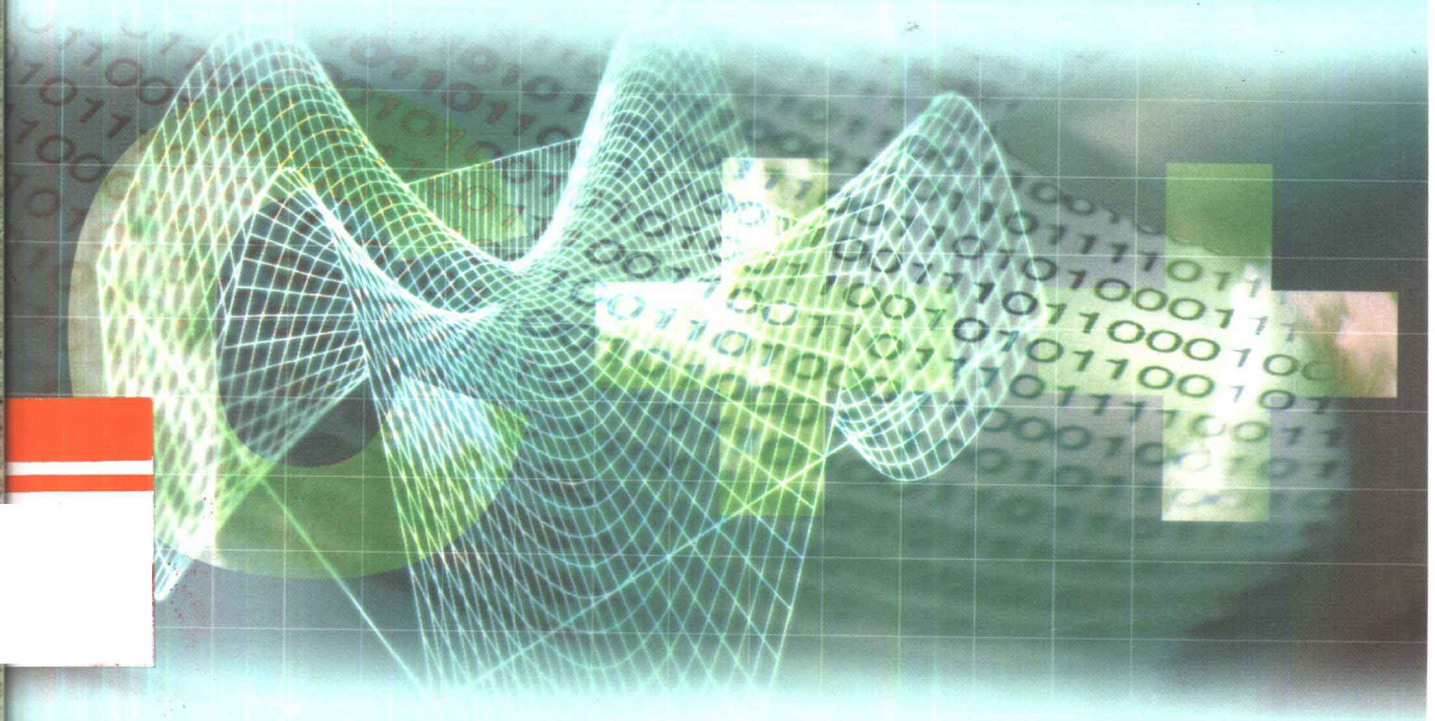
高等学校教材

C/C++程序设计题解与实验教程

谭浩强 张基温 主编



高等教育出版社

高等学校教材

C/C++程序设计题解与实验教程

谭浩强　张基温　主编

高等教育出版社

图书在版编目(CIP)数据

C/C ++程序设计题解与实验教程/谭浩强,张基温主编.—北京:高等教育出版社,2001.1
ISBN 7-04-009202-6

Ⅰ.C… Ⅱ.①谭… ②张… Ⅲ.C语言-程序设计-教材 Ⅳ.TP312

中国版本图书馆CIP数据核字(2000)第77107号

C/C ++程序设计题解与实验教程
谭浩强 张基温 主编

出版发行 高等教育出版社
社 址 北京市东城区沙滩后街55号 邮政编码 100009
电 话 010-64054588 传 真 010-64014048
网 址 http://www.hep.edu.cn
http://www.hep.com.cn

