



新编21世纪高职高专计算机系列规划教材

C语言程序设计

北京希望电子出版社 总策划
成奋华 陆慧民 主 编
刘长生 唐 强 周艳秀 副主编



科学出版社
www.sciencep.com



教育部“十三五”普通高等教育规划教材

C语言程序设计

主 编 王 健 副主编 王 健
参 编 王 健 王 健 王 健
参 编 王 健 王 健 王 健

清华大学出版社

新编21世纪高职高专计算机系列规划教材

C语言程序设计

北京希望电子出版社 总策划
成奋华 陆慧民 主 编
刘长生 唐 强 周艳秀 副主编

内容简介

本书严格遵循 C 语言标准,全面、系统地阐述了 C 语言的基本概念、语法和语义,以及 C 语言进行程序设计的方法和技术。内容包括数据类型、运算符和表达式、流程控制、数组、函数、指针、构造类型、文件和 C++ 基础。其中,对 C 语言的表达式、类型转换、函数和指针的用法等重点和难点均作了明确而详细的介绍。

本书概念清楚、叙述详实、内容新颖实用、实例典型丰富、要点清晰明了。实例程序均经过上机调试运行,大部分给出了算法的思路分析。每章末均附有本章小结及精选的习题,便于教学。

本书是一本准确而又较全面、深入反映标准 C 的程序设计教材,适合作为高职高专院校计算机、通信、电子、自动化等信息学科的教学使用,可作为全国计算机二级考试 C 语言的辅导用书,也适合广大科技人员和自学者参考。

需要本书或技术支持的读者,请与北京清河 6 号信箱(邮编:100085)发行部联系,电话:010-82702660,62978181(总机)传真:010-82702698, E-mail: tbd@bhp.com.cn。

图书在版编目(CIP)数据

C 语言程序设计 / 成奋华, 陆慧民主编. —北京: 科学出版社, 2006.9

(新编 21 世纪高职高专计算机系列规划教材)

ISBN 7-03-017715-0

I. C ... II. ①成... ②陆... III. C 语言—程序设计
IV. TP312

中国版本图书馆 CIP 数据核字(2006)第 083001 号

责任编辑: 方红琴

/ 责任校对: 马 君

责任印刷: 媛 明

/ 封面设计: 刘孝琼

科学出版社 出版

北京东黄城根北街 16 号

邮政编码: 100717

<http://www.sciencep.com>

北京媛明印刷厂印刷

科学出版社发行 各地新华书店经销

*

2006 年 9 月第 一 版 开本: 787×1092 1/16

2006 年 9 月第一次印刷 印张: 21 7/8

印数: 1—3000

字数: 499 434

定价: 34.00 元

新编 21 世纪高职高专计算机系列规划教材编委会

主 任： 沈复兴 全国高等师范学校计算机教育研究会副理事长
北京师范大学信息科学学院院长

副主任： 桂卫华 中南大学信息科学与工程学院院长

李仁发 湖南大学计算机与通信学院院长

唐汝元 湖南张家界航空职业技术学院副院长

刘小芹 湖北武汉职业技术学院副院长

陆卫民 中国科学出版集团北京希望电子出版社社长

委 员： （按姓氏笔画为序）

于小川	牛军涛	王 立	王 虹	王亚林	冯玉东	石 磊
刘其群	刘晓魁	向长喜	曲宏山	朱乃立	朱志伯	朱国军
齐英兰	张凌雯	张惠敏	李 晨	李节阳	李国安	杨成卫
狄文辉	肖 衍	苏 玉	连卫民	陆立松	陈利军	陈桂生
周承华	易著梁	罗立红	郑明红	胡远萍	骆 刚	徐刚强
秦国防	崔清民	黄贻彬	黄振中	彭 勇	曾凡秩	蒋本立

秘 书： 李节阳

总 序

一本好书，是人生前进的阶梯；一套好教材，就是教学成功的保证。为满足培养应用型人才的需要，我们成立了本编委会。在明确高职高专应用型人才培养模式、培养目标、教学内容和课程体系的框架下，我们组织编写了本套规划教材。

为了使本套教材能够达成目标，编委会做了大量的前期调研工作，在广泛了解各高职高专学校的教学现状、学生水平、培养目标的情况下，认真探讨了课程设置，研究了课程体系。为了编写出符合教学需求的好教材，我们除了聘请一批计算机知名专家、教授作为本套教材的主审和编委外，还组织了一批具备较高的学术水平、丰富的教学经验、较强的工程实践能力的学术带头人和骨干教师来承担具体编写工作，从而编写出特色鲜明、适用性强的教材，以真正满足目前高职高专学校应用型人才培养的需要。教材编写采用整体规划、分步实施、在实践中检验提高的方式，分期分批地启动编写计划。编写大纲以及教材编写方式的确定均经过编委会多次认真讨论，以确保该套教材的高质量和实用性。

