

**课 程 实 验 报 告**

**课程名称： C语言程序设计实验**

**专业班级： 信安2班**

**学 号： U201614836**

**姓 名： 林晓斌**

**指导教师： 祝 建 华**

**报告日期：**

**计算机科学与技术学院**

目录

[1 表达式和标准输入与输出实验 2](#_Toc484993653)

[1.1 实验目的 2](#_Toc484993654)

[1.2 实验内容 2](#_Toc484993655)

[1.2.1 源程序改错 2](#_Toc484993656)

[1.2.2 源程序修改替换 5](#_Toc484993657)

[1.2.3程序设计 6](#_Toc484993658)

[1.3实验心得 10](#_Toc484993659)

1 表达式和标准输入与输出实验

1.1 实验目的

（1）熟练掌握各种运算符的运算功能，操作数的类型，运算结果的类型及运算过程中的类型转换，重点是C语言特有的运算符，例如位运算符，问号运算符，逗号运算符等；熟记运算符的优先级和结合性。

（2）掌握getchar, putchar, scanf 和printf 函数的用法。

（3）掌握简单C程序（顺序结构程序）的编写方法。

1.2 实验内容

1.2.1 源程序改错

在这个例子程序中存在若干语法和逻辑错误。要求参照1.3和1.4的步骤对下面程序进行调试修改，使之能够正确完成指定任务。

下面给出了一个简单C语言程序例程，用来完成以下工作：

（1）输入华氏温度f，将它转换成摄氏温度c后输出；

（2）输入圆的半径值ｒ，计算并输出圆的面积ｓ；

（3）输入短整数ｋ、ｐ，将ｋ的高字节作为结果的低字节，ｐ的高字节作为结果的高字节，拼成一个新的整数，然后输出；

1. #include<stdio.h>

2 #define PI 3.14159;

3 voidmain( void )

4 {

5 int f ;

6 short p, k ;

7 double c , r , s ;

8 /\* for task 1 \*/

9 printf(“Input Fahrenheit:” ) ;

10 scanf(“%d”, f ) ;

11 c = 5/9\*(f-32) ;

12 printf( “ \n %d (F) = %.2f (C)\n\n ”, f, c ) ;

13 /\* for task 2 \*/

14 printf("input the radius r:");

15 scanf("%f", &r);

16 s = PI \* r \* r;

17 printf("\nThe acreage is %.2f\n\n",&s);

18 /\* for task 3 \*/

19 printf("input hex int k, p :");

20 scanf("%x %x", &k, &p );

21 newint = (p&0xff00)|(k&0xff00)<<8;

22 printf("new int = %x\n\n",newint);

}

**解答**：本实验共有9个错误，修改数为10。

第2行#define定义符号常量指令行末尾不加分号。改为：#define PI 3.14159

第3行void main是非标准语句，需改为int main，同时最后加上return 0.

第10行scanf函数格式错误，应为：scanf("%d",&f);

第11行的5与9都是整型数，相除之后结果为0，故应将数字类型转换，改为：c=5.0/9 \*(f-32)

第15行r为双精度变量，故应改为：scanf(“%lf”,&r);

第17行printf语句不需要加地址符&，故改为：printf("\nThe acreage is %.2f\n\n",s);

第20行应取作16进制运算，故改为：scanf("%hx %hx", &k, &p );

newint变量并没有定义，故应在第6行声明它的定义：short p, k ,newint;

第21行转换式子出错，应为：newint = (p&0xff00)|(k&0xff00)>>8;

以下为经过修改后的源程序。

#include <stdio.h>

#include <stdlib.h>

#define PI 3.14159

int main(void)

{

int f;

short p,k,newint;

double c,r,s;

/\*for task1\*/

printf("Input Fahrenheit:");

scanf("%d",&f);

c=5.0/9\*(f-32);

printf("\n%d(F)=%.2f(C)\n\n",f,c);

/\*for task2\*/

printf("inout the radius r:");

scanf("%lf",&r);

s=PI\*r\*r;

printf("\nThe acreage is %.2f\n\n",s);

