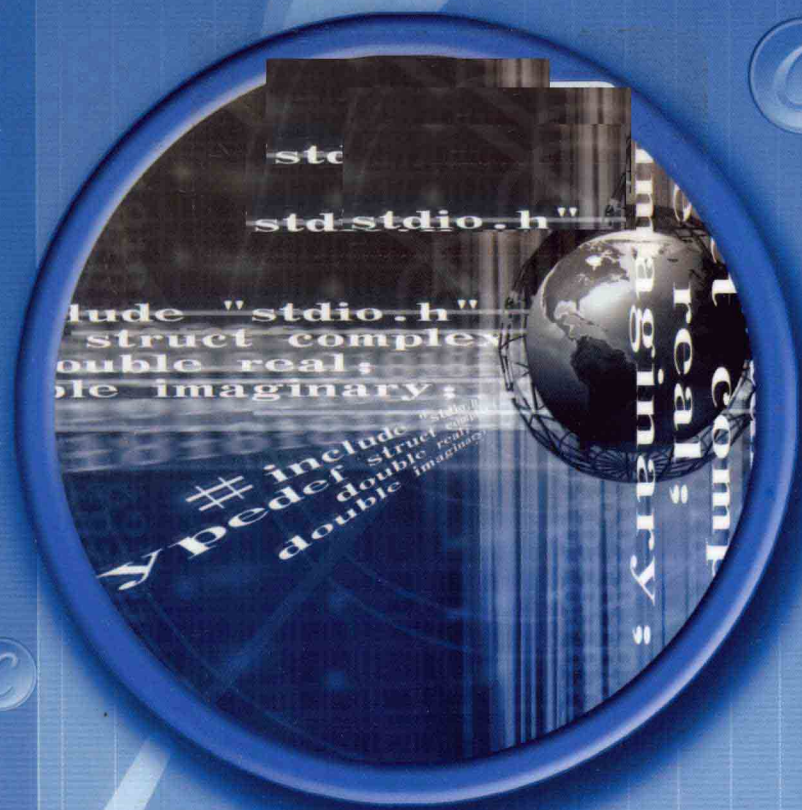


C语言程序设计

上机指导与习题集



高维春 主 编

贺敬凯 汤晓乐 张竞丹 副主编



人民邮电出版社
POSTS & TELECOM PRESS

世纪英才高等职业教育课改系列规划教材（计算机类）

C 语言程序设计上机指导与习题集

高维春 主 编

贺敬凯 汤晓乐 张竞丹 副主编

人 民 邮 电 出 版 社

北 京

图书在版编目 (CIP) 数据

C语言程序设计上机指导与习题集 / 高维春主编. --
北京: 人民邮电出版社, 2010.5

(世纪英才高等职业教育课改系列规划教材. 计算机类)

ISBN 978-7-115-22385-2

I. ①C… II. ①高… III. ①

C语言—程序设计—高等学校: 技术学校—教学参考资料

IV. ①TP312

中国版本图书馆CIP数据核字(2010)第036732号

内 容 提 要

本书是与《C语言程序设计项目教程》配套的辅导教材, 包括上机指导、习题解答和习题集3篇内容。其中, 上机指导介绍了 Visual C++ 6.0 系统使用环境及在该系统上运行 C 语言程序的方法, 根据教材中的知识点精心安排了 9 个实验; 习题解答是对《C语言程序设计项目教程》中全部习题的解答; 为了进一步理解教材中的知识点, 除了各章的习题外, 在习题集中又增设了 200 道习题, 包括 150 道单项选择题和 50 道程序设计题。相关教学资料可从人民邮电出版社网站下载。

本书可作为高职高专院校 C 语言程序设计课程的辅导教材, 也可供广大 C 语言程序设计爱好者练习使用。

世纪英才高等职业教育课改系列规划教材 (计算机类)

