



普通高等教育应用型规划教材·计算机类

C语言程序设计

(第三版)

罗朝盛 张银南 ■ 主编



章节结构体系创新，内容安排由浅入深，通俗易懂
例题丰富实用，举一反三，培养学生语言应用能力
以任务驱动方式设计实验CAI系统，改善实验效果
教学资源丰富，配套讲解视频，可扫描二维码观看



科学出版社

普通高等教育应用型规划教材·计算机类

C 语言程序设计

(第三版)

罗朝盛 张银南 主 编

魏 英 白宝钢 副主编

科学出版社

北京

内 容 简 介

本书共 10 章, 主要内容包括 C 语言程序设计概述, 数据类型与常用库函数, 运算符与表达式, 控制结构程序设计, 数组、字符串与指针, 函数, 编译预处理, 结构体、共用体与枚举类型, 文件操作, C++ 程序设计初步等。

本次再版融入“互联网+教育”的理念, 除配有大量例题和习题外, 本书提供了重点教学内容和大部分例题的讲解视频, 读者可以扫描书中的二维码进行观看; 还提供了与教材配套的 PPT 课件和“C 语言程序设计实验 CAI 系统”, 使用本 CAI 系统进行上机实验, 可以帮助学生更加明确上机实验的目的, 明显改善实验效果, 减轻教师指导学生实验的工作量。

本书可以作为各类高等院校理工科学生学习“C 语言程序设计”课程的教材, 也可作为广大计算机爱好者学习 C 语言程序设计的参考书。

图书在版编目 (CIP) 数据

C 语言程序设计/罗朝盛, 张银南主编. —3 版. —北京: 科学出版社, 2018.6

ISBN 978-7-03-057502-9

I. ①C… II. ①罗…②张… III. ①C 语言-程序设计-高等学校-教材
IV. ①TP312

中国版本图书馆 CIP 数据核字 (2018) 第 110163 号

责任编辑: 孙露露 常晓敏 / 责任校对: 王 颖

责任印制: 吕春珉 / 封面设计: 曹 来

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>

三河市良远印务有限公司印刷

科学出版社发行 各地新华书店经销

*

2006 年 11 月第 一 版 开本: 787×1092 1/16

2012 年 2 月第 二 版 印张: 18 1/4

2018 年 6 月第 三 版 字数: 413 000

2018 年 6 月第十八次印刷

定价: 46.00 元

(如有印装质量问题, 我社负责调换〈良远〉)

销售部电话 010-62136230 编辑部电话 010-62135397-2010

版权所有, 侵权必究

举报电话: 010-64030229; 010-64034315; 13501151303

前言

本书曾被中国科学院教材建设专家委员会评为信息技术类“优秀教材(部级)二等奖”,并获浙江省“十一五”重点建设教材。第二版自2012年2月由科学出版社出版以来,先后多次重印,受到广大读者的欢迎,被全国多所高等院校选为教材,得到了不少专家、教师和学生的好评。几年来,我们收到不少读者和教师来信,他们在肯定本书的同时,也提出了不少有益的意见和建议。我们根据读者反馈的意见,结合这几年来使用本书的教学实践,进行了再次修订,推出第三版。第三版保持了第一、二版的写作风格和特色,对部分章节内容进行了删减、调整和充实。

第三版与第二版相比较,主要做了以下几个方面的调整。

1) 根据实际教学需要,删减、调整、充实了部分章节。将原“4.2 节算法及算法的表示”调整到第1章的1.4节,新的第4章修改为“控制结构程序设计”。

2) 对各章内容和文字进行了细致地修改,使读者更容易理解与掌握。

3) 精选和充实了教材的例题。例题是为帮助读者理解、掌握教学内容而设计的程序范例,此次修改更换了部分例题,对原有例题的程序代码进行了优化,使读者更容易阅读理解。对大部分例题给出了编程分析,在例题最后,针对一些要求学生掌握并容易出错的问题设计了“思考与讨论”,使读者通过学习能够做到举一反三,加深对所学内容的理解和掌握,提高编程能力。