/\*for task3\*/

printf("input hex int k,p:");

scanf("%hx%hx",&k,&p);

newint=(p&0xff00)|((k&0xff00)>>8);

printf("newint=%x\n\n",newint);

return 0;

}

对修改后的程序进行一次简单的数据检验，如图1.1所示。

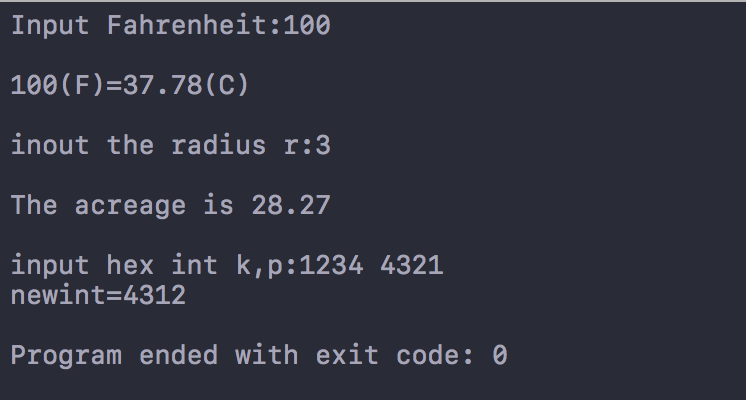


图 1.1 修改后的程序运行结果

1.2.2 源程序修改替换

下面的程序利用常用的中间变量法实现两数交换，请改用不使用第3个变量的方法实现。该程序中t是中间变量，要求将定义语句中的t删除，修改下划线处的语句，使之实现两数对调的操作。

#include<stdio.h>

void main( )

{

int a, b, t;

printf(“Input two integers:”);

scanf(“%d %d”,&a,&b);

t=a ；a=b；b=t；

prinf(“\na=%d,b=%d”,a,b);

}

**解答**：利用按位加运算可以完成，程序如下

#include <stdio.h>

#include <stdlib.h>

int main()

{

int a,b;

printf("Input two integers:");

scanf("%d%d",&a,&b);

a^=b;

b^=a;

a^=b;

printf("\na=%d,b=%d",a,b);

return 0;