本套规划教材的主要特点是：

(1) 以服务教学为最高宗旨，认真做好教学内容的取舍、教学方法的选取、教学成果的检验工作。本套教材在教学过程中的有益反馈，都将及时体现在后续版本。

(2) 面向应用型高职高专，在保证学科体系完整的基础上把握好理论的深度和难度。注重理论知识与实践相结合，使学生通过实践深化对理论的理解，学会并掌握理论方法的实际运用。从而较好地培养学生的专业技能和实施工程的实用技术能力。

(3) 教材在内容编排上，力求由浅入深，循序渐进；举一反三，突出重点；语言简练，通俗易懂。采用模块化结构，兼顾不同层次的需求，在具体授课时可根据具体教学计划适当取舍内容。

(4) 教材采用“任务驱动”的编写方式，以实际问题引出相关原理和概念，在讲述实例的过程中将本章的知识点融入，通过分析归纳，介绍解决工程实际问题的思想和方法，同时，引入案例教学和启发式教学方法，便于激发学习兴趣。

(5) 在教材中加大实训部分的比重，使学生能比较熟练地应用计算机知识和技术解决实际问题，既注重培养学生分析问题的能力，也注重培养学生解决问题的能力。

(6) 大部分教材配有电子教案，从而更好地服务教学。

为编写本套教材，作者们付出了艰辛的劳动，编委会的各位专家进行了悉心的指导和认真的审定。书中参考、借鉴了国内外同类的优秀教材和专著，在此一并表示感谢。

我们衷心希望更多的优秀教师参与到教材建设中来，真诚希望广大教师、学生与读者朋友在使用本丛书过程中提出宝贵意见和建议。

若有投稿或建议，请发电子邮件到 textbook@bhp.com.cn。谢谢！

前 言

C 语言具有语言简洁紧凑,运算符、表达式类型及数据类型丰富,表达能力强,使用方便灵活,支持结构化程序设计,可移植性好,目标代码运行效率高,兼容高、低级语言功能等众多优点,是目前世界上最受欢迎、应用最广泛的程序设计语言之一,成为目前几乎所有计算机专业学生学习计算机语言的入门课程,很多高等学校都把 C 语言作为计算机语言的首选教学语言,并把它作为数据结构和各种操作算法的描述工具。

为便于计算机专业学生在有限的教学时间内尽快掌握 C 语言的主要内容,经新编 21 世纪高职高专计算机系列规划教材编委会策划,我们编写了本书。

本书编写的总原则是:根据专业人才培养规格的需要,突出职业素质教育和技术应用能力教育主线,强调理论与实践教学相结合,注重创新精神、综合素质、实践能力和可持续发展能力的培养。编写时力求“基础知识够用,注重应用能力”,各章主体内容均以需求实例为主线,首先介绍相关的基本知识和基本规则,重点介绍其应用方法与注意事项,继而解决需求实例(书中完整的例题都在 Turbo C 2.0 和 Visual C++ 6.0 环境下调试通过,便于读者直接上机验证),最后辅以上机实训指导,使基本概念介绍、应用方法学习、上机实训操作一气呵成,即学即用,帮助初学者一步一个脚印地踏实前进。

本书参照当前最新的 2002 年版全国计算机等级考试二级 C 语言程序大纲安排章节,内容涵盖该大纲的要求并有较大延伸。编写时注意对该大纲要求的内容从概念到应用均作重点介绍,以期望对初学者参加全国计算机等级考试有较大帮助。

本书内容可分为四部分:第 1 章到第 5 章为第一部分,主要包括程序设计基础知识, C 语言的特点、开发过程与开发环境, C 语言源程序的组成, C 语言的数据类型、运算符与表达式、基本语句和控制结构等,属于基本概念部分,以使读者初步掌握用 C 语言进行程序设计所需的基本知识;第 6 章到第 10 章为第二部分,主要包括 C 语言的数组应用、函数应用、指针应用、构造数据类型应用与文件操作等,属于应用基础部分,以使读者初步掌握用 C 语言进行程序设计的基本方法和常用算法,进而具有用 C 语言解决实际问题的初步能力;第 11 章为第三部分,简要介绍 C++ 基础,以使读者对面向对象这一当前软件开发的主流方法有个初步印象;附录为第四部分,供读者在编制 C 语言程序解决问题时参考。

本书内容安排由简到繁、循序渐进、深入浅出、强调实用,既考虑到系统性、严密性、先进性,又兼顾目前学生的整体水平和初学者的接受能力。因此,本书不仅可作为高职高专计算机专业的教材,也可作为计算机专业的成人教育、自学考试和培训辅导教材,还可作为中职中专学校计算机教学参考书及有关人员的自学参考书。