经 销 新华书店北京发行所
印 刷 北京人卫印刷厂

开 本 787×1092 1/16 版 次 2001年1月第1版
印 张 15.75 印 次 2001年1月第1次印刷
字 数 380 000 定 价 14.50元

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换。

内 容 提 要

本书是谭浩强、张基温编著的《C/C++程序设计教程》一书的配套辅导教材。全书共包括4大部分，第一部分是根据教程中的习题而编写的参考答案以及相关知识的讲解；第二部分是C/C++程序设计实验，它既可以供学生上机进行自学使用，也可以供教师作为教学的实验辅导题；第三部分对C/C++上机环境进行了全面的介绍，可使读者对语言的环境有一个形象的认识；第四部分详细地列出了C/C++中的出错信息，是读者调试程序的有利工具。

本书例题、习题丰富，讲解通俗易懂；实验面向测试、调试，具有启发性。可作为高等学校C/C++程序设计课程的教学辅导书，也可供应用开发人员学习、参考。

前　言

本书是与《C/C ++ 程序设计教程》(谭浩强、张基温编著,高等教育出版社出版)一书配套使用的辅导教材。

书中共包括4个方面的内容:

(1) 第一部分是配套教材中各章大部分习题的参考答案(1～6章由董惠丽编写,第7章由张基温编写),其中部分能够直接从教材中或运行程序即可得到的习题答案,没有给出参考解答。此外,有几章中的少量习题也没有提供参考解答,留给读者独立完成,进行自我检测。

书中的一些编程题只给出了程序,而没有作解释,并且不给出算法或流程图,目的是想给读者留更多的思考空间。

(2)第二部分是学习本课程应进行的实验(由张基温编写)。考虑到教学的要求,共安排了12个实验。在每个实验中除了提出实验内容外,还对程序的测试和调试提出要求,并且列出了需要进一步深入分析和思考的问题。

(3)第三部分介绍 C/C ++ 程序开发环境,即运行 C/C ++ 程序的环境(由张基温、贾中宁编写)。本部分中比较详细地介绍了 Turbo C 的使用方法,同时简要地介绍了 Borland C ++ 的集成环境。

(4)第四部分介绍 C/C ++ 的出错信息(由张基温、张秋菊收集),可供读者上机调试程序时参考。

全书由谭浩强教授修改定稿。

本书内容具有通用性,因此不仅可以作为配套教材的参考书,也可以作为学习 C/C ++ 程序设计课程的参考读物。

书中若有缺点或错误,敬请读者提出宝贵意见。

作　者

2000年5月

目　录

第一部分　C/C++程序设计习题参考答案

第二部分　C语言实验

第三部分　C/C++程序开发环境

第四部分　C/C++出错信息

第一部分

C/C ++ 程序设计
习题参考答案

第1章　C语言程序设计初步

1.1　试举例说明日常生活中的“程序”。(解略)

1.2　如何理解面向过程的程序设计方法和面向对象的程序设计方法。(解略)

1.3　用C语言进行程序设计,要经过哪些步骤?(解略)

1.4　试在你使用的C语言程序开发环境中调试下面的程序,并指出程序中每个语句的功能。

(1)

```
# include <stdio.h> /*  嵌入头文件“stdio.h”,该文件含有
                        使用函数 printf()必须的信息 */
main()
{
    printf("I have a computer."); /* 输出字符串“I have a computer.” */
}
```

解:

功能:在显示器上原样输出字符串“I have a computer.”