4) 考虑到互联网在教育教学中的应用,本次再版融入了“互联网+教育”的理念,提供了重点教学内容和大部分例题的讲解视频,读者可以扫描书中的二维码进行观看;还提供了与教材配套的PPT课件和“C语言程序设计实验CAI系统”,方便读者更好地利用本书进行教学与学习。

本次再版由罗朝盛提出修改思路、修订方案并最后修改定稿。参与再版工作的有罗朝盛、张银南、魏英、马杨琿、朱梅、琺洁慧、庄儿、龚婷、楼宋江、陈琰宏、白宝钢等,马杨琿、朱梅、琺洁慧、龚婷、庄儿等为本书提供了视频资源,最后由罗朝盛、张银南负责统稿。

在本次再版过程中,我们得到了浙江科技学院信息与电子工程学院计算机基础教学部全体教师的帮助和支持,吸取了不少读者反馈的意见和建议,以使教材内容能更加符合读者需要,在此对他们表示衷心的感谢。

本书虽经多次讨论并反复修改,但限于作者水平,不当之处仍在所难免,谨请广大读者批评指正。

目 录

第1章 C语言程序设计概述	1
1.1 计算机程序设计概述	1
1.1.1 程序与程序设计语言	1
1.1.2 程序设计方法概述	3
1.2 C语言简介	4
1.2.1 C语言的发展	4
1.2.2 C语言的特点	5
1.2.3 C语言的字符集	6
1.2.4 C语言的标识符	7
1.3 C语言的程序结构	8
1.3.1 几个简单的C语言程序实例	8
1.3.2 C语言程序的结构特点	10
1.3.3 C语言函数的结构	10
1.3.4 C语言程序的书写风格	11
1.4 算法及算法的表示	11
1.4.1 算法概述	11
1.4.2 算法的特性	12
1.4.3 算法的表示	12
1.4.4 用N-S流程图表示算法	14
1.4.5 程序设计过程	16
1.5 运行一个C语言程序	16
1.5.1 C语言程序运行的一般步骤	16
1.5.2 在Microsoft Visual C++环境下编辑、编译及运行C语言程序	17
小结	20
习题	21
第2章 数据类型与常用库函数	23
2.1 C语言的数据类型	23
2.1.1 C数据类型概述	24
2.1.2 基本数据类型	24
2.1.3 各类数据在内存中的存放方式	25
2.2 常量	28
2.2.1 数值常量	28
2.2.2 字符常量	29

2.2.3	字符串常量	30
2.2.4	转义字符	30
2.2.5	符号常量	31
2.3	变量	32
2.3.1	变量的定义	32
2.3.2	变量的赋初值	32
2.4	指针变量	33
2.4.1	地址与指针的概念	33
2.4.2	变量的存储与访问	33
2.4.3	指针变量的定义	34
2.4.4	指针变量的初始化	35
2.4.5	指针变量的访问	35
2.5	标准输入/输出函数用法	36
2.5.1	格式化输出函数 printf()	36
2.5.2	格式化输入函数 scanf()	40
2.5.3	字符数据非格式输入/输出函数	43
2.6	常用库函数	44
2.6.1	数学函数	45
2.6.2	随机函数	45
2.6.3	字符函数	46
小结		46
习题		47
第3章	运算符与表达式	49
3.1	运算符概述	49
3.1.1	C 语言运算符分类	49
3.1.2	运算符的优先级和结合性	50
3.2	算术运算符与算术表达式	50
3.2.1	算术运算符	50
3.2.2	算术表达式	52
3.3	赋值运算符和赋值表达式	53
3.3.1	简单赋值运算符和表达式	53
3.3.2	复合赋值符及表达式	54
3.4	关系运算与逻辑运算	54
3.4.1	关系运算符与关系表达式	54
3.4.2	逻辑运算符与逻辑表达式	55
3.5	逗号运算符和逗号表达式	57
3.6	位运算符	57
3.7	指针变量的运算	61
3.7.1	指针运算符	61
3.7.2	指针变量的运算	61