虽然本书的完整例题都在 Turbo C 2.0 和 Visual C++ 6.0 环境下调试通过,但随着 Windows 环境下的 C++ 及 C# 的迅速崛起,面向对象与可视化编程已成为当前软件开发的主流, DOS 环境下的 Turbo C 已较少使用,即使学习 C 语言的人们目前也多在 C++ 环境中上机调试运行。为此,编者推荐读者在 Visual C++ 6.0 开发环境下学习 C 语言。在一些 Turbo

C 2.0 和 Visual C++ 6.0 两种环境下差别较大的地方，书中给出了简要说明，以便读者尽快熟悉 Visual C++ 6.0 环境，有利于读者在后续学习中向面向对象与可视化编程方向发展。

本书的参考教学时间是 72 课时，其中上机 18 学时。如果作为中职中专和培训辅导教材，教学时可酌情对一些章节进行删减。

本书由长期工作在教学一线、具有丰富教学经验、熟悉高职高专教学模式的教师编写，他们是：陆惠民、成奋华、刘长生、唐强、周艳秀、陆海霞、张敏、黄婕、杨光祖、欧阳牡瑶、刘钢等教师。

在本书的编写出版发行过程中，相关单位的许多领导和同仁给予了大量关心、指导和帮助。在此，谨向为本书付梓提供了大量帮助和便利的湖南涉外经济学院、湖南科技职业学院、湘潭职业技术学院、湖南工业职业技术学院、长沙航空职业技术学院、长沙民政职业技术学院、长沙环保职业技术学院，新加坡国立大学博士后叶施仁等表示衷心的感谢和诚挚的祝福。

虽然我们力求准确无误、力求尽善尽美，但由于时间仓促，加之计算机技术的飞速发展，书中的内容取舍难免顾此失彼，也难免有错误或不妥之处，恳请专家和读者批评指正。您的使用是对我们工作最好的鼓励，您的批评是对我们工作最好的帮助。

编 者

目 录

第1章 C语言程序设计概述.....1	2.4.3 实型变量.....34
1.1 程序设计基础知识.....1	2.4.4 字符变量.....35
1.1.1 程序与程序设计语言.....1	2.4.5 变量的定义与赋值.....37
1.1.2 程序算法描述.....3	2.5 运算符与表达式概述.....38
1.1.3 程序的3种基本结构.....8	2.5.1 运算符与表达式的分类.....38
1.2 C语言的历史及特点.....9	2.5.2 运算符的优先级与结合性.....39
1.2.1 C语言的产生与发展概况.....9	2.6 算术运算符与算术表达式.....41
1.2.2 C语言的特点.....9	2.6.1 基本算术运算符.....41
1.3 C语言源程序的组成、书写规则与风格...10	2.6.2 负值运算符、自增运算符与 自减运算符.....41
1.3.1 C语言源程序的组成.....10	2.6.3 算术表达式.....44
1.3.2 C语言源程序书写规则与风格...13	2.7 赋值运算符与赋值表达式.....44
1.4 C语言的字符集、标识符与关键字.....14	2.7.1 基本赋值运算符.....44
1.4.1 C语言的基本字符集与转义字符...14	2.7.2 复合赋值运算符.....45
1.4.2 C语言的标识符.....15	2.7.3 赋值表达式.....46
1.4.3 C语言的关键字.....16	2.8 逻辑运算符与逻辑表达式.....46
1.5 C语言程序的开发过程与开发环境.....16	2.8.1 逻辑运算符.....47
1.5.1 C语言程序的开发过程.....16	2.8.2 逻辑表达式.....48
1.5.2 Turbo C 2.0 集成开发环境简介..17	2.9 关系运算符与关系表达式.....49
1.5.3 Visual C++ 6.0 开发环境简介.....21	2.9.1 关系运算符.....49
小结.....23	2.9.2 关系表达式.....50
习题.....24	2.10 位运算符与位运算表达式.....51
第2章 C语言的数据类型与运算.....26	2.10.1 位运算符与位运算.....51
2.1 C语言的数据类型分类.....26	2.10.2 位运算赋值运算符.....54
2.1.1 数据类型的基本概念.....26	2.10.3 位运算表达式.....54
2.1.2 数据类型的分类.....26	2.11 其他运算符与表达式.....55
2.2 基本数据类型.....28	2.11.1 条件运算符和条件表达式.....55
2.2.1 整型.....28	2.11.2 逗号运算符和逗号表达式.....56
2.2.2 实型.....28	2.11.3 求字节运算符.....57
2.2.3 字符型.....28	2.12 不同类型数据的混合运算与类型转换...58
2.3 常量.....28	2.12.1 混合运算与数据类型 转换的分类.....58
2.3.1 整型常量.....28	2.12.2 自动转换.....58
2.3.2 实型常量.....29	2.12.3 赋值转换.....60
2.3.3 字符常量.....30	2.12.4 强制转换.....61
2.3.4 字符串常量.....31	小结.....62
2.3.5 直接常量与符号常量.....31	习题.....62
2.4 变量.....32	第3章 C语言基本语句和顺序结构程序设计...64
2.4.1 变量的含义与分类.....32	
2.4.2 整型变量.....33	