}

对完成的程序进行一次简单的程序验证，如图1.2所示。

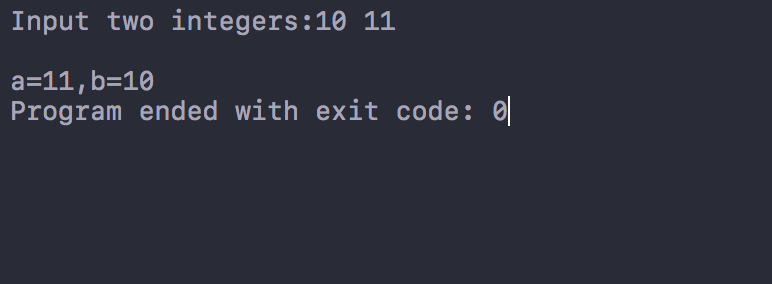


图1.2简单的程序验证

1.2.3程序设计

1编写一个程序，输入字符c，如果c是大写字母，则将c转换成对应的小写，否则c的值不变，最后输出c。

**解答**:首先判断是否为大写字母，若为大写字母，则将该字母的ASCII码减去32，若不是，则直接输出。转化为流程图（图1.3）

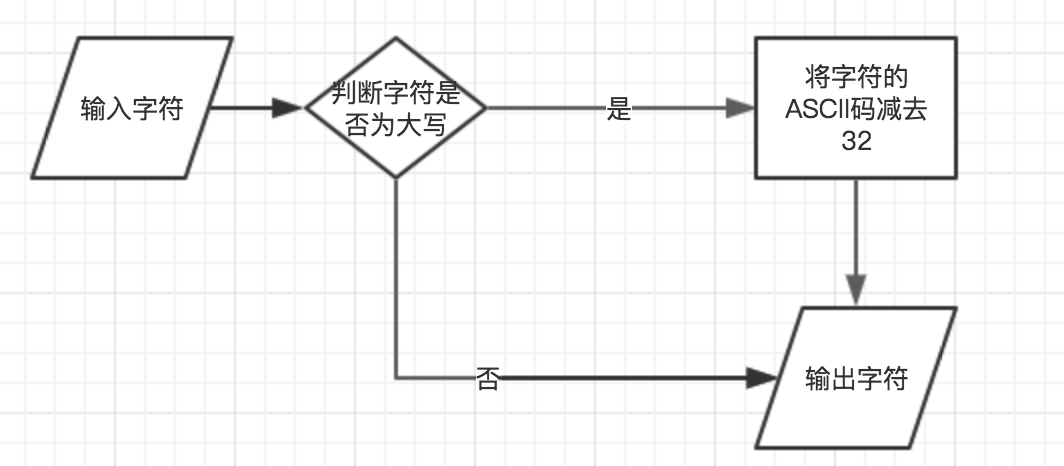


图1.3程序设计第一题的流程图

利用流程图思路写成源程序如下

#include <stdio.h>

#include <stdlib.h>

int main()

{

char x;

printf("Input your char:");

scanf("%c",&x);

if(x>='a'&&x<='z')

{

printf("%c\n",x);

}

else if(x>='A'&&x<='Z')

{printf("%c\n",x+32);}

else printf("please input right char");

return 0;

}

向程序输入小写的“a”，应该输出也是小写的“a”，实验结果（图1.4）与预设一致

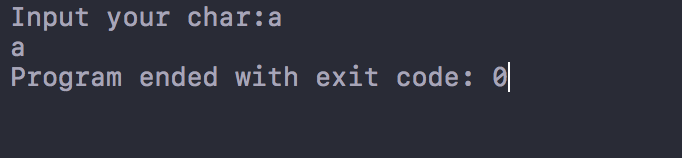


图1.4程序设计第一题的输出结果

2编写一个程序，输入无符号短整数x，ｍ，ｎ（0 ≤ｍ≤ 15, 1 ≤ ｎ≤ 16-ｍ）,取出x从第ｍ位开始向左的ｎ位（ｍ从右至左编号为0～15），并使其向左端（第15位）靠齐。

**解答**：解题思路主要通过判断语句完成。

1.输入x，m，n，为了方便分析测试结果，x的输入采用16进制

2.如果0 ≤ｍ≤ 15, 1 ≤ ｎ≤ 16-ｍ，转2.1，否则转3.

2.1 首先x>>m，将要处理的n位移动到最右；

2.2 再将上一步的结果左移16-n位，即：x<<16-n;

2.3 用16进制输出结果并转4.

3. 显示输入错误信息；

4. 结束

依照解题思路写出源程序

#include <stdio.h>

#include <stdlib.h>

int main()

{

unsigned short int x,m,n;

printf("Please enter x(16进制,m(0~15),n(1~16-n):\n" );

scanf("%hu%hu%hu",&x,&m,&n);

if ((m>=0&&m<=15)&&(n>=0&&n<=15-m))

{

x>>=m;

x<<=(16-n);

printf("%hx",x);

}

else printf("Error!\n");

return 0;

}

分三种不同的情况对源程序进行测试，对于输入的n、m值，分为m超边界条件，n超边界条件，m、n均不超出边界条件进行讨论。如表1-1所示

表1-1 编程题2的测试数据

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试  用例 | 程 序 输 入 | | | 理 论 结 果 | 运 行 结 果 |
| X | m | N |
| 用例1 | 0100 0110 1000 0000（4680） | 7 | 4 | 计算结果1101 0000 0000 0000 即D000 | D000 或 图1.5 |
| 用例2 | 1101 0101 1000 0011（D583） | 16 | 1 | 输入错误（m值超范围） | 图1.6 |
| 用例3 | 1101 0101 1000 0011（D583） | 13 | 5 | 输入错误（n值超范围） | 图1.7 |

