# 实验1 表达式和标准输入与输出实验

* 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**

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；

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。

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);

}

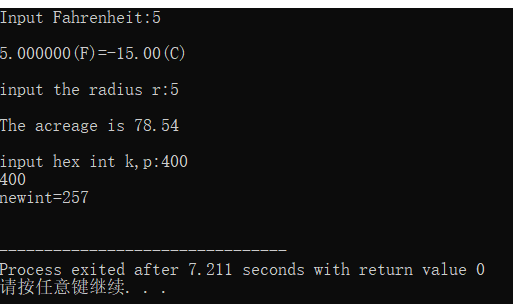
**解答：**

**1）错误修改**

（1） #define PI 3.14159;  
错误原因:宏定义在预处理时会将后续程序中的标示符展开为对应全部字符串，而PI不需要分号。  
改正:去掉其后分号

（2）voidmain(void)   
错误原因:函数返回值与函数名之间应加空格，C90标准建议main函数应返回一个int作为状态值  
改正: int main(void)

1. short p,k;   
   错误原因:后续用到了newint变量但并未声明改正: short p,k,newint
2. scanf("%d", f);  
   错误原因;应该把要读入的变量的地址传递给scanf才能正确写入改正: scanf("%d",&f);
3. c= 5/9\*(f-32);  
   错误原因: 5和9是整型常数，相除会截断小数部分，应该将其中任意一个输表示为或转换为浮点数类型  
   改正: C= 5.0/9\*(f-32);
4. scanf("%f", &r);  
   错误原因:double类型对应的占位符是%f改正: scanf("%f",&r);
5. print("\nThe acreage is % 2f\n\n",&s);  
   错误原因:要输出s的值应该将s作为然数传递给printf, 不应传递其地址  
   改正: print("\nThe acreage is % 2f\n\n",s);
6. scanf("%x %x", &k, &p );printf("new int = %x\n\n",newint);  
   错误原因; k,p 为short 类型，对应十六进制占位符应为%hx 改正: scanf("%hx %hx" ,&k,&p);
7. Newint=(p&0xff00)|(k&0xff00)<<8;  
   错误原因：要求将k的高字节作为结果的低字节，应该将k的高字节提取出后右移至低字节。  
   改正：newint=(p&0xff00)|(((k&0xff00)>>8)&0x00ff);
8. **错误修改后运行结果**

****

**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。  
程序代码:  
#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);  
[分析]将a,b中的任何一个看做原码，由此使用异或运算进行交换。

替换后的程序如下所示：

#include<stdio.h>

#include<stdlib.h>

int mainO{  
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;  
  
(测试1如果输入a= 1，b=7,应该输出: a=7, b=1,程序运行结果  
如下，程序正确。  


**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：**

1） 算法流程如图1.1所示：

开始

输入字符c

结束

输出c

N

输出c-’A’+’a’

Y

c是大写字母吗？

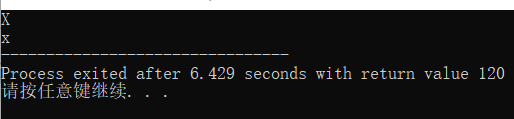
2）源程序清单

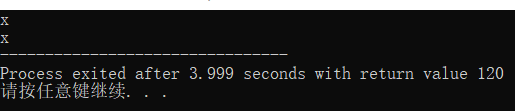
【程序】  
#include <stdio.h>  
int main（）  
{printf(“Input your char x:\n”);

char x;  
scanf( %c",&x);

if(x>='a'&&x<='z')  
printf(" %c\n" ,x);  
else if(x>='A'&&x<='Z')  
printf(" %c\n" x-’A’+’a’);  
else  
printf("Please input the right char.");  
  
return 0;  
}  
【测试】

1. 测试数据：大写字母’X’,小写字母’x’.
2. 对应测试数据的运行结果截图





（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：**

1) 解题思路：

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>>m)<<(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

2）程序清单

#include<stdio.h>

int main( void )

{

unsigned short x,m,n;

printf("输入x（16进制）、m（0~15）和n（1~16-m）：\n ");

scanf("%hx%hd%hd",&x,&m,&n);

if (m>=0&&m<=15&&n>=1&&n<=16-m)

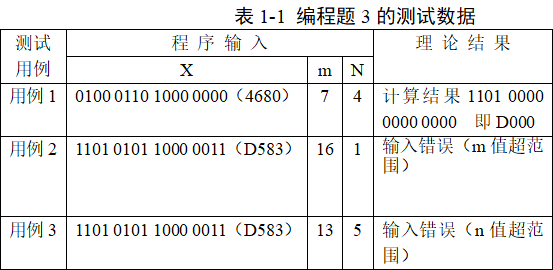
printf("ans=%hx\n",(x>>m)<<(16-n));

return 0;

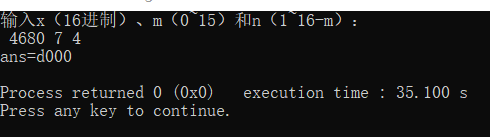
}

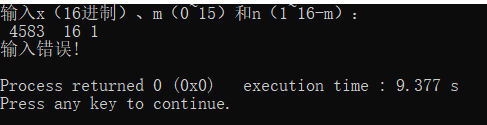
1. 测试
2. 测试数据

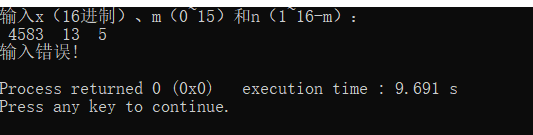
|  |
| --- |
|  |



(b)测试结果







说明上述的运行结果与理论分析吻合，验证了程序的正确性。

**1.3自设题**

**1）** 自设实验题目：实型数的浮点表示误差的简单验证

**（2）** 实验目的：通过设计实验程序，理解实数的浮点数表示上的误差，浮点数不宜做等值比较。

**（3）**实验程序：

#include "stdio.h"

int main()

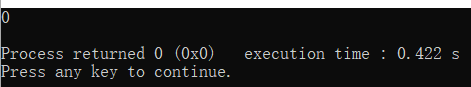
{

float a=10.33;

double b=10.33;

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

}

****

1. 实验结论：结果为0，即a==b的布尔值为0，即a与b不相等。说明浮点数的数据类型不同，值可能不同，不适合做比较。应该用一个较小的值来规定在该误差范围内的两个浮点数相等。

## 1.4 实验小结   在做改错题时，先浏览一遍，找出明显的错误，然后将初步修改后的代码输入编译器并编译，再根据编译器的警告与错误提示修改程序至完全正确，这个过程很需要细心和耐心，这个过程能够很好的帮助我再今后不犯类似错误。源程序修改与替换题锻炼我们举一反三的能力，我们在平时写程序时就应该思者对于一个问题有没有多钟解决方案，不使用中间变量交换两个整数是很常见的问题，我利用一个数异或自身等于0, 0异或任意位保留原位的原理来交换。程序 设计题提高了我们对所学知识的熟练度和理解程度，只有实践才能学会C语言，在亲手编写代码的过程中发现问题，理解概念。