3.1 C 语言基本语句.....	64	6.2.1 一维数组的定义.....	114
3.1.1 流程控制语句.....	64	6.2.2 一维数组的初始化.....	114
3.1.2 表达式语句.....	66	6.2.3 一维数组的引用.....	115
3.1.3 函数调用语句.....	66	6.2.4 一维数组程序举例.....	116
3.1.4 空语句.....	67	6.3 多维数组.....	119
3.1.5 复合语句.....	67	6.3.1 多维数组的定义.....	119
3.2 数据输入与输出.....	67	6.3.2 多维数组的初始化.....	120
3.2.1 格式输入与输出.....	67	6.3.3 多维数组的引用.....	120
3.2.2 字符数据输入与输出.....	73	6.3.4 多维数组程序举例.....	121
3.3 顺序结构程序举例.....	74	6.4 字符数组与字符串.....	123
小结.....	76	6.4.1 字符数组的定义.....	123
习题.....	76	6.4.2 字符数组的初始化.....	123
第4章 选择结构程序设计.....	79	6.4.3 字符数组的引用.....	124
4.1 选择结构的需求实例.....	79	6.4.4 字符数组的输入与输出.....	125
4.2 if 语句.....	80	6.4.5 字符串的特殊性.....	125
4.2.1 if 语句的3种形式与使用.....	80	6.4.6 常用字符串处理函数.....	125
4.2.2 嵌套if语句的二义性与解决方法.....	83	6.4.7 字符数组应用程序举例.....	129
4.3 用switch语句实现选择.....	85	小结.....	130
4.4 用条件运算符实现选择.....	88	习题.....	130
4.5 几种选择结构实现方法的比较.....	88	第7章 函数.....	134
4.6 选择结构程序举例.....	91	7.1 概述.....	134
小结.....	93	7.1.1 函数应用的需求举例.....	134
习题.....	93	7.1.2 函数的概念与意义.....	135
第5章 循环结构程序设计.....	96	7.1.3 模块化程序设计与函数.....	136
5.1 while 语句.....	96	7.1.4 C语言函数的分类.....	136
5.2 do...while 语句.....	97	7.2 函数的定义.....	136
5.3 for 语句.....	99	7.2.1 函数的组成和函数定义的 一般形式.....	136
5.4 break、continue、goto 语句.....	101	7.2.2 函数的类型与函数值.....	139
5.4.1 break 语句.....	101	7.3 有参函数的参数与参数值传递.....	141
5.4.2 continue 语句.....	102	7.3.1 形式参数与实际参数.....	141
5.4.3 goto 语句.....	103	7.3.2 各种类型的实参.....	142
5.5 循环的嵌套.....	104	7.3.3 参数值的传递.....	146
5.6 复合结构程序举例.....	104	7.4 函数的调用.....	148
小结.....	108	7.4.1 函数调用的一般形式.....	148
习题.....	109	7.4.2 函数调用的方式.....	149
第6章 数组.....	112	7.4.3 库函数的调用.....	149
6.1 概述.....	112	7.4.4 函数原型与自定义函数的调用.....	150
6.1.1 数组应用需求实例.....	112	7.5 函数的嵌套调用.....	152
6.1.2 数组的概念与分类.....	113	7.5.1 函数嵌套调用的过程.....	152
6.1.3 数组名的特殊意义.....	113	7.5.2 函数嵌套调用程序举例.....	153
6.2 一维数组.....	113		