C 语言程序设计上机指导与习题集

◆ 主 编 高维春

副 主 编 贺敬凯 汤晓乐 张竞丹

责任编辑 丁金炎

执行编辑 洪 婕

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号

邮编 100061 电子函件 315@ptpress.com.cn

网址 <http://www.ptpress.com.cn>

北京昌平百善印刷厂印刷

◆ 开本: 787×1092 1/16

印张: 10

字数: 217 千字

2010 年 5 月第 1 版

印数: 1—3 500 册

2010 年 5 月北京第 1 次印刷

ISBN 978-7-115-22385-2

定价: 19.00 元

读者服务热线: (010)67129264 印装质量热线: (010)67129223

反盗版热线: (010)67171154

前言 Foreword

Foreword

本书是《C 语言程序设计项目教程》教材的配套用书。学生使用教材《C 语言程序设计项目教程》学习 C 语言程序设计的相关内容后,可通过本书强化、融会贯通所学知识,进一步巩固和提高编程能力、动手能力、技术应用能力,达到理论与实践互相渗透、有机结合的目的。

本书有以下特色。

(1) 本书作为辅助教材,包括 C 语言程序设计上机指导、习题解答和习题集 3 篇。本书第一篇为上机指导,随教学进度精心安排了 9 个实验。每个实验都有具体的要求,安排了至少一道程序分析题,学生上机前先要分析结果;第二篇习题解答中列出了教材中各任务的习题及答案;为了进一步理解教材中的各知识点,第三篇习题集中又增设了 200 道习题。

(2) 注重程序设计,突出算法分析。通过问题分析,将实际问题的解决同 C 语言程序设计的知识和语法有机结合,符合高等职业院校教育的培养目标。

(3) 例题量大。为了进一步理解教材中的各知识点,在习题集中增设了 200 道习题,包括 150 道单项选择题,50 道程序设计题。

本书由高维春统稿。由河南省职业技术教育教研室汤晓乐编写第一篇中实验三、实验四、实验五、实验六、实验七、实验九,第二篇中任务六习题、任务七习题、任务八习题;由深圳信息职业技术学院张竞丹编写第一篇中实验一、实验二、实验八;由深圳信息职业技术学院贺敬凯编写第二篇中开篇导读习题;由深圳信息职业技术学院高维春编写第二篇中任务一习题、任务二习题、任务三习题、任务四习题、任务五习题、第三部分习题集。由于编者水平所限,书中不足甚至错误之处在所难免,敬请读者多提宝贵意见。

编者

第一篇 C 语言程序设计上机指导	1
实验一 C 语言程序的基础知识	1
实验二 顺序结构	3
实验三 分支结构	5
实验四 循环结构	7
实验五 数组	11
实验六 函数	17
实验七 结构体	20
实验八 指针	23
实验九 文件	28
第二篇 习题解答	32
开篇导读 C 语言基础知识习题	32
任务一 学生成绩管理系统界面设计习题	35
任务二 学生成绩管理系统主菜单功能实现习题	38
任务三 学生成绩管理系统主菜单重复选择的实现习题	42
任务四 用数组实现学生成绩管理系统习题	48
任务五 用函数改善学生成绩管理系统习题	53
任务六 用结构体优化学生成绩管理系统习题	59
任务七 用指针实现查询、修改、添加、删除学生成绩习题	65
任务八 用文件完善学生成绩管理系统习题	73
第三篇 习题集	78
第一部分 单项选择题	78
第二部分 编程题	117
参考文献	154

第一篇 C 语言程序设计上机指导

实验一 C 语言程序的基础知识

实验目的

- (1) 掌握常量、符号常量的应用。
- (2) 掌握变量的定义、初始化的方法。
- (3) 掌握算术运算符及其优先级。

实验内容

1. 在屏幕上输出 “This is a C program!”。

编程分析:

- (1) C 程序由函数构成, 但有且只有一个主函数 main();
- (2) 函数=函数名 (参数) { };
- (3) printf 输出函数, 将程序运行结果显示到输出设备上 (显示器);
- (4) '\n' 是一种转义字符, 表示换行;
- (5) 一条语句结束必须用 “;”。

程序代码:

```
#include <stdio.h>          /* printf 函数所在头文件 */
void main( )                /* 主函数 */
{
    printf("This is a C program!\n"); /* 输出一句话 */
}
```

运行结果:

This is a C program!