运行结果:

```
I have a computer.
```

(2)

```
# include <stdio.h>
main()
{
    int x,y,z;              /* 声明 x,y,z 为整型变量 */
    x=5;                    /* 将 5 送给变量 x */
    y=3;                    /* 将 3 送 给变量 y */
    z=x+y;                  /* 将 x 和 y 之和送给变量 z */
    printf("z=%d", z);      /* 输出 z 变量的值 */
}
```

解:

功能:计算变量x和y的和。

运行结果:

```
z=8
```

(3)

```
# include  <stdio.h>
main()
{
    int x=5,y=3,z;                  /* 声明变量 x,y,z 为整型变量,同时初始化 x,y */
```

```
    z = x;                                  /* 将 x 的值送给变量 z,使变量 x 和 z 的值同时为 5 */
    printf(" x = %d,y = %d,z = %d \n ",x,y,z);  /* 输出变量 x,y,z 的值 */
    x = y;                                  /* 将 y 的值送给 x,使 x 和 y 的值同时为 3 */
    printf(" x = %d,y = %d,z = %d \n ",x,y,z);
    y = z;                                  /* 将 z 的值送给 y,使 z 和 y 的值同时为 5 */
    printf(" x = %d,y = %d,z = %d \n ",x,y,z);  /* 输出 x,y,z 的值 */
}
```

解:

功能:交换整型变量 x 和 y 的值。

运行结果:

```
x = 5,y = 3,z = 5
x = 3,y = 3,z = 5
x = 3,y = 5,z = 5
```

(4)

```
# include < stdio.h >
main()
{
    float a,b,max;                  /* 声明 a,b,max 为 3 个实型变量 */
    printf(" \nPlease input a & b:");  /* 提示用户从键盘键入 a 和 b 的值 */
    scanf(" %f %f ",&a,&b);         /* 调用函数 scanf()输入 a 和 b 的值 */
    if(a > b)                       /* 比较变量 a 和 b 的大小 */
        max = a;                    /* 如果 a > b,把 a 中的数送给变量 max */
    else                            /* 否则,即如果 a < = b */
        max = b;                    /* 把 b 中的数送给变量 max */
    printf(" \nmax = %f ",max);
}
```

解:

功能:输出 a 和 b 中较大者。

运行结果:

```
Please input a & b:3 5↙①  (注意:键入的数据之间用空格隔开)
max = 5
```

1.5　编写下面的程序,并请上机编辑、编译、链接、调试。

(1)编写 C 语言程序,在屏幕上显示一串字符。

I am a student.

解:

程序如下:

```
# include   < stdio.h >
main()
{
    printf("I am a student.");
```

① 带下划线内容表示此部分由键盘输入。

```
}
```

(2)编写一个交换字符型变量值的 C 语言程序。

解:

程序如下:

```
# include   < stdio.h >
main()
{
    char c1,c2,c3;
    c1 = 'a';c2 = 'b';
    printf(" \n交换前:");
    printf(" \nc1 = \' %c \',c2 = \' %c \'",c1,c2);
    c3 = c1;c1 = c2;c2 = c3;
    printf(" \n交换后:");
    printf(" \nc1 = \' %c \',c2 = \' %c \'",c1,c2);
}
```

运行结果:

```
交换前:
c1 = 'a',c2 = 'b'
交换后:
c1 = 'b',c2 = 'a'
```

(3)编写一个 C 语言程序,从两个整数中选出小者。

解:

程序如下:

```
# include   < stdio.h >
main()
{
    int x,y,min;
    printf(" \nPlease input x & y:");
    scanf(" %d,%d",&x,&y);
    if (x < y)
      min = x;
    else
       min = y;
    printf("min = %d",min);
}
```

运行结果:

```
Please input x & y:5,3↙
min = 3
```

1.6　测试你使用的 C 语言系统中各数值数据的存储长度。

解:

程序如下:

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

```
main()
{
    printf(" char   :  %dbytes \n ",sizeof(char));
    printf(" short  :  %dbytes \n ",sizeof(short));
    printf(" int    :  %dbytes \n ",sizeof(int));
    printf(" long   :  %dbytes \n ",sizeof(long));
    printf(" float  :  %dbytes \n ",sizeof(float));
    printf(" double :  %dbytes \n ",sizeof(double));
}
```

运行结果:

```
char   :  1bytes
short  :  2bytes
int    :  2bytes
long   :  4bytes
float  :  4bytes
double :  8bytes
```

1.7　将下列代数式写成 C 语言表达式。

(1) $y = ax^2 + bx + c$　(2) $z = (x + y)^3$　(3) $z = (a + b) \div (c - d)$

(4) $x = a - \frac{ab}{c + d}$　(5) $x = \frac{a^b}{c - d}$　(6) $z = \frac{1}{2}(xy + \frac{2}{x^y})$