7.6 函数的递归调用	154	函数的参数	197
7.6.1 递归的概念、过程与优缺点	154	8.3.4 指向一维数组的指针变量的 程序举例	199
7.6.2 函数递归调用程序设计举例	157	8.4 指针与多维数组	201
7.7 函数应用程序举例	160	8.4.1 二维数组各元素的地址及 值的表示方法	201
7.7.1 无参函数应用程序设计举例	160	8.4.2 指向二维数组的指针 变量的引用	202
7.7.2 有参函数应用程序设计举例	161	8.4.3 指向二维数组的指针 变量作函数参数	203
7.8 变量的作用域、生存期、 存储类别及其声明	162	8.4.4 指向二维数组的 指针变量的程序举例	205
7.8.1 概述	162	8.5 指针与字符串	206
7.8.2 局部变量和全局变量及其 作用域	163	8.5.1 指向字符串指针的 定义与赋值	206
7.8.3 动态存储变量和静态存储变量 及其生存期	165	8.5.2 指向字符串指针变量的引用	207
7.8.4 变量声明与变量定义的区别	172	8.5.3 用指向字符串的指针变量 作为函数的参数	208
7.9 内部函数与外部函数	173	8.5.4 指向字符串指针 变量的程序举例	210
7.9.1 内部函数	173	8.6 指针与函数	211
7.9.2 外部函数	174	8.6.1 指向函数的指针的定义与赋值	211
7.9.3 外部函数应用程序举例	175	8.6.2 用指向函数的指针变量 作为函数的参数	212
7.10 编译预处理	176	8.6.3 用指向函数的指针变量 调用函数的程序设计举例	214
7.10.1 文件包含	176	8.7 指针类型的函数	215
7.10.2 宏定义	177	8.7.1 指针类型函数的概念、 作用与定义	215
7.10.3 条件编译	179	8.7.2 指针类型函数的程序举例	216
小结	181	8.8 指针数组	218
习题	181	8.8.1 指针数组的概念及定义	218
第8章 指针	185	8.8.2 指针数组的应用及程序举例	219
8.1 概述	185	8.9 指向指针的指针	221
8.1.1 指针应用的需求实例	185	8.9.1 指向指针的指针的概念、 定义与赋值	221
8.1.2 变量地址、指针、指针 变量的概念	187	8.9.2 指向指针的指针的程序举例	222
8.1.3 指针变量的定义与赋值规则	187	8.10 main 函数的命令行参数	224
8.1.4 地址运算符和指针运算符	189	8.10.1 概念与一般形式	224
8.1.5 指针变量运算及注意事项	190	8.10.2 指针数组用作 main 函数的 形参及程序举例	224
8.2 指针与简单变量	191		
8.2.1 指向简单变量的指针变量	191		
8.2.2 指针变量作为函数的参数	191		
8.2.3 程序举例	193		
8.3 指针与一维数组	195		
8.3.1 指向一维数组的指针	195		
8.3.2 指向一维数组的指针 变量的引用	196		
8.3.3 用数组名或指针变量作为			

小结	225	10.3.5 读写一个字符串	263
习题	226	10.3.6 按指定格式读写数据	264
第 9 章 构造数据类型	230	10.4 文件的定位	264
9.1 结构体	230	10.4.1 移动文件指针到文件头部	264
9.1.1 结构体类型定义	230	10.4.2 移动文件指针到指定位置	265
9.1.2 结构体类型变量定义	231	10.4.3 获得流中文件指针的位置	266
9.1.3 结构体类型变量的引用	232	10.5 文件状态检测	266
9.1.4 结构体数组	233	10.6 文件操作的程序实例	267
9.1.5 指向结构体类型数据的指针	234	小结	269
9.2 链表	237	习题	270
9.2.1 概述	237	第 11 章 C++基础	271
9.2.2 创建静态链表	239	11.1 C++概述	271
9.2.3 创建动态链表	240	11.2 C++对 C 的扩充	272
9.3 共用体	243	11.2.1 C++的输入输出	273
9.3.1 共用体的概念	243	11.2.2 C++的行注释	275
9.3.2 共用体类型的定义	243	11.2.3 const 常量定义与使用	275
9.3.3 共用体变量的定义和引用	243	11.2.4 局部变量的定义与全局 变量作用域运算符	277
9.4 枚举类型	245	11.2.5 变量的引用	277
9.5 定义已有类型的别名	246	11.2.6 带缺省参数的函数	279
9.6 位段	247	11.2.7 内联函数	281
小结	249	11.2.8 new 与 delete 运算符	282
习题	250	11.3 C++面向对象程序设计	283
第 10 章 文件	252	11.3.1 类与对象	283
10.1 概述	252	11.3.2 构造函数与析构函数	287
10.1.1 文件应用的需求举例	252	11.3.3 静态成员	288
10.1.2 文件的概念、文件名和 文件属性	252	11.3.4 友元	289
10.1.3 缓冲文件系统与非缓冲 文件系统	253	11.3.5 重载	290
10.1.4 流式文件、文本流与 C 语言 文件分类	253	11.3.6 继承与派生	292
10.1.5 文件类型指针及其定义	254	11.3.7 多态性与虚函数	295
10.2 文件的打开与关闭	254	11.3.8 I/O 类库与文件输入输出	297
10.2.1 文件的打开	255	11.3.9 模板	298
10.2.2 文件的关闭	256	小结	299
10.3 文件的读写	257	习题	300
10.3.1 读写一个字符	257	附录 A ASCII 字符一览表	301
10.3.2 读写两个字节	259	附录 B 全国计算机等级考试二级 (C 语言程序设计) 考试大纲	303
10.3.3 读写一行字符	260	附录 C Turbo C 2.0 常用库函数	307
10.3.4 读写一个数据块	261	附录 D 上机实验	314