3.8 数据类型隐式转换与强制转换	62
小结	63
习题	64
第4章 控制结构程序设计	67
4.1 C语句结构	67
4.2 顺序结构	69
4.3 选择结构	70
4.3.1 if条件语句	70
4.3.2 条件运算符和条件表达式	74
4.3.3 switch语句(开关语句)	75
4.3.4 选择结构的嵌套	77
4.3.5 选择结构程序举例	78
4.4 循环结构	80
4.4.1 while语句	80
4.4.2 do...while语句	81
4.4.3 for循环语句	83
4.4.4 循环的嵌套——多重循环结构	86
4.4.5 3种循环语句比较	87
4.5 其他控制语句	88
4.5.1 break语句	88
4.5.2 continue语句	89
4.5.3 goto语句	89
4.6 应用程序举例	90
4.6.1 素数与哥德巴赫猜想	90
4.6.2 穷举法	92
4.6.3 迭代法	93
小结	94
习题	95
第5章 数组、字符串与指针	100
5.1 数组概述	100
5.2 一维数组	101
5.2.1 一维数组的定义	101
5.2.2 一维数组元素赋值	102
5.2.3 一维数组元素的引用	103
5.2.4 一维数组的基本操作	104
5.2.5 一维数组的应用——排序	105
5.3 二维数组与多维数组	107
5.3.1 二维数组的定义	107
5.3.2 二维数组元素的引用	108
5.3.3 二维数组的初始化	108

5.3.4	二维数组的基本操作	109
5.3.5	二维数组应用举例	110
5.3.6	多维数组的定义和引用	111
5.4	数组与指针	111
5.4.1	指向一维数组的指针	112
5.4.2	指向二维数组的指针	115
5.5	字符数组与字符串	118
5.5.1	字符数组与初始化	118
5.5.2	字符数组的引用	119
5.5.3	字符串与字符数组	119
5.5.4	字符数组的输入/输出	120
5.6	字符串与字符指针	121
5.6.1	指向字符串的指针	121
5.6.2	使用字符串指针变量与字符数组的区别	123
5.6.3	字符串处理函数	124
5.7	指针数组与多级指针变量	127
5.7.1	指针数组	127
5.7.2	指向指针的指针	128
5.8	应用程序举例	130
5.8.1	数据查找	130
5.8.2	插入法	132
5.8.3	字符串的处理	134
小结		137
习题		137
第 6 章	函数	144
6.1	函数概述	144
6.1.1	模块化程序设计方法	144
6.1.2	C 语言中函数的分类	146
6.2	函数的定义	147
6.2.1	函数的定义形式	147
6.2.2	函数返回值	147
6.2.3	形式参数的设计	148
6.2.4	函数原型	149
6.3	函数调用与参数传递	150
6.3.1	函数的调用方式	150
6.3.2	参数传递	151
6.3.3	指针变量作参数	152
6.3.4	数组作参数	154
6.3.5	函数的嵌套调用	159
6.3.6	函数的递归调用	161

6.4	函数与指针	162
6.4.1	返回指针值的函数	162
6.4.2	函数的指针	164
6.4.3	指向函数的指针变量作参数	165
6.5	main 函数的参数	166
6.6	函数与变量的作用域	167
6.6.1	局部变量与全局变量	167
6.6.2	动态存储变量与静态存储变量	171
6.6.3	内部函数与外部函数	172
6.7	应用程序举例	173
	小结	175
	习题	176
第7章	编译预处理	182
7.1	宏定义	182
7.1.1	无参宏定义	182
7.1.2	带参宏定义	184
7.1.3	带参宏与函数的比较	186
7.2	文件包含	187
7.3	条件编译	188
7.4	应用程序举例	189
7.4.1	建立自己的头文件	189
7.4.2	多个源文件组成的 C 语言程序	190
	小结	191
	习题	192
第8章	结构体、共用体与枚举类型	194
8.1	结构体类型与结构体变量	194
8.1.1	结构体概述	194
8.1.2	结构体的声明	195
8.1.3	结构体变量的定义	196
8.1.4	结构体变量的引用	197
8.1.5	结构体变量的初始化	199
8.1.6	指向结构体的指针	199
8.2	结构体数组	200
8.3	结构体与函数	203
8.4	动态数据结构——链表	205
8.4.1	问题的提出	205
8.4.2	链表的基本结构	206
8.4.3	链表的基本操作	206
8.5	共用体	214
8.6	枚举类型	217