解:

数学表达式	C 语言表达式
(1) $y = ax^2 + bx + c$	y = a * pow(x,2) + b * x + c
(2) $z = (x + y)^3$	z = pow(x + y,3)
(3) $z = (a + b) \div (c - d)$	z = (a + b)/(c - d)
(4) $x = a - \frac{ab}{c + d}$	x = a - (a * b/(c + d))
(5) $x = \frac{a^b}{c - d}$	x = pow(a,b)/(c - d)
(6) $z = \frac{1}{2}(xy + \frac{2}{x^y})$	z = (1/2) * (x * y + 2/pow(x,y))

说明:pow(x,y)是 C 语言中的一个库函数,用来计算 x^y。

1.8　先阅读下面程序,指出输出结果,然后上机验证、比较,分析自己的判断与机器执行结果不同的原因。

(1)

```
# include   < stdio.h >
main()
{
    int a,b;
    long c,d;
    unsigned e,f;
    a = 32767;          b = a + 1;
    c = 2147483647;     d = c + 1;
    e = 65535;          f = e + 1;
```

```
    printf("\nint:a=%d,b=%d",a,b);
    printf("\nlong:c=%ld,d=%ld",c,d);
    printf("\nunsigned:e=%u,f=%u",e,f);
}
```

解：

运行结果：

```
int:a=32767,b=-32768
long:c=2147483647,d=-2147483648
unsigned:e=65535,f=0
```

程序说明：对于存储空间为两个字节的 int 型变量，它的取值范围为 -32 768 ~ 32 767。变量 a 是一个 int 型变量，其值 32 767 在内存存储时的二进制数为 0111111111111111，即除最高位外，其余各位都是 1，若 0111111111111111 + 1（十进制的 32 768），结果就是 1000000000000000，而该数是十进制数 -32 768 的补码形式，因此 b = -32 768。long 型变量的情况与 int 型类似。unsigned 型是不带符号的数，程序中没有声明其类型时默认为 unsigned int 型，两个字节的不带符号的 int 型能表达的最大的数是 65 535（二进制为 16 个 1），若再加 1，进位后只留低 16 位，即 0，所以 f = 0。

(2)

```
#include <stdio.h>
main()
{
    int x=22;
    printf("\nx1=%d",x);
    {
        int x=333;        /* x为局部变量,作用域只在定义该变量的语句括号内 */
        printf("\nx2=%d",x);
    }
    printf("\nx1=%d",x);
    {
        int x=55555;      /* 超出int型所能取值范围 */
        printf("\nx3=%d",x);
    }
    printf("\nx1=%d",x);
}
```

解：

运行结果：

```
x1=22
x2=333
x1=22
x3=-9981
x1=22
```

程序说明：变量定义“int x = 55555;”给 x 赋的值 55555 超出了 int 型所能表示的范围，因此

在赋值时,有一个隐含的类型转换,只将数据 55555 的低 2 个字节送给变量 x。这样变量 x 里存储的是二进制数 1101100100000011,由于最高位是 1(最高位是符号位),系统将把该数作为负数处理(在系统里负数以补码的形式出现,另外负数的表示形式还有原码和反码。负数的原码其符号位为 1,反码除符号位外,其余各位按位取反。求负数的补码的方法是:将该数的绝对值的二进制形式按位取反再加 1)。因此,除最高位外,其余位先减 1 再按位取反变成该数的原码,即是十进制数 - 9981,所以有 x3 = - 9981。

(3)

```
main()
{
    char ch1 = 65,ch2 = - 65;
    printf(" \n(char)ch1 = %c \t(char)ch2 = %c" ,ch1,ch2);
    printf(" \n(int)ch1 = %d \t(int)ch2 = %d" ,ch1,ch2);  /* 十进制 */
    printf(" \n(int)ch1 = %o \t(int)ch2 = %o" ,ch1,ch2);  /* 八进制 */
    printf(" \n(int)ch1 = %x \t(int)ch2 = %x" ,ch1,ch2);  /* 十六进制 */
}
```

