



21 世纪高等学校规划教材
Textbook Series of 21st Century

C 语言程序设计

寒 枫 赵文清 崔克彬 编 著



中国电力出版社
<http://jc.cepp.com.cn>



21 世纪高等学校规划教材
Textbook Series of 21st Century

C 语言程序设计

编 著 寒 枫 赵文清 崔克彬
主 审 杨国兴



中国电力出版社
<http://jc.cepp.com.cn>

内 容 提 要

本书为 21 世纪高等学校规划教材。

全书共分为 13 章, 主要内容包括程序设计基础知识、C 语言入门、数据类型和表达式、C 语言程序结构、数组、函数、指针、构造类型、输入和输出、位运算、文件、综合实例、常见错误及调试。书后还附有常用字符与 ASCII 码对照表、C 语言上机操作指导等内容。

本书主要作为高等院校各专业“C 语言程序设计”课程的本科教材, 也可作为高职高专和函授的相关课程教材, 同时可作为计算机爱好者的参考用书。

图书在版编目 (CIP) 数据

C 语言程序设计/寒枫编著. —北京: 中国电力出版社, 2006

21 世纪高等学校规划教材

ISBN 7 - 5083 - 4608 - 4

I. C... II. 寒... III. C 语言—程序设计—高等学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字 (2006) 第 089484 号

中国电力出版社出版、发行
(北京三里河路 6 号 100044 <http://jc.cepp.com.cn>)

航远印刷有限公司印刷
各地新华书店经售

*

2006 年 9 月第一版 2006 年 9 月北京第一次印刷
787 毫米×1092 毫米 16 开本 20.75 印张 509 千字
印数 0001—3000 册 定价 29.80 元

版 权 专 有 翻 印 必 究

(本书如有印装质量问题, 我社发行部负责退换)

前 言

随着计算机科学的日益普及,无论是计算机专业还是非计算机专业大都将 C 语言作为重要的基础课程。

C 语言是在实用中逐步发展起来的语言。最初的 C 语言主要是为了描述和实现 UNIX 操作系统而研制出来的,其前身是出现在 1960 年的 ALGOL 60 语言。1972~1973 年间,贝尔实验室的 D. M. Ritchie 设计出了最早的 C 语言。后来, C 语言经过多次改进,但主要是在贝尔实验室内部使用。1983 年,美国国家标准化协会(ANSI)根据 C 语言问世以来的各种版本对 C 语言进行改进和扩充,制定了新的标准,称为 ANSI C。1987 年, ANSI 又公布了新标准——87 ANSI C。1990 年,国际标准化组织 ISO 接受 87 ANSI C 为 ISO C 的标准(ISO 9899—1990)。目前流行的 C 编译系统都是以它为基础的。

C 语言因编程灵活方便,语言简洁紧凑,运算符丰富,可以对变量地址进行运算等特点而受到使用者的喜爱。

本书全面讨论了 C 语言程序设计的有关概念。内容由浅入深地逐步展开,力图使初学者容易理解。同时,书中设计了大量的例题,帮助读者理解语法和概念的应用方法。所有的例题都在 Turbo C2.0 系统中运行通过。

C 语言程序设计方面的教材在市场上有各种版本,就其内容来看各有所长。我们组织编写的这本《C 语言程序设计》和它们相比有以下特点:

- (1) 教材的编写者来自教学第一线,有着丰富的讲授 C 语言的经验。在教材中加入了大量的生动形象的语言和实例,教材通俗易懂。
- (2) 教材大量采用图形和文字相结合的形式,将晦涩难懂的 C 语言运算符和语句、函数、指针、结构体、联合和文件等转化成易于理解的图示,便于学生自学。
- (3) 书中的例题都有详细的说明,并给出实际运行结果图示。
- (4) 每章结束都安排了适当的习题,便于学生理解和巩固所学的知识。