第 1 章 C 语言程序设计概述

教学目标

- 了解程序、程序设计、程序设计语言的概念及程序设计语言的分类。
- 了解算法的概念、特性与基本要求，掌握用流程图和伪码描述算法的方法。
- 理解程序的 3 种基本结构。
- 了解 C 语言的特点，掌握 C 语言程序的基本构成和书写规则。
- 掌握 C 语言的字符集、标识符与关键字。
- 掌握 C 语言程序的开发过程与 Turbo C 2.0 和 MS Visual C++ 6.0 两种常用开发环境下开发 C 语言程序的方法。

计算机系统是一种能自动、高速、精确地进行信息处理的现代工具。与人类历史上拥有的其他工具相比，计算机系统具有运算速度快、记忆能力强、有逻辑判断能力、能自动进行计算等基本特点，这些特点根源于计算机系统的组成。

计算机系统的组成包括两大部分：硬件系统和软件系统。计算机硬件系统可以定义为“组成计算机系统的各种物理元器件”；软件系统则可以定义为“计算机程序及说明程序的各种文档”。软件系统与硬件系统一起构成完整的计算机系统，硬件系统是载体，软件系统是灵魂，它们相互依存、缺一不可。

“程序”是关于计算任务的处理对象和处理规则的描述；“文档”则是关于计算机程序功能、设计、编制、使用的各种资料的集合。

要让无生命的计算机系统运行起来，为人类完成各种工作，就必须让它执行相应的程序。这些程序都是采用程序设计语言结合相应的数据结构编制而成，且预先存储在计算机的存储器中，是使计算机完成任务的关键。

1.1 程序设计基础知识

1.1.1 程序与程序设计语言

要让计算机按人的意图处理事务，必须预先编制好相应的程序并将其存放在计算机存储器中。计算机的工作过程就是自动、连续、有序地读取程序指令，按照指令执行相应操作的过程。

1. 程序、程序设计与程序设计语言的概念

程序是算法与数据结构的和谐结合，由有限的指令序列构成，它的作用是告诉计算机当前的处理对象与处理步骤，使计算机完成相应的任务。

程序设计是指人们借助程序设计语言，将某一任务的处理对象、处理方法及处理步骤转化为计算机能够处理的符号序列（语句、命令、指令），以及指挥计算机完成相应任务的过程。

程序设计语言是建立在一定语法规则之上、用于向计算机传递程序设计人员思想的一

批特殊符号的集合。它与人类自然语言的根本区别在于它能被计算机硬件识别并执行，或经相应的工具软件（编译程序、解释程序、汇编程序等）处理后能被计算机硬件识别并执行。自 20 世纪 40 年代电子计算机诞生至今，已有许多种程序设计语言面世，且发展非常迅速，新的程序设计语言层出不穷，其功能越来越强大，人机界面越来越友好。

2. 程序设计语言的分类

按照其与计算机硬件联系的紧密程度或与人类自然语言的接近程度，程序设计语言可以大致分为低级语言（如机器语言、汇编语言）和高级语言（如 FORTRAN 语言、PASCAL 语言、BASIC 语言、C 语言等）两大类。由于 C 语言兼有高级语言的成分和低级语言的功能，也有人将 C 语言称为中级语言。按照其发展阶段与编程特性，可以分为面向机器的语言、面向过程的语言、面向任务的语言及当今最为流行的面向对象的语言等几大类。

(1) 面向机器的程序设计语言。该类语言与硬件密切关联，执行速度快，占用空间少，可以直接操作硬件以完成高级语言无法完成的特殊功能，适合于编制对程序执行时间要求高的软件系统（如实时处理系统），属于低级语言范围。该类语言主要有机器语言和汇编语言。机器语言是由 0 和 1 组成的符号序列表示的机器指令的集合（指令系统），是惟一能被硬件直接理解和执行的语言。用机器语言编制软件难写、难读、易出错、不通用；汇编语言采用助记符表示机器指令程序设计语言，其指令与机器指令一一对应，但并不能被硬件直接理解和执行。用汇编语言源程序比机器语言程序较为便书写、识别和记忆，是迄今为止许多实时处理系统的常用程序设计语言。

(2) 面向过程的设计语言。又称算法语言，不依赖于机器硬件，形式上较为接近人类自然语言，属于高级语言范围。与低级语言比较，用该类语言编制的程序较为易写、易读、易记忆、可移植，程序设计人员编程时不需熟悉繁杂难记的计算机指令系统，可以把精力集中在解决问题的算法上，从而提高程序设计的效率和质量。该类语言的代表有 FORTRAN 语言、PASCAL 语言、C 语言等。