2. 已知圆的半径 $r=1.5$, 求圆的周长、面积和球的体积。

编程分析:

- (1) 确定题目中各个变量的数据类型, 并为圆的半径赋值 $r=1.5$;
- (2) 分析求解问题的方法。

$$c = 2\pi r$$

$$s = \pi r^2$$

$$v = \frac{4}{3} \pi r^3$$

程序代码:

```
#include <stdio.h> /* printf 函数所在头文件 */
void main()
{
    double r=1.5,c,s,v;
    double PI=3.1415926;
    c=2*PI*r;
    s=PI*r*r;
    v=4/3*PI*r*r*r;
    printf("c=%fns=%f\nv=%f\n",c,s,v);
}
```

运行结果:

```
c=9.424778
s=7.068583
v=10.602875
```

思考:

(1) 为什么 v 的计算结果 (10.602875) 与手工计算结果 (14.137167) 不同? 该如何解决?

(2) 变量 PI 的值在整个程序中未发生改变, 如何声明为符号常量?

(3) 式 $s=PI*r*r$ 和 $v=4/3*PI*r*r*r$ 中 r 的连乘是否可用数学函数表述?

3. 有以下程序代码:

```
#include <stdio.h> /* printf 函数所在头文件 */
void main()
{
    int i=40;
    printf("%d ",i++);
    printf("%d ",++i);
    printf("%d ",i--);
    printf("%d ",--i);
}
```

运行结果:

```
40 42 42 40
```

实验二 顺序结构

实验目的

- (1) 掌握算法的两种描述方法——传统流程图和 NS 流程图的画法。
- (2) 掌握赋值语句和赋值表达式的相关知识。
- (3) 掌握字符数据的输入输出——getchar 函数和 putchar 函数。
- (4) 掌握简单的格式输入与输出——scanf 函数和 printf 函数。

实验内容

1. 编写程序输入三个数，求它们的平均值并输出，用浮点数据处理。

编程分析：

- (1) 以 scanf 函数读入三个单精度实型数，并用 printf 函数输出结果；
- (2) 按题目要求计算。

程序代码：

```
#include <stdio.h> /* printf scanf 函数所在头文件 */
void main()
{
    float f1,f2,f3,f_mean;
    printf("输入三个浮点数:");
    scanf("%f,%f,%f",&f1,&f2,&f3);
    f_mean=(f1+f2+f3)/3;
    printf("输出结果:%f",f_mean);
}
```

运行结果：

输入三个浮点数: 4.0,5.3,6.7✓

输出结果: 5.333333

2. 用 getchar 函数读入两个字符给 c1、c2，然后分别用 putchar 函数和 printf 函数输出这两个字符。

编程分析：

- (1) 字符输入函数 getchar 的一般形式：getchar();
- (2) 字符输出函数 putchar 的一般形式：putchar (字符变量);
- (3) 格式输出函数 printf 的一般形式：printf (“格式控制字符串”，输出表列)。

程序代码：

```
#include <stdio.h>                                /* printf.getchar.putchar 函数所在头文件 */
void main()
{
    char c1,c2;
    printf("输入两个任意字符:");
    c1=getchar();
    c2=getchar();
    printf("输出结果:");
    putchar(c1);
    printf("%c",c2);
}
```

运行结果:

输入两个任意字符: s3✓

输出结果: s3

思考:

比较用 printf 和 putchar 函数输出字符的特点。

3. 求 $ax^2+bx+c=0$ 方程的根。a、b、c 由键盘输入, 设 $b^2-4ac>0$ 。

编程分析:

(1) 由于 $b^2-4ac>0$, 方程 $ax^2+bx+c=0$ 有两个不等的实根, 分别为

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \quad x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

如已知 a,b,c 的值, 可以求 x_1, x_2 的值;

(2) 通过 scanf 函数输入 a, b, c 的值。调用数学函数 sqrt 求一个数开根号, 调用数学函数 pow 求一个数的平方, 必须包含 math.h 头文件。计算 x_1, x_2 的值, 并以 printf 函数输出结果。

程序代码:

```
#include <stdio.h>                                /* printf scanf 函数所在头文件 */
#include <math.h>
void main()
{
    double a,b,c,x1,x2;
    printf("输入 a,b,c 的值:");
    scanf("%lf %lf %lf",&a,&b,&c);
    x1=(-b+sqrt(pow(b,2)-4*a*c))/(2*a);
    x2=(-b-sqrt(pow(b,2)-4*a*c))/(2*a);
    printf("输出结果:x1=%f x2=%f",x1,x2);
}
```