解:

运行结果:

```
(char)ch1 = A    (char)ch2 = ┐
(int)ch1 = 65    (int)ch2 = - 65
(int)ch1 = 101   (int)ch2 = 177677
(int)ch1 = 41    (int)ch2 = ffbf
```

程序说明:

①ch2 被赋值 - 65,二进制数为 - 01000001,其补码为 10111111,由于系统用 1 个字节作为扩展后的字符 ASCII 编码(最高位不再当符号位处理),所以 10111111 是一个正数,十进制为 191,是字符"┐"的 ASCII 编码。

②当一个字节的字符型数据,按 int 型的格式输出时,要进行数据类型的隐式的输出转换。在数据的隐式转换中要特别注意以下几种情况:

·负数转换为无符号数:符号位将作为数值的一部分(而不作为符号)处理。

·较长类型转换为较短类型时:丢掉高字节,只留低字节。

·较短类型转换为较长类型时:在增加的字节中,各位的状态与原来较短的数据中的符号位相同,即符号位扩展。

本程序为较短类型转换为较长类型。

ch1 = 65,符号位是 0,转换为 int 型时,增加的 1 个字节中,各位全为 0,所以不论是哪一种进制的数,其值均不变。而 ch2 = - 65,转换为 int 型时,增加的 1 个字节中,各位全为 1,- 65 的二进制补码为 10111111,增加 1 个字节后二进制为 1111111110111111,最高位是 1,当格式符为"%d"时,为带符号数,按负数处理,除最高位作为符号位外,其余各位减 1 再按位取反是 - 65。当格式符为"%o"或"%x"时,按不带符号的数对待,最高位的 1 当作数值计算,不再是符号位,二进制数 1111111110111111 转换为八进制数是 177677,十六进制数是 ffbf。

(4)

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

```
main()
{
    int x,y,z,a,b,c;
    x = 1;y = 2;
    z = ++x + (++y);
    printf("x = %d \n",x);
    printf("y = %d \n",y);
    a = -1;b = -2;
    c = ++a - b++;
    printf("a = %d \n",a);
    printf("b = %d \n",b);
}
```

解：

运行结果：

```
x = 2
y = 3
a = 0
b = -1
```

(5)

```
#include  <stdio.h>
main()
{
    float a = 123456789e5,b;
    b = a + 20;
    printf("%f \n",b);
}
```

解：

运行结果：

```
12345679020032.000000
```

(6)

```
#include  <stdio.h>
main()
{
    float a = 123.456;
    double b = 8765.4567;

    printf("\n(1)%f \n",a);
    printf("(2)%14.3f \n",a);
    printf("(3)%6.4f \n",a);
    printf("(4)%lf \n",b);
    printf("(5)%14.3f \n",b);
    printf("(6)%8.4f \n",b);
```

```
    printf("(7)%.4f \n",b);
}
```

解:

运行结果:

```
(1)123.456001
(2)        123.456
(3)123.4560
(4)8765.456700
(5)        8765.457
(6)8765.4567
(7)8765.4567
```

(7)

```
#define AMT1 a+a+a
#define AMT2 AMT1-AMT1
main()
{
    int a=2;
    printf("%d \n",AMT2);
}
```

解:

运行结果:

```
8
```

程序说明:解题时我们可以用 AMT1 - AMT1 替换语句 printf(" %d \n",AMT2)中的 AMT2,替换后该语句变为 printf("%d \n",AMT1 - AMT1),我们再一次用 a + a + a 去替换 AMT1,最后该语句为:printf("%d \n",a + a + a - a + a + a)。

1.9 有以下的变量定义

```
int *pa=&a;
int **ppa=&pa;
float b=1.23;
float *pb=&b;
float **ppb=&pb;
```

(1)试编写一个小的 C 程序,计算下列表达式的值:

```
*(pa+1),*(pa+2),**ppa,*ppa,*(pb+1),*(pb+2),**ppb,*ppb
```

解:

程序如下:

```
#include <stdio.h>
main()
{
    int a=5;
    int *pa=&a;
    int **ppa=&pa;
    float b=1.23;
```