本书在编写过程中得到了华北电力大学计算机学院领导和同仁的大力支持,在此表示感谢。

此外,还要特别感谢华北电力大学参加课程学习的同学们和参加课程辅导的研究生们,是他们的思考和问题给了我们许多启示。特别是王延明、张燕、楚红涛同学做了大量程序调试的校对工作,在此表示感谢。

北京科技大学的杨国兴教授审校了全书,并提出了宝贵意见,在此表示感谢。

由于时间、水平和能力的限制,本书难免有疏漏和不当之处,恳请读者多加批评指正,也希望同行们对本书提出宝贵意见。

编 者

2006 年 6 月

目 录

前言

第 1 章 程序设计基础知识	1
1.1 程序设计语言发展史	1
1.2 算法的描述方法	2
小结	5
习题	6
第 2 章 C 语言入门	8
2.1 C 语言发展简史	8
2.2 C 语言构成和特点	9
2.3 C 语言源程序结构形式	11
2.4 C 语言程序执行过程	12
小结	13
习题	13
第 3 章 数据类型和表达式	14
3.1 C 语言的基本数据类型	14
3.2 整型数据	17
3.3 实型数据	20
3.4 字符型数据	22
3.5 运算符和表达式	25
小结	34
习题	35
第 4 章 C 语言程序结构	39
4.1 顺序结构	40
4.2 选择结构	41
4.3 循环结构	50
4.4 C 语言预处理命令	64
小结	73
习题	73
第 5 章 数组	75
5.1 一维数组的定义和引用	75
5.2 二维数组的定义和引用	80
5.3 字符数组和字符串	86
小结	95
习题	96

第6章 函数	99
6.1 C语言程序结构	99
6.2 C语言的函数	101
6.3 函数的定义和说明	102
6.4 函数的调用	105
6.5 常用库函数	127
小结	136
习题	137
第7章 指针	141
7.1 指针的基本概念	141
7.2 指针与变量	142
7.3 指针与数组	152
7.4 指针与函数	163
7.5 指针与字符串	167
7.6 指针数组	171
7.7 多级指针	173
7.8 带参数的主函数 main	175
小结	176
习题	178
第8章 构造类型	180
8.1 结构体	180
8.2 链表	200
8.3 联合	210
8.4 枚举类型	213
8.5 自定义类型标识符 typedef	216
小结	217
习题	217
第9章 输入和输出	219
9.1 C语言的标准输入和标准输出	219
9.2 格式化输出函数 printf()	219
9.3 格式化输入函数 scanf()	223
9.4 单个字符输入输出函数 putchar()和 getchar()	227
9.5 字符串输入输出函数 gets()和 puts()	228
小结	229
习题	229
第10章 位运算	233
10.1 原码、反码和补码	233
10.2 位运算符的使用	235
10.3 位运算应用实例	241

小结	243
习题	243
第 11 章 文件	246
11.1 C 文件的概念	246
11.2 文件类型和文件指针	246
11.3 文件的打开与关闭	248
11.4 文件的读写	249
11.5 文件的定位与随机读写	259
11.6 文件操作的出错检错	262
习题	268
第 12 章 综合实例	270
12.1 程序功能	270
12.2 程序设计目的	270
12.3 程序设计	270
第 13 章 常见错误及调试	286
13.1 常见错误	286
13.2 程序调试	294
小结	296
附录 A 常用字符与 ASCII 代码对照表	297
附录 B C 语言上机操作指导	298
B.1 运行一个 C 语言程序的一般过程	298
B.2 Turbo C 的启动、退出与常用命令菜单	298
B.3 编辑并保存一个 C 语言程序	300
B.4 编译、链接单个源程序文件	301
附录 C C 语言中的关键字	305
附录 D 运算符和结合性	306
附录 E C 语言常用语法提要	308
附录 F Turbo C (V2.0) 编译错误信息详表	312
附录 G 库函数	318
参考文献	325

第1章 程序设计基础知识

本章介绍程序设计语言的发展简史与特点、程序设计语言的应用环境以及开发应用程序的主要步骤。同时,在程序设计的基本知识里还要介绍描述算法的基本方法。通过本章的学习,对程序设计语言和算法有基本的了解。