(3) 面向任务的设计语言。该类语言以各种数据库管理系统（DBMS）为代表，具有算法语言的优点，而且是一种非过程化的语言，具有自动解决部分算法过程的能力，编程时不需详细描述问题的求解过程，而只需描述需要求解的问题即可。例如，在支持结构化查询语言（SQL）的 DBMS 中，从数据表“学生表”中查找“计软 0501”班学生的姓名、年龄信息，使用 SELECT 语句作如下描述：

```
SELECT 姓名, 年龄 FROM 学生表 WHERE 班级='计软 0501'
```

即可得当所要信息，而具体怎样从数据表中筛选出“计软 0501”学生及如何将所要的信息提取出来等具体算法过程均不必详细描述。

(4) 面向对象的程序设计语言。面向对象的方法把数据、操作封装在一起形成类，以对象和类间的抽象/细化为核心，通过派生、继承、重载和多态性等重要特色机制实现了较好的抽象性、封装性和共享性，极大地改善了结构化方法开发的软件的可维护性、可扩充性、可重用性较差等致命缺陷，适合于开发大型复杂的软件，已成为当今程序设计的主流方法。目前该类语言较多，由 C 语言发展而来的 C++ 以及 C#、JAVA 和 VB.NET 等是其代表。

1.1.2 程序算法描述

1. 算法的概念、特性与基本要求

算法 (Algorithm) 是对特定问题求解方法与步骤的描述, 是程序求解问题的核心内容。著名的瑞士计算机科学家、PASCAL 语言发明者 Niklaus Wirth 教授曾提出程序定义的著名公式: 程序=算法+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, 同时也说明数据结构的选择也十分重要。

算法由有限的指令 (语句) 序列构成, 其中每条指令 (语句) 表示一或多个操作。它有 5 个重要特性。

- (1) 有穷性: 对任何合法输入值, 算法必须在有穷时间内、执行有穷步之后结束。
- (2) 确定性: 算法中的每一个步骤必须有确切含义, 不产生二义性, 任何条件下对相同的输入只能得出相同的输出。
- (3) 可行性: 算法中描述的操作都是可通过已实现的基本运算执行有限次实现, 任何条件下都不允许出现算法不能进行的现象。
- (4) 输入: 输入指算法执行时从外界取得的信息。一个算法可有零个或多个输入。
- (5) 输出: 输出是指算法执行后得到的结果。一个算法可以没有输入, 但必须有输出, 无输出的算法是无意义的。

在程序设计语言中, 与算法密切相关的便是语句, 包括功能处理语句 (如输入输出语句、赋值语句、调用语句等) 和与流程控制语句 (如条件转移语句、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特定功能。

一个好的算法应当达到以下要求。

- (1) 正确性: 算法应满足具体问题的需求, 除了不含语法错误这一基本要求外, 还应当对精心选择的典型、苛刻而带有刁难性的几组输入数据能够得出满足用户提出的规格说明要求的结果。正确性是对算法的最基本要求, 不具备正确性的算法毫无意义。
- (2) 可读性: 算法主要是为了人的阅读与交流, 其次才是机器执行。良好的可读性是根据算法编制的软件具有较好的可维护性的前提, 晦涩难懂的算法易隐藏较多错误且难以调试和修改。
- (3) 健壮性: 算法对来自程序外部的错误 (如非法输入数据) 和内部的缺陷 (如未能考虑到的运算条件) 应能适当作出反应或处理, 不能出现莫名其妙的结果或停机等现象。
- (4) 高效性: 在保证正确性和可读性的前提下, 算法应尽可能追求较少的时间与空间占用。

2. 简单算法举例

【例 1-1】 求解自然数前 10 项的代数和。

算法 1: 采用最原始的方法求解。

- 第 1 步: 求 $0+1$, 得结果 1;
- 第 2 步: 将第 1 步的结果再加上 2, 得结果 3;
- 第 3 步: 将第 2 步的结果再加上 3, 得结果 6;
- 第 4 步: 将第 3 步的结果再加上 4, 得结果 10;

...