8.7 自定义类型标识符	218
小结	219
习题	220
第9章 文件操作	227
9.1 文件系统的概念	227
9.1.1 文本文件与二进制文件	227
9.1.2 缓冲文件系统	228
9.1.3 文件类型指针	228
9.2 文件的打开与关闭	229
9.2.1 文件的打开	229
9.2.2 文件的关闭	230
9.3 文件的读写	231
9.3.1 文件读写函数概述	231
9.3.2 字符读写函数	231
9.3.3 字符串读写函数	233
9.3.4 格式读写函数	234
9.3.5 数据块读写函数	238
9.3.6 随机读写文件	239
9.4 应用举例	241
9.4.1 文件的加密和解密	241
9.4.2 文件的拆分与连接	242
小结	243
习题	244
第10章 C++程序设计初步	249
10.1 面向对象的概念	249
10.2 C++的输入与输出	250
10.2.1 使用 cout 进行输出	251
10.2.2 使用 cin 进行输入	252
10.3 函数重载	253
10.4 类与对象	254
10.4.1 类与对象的概念	254
10.4.2 对象的创建	256
10.4.3 构造函数	257
10.4.4 析构函数	259
10.5 继承与派生	260
10.5.1 继承的基本概念	260
10.5.2 派生类	261
小结	264
习题	265
参考文献	268

附录	269
附录 A ASCII 字符集	269
附录 B 运算符的优先级和结合性	270
附录 C 标准 C 库函数	270
附录 D C 语言程序设计实验 CAI 系统	275

本书学习建议

理解本书设计的意图,熟悉本书体系的执行过程

了解本书的编写目的

了解本书的编写原则和编写风格

了解本书的编写方法和编写工具

了解本书的编写人员

了解本书的编写过程

1.1 计算机程序设计概述

1.1.1 程序与程序设计语言

计算机是由硬件系统和软件系统两大部分组成。硬件是物理基础,而软件(程序)则是计算机的灵魂。没有硬件,计算机只是一台“裸机”;没有软件,计算机是“死”的。只有硬件和软件相结合,计算机才能“活”起来。所有的软件,都是程序员用程序设计语言编写的。

程序设计语言是计算机语言的一种,它是为了解决某一任务而制定的一组操作步骤。这些操作步骤是按照一定的顺序执行的。例如,要完成一个计算“1+2+3+4+5”的任务,可以设计如下程序:第一步,将数字1输入计算机;第二步,将数字2输入计算机;第三步,将数字3输入计算机;第四步,将数字4输入计算机;第五步,将数字5输入计算机;第六步,将数字1、2、3、4、5相加,得到结果15。显然,上面这个程序是按照一定的顺序执行的。任何人在看这个程序时,都能按照程序完成这个任务。因为这段程序是用自然语言编写的,很容易理解。同样,计算机能完成各种数据处理任务,我们可以用自然语言编写程序,即把每一步操作步骤,按计算机可以执行的步骤进行,完成某个数据处理任务。在自然语言中,每一步操作步骤,都必须用自然语言来描述。在计算机中,每一步操作步骤,都必须用计算机语言来描述。计算机语言就是计算机程序。即为计算机程序。

1.1.2 程序与程序设计语言

计算机语言是计算机语言的一种,它是为了解决某一任务而制定的一组操作步骤。这些操作步骤是按照一定的顺序执行的。例如,要完成一个计算“1+2+3+4+5”的任务,可以设计如下程序:第一步,将数字1输入计算机;第二步,将数字2输入计算机;第三步,将数字3输入计算机;第四步,将数字4输入计算机;第五步,将数字5输入计算机;第六步,将数字1、2、3、4、5相加,得到结果15。显然,上面这个程序是按照一定的顺序执行的。任何人在看这个程序时,都能按照程序完成这个任务。因为这段程序是用自然语言编写的,很容易理解。同样,计算机能完成各种数据处理任务,我们可以用自然语言编写程序,即把每一步操作步骤,按计算机可以执行的步骤进行,完成某个数据处理任务。在自然语言中,每一步操作步骤,都必须用自然语言来描述。在计算机中,每一步操作步骤,都必须用计算机语言来描述。计算机语言就是计算机程序。即为计算机程序。