1.1 程序设计语言发展史

在计算机普及的今天,计算机的应用已经深入到许多领域。而计算机所用的程序设计语言也从最早面向计算机的低级语言发展到今天的面向使用者的高级语言。程序设计语言的发展大致经历了如下几个阶段:

1. 机器语言 (第一代语言)

一种 CPU 的指令系统,称为这种 CPU 的机器语言。由 CPU 可以识别的一串由 0 和 1 组成的指令码,称为机器语言。

用机器语言编程序,就是把由计算机来解决问题的步骤用机器指令的序列来进行描述。其特点是,这种机器语言编写的程序计算机可以直接理解、执行。但是,对于使用机器语言编写程序的人来说,机器语言难记忆、难理解、难查错、难修改。只有很少数的专业人员才可以掌握。

2. 汇编语言 (第二代语言)

随着计算机的普及,程序员使用机器语言编程难度太大,效率很低又很繁琐。之后,产生了由类似英文缩写的助记符构成的程序设计语言,称为汇编语言。机器不能直接识别、理解和执行汇编语言编制的程序,要先通过汇编器的翻译转换为机器语言才能理解和执行。随着汇编语言的出现,计算机的应用迅速增加,但是,即使最简单的任务,也需要许多条指令才能完成。

机器语言和汇编语言都是根据 CPU 的不同而异的,它们都属于面向机器的语言。使用它们编程的时候,不仅要考虑解题思路,还要熟悉机器的内部结构,对内存要进行手工分配,这种形式的编程劳动强度大,妨碍计算机的普及和推广。

3. 面向过程的高级语言 (第三代语言)

为了提高编程速度,改善程序员的编程环境,人们开发了高级语言,可以用一条语句完成大量的任务。高级语言通过编译器翻译成相应的机器语言。高级语言的程序不再面向具体的机器,而是面向解题过程,即用人们易于理解的形式来表示解题的过程,称为面向过程的高级语言,也称为算法语言。从程序员角度看,高级语言比机器语言和汇编语言好许多,C 语言就属于广泛使用的高级语言。

4. 新一代的高级语言 (第四代语言)

使用过程化语言编程的特点就是把要用计算机解什么题,转换成计算机怎么做才能解这个题,即把“做什么?”转换成“怎么做?”,也就是说把解题过程编写成高级语言程序。而

第四代语言是非过程化的语言，是面向应用层的，对一个应用只要说明“做什么？”，不需要再说明“如何做？”，这样的语言一般称为“查询语言”。

第四代语言虽然没有确切的定义，但是，有一些公认的特征，例如，大部分第四代语言是采用非过程化的机制。第四代语言和数据库的关系密切，你可以很方便地查询到数据库的数据。第四代语言还有许多其他的特征，这里不再一一列举。

简而言之，第四代语言是用来快速开发事务处理系统的高效软件工具。第四代语言不支持应用软件开发的全过程，而是侧重于支持应用软件开发过程的设计阶段和实现阶段的工作。

1.2 算法的描述方法

程序设计指的是用计算机语言编制解决某一问题的程序。我们可以把程序看作是：

程序 = 算法 + 数据结构 + 程序设计方法 + 语言工具和环境

从程序包含的几个部分看，要想得到一个正确的程序，算法是很重要的一个部分。任何计算问题都可以通过按特定顺序执行一系列操作而完成。解决问题的过程称为算法。算法包括要执行的操作和执行操作的顺序两部分。

算法的特点是：

- (1) 解题的操作序列是个有穷序列。
- (2) 操作序列只有一个初始操作动作。
- (3) 序列中每一个操作动作仅有一个后继动作。
- (4) 操作序列结束表示问题得到或没有得到解答。