运行结果:

输入 a, b, c 的值: 4 4 1✓

输出结果: $x_1 = -0.500000$ $x_2 = -0.500000$

实验三 分支结构

实验目的

- (1) 学会正确使用逻辑运算符和逻辑表达式。
- (2) 熟练掌握 if 语句和 switch 语句的使用。

实验内容

1. 输入一个数, 判别能否被 3 和 5 整除, 且不能被 7 整除, 如果符合条件, 输出“YES”, 否则输出“NO”。

编程分析:

- (1) 使用 scanf 语句得到键盘输入;
- (2) 这里有两个显然的分支, 但是要注意分支中条件的书写。

程序代码:

```
#include <stdio.h>
void main()
{
    int x;
    printf("请输入一个整数:");
    scanf("%d", &x);
    if(x % 3 == 0 && x % 5 == 0 && x % 7 != 0)    printf("YES\n");
    else    printf("NO\n");
}
```

运行结果:

请输入一个整数: 15✓

YES

2. 有如下分段函数, 编写程序, 输入 x 值, 输出 y 值。

$$y = \begin{cases} 10 - x & (x < -10) \\ x^2 & (-10 \leq x \leq 10) \\ x^3 + 2x & (x > 10) \end{cases}$$

编程分析:

对 x 进行分段处理, 依据其值的不同, 有 3 个分支, 要注意表达式的书写。

程序代码:

```
#include <stdio.h>
void main()
{
    float x, y;
    printf("输入 x:");
    scanf("%f", &x);
    if(x < -10) /* 当 x < -10 时, 求对应 y 值 */
        y = 10 - x;
    else if(x <= 10) /* 当 -10 ≤ x < 10 时, 求对应 y 值 */
        y = x * x;
    else /* 当 x > 10 时, 求对应 y 值 */
        y = x * x * x + 2 * x;
    printf("x = %f, y = %f\n", x, y);
}
```

运行结果:

输入 x: 5.5 ✓

x = 5.500000, y = 30.250000

3. 输入一个不多于 5 位的正整数, 反序输出, 如输入 1234, 输出 4321。

编程分析:

(1) 要反序输出, 就得知道其中每位上的数字是几, 如 1234 可以这样分解出每一个位的数字来:

位	值	计 算 式	位	值	计 算 式
千位	1	1234 / 1000	十位	3	1234 / 10 % 10
百位	2	1234 / 100 % 10	个位	4	1234 % 10

(2) 输出时, 并不知道有几位要输出, 所以还得确定出整数有 w 位, 可以通过将其与 10^w 进行比较;

(3) 最后可以根据 w 的值, 进行输出。

程序代码:

```
#include <stdio.h>
void main()
{
    int n;
    /* 下面定义的各变量, 分别代表个位, 十位, 百位, 千位, 万位, 以及位数 */
    int g, s, b, q, w, width;
    printf("请输入一个整数(0 ~ 99999):");
    scanf("%d", &n);
```

```

/* 判断变量 n 的位数 */
if(n > 9999)      width = 5;
else if(n > 999)   width = 4;
else if(n > 99)    width = 3;
else if(n > 9)     width = 2;
else              width = 1;
/* 求出 n 在各位上的值 */
w = n / 10000;
q = n / 1000 % 10;
b = n / 100 % 10;
s = n / 10 % 10;
g = n % 10;
/* 根据变量 n 的位数, 并根据位数做出相应的输出 */
switch(width)
{
case 1:  printf("%d\n", g);          break;
case 2:  printf("%d%d\n", g, s);     break;
case 3:  printf("%d%d%d\n", g, s, b); break;
case 4:  printf("%d%d%d%d\n", g, s, b, q); break;
case 5:  printf("%d%d%d%d%d\n", g, s, b, q, w); break;
default: printf("输入错误.\n");
}
}

```

运行结果:

请输入一个整数(0 ~ 99999): 12034 ✓
43021

思考:

- (1) 本例中没有考虑到输入的数可能为负数的情况, 请补充完整。
- (2) 有没有其他得到各位上数字的方法? 请加以总结。

实验四 循环结构

实验目的

- (1) 掌握用 while 语句、do-while 语句和 for 语句实现循环的方法。
- (2) 掌握良好的代码书写格式。

实验内容

1. 用下面的公式求 π 的近似值, 直到最后一项的绝对值小于 10^{-6} :

$$\frac{\pi}{4} \approx 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \dots$$

编程分析:

(1) 此公式的和值乘以 4 即可得出所要求的值;

(2) 每项的分子均是 1, 分母累加 2, 最后一项的值要小于 10^{-6} ($1e-6$), 但是因为最后一项可能为正也可能为负, 故需要将其绝对值与 10^{-6} 比较, 可以使用求定义在 math.h 头文件中的求实型绝对值的函数 fabs;

(3) 通过观察, 可以看出正负号交替出现, 所以可以考虑加上一个初值为 1 的符号变量 sgn, 每到下一项时取 sgn 的相反数赋给自身。

程序代码:

```
#include <stdio.h>
#include <math.h>
void main()
{
    int n;                /* sgn 代表符号位, 即分子, n 代表分母的值 */
    float sgn, pi, t;      /* pi 代表  $\pi$  的值, t 代表每一项的值 */
    sgn = 1;    n = 1;    pi = 0;    t = 1;    /* 给变量赋初始值 */
    for(;;)              /* 等同于 while(1) */
    {
        if(fabs(t) < 1e-6) break;
        pi = pi + t;
        sgn = -sgn;
        n += 2;
        t = sgn / n;
    }
    pi = pi * 4;
    printf("PI: %f\n", pi);
}
```

运行结果:

PI: 3.141594

思考:

(1) 这 3 种循环的功能一致, 不过有时要考虑循环的执行顺序。如在 for 循环中, 如果把语句 “n+=2”; 放在第二个分号后面, 结果又将如何?

(2) 如何用 while、do-while 语句来改写此程序?

2. 输出 Fibonacci 数列中的前 10 个偶数, 每 5 个数输出一行。

编程分析:

- (1) 要先得到 Fibonacci 数列的值;
- (2) 判断每一项值是否为偶数, 如果是则输出;
- (3) 需要输出 10 个, 用 count 进行统计, 且每 5 个数后输出一个换行符;
- (4) 程序在输完 10 个偶数后结束, 在这之前一直运行, 可以用 while(1) 来表示, 当到达结束点时, 跳出程序, 以避免死循环。

程序代码:

```
#include <stdio.h>
void main()
{
    int count = 0, a = 1, b = 1, c;    /* a、b、c 为连续的 3 项, count 计数 */
    while(1)
    {
        c = a + b;
        if(c % 2 == 0)
        {
            count++;
            if(count > 10) break;
            if((count-1) % 5 == 0) putchar('\n');
            printf("%8d", c);
        }
        a = b; b = c;
    }
    putchar('\n');
}
```

运行结果:

```
2      8      34     144     610
2584   10946   46368   196418  832040
```

3. 以下面的样式显示九九乘法表。

九九乘法表

x \ y	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9
2	2	4	6	8	10	12	14	16	18
3	3	6	9	12	15	18	21	24	27
4	4	8	12	16	20	24	28	32	36
5	5	10	15	20	25	30	35	40	45
6	6	12	18	24	30	36	42	48	54
7	7	14	21	28	35	42	49	56	63
8	8	16	24	32	40	48	56	64	72
9	9	18	27	36	45	54	63	72	81

编程分析:

- (1) 九九乘法表中需要二重循环;
 (2) 注意前两行和前两列的输入方法: 每一列对应 4 个字符的宽度, 在第二行第 4 个字符处是个 “+” 号, “九九乘法表” 几个字处于居中的位置, 在前面有 15 个空格字符。

程序代码:

```
#include <stdio.h>
void main()
{
    int digit, n, i, j;
    printf("\n          九九乘法表\n\n");      /* "九"前面是 15 个空格符 */
    printf(" X |");                          /* "X"前后各是一个空格符 */
    for(digit = 1; digit < 10; digit++)      /* 输出第一行数字字符 */
        printf("%4d", digit);
    printf("\n");
    for(n = 1; n <= 45; n++)                  /* 输出第二行字符 */
    {
        if(n == 4)    printf("+");
        else          printf("-");
    }
    printf("\n");
    for(i = 1; i < 10; i++)                    /* 输出第三行到末行的字符 */
    {
        printf("%2d ", i);                    /* 输出每行的首列数字字符 */
        printf("|");
        for(j = 1; j < 10; j++)                /* 输出每列的字符, 宽度为 4 */
            printf("%4d", i*j);
        printf("\n");
    }
}
```