计算机语言是计算机语言的一种,它是为了解决某一任务而制定的一组操作步骤。这些操作步骤是按照一定的顺序执行的。例如,要完成一个计算“1+2+3+4+5”的任务,可以设计如下程序:第一步,将数字1输入计算机;第二步,将数字2输入计算机;第三步,将数字3输入计算机;第四步,将数字4输入计算机;第五步,将数字5输入计算机;第六步,将数字1、2、3、4、5相加,得到结果15。显然,上面这个程序是按照一定的顺序执行的。任何人在看这个程序时,都能按照程序完成这个任务。因为这段程序是用自然语言编写的,很容易理解。同样,计算机能完成各种数据处理任务,我们可以用自然语言编写程序,即把每一步操作步骤,按计算机可以执行的步骤进行,完成某个数据处理任务。在自然语言中,每一步操作步骤,都必须用自然语言来描述。在计算机中,每一步操作步骤,都必须用计算机语言来描述。计算机语言就是计算机程序。即为计算机程序。

计算机语言是计算机语言的一种,它是为了解决某一任务而制定的一组操作步骤。这些操作步骤是按照一定的顺序执行的。例如,要完成一个计算“1+2+3+4+5”的任务,可以设计如下程序:第一步,将数字1输入计算机;第二步,将数字2输入计算机;第三步,将数字3输入计算机;第四步,将数字4输入计算机;第五步,将数字5输入计算机;第六步,将数字1、2、3、4、5相加,得到结果15。显然,上面这个程序是按照一定的顺序执行的。任何人在看这个程序时,都能按照程序完成这个任务。因为这段程序是用自然语言编写的,很容易理解。同样,计算机能完成各种数据处理任务,我们可以用自然语言编写程序,即把每一步操作步骤,按计算机可以执行的步骤进行,完成某个数据处理任务。在自然语言中,每一步操作步骤,都必须用自然语言来描述。在计算机中,每一步操作步骤,都必须用计算机语言来描述。计算机语言就是计算机程序。即为计算机程序。

第1章 C语言程序设计概述

本章学习要求

- 理解程序设计的基本概念和程序的执行过程
- 了解C语言的特点
- 掌握C语言中标识符的命名规则
- 掌握C语言程序的基本组成结构
- 理解算法及算法的表示
- 掌握开发和运行一个C语言程序的过程

1.1 计算机程序设计概述

1.1.1 程序与程序设计语言

计算机系统由硬件系统和软件系统两大部分组成。硬件是物质基础，而软件（程序）可以说是计算机的灵魂，没有软件的计算机只是一台“裸机”，什么也不能干，有了软件，才能灵动起来，成为一台真正的“电脑”。所有的软件，都是用计算机语言编写的。

1. 程序

什么是程序？广义地讲，程序就是为完成某一任务而制定的一组操作步骤。按该操作步骤执行，就完成程序所规定的任务。譬如，要完成一个评选“优秀团员”的任务，可以为该任务设计程序：第1步，发通知让同学申报或同学推荐；第2步，召开评审会议；第3步，将申报或同学推荐候选人材料交评委评阅，并投票评选出“优秀团员”。显然上面程序所规定的3个操作步骤，任何人看了都能按照程序规定的步骤完成该任务。因为这段程序是用自然语言书写，任务执行者是人。同样，计算机能完成各种数据处理任务，我们可以设计计算机程序，即规定一组操作步骤，使计算机按该操作步骤执行，完成某个数据处理任务。但是迄今为止，在为计算机设计程序时，尚不能用自然语言来描述操作步骤，必须用特定的计算机语言描述。用计算机语言设计的程序，即为计算机程序。