描述算法的方法很多，例如，程序流程图、N-S图、PAD图及自然语言描述算法的方法。下面简单介绍三种常用算法描述方法。

1. N-S图^①

N-S图是一种常用的描述算法的图形工具。它的特点就是没有连接线，只有三种基本结构。不论什么样的操作过程，都是用这三种基本结构来描述。

N-S图的基本结构如图 1-1 所示。

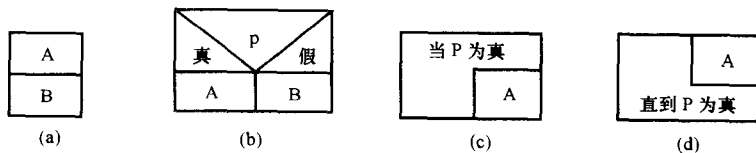


图 1-1 N-S图基本结构

(a) 顺序结构；(b) 选择结构；(c) 当型循环结构；(d) 直到型循环结构

2. PAD图

PAD图 (Problem Analysis Diagram, 问题分析图) 是一种常用的描述算法的图形工

^① N-S图是编程过程中常用的一种分析工具，N-S图是由两个人合作完成的，他们就是 Ike Nassi 和 Ben Schneiderman。在大洋彼岸的美国的 N-S图之父 Ike Nassi，他有一个非常中国的名字：王大西。他的网页是：
www.nassi.com

具。它的特点是在 PAD 图左侧的一根组织线上连接顺序、选择、循环结构组成的一个二维关系图，自上而下，从左向右的描述层次关系。

PAD 图的三种基本结构分别为：顺序结构（见图 1-2）、选择结构（见图 1-3）、循环结构（见图 1-4）。

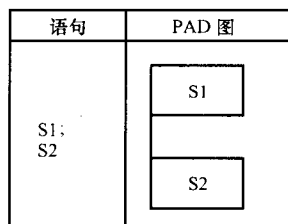


图 1-2 顺序结构 PAD 图

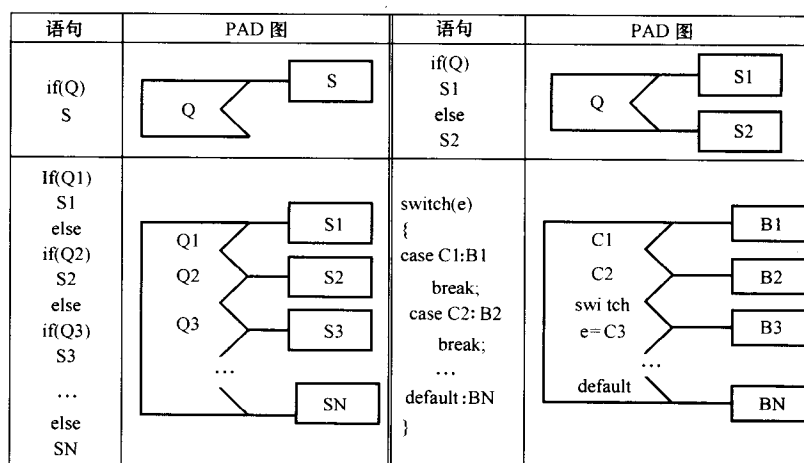


图 1-3 选择结构 PAD 图

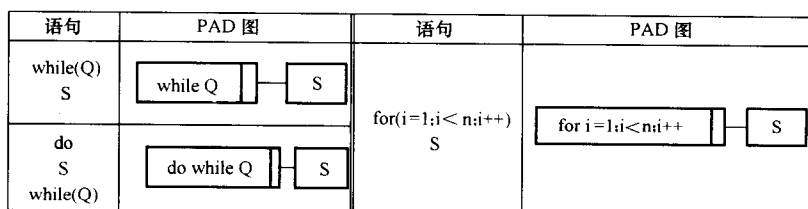


图 1-4 循环结构 PAD 图

3. 伪代码

图形的算法工具有直观、易懂的优点，也有修改不易的不足。而用自然语言描述算法理解起来不易，很容易产生二义性。伪代码是介于自然语言与计算机语言之间的利用文字和符号描述算法的工具。主要特点是借助高级语言的控制结构和自然语言混合的方法共同描述算法。