4. 输入一行字符, 分别统计出其中英文字母、空格、数字和其他字符的个数。

编程分析:

- (1) 输入字符可以用 getchar 语句来得到一个字符;
 (2) 将得到的字符进行分类判断, 最后输出。

程序代码:

```
#include <stdio.h>
void main()
```

```

{ char c;
  int letter = 0, space = 0, digit = 0, other = 0;
  printf("请输入一行字符, 按 Enter 键结束:\n");
  while((c = getchar()) != '\n')      /* 读取当前字符, 如不为回车符则进行统计 */
  {
    if(c >= 'a' && c <= 'z' || c >= 'A' && c <= 'Z')    letter++;
    else if(c == ' ')                                   space++;
    else if(c >= '0' && c <= '9')                       digit++;
    else                                                other++;
  }
  printf("字母:%d, 空格:%d, 数字:%d, 其他:%d\n", letter, space, digit, other);
}

```

运行结果:

请输入一行字符, 按 Enter 键结束:

I call Li old big. Toy ear 25. One see you, my heart flower angry open! ✓

字母: 50, 空格: 14, 数字: 2, 其他: 4

思考:

注意循环语句中的 `getchar` 语句, 它相当于哪几条语句, 如果写成下面形式, 可不可以, 为什么?

```
while(c = getchar() != '\n')
```

实验五 数 组

实验目的

- (1) 掌握一维数组和二维数组定义、赋值和输入输出的方法。
- (2) 掌握字符数组和字符串函数的使用。
- (3) 掌握与数组有关的算法。

实验内容

1. 将一个数组中的值按逆序重新存放。例如, 原来顺序为 1, 2, 3, 4, 5, 要求改为 5, 4, 3, 2, 1。

编程分析:

可以先将第一个与最后一个交换, 然后第二个与倒数第二个交换, 这样依此类推, 直到中间一个数组元素为止。

程序代码:

```
#include <stdio.h>
void main()
{
    int a[5] = {1, 2, 3, 4, 5}, i, t;
    printf("原始数组:");
    for (i=0; i<5; i++)          /* 输出原始数组 */
        printf("%d ", a[i]);
    /* 对数组前一半元素, 与后一半相应位置上的元素调换 */
    for (i=0; i<3; i++)
    {   t = a[i];   a[i] = a[4-i];   a[4-i] = t;   }
    printf("\n 逆序数组:");
    for (i=0; i<5; i++)          /* 输出逆序后的数组 */
        printf("%d ", a[i]);
    printf("\n");
}
```

运行结果:

原始数组: 1 2 3 4 5

逆序数组: 5 4 3 2 1

2. 有一个已排好序的数组, 输入一个数, 要求按原排序规律将它插入到数组中。

编程分析:

此例与插入排序很相似。如数组 $a[N]$, 前面 $N-1$ 个已经有序, 当插入一个数 m 时, 需要将它与有序的最后一个元素 $a[N-2]$ 比较大小, 分成两种情况:

(1) 如果 $m \geq a[N-2]$, 直接将待插入的数放到最后一个, 即 $a[N-1] = m$;

(2) 否则, $a[N-2]$ 向后移动一个位置, 到 $a[N-1]$ 处, 将它自身位置空出, 然后再将 m 与 $a[N-3]$ 比较, 如果 $m \geq a[N-3]$, 直接将它存入到 $a[N-2]$ 里, 否则, 再向前比较, 直到找到合适的位置, 将其存入进去。

程序代码:

```
#include <stdio.h>
#define N 6
void main()
{
    int a[N], t, i, j, m;
    printf("输入 5 个数:");          /* 数组初始化时要预留一位置 */
    for(i=0; i<5; i++)               scanf("%d", &a[i]);
    printf("输入要插入的数:");
```