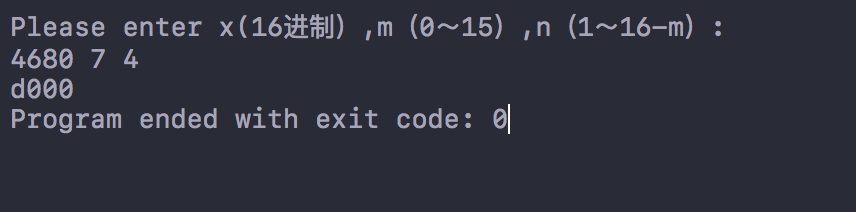


图 1.5 编程题2，m,n均不超出边界条件情况

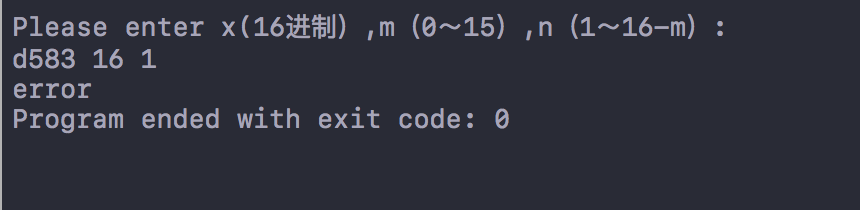


图 1.6编程题2，m超出边界条件情况

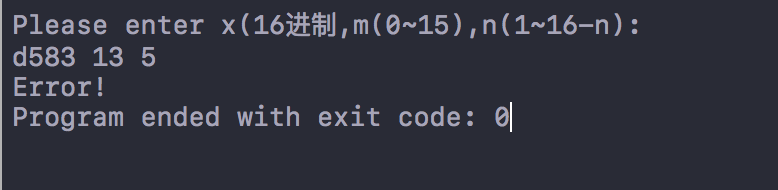


图 1.7编程题2，n超出边界条件情况

3 IP地址通常是4个用句点分隔的小整数（即点分十进制），如32.55.1.102。这些地址在0机器中用无符号长整数表示。编写一个程序，以机器存储的形式读入一个互联网IP地址，对其译码，然后用常见的句点分隔的4部分的形式输出。例如：整形676879571的二进制表示是00101000 01011000 01011100 11010011.按照8位一组可表示40 88 92 211.由于CPU处理数据的差异，他的顺序是颠倒的，所以最终格式为211.92.88.40。

**解答**：本题依据mask的遮盖作用可以将IP地址逐位提出然后再将其反向重新组装，既得到答案。

#include <stdio.h>

#include <stdlib.h>

int main()

{

unsigned long IP;

unsigned short one,two,three,four;

printf("Please enter your IP:");

scanf("%lu",&IP);

one=(IP&0xFF000000)>>24;

two=(IP&0xFF0000)>>16;

three=(IP&0xFF00)>>8;

four=(IP&0xFF);

printf("%hu.%hu.%hu.%hu",four,three,two,one);

return 0;

}

对源程序进行测试，整形676879571的二进制表示是00101000 01011000 01011100 11010011.按照8位一组可表示40 88 92 211.由于CPU处理数据的差异，他的顺序是颠倒的，所以最终格式为211.92.88.40。故输出应为211.92.88.40，结果如图1.8.

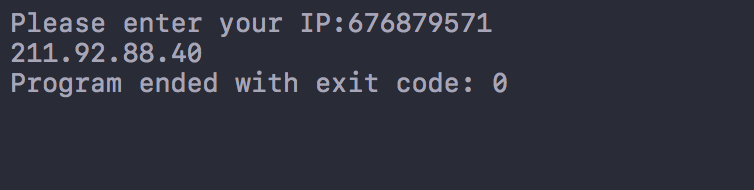


图 1.8 编程题3的输出结果

1.3实验心得

这次c上机实验给了我很多心得体会。其中，最为深刻的是程序的界面友好很关键。众所周知如果你不加以阐释，c语言的运行界面就是一片漆黑，别人根本不知道要输入什么数字，也不知道这些数字会被用来做什么。所以，一个良好的提示界面就显得尤为重要。以及，适当地在自己的代码后加上一点注释，有助于别人理解你的程序，自己以后打开也不会迷茫。

还有就是细心极为重要，这次改错的许多错误也是我经常犯的，比如语句打不打分号的问题等等。对于位运算，以前一直认为其作用不大，今天做了实验题之后我才真正了解到了位运算的简便以及高效，包括mask以及按位加的作用，位运算虽然比较费脑筋但是比起简单的循环确实是高效了许多。