使用伪代码描述算法，通常是先用自然语言，从问题域出发，给出一个高度抽象和概括

的算法，目的是和用户进行很好的沟通；之后，再对上述算法细化。即把算法的每一个局部（也称子部分）细化，给出高级语言描述的局部结构。细化的过程根据问题的复杂程度而不同。最终，当算法全部用高级语言所描述时，也就完成了这个算法的程序设计。

用计算机解题的过程就是描述算法并且把算法按照一定规则转换成计算机能够识别的高级语言程序的过程。下面举例说明。

【例 1-1】 求 1~100 整数之和。

算法：（伪代码描述）

- ①用 sum 变量存放求和的结果，给 sum 赋初值，sum=0；
- ②用循环的方法，给 sum 加 1，即 sum=sum+1；
- ③输出最后结果。

细化：

- ①int sum=0；
- ②for (i=0; i<=100; i++) sum=sum+i；
- ③printf("sum=%d\n", sum)；

算法转换成 C 程序：

这个简单的求和问题，算法简单，转换为 C 语言程序也很容易。

```
#include <stdio.h>
```

```
void main( )
```

```
{   int i,sum=0;                        /* 将 sum 初始化为 0 */
    for( i=0; i<=100; i++ ) sum=sum+i; /* 利用 for 循环求和 */
    printf("sum=%d\n", sum);          /* 打印求和结果 sum */
}
```

程序运行结果：

```
sum=5050
```

程序的 N-S 图和 PAD 图如图 1-5、图 1-6 所示。

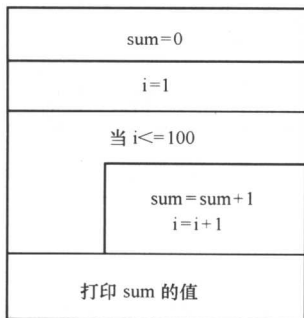


图 1-5 【例 1-1】 N-S 图

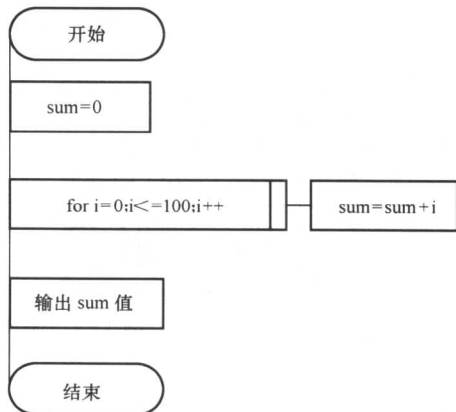


图 1-6 【例 1-1】 PAD 图

【例 1-2】 输入整数 m，求 m!。

算法：（伪代码描述）

- ①输入 m 的值。

②利用 for 循环求 $m!$ 。

③输出 $m!$ 的结果。

细化：

①输入 m 的值。scanf ("%d", &m);

②设计求 $m!$ 。

$n=1$;

for(int i=2; i<=m; i++) $n=n*i$;

③输出结果

printf("%d! is %d ",m, n);

算法转换为 C 语言程序：

#include <stdio. h>

void main()

{

int i,m,n=1;

printf("input:");

scanf("%d",&m);

/* 输入 m 的值 */

for(i=2; i<=m; i++) $n=n*i$;

/* 利用 for 循环求 $m!$ */

printf("%d! is %d ",m, n);