2. 程序设计语言

人和计算机交流信息使用的语言称为计算机语言或程序设计语言。计算机程序设计语言的发展经历了从机器语言（machine language）、汇编语言（assemble language）到高级语言（high level language）的历程。

（1）机器语言

计算机能直接识别和执行的语言是机器语言，机器语言以二进制数表示，即以“0”和

“1”的不同编码组合来表示不同指令的操作码和地址码，它是第 1 代计算机语言。用机器语言编写的程序称为计算机机器语言程序，这种程序不便于记忆、阅读和书写。但由于使用的是针对特定型号计算机的语言，故而运算效率是所有语言中最高的。

(2) 汇编语言

为了克服使用机器语言编程的难记、难读等缺点，人们进行了一种有益的改进，用一些简洁的英文字母、符号串（称为助记符）来替代一个特定的二进制串指令，比如，用“ADD”代表加法指令，“MOV”代表数据传送指令等，这样就很容易读懂和理解程序在干什么，程序维护也就变得方便了，这种程序设计语言称为汇编语言。

汇编语言属于第 2 代计算机程序语言。汇编语言适用于编写直接控制机器操作的低层程序，它与机器密切相关，移植性不好，但效率仍十分高。针对计算机特定硬件而编制的汇编语言程序，能准确发挥计算机硬件的功能和特长，程序具有占用内存空间少、执行速度快等特点，所以至今仍是一种常用而强有力的软件开发工具。当然，对于算法的描述，汇编语言程序不如高级语言直观。

汇编语言和机器语言都是面向机器的程序设计语言，一般称为低级语言。

(3) 高级语言

高级语言是一种与硬件结构及指令系统无关，表达方式比较接近自然语言和数学表达式的一种计算机程序设计语言。其优点是：描述问题能力强，通用性、可读性、可维护性都较好。其缺点是：执行速度较慢，编制访问硬件资源的系统软件较难。

高级语言的发明是计算机发展史上颇为惊人的成就之一。1954 年，第一个完全脱离机器硬件的高级语言——FORTRAN 问世了。50 多年来，共有几百种高级语言出现，目前在计算机中广泛使用的高级语言有几十种，影响较大、使用较普遍的有 FORTRAN、ALGOL、COBOL、BASIC、FoxBASE、LISP、NOBOL、PL/1、PASCAL、C、PROLOG、Ada、C++、C#、Visual C++、Visual Basic、Visual FoxPro、Delphi、Java 等。

C 语言是一种具有很高灵活性的高级语言，适用于系统软件、数值计算、数据处理等，使用非常广泛。面向对象的 C++ 语言及 Visual C++、Borland C++、C# 等集成开发工具，也深受程序开发者的青睐。

3. 程序的执行

用汇编语言和高级语言编写的程序称为“源程序”，计算机不能直接识别和执行。要把源程序翻译成机器指令，才能执行。

(1) 汇编程序的执行

汇编语言编写的程序必须由“汇编程序”（或汇编系统）将这些符号翻译成二进制数的机器语言才能运行。这种“汇编程序”就是汇编语言的翻译程序。汇编语言程序的执行过程如图 1-1 所示。



图 1-1 汇编语言程序的执行过程

(2) 高级语言程序的执行

高级语言编写的程序通常有编译和解释两种执行方式。

编译执行过程是将源程序整个编译成等价的、独立的目标程序，然后通过连接程序将目标程序连接成可执行程序，运行执行文件输出结果。其执行过程如图 1-2 所示。



图 1-2 高级语言编译执行过程

解释执行过程是将源程序逐句翻译，翻译一句执行一句，边翻译边执行，不产生目标程序。整个执行过程，解释程序都一直在内存中。解释方式执行过程如图 1-3 所示。



图 1-3 高级语言解释执行过程

目前，高级语言基本提供了集成开发环境，它集源程序编辑、编译（解释）、执行为一体，非常方便用户使用，如 Turbo C、Visual Basic、Visual C++、Delphi 等。

