 30 项的值,则依据公式,从初始条件 $F(1)=1, F(2)=1$ 出发,可逐步求出 $F(3), F(4), \dots$,直到求出 $F(30)$ 。

除此之外,还有回溯法(一种基本策略),分治法(问题分解成较小部分,求解再组合)等其他的常见算法,在此不一一阐述。

1.3 C语言的发展及特点

1.3.1 C语言的发展概况

C语言是当今社会应用广泛,并受到众多用户欢迎的一种计算机算法语言。它既可作为系统软件描述语言,也可用来开发应用软件。