/* 输出 $m!$ 的值 */

}

程序运行结果：

```
input:5
5! is 120
```

程序对应的 N-S 图和 PAD 图如图 1-7、图 1-8 所示。

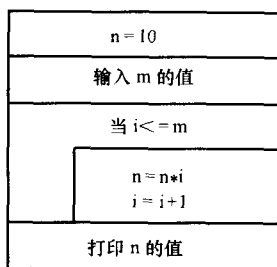


图 1-7 【例 1-2】N-S 图

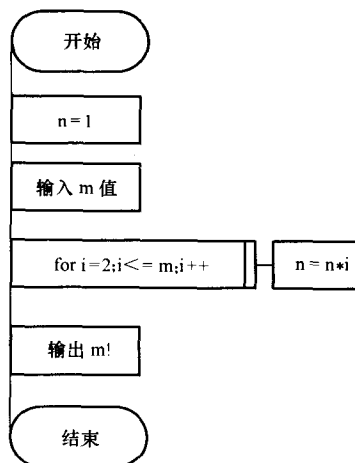


图 1-8 【例 1-2】PAD 图

这里简单介绍了常用的程序设计过程中的算法描述方法，初学者可以根据具体的实际问题，逐步练习使用这些方法来描述一个算法，为编程打下良好的基础。

小 结

本章介绍了常用的算法描述工具 N-S 图和 PAD 图。通过对 N-S 图和 PAD 图的了解，

熟练掌握它们的使用方法,能对读者尽快学会和应用 C 语言进行程序设计起到事半功倍的效果。对于初次学习程序设计的读者,掌握一定的描述算法的理论和工具是大有裨益的。

习 题

1. 请叙述程序发展的几个阶段及其特点。
2. 什么是算法?试从日常生活中找出几个例子,分别描述它们的算法。
3. 常用算法描述方法有哪几种?分别是什么?
4. 有两个瓶子 A 和 B,分别盛放酱油和醋,要求将它们互换,请用 N-S 图表示这个算法。
5. 依次将 10 个数输入,要求按大小顺序把它们打印出来,请用伪代码表示此算法。
6. 有 3 个数 a、b、c,要求按大小顺序把它们打印出来,请用 N-S 图表示此算法。
7. 求 $1+2+3+\cdots+100$,请用伪代码表示此算法。
8. 判断一个数 N 能否同时被 3 和 5 整除,请用 N-S 图表示此算法。
9. 将 100~200 之间的素数打印出来,请用伪代码表示此算法。
10. 求两个数 M 和 N 的最小公倍数,请用 N-S 图表示此算法。
11. 求方程式 $ax^2+bx+c=0$ 的根。分别考虑:①有两个不等的实根;②有两个相等的实根,请用伪代码表示此算法。
12. 用自顶向下、逐步细化的方法进行以下算法的设计:
 - (1) 打印出 1900~2000 年中是闰年的年份,闰年的条件是:①能被 4 整除但不能被 100 整除;②能被 100 整除且能被 400 整除。

(2) 求 $ax^2+bx+c=0$ 的根。分别考虑 $d=b^2-4ac$ 大于 0、等于 0 和小于 0 三种情况。

(3) 输入 10 个数,找出最大的一个数,并打印出来。

13. 分别用 N-S 图、PAD 图描述下列程序,并写出程序的执行结果。

(1)

```
void main( )
{
    int value1,value2,sum;
    value1=50;
    value2=25;
    sum=value1+value2;
    printf("The sum of %d and %d is %d\n",value1,value2,sum);
}
```

(2)

```
void main( )
{
    int a;
    scanf("%d",&a);
    if(a>=0)
        printf("plus\n");
    else
        printf("minus\n");
}
```

14. 请设计下列算法：

- (1) 有两盘磁带，A 录英语，B 录音乐，把它们交换过来。
- (2) 依次输入 10 个数，找出最大数。
- (3) 把三个任意数按大小顺序排列。
- (4) 判断一个整数 n 能否被 3 和 5 整除。
- (5) 求两个正整数 m 和 n 的最大公约数。