1.1.2 程序设计方法概述

程序设计是一门技术，需要相应的理论、技术、方法和工具来支持。就程序设计方法和技术的发展而言，可以划分为以下 3 个阶段。

1. 早期的程序设计

计算机发展的初期，由于 CPU 运行速度慢、内存容量小，因此衡量程序质量优劣的标准是占用内存的大小和运行时间的长短，这就导致了程序设计人员不得不把大量的精力耗费在程序设计技巧上。反映在程序结构上，对程序的流程没有严格的限制，程序员可以随心所欲地令流程转来转去，程序流程变得毫无规律，读者要花费很大的精力去追踪流程，使得程序很难修改和维护。

传统方法开发软件时间长、成本高、可靠性低、难以修改和维护等问题日益突出，这就出现了当时的“软件危机”。软件危机引起了人们的高度重视，不少计算机专家着手研究探讨产生软件危机的原因并探索解决软件危机的途径。于是，出现了对软件工程、软件管理、软件可靠性及程序设计方法（设计、编制、调试及维护程序的方法）等问题的研究。

2. 结构化程序设计

20 世纪 60 年代末，著名学者 Dijkstra 首先提出了“结构化程序设计（structured programming, SP）”的思想。这种方法要求程序设计者按照一定的结构形式来设计和编写程序，使程序易阅读、易理解、易修改和易维护。这个结构形式主要包括两方面的内容。

1) 在程序设计中，采用自顶向下、逐步细化的原则。按照这个原则，整个程序设计过程应分成若干层次，逐步加以解决。每一步是在前一步的基础上，对前一步设计的细化。这样，一个较复杂的大问题，就被层层分解成为多个相对独立的、易于解决的小模块，有利于程序设计工作的分工和组织，也便于开展调试工作。

2) 在程序设计中，编写程序的控制结构仅由 3 种基本的控制结构（顺序结构、选择结

构和循环结构，它们的算法表示将在 1.4 节介绍）组成，避免使用可能造成程序结构混乱的 goto 转向语句。

按照结构化程序设计方法，使设计编写的程序的控制结构由上述 3 种结构组成，这样的程序就是结构化程序。C 语言是一种结构化程序设计语言。

结构化程序设计技术虽已使用了几十年，但软件质量的问题仍未得到很好的解决。这是因为面向过程的程序设计方法仍然存在与人的思维方式不协调的地方，所以很难自然、准确地反映真实世界，因而用此方法开发出来的软件的质量还是很难保证，甚至需要进行重新开发。另外，该方法在实现中只突出了实现功能的操作方法（模块），而被操作的数据（变量）处于实现功能的从属地位，即程序模块和数据结构是松散地结合在一起。因此，当应用程序比较复杂时，容易出错，难以维护。

为适应现代化软件开发的需要，一种全新的软件开发技术应运而生，这就是面向对象的程序设计（object oriented programming, OOP）。

3. 面向对象的程序设计

面向对象的程序设计在 20 世纪 80 年代初就提出了，它起源于 Smalltalk 语言。用面向对象的方法解决问题，不再将问题分解为过程，而是将问题分解为对象。对象是现实世界中可以独立存在、可以被区分的实体，也可以是一些概念上的实体，现实世界是由众多对象组成的。对象有自己的数据（属性），也包括作用于数据的操作（方法），对象将自己的属性和方法封装成一个整体，供程序设计者使用。对象之间的相互作用通过消息传送来实现。这种“对象+消息”的面向对象的程序设计模式将取代“数据+算法”的面向过程的程序设计模式。

但要注意到，面向对象的程序设计并不是要抛弃结构化程序设计方法，而是要“站”在比结构化程序设计更高、更抽象的层次上去解决问题。当它被分解为低级代码模块时，仍需要结构化编程的方法和技巧，只是它分解一个大问题为小问题时采取的思路与结构化方法是不同的。结构化的分解突出过程，强调的是如何做（How to do?），代码的功能如何完成；面向对象的分解突出现实世界和抽象的对象，强调的是做什么（What to do?），它将大量的工作交由相应的对象来完成，程序员在应用程序中只需说明要求对象完成的任务。