第 10 步：将第 9 步的结果再加上 10，得最后结果 55；
第 11 步：输出最后结果 55。

【例 1-1】中算法 1 是正确的，但太繁琐，如果要求解自然数前 1000 项的代数和，则需要写 1001 个步骤，显然不可取。可以改用算法 2。

算法 2：采用循环的方法求解。设变量 add 表示加数（兼作循环控制变量），变量 sum 开始时存放第 1 个数 0，然后存放和。

第 1 步：sum=0；
第 2 步：add=1；
第 3 步：sum=sum+add；
第 4 步：add=add+1；
第 5 步：如果 add 不大于 10，返回到第 3 步，重复执行第 3 步、第 4 步和第 5 步（循环）；否则 sum 中就是最后结果；
第 6 步：输出 sum。

相对于算法 1，算法 2 有了很大的进步，如果要求解自然数前 1000 项的代数和只要将第 5 步中的“add 不大于 10”修改成“add 不大于 1000”即可。

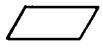
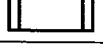
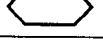
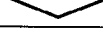
3. 算法的描述

算法考虑的是特定问题的求解方法与步骤，是在较高、较抽象层次上的解决问题的框架流程；编程则是根据算法给出的框架流程进行语言细化，实现问题求解的具体过程。因此，算法的描述不像编写程序那样有严格的、受限于具体语言的语法。可以使用下面几种工具描述算法。

（1）自然语言。即人们日常进行交流的所使用的语言，如英语、汉语等。虽然用自然语言描述算法对用户之间的交流较为方便，但由于容易出现歧义性、不方便描述包含分支和循环结构、目前尚难于直接在计算机上实现等原因，所以一般不用自然语言描述算法。

（2）流程图。流程图是用一些特定的图形符号表示相应操作及其顺序的图表。表 1-1 列举了常用的一些流程图符号。

表 1-1 常用流程图符号

符 号	名 称	说 明
	开始框	程序或算法开始
	结束框	程序或算法结束
	输入/输出框	输入/输出数据，可注明数据名称、来源、用途或其他文字说明
	处理框	能改变数据值或位置的各种处理，可注明处理名称或其简要功能
	特定处理框	可注明特定处理名称或简要功能，表示已命名的处理
	判断框	可注明判断的条件。只有一个入口，但可有若干个可供选择的出口
	流程线	连接其他符号，指明程序流向

(续表)

符 号	名 称	说 明
○	连接点	转到图的另一部分或从图的另一部分转来；通常在同一页上
□	换页连接	转到另一页图上或由另一页图转来
□	文档	通常表示打印输出，也可表示用打印终端输入数据
□	磁盘	磁盘输入或输出，也可表示存储在磁盘上的文件或数据库

【例 1-2】 用流程图描述【例 1-1】中的算法 2。

流程图如图 1-1 所示。

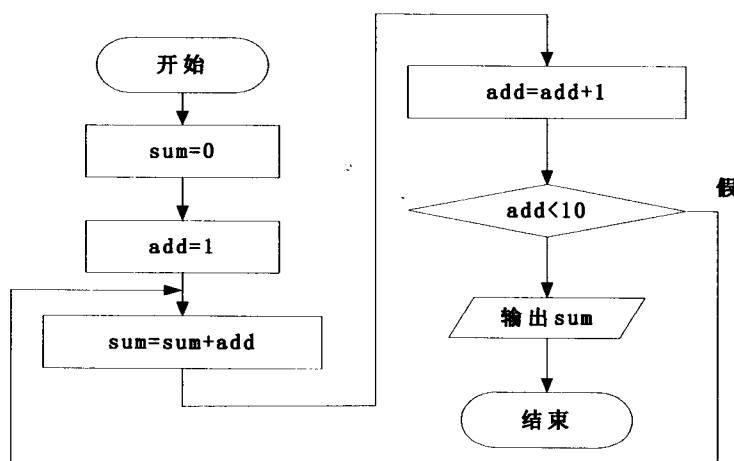


图 1-1 流程图示例

一个流程图至少包括 3 部分：表示相应操作的框、流程线和框内外必要的文字说明。应特别注意的是流程线必须带箭头以指明流程图上各框所示操作执行的先后次序。

流程图的优点是直观清晰、易于使用，是开发者普遍采用的工具，但它有严重缺点。

①由于流程线的存在，设计时可随心所欲地画流程线的流向，容易造成非结构化的程序结构。编码时势必频繁使用 goto 语句，导致基本控制块多入口多出口，使流程图难于理解，继而影响软件质量。

②不易反映逐步求精过程，往往反映的是最后结果。

③不易表示数据结构。

为解决流程线导致程序流程随意流转问题，可以采用 N-S 图、PAD 图等来描述算法流程。

(3) N-S 流程图。简称 N-S 图，由美国学者 I.Nassi 和 B.Shneiderman 提出的一种结构化流程图。N-S 图没有流程线，而是针对程序的 3 种基本结构，使用 4 种基本框图加以表示，全部算法均写在 1 个矩形框内。由于没有流程线，N-S 图从根本上杜绝了由流程线导致的非结构化程序流程，非常适合于结构化程序设计。

图 1-2、图 1-3、图 1-4 分别是 N-S 图的 4 种基本框图。其中图 1-2 的“A”、“B”两个