第 2 章 C 语言入门

本章主要介绍 C 语言的发展简史、特点, 程序设计中的一些基本知识以及 C 语言程序执行的过程。

2.1 C 语言发展简史

C 语言的发展历史比较独特, 它是在实用中逐步发展起来的语言。C 语言和 UNIX 操作系统从一开始就紧密地联系在一起。最初的 C 语言主要是为了描述和实现 UNIX 操作系统而研制出来的, 其前身是出现在 1960 年的 ALGOL 60 语言。ALGOL 60 是一种面向问题的高级语言。不过, 由于它与硬件联系较远, 不适合编制系统程序。1963 年, 剑桥大学在 ALGOL 60 的基础上设计了 CPL (Combined Programming Language) 语言, 但没有实现。1967 年, 剑桥大学的 Martin Richards 对 CPL 语言进行简化, 推出 BCPL (Basic Combined Programming Language) 语言。1970 年, 美国贝尔实验室的 Ken Thompson 在 BCPL 的基础上, 进一步简化并设计出了更简单、更接近硬件的 B 语言 (取 BCPL 的第一个字母), 并用 B 语言在 PDP-7 上编写了 UNIX 操作系统。1971 年, 又在 PDP-11/20 上实现了 B 语言, 同时编写了 UNIX 操作系统。由于 B 语言过于简单, 功能也有限, 1972~1973 年间, 贝尔实验室的 D.M. Ritchie 在 B 语言的基础上设计出了最早的 C 语言 (取 BCPL 的第二个字母)。C 语言保持了 B 语言的优点, 又克服了它们过于简单、没有数据类型的缺点。1973 年, 又用 C 语言重写了 UNIX, 即 UNIX 第 5 版。

后来, C 语言经过多次改进, 但主要是在贝尔实验室内部使用。1975 年 UNIX 第 6 版公布后, C 语言的突出优点引起了人们的注意。1977 年出现了不依赖具体机器的 C 语言编译文本——可移植 C 语言编译程序, 使 C 语言移植到其他机器上较为容易。随着 UNIX 的日益广泛使用, C 语言也迅速得到推广。1978 年, Brian W. Kernighan 和 Dennis M. Ritchie (合称 K&R) 合著了影响深远的名著《The C Programming Language》。这本书中介绍的 C 语言成为后来广泛使用的 C 语言版本的基础, 它被称为标准 C。1983 年, 美国国家标准化协会 (ANSI) 根据 C 语言问世以来的各种版本对 C 进行改进和扩充, 制定了新的标准, 称为 ANSI C。1987 年, ANSI 又公布了新标准——87 ANSI C。1990 年, 国际标准化组织 ISO 接受 87 ANSI C 为 ISO C 的标准 (ISO/IEC 9899-1990), 目前流行的 C 编译系统都是以它为基础的。

20 世纪 90 年代以来, 随着程序设计理论和技术的不断发展, C 语言中引进了类的概念, 就是后来的 C++ 语言的雏形。C++ 语言改进了 C 语言的不足, 形成了既具有 C 语言的简洁风格, 又支持面向对象技术的综合性语言, 并得到越来越广泛的应用。

2.2 C 语言构成和特点

C 语言有非常简洁和层次分明的程序结构，有自己独特的字符集和丰富的运算符，有其鲜明的特点，从而被广泛应用。下面对 C 语言的构成和特点做详细的介绍。

2.2.1 C 语言构成

每种计算机语言，都有自己的语法规则和表现形式，有自己的程序结构规则和特定的书写方式。下面给出 C 语言允许使用的字符集和标识符。

1. 字符集

字符集包括大写字母、小写字母、阿拉伯数字和一些特殊字符。特殊字符如下：

+ - * / % _ (下划线) = < > & ~ () [] . { } : ? ;
" ' ! # \$ % & ' () * + , - . / : ;
_ { | } ^ & * ~

2. 标识符

标识符是用来表示常量、变量、函数、类型等的名字。有保留字、预定义标识符、用户定义标识符三类。

保留字是 C 语言具有特殊含义的标识符，不允许用作程序的变量名称。常见的保留字包括类型说明符和语句标识符等。C 语言的保留字一般用小写字母表示，如下所示：

auto	break	case	char	const	continue	default	do
double	else	enum	extern	float	for	goto	if
int	long	register	return	short	signed	sizeof	static
struct	switch	typedef	union	unsigned	void	volatile	while