目前，常用的面向对象的程序设计语言有 Borland C++、Visual C++、Visual FoxPro、Visual Basic、Java、Delphi、Visual Fortran 等。它们虽然风格各异，但都有共同的概念和编程模式。

1.2 C 语言简介

1.2.1 C 语言的发展

C 语言是在 20 世纪 70 年代初问世的。1973 年，美国电话电报公司（AT&T）贝尔实验室 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（basic combined programming language）的第 2 个字母作为这种语言的名字，这就是 C 语言。1978 年，由 Kernighan 和 Ritchie 出版的名著《The C Programming Language》一书，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，10 年后

(即 1983 年)由美国国家标准学会在此基础上制定了一个 C 语言标准,通常称为 ANSI C。

早期的 C 语言主要是用于 UNIX 系统。由于 C 语言的强大功能和各方面的优点逐渐为人们认识,到了 20 世纪 80 年代, C 开始进入其他操作系统,并很快在各类大、中、小和微型计算机上得到了广泛的使用,成为当代非常优秀的程序设计语言之一。

在 C 的基础上,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。目前,流行的 C++ 是 Borland C++、Symantec C++ 和 Microsoft Visual C++。C++ 提出了一些更为深入的概念,它所支持的这些面向对象的概念容易将问题空间直接地映射到程序空间,为程序员提供了一种与传统结构程序设计不同的思维方式和编程方法,因而也增加了整个语言的复杂性,掌握起来有一定难度。

C 是 C++ 的基础,C++ 语言和 C 语言在很多方面是兼容的。因此,掌握了 C 语言,再进一步学习 C++ 就能以一种熟悉的语法来学习面向对象的语言,从而达到事半功倍的目的。

1.2.2 C 语言的特点

C 语言发展如此迅速,而且成为颇受欢迎的语言之一,主要因为它具有强大的功能。许多著名的系统软件,如 FoxBASE、DBASE III 等都是用 C 语言编写的。用 C 语言加上一些汇编语言子程序,就更能显示 C 语言的优势,如 PC-DOS、Word 等就是用这种方法编写的。归纳起来,C 语言具有下列特点。

1. 简洁紧凑、灵活方便

标准 C 语言一共有 32 个关键字,9 种控制语句,程序书写自由,一个 C 语句可以写在一行上,也可分多行书写,主要用小写字母表示。

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,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 语言是结构化程序设计语言

结构化程序设计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。这些函数可以方便地调用,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 语言语法限制不太严格、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. C 语言允许直接访问物理地址，可以直接对硬件进行操作

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，可以用来写系统软件。所以，有人把 C 语言称为中级语言。

7. C 语言程序生成代码质量高，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. C 语言适用范围大

C 语言还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。C 语言既适合编写大型的系统软件，也适合编写应用软件。

C 语言的以上特点，读者现在也许还不能深刻理解，待学完 C 语言以后再回顾一下，就会有比较深的体会。

1.2.3 C 语言的字符集

字符是组成语言的最基本的元素。C 语言字符集由字母、数字、空格、标点和特殊字符组成。在字符常量，字符串常量和注释中还可以使用汉字或其他可表示的图形符号。

1) 字母：小写字母 a~z 共 26 个，大写字母 A~Z 共 26 个。

2) 数字：0~9 共 10 个。

3) 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。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。

4) 标点和特殊字符。

C 语言编程中可以使用的标点和特殊字符共有 30 个，如表 1-1 所示。

表 1-1 标点和特殊字符

符号	说明	符号	说明
%	百分号（整型数据求余运算符或输入格式开始符）	~	波浪符（位取反运算符）
&	和号（取地址或位与运算符）	=	等于号（关系运算符、赋值运算符）
!	感叹号（逻辑非）	(	左圆括号
#	磅号（预处理命令开始符）	)	右圆括号
+	加号	'	单引号（字符常量定界符）
-	减号	"	双引号（字符串常量定界符）
*	星号（乘法或取指针指向内容运算符）	,	逗号（运算符或分隔符）