预定义标识符是包括预编译命令和 C 编译系统提供的库函数名。预编译命令有 define 和 include 等。

用户定义标识符是程序员根据要解决的问题在程序里定义的变量名、文件名、函数名等标识符。

用户定义标识符由字母、数字、下划线组成，并且只能由字母和下划线开头。

3. 标识符使用注意事项

(1) 变量名称要由含义明确的词或公认的缩写形式来表示，使读者见词就能够明意。

(2) 标识符一般采取“常用取简专用取繁”的原则。同时也要注意“约定俗成”的习惯。

(3) 如果一个标识符需要由一组词来描述，建议在词与词之间用下划线连接。

(4) 可以用特定的字符作其前缀标识或者后缀标识来表示某类数据，提高可读性。

(5) C 语言内部的各种标识符一般是以下划线开头的，用户的标识符尽量避免以下划线开头，以免造成系统错误。

(6) 注意字母的大小写问题。C 语言是区分大小写的。

(7) 标识符的长度要符合 C 语言的规定。不同的 C 语言版本，支持标识符的长度不同。

标识符举例如下。

正确的标识符：

PI 表示圆周率用标识符, R 表示半径, student 表示学生, my_file 表示我的文件, student_name 表示学生姓名, student_dept 表示学生所在系, A1 和 a1 在 C 语言中是两个不同的变量, 因为 C 语言区分大小写字母。

错误的标识符:

for C 语言保留字, 不能作变量

66abc 不能以数字开头

"TEL" 有非法字符引号

-1-2A 标识符只能用下划线, 不能用减号

2.2.2 C 语言特点

1. C 语言比其他高级语言更接近硬件

C 语言允许直接访问物理地址, 能进行位 (bit) 操作, 能实现汇编语言的大部分功能, 可以直接对硬件进行操作。因此, C 既具有高级语言的功能, 又具有低级语言的许多功能, 可用来编写系统软件。C 语言的这种双重性, 使它既是成功的系统描述语言, 又是通用的程序设计语言。有人把 C 称为“高级语言中的低级语言”或“中级语言”, 意为兼有高级和低级语言的特点。一般仍习惯将 C 语言称为高级语言, 因为 C 程序也要通过编译、链接才能得到可执行的目标程序, 这和其他高级语言是相同的。

2. C 语言是结构化的程序设计语言

C 语言是理想的结构化程序设计语言, 符合现代程序风格的要求。

C 语言提供了编制结构化程序的控制语句。按照结构化方法规定: 一个程序只能由三种基本控制结构或它们的派生结构组成。这三种结构是: 顺序结构、选择结构、循环结构。C 语言为这三种结构提供了具有结构化的控制语句 (如 if—else 语句、while 语句、do—while 语句、switch 语句、for 语句)。

3. C 语言是模块化的程序设计语言

用函数作为程序的模块单位, 便于实现程序的模块化。当计算机在处理较大的复杂任务时所编写的应用程序经常由上万条语句所组成, 需要由许多人共同来完成。这时常常把复杂的大任务分解为若干个子任务, 每个子任务又分成很多个小子任务, 每个小子任务只完成一项简单的功能。在程序设计时, 用一个个小模块来实现这些功能。程序设计人员分别完成一个或多个小模块。我们称这样的程序设计方法为“模块化”的方法, 由一个个功能模块构成的程序结构为模块化结构。

4. C 语言具有较好的移植性和生成较短的代码

用 C 语言写的程序可移植性好 (与汇编语言比), 基本上不做修改就能用于各种型号的计算机和各种操作系统。C 语言程序生成目标代码质量高, 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%。

5. C 语言具有类型丰富和使用灵活的运算符

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、联合类型等。能用来实现各种复杂的数据结构 (如链表、树、栈等) 的运算, 尤其是指针类型数据, 使用起来更为灵活、多样。

C 的运算符包含的范围很广泛, 共有 34 种运算符 (见附录 D)。C 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