# 1 表达式和标准输入与输出实验

## 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程序的编写方法。

（4）熟悉C语言程序的开发环境，并学会调试程序的方法。

## 1.2 实验内容

**1.2.1 程序改错与跟踪调试**

下面的实验1-1程序用来完成以下任务：

（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）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，拼成一个新的整数后输出。

在这个程序中存在若干语法和逻辑错误，要求先编译程序改正语法错误，再采用单步执行的方式调试程序找出逻辑错误。在单步执行程序的过程中，观察以下变量值：

（1）执行完c = 5/9 \* (f-32)，c的值为多少？

（2）执行完scanf(“%f”, &r)，r的值为多少？

（3）执行完newint = p&0xff00|k>>8，newint的值是多少？表达式k>>8的值是多少？

根据观察结果分析代码并修改程序，使之能够正确完成指定任务。

/\*实验1-1程序改错与跟踪调试题源程序\*/

1 #include<stdio.h>

2 #define PI 3.14159;

3 int main( void )

4 {

5 int f ;

6 short p, k ;

7 double c, r, s ;

/\* 任务1 \*/

8 printf("Input Fahrenheit: " ) ;

9 scanf("%d", f ) ;

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

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

/\* 任务2 \*/

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

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

14 s = PI \* r \* r;

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

/\* 任务3 \*/

16 k = 0xa1b2, p = 0x8432;

17 newint = p&0xff00|k>>8;

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

19 return 0;

20 }

**解答：**

（1）语法错误修改：

1) 第2行的符号常量定义后不能有分号，正确形式为：

#define PI 3.14159

2) 第6行的short p, k ;，未声明变量newint，正确形式为：

short p, k, newint ;

3) 第11行的scanf(“&d”,f );，未指明变量的地址，正确形式为：

scanf(“%d”,&f)

4) 第17行的scanf(“%f”,&r) ;，r为double类型，在scanf()函数中占位符应为1f，正确形式为：

scanf(“%1f”,&r) ;

5) 第19行的printf("\nThe acreage is %.2f\n\n",&s);中，不应使用地址，正确形式为：

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

（2）修改完以上错误后，在单步执行过程中，观察以下变量值：

1. 执行完c = 5/9 \* (f-32)，c的值为：0
2. 执行完scanf(“%f”, &r)，r的值为：0
3. 执行完newint = p&0xff00|k>>8，newint的值是0xffffffa1。表达式k>>8的值是0xa1。

（3）基于以上的单步执行观察结果，说明还存在如下逻辑错误：

1）第5行的c = 5/9 \* (f-32) ;，整数直接相除会损失精度，需要改为浮点型，应该修改成：c = 5.0/9.0 \* (f-32) ;

2）第6行的short p, k ;变量类型错误，应为unsigned short，应该修改成：unsigned short p, k, newint ;

修改后，源程序清单如下：

#include<stdio.h>

#define PI 3.14159

int main( void )

{

int f ;

unsigned short p, k, newint ;

double c, r, s ;

/\* 任务1 \*/

printf("Input Fahrenheit: " ) ;

scanf("%d", &f ) ;

c = 5.0/9.0 \* (f-32) ;

printf("%.2f",c) ;

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

/\* 任务2 \*/

printf("input the radius r:") ;

scanf("%lf", &r) ;

s = PI \* r \* r ;

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

/\* 任务3 \*/

k = 0xa1b2, p = 0x8432 ;

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

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

return 0 ;

}

（4）错误修改后运行结果如图1-1所示。

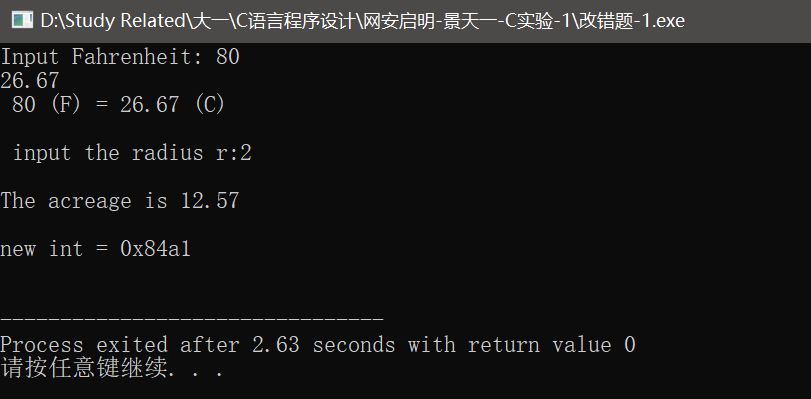
****

图1-1 实验1-1修改后运行结果截图

**1.2.2 源程序修改替换**

下面的实验1-2程序用“更相减损”法求m与n的最大公约数。

/\* 实验1-2程序分析与修改题源程序 \*/

#include <stdio.h>

int main( )

{

int m, n, k, p, i, d;

printf("input m, n \n");

scanf("%d%d", &m, &n);

if (m<n) /\* 交换m和n \*/

{

int t;

t = m;

m = n;

n = t;

}

k = 0;

while (m%2 == 0 && n%2 == 0) /\* m和n均为偶数 \*/

{

m /= 2; /\* 用2约简m和n \*/

n /= 2;

k++;

}

for (p = 1,i = 0; i<k; i++) p \*= 2; /\* 求p=2k \*/

while((d=m-n)!=n)

{

if(d>n) m = d;

else

{

m = n;

n = d;

}

}

d \*=p;

printf("the greatest common divisor : %d", d);

return 0;

}

1. 分析程序采用的“更相减损”法的算法步骤。

**解答：**

1. 程序采用的“更相减损”法的算法步骤如下：
   1. 输入被除数m和除数n
   2. 若m<n时，互换m, n的值
   3. 当m, n均可被2整除时，同时除以2直到其中一个数不能被2整除；
   4. 求数字p，使得p=2^k，k为上一步约简次数；
   5. 求m-n的值；
   6. 循环直至m-n的值等于除数n，计此时该数字为d；
   7. 最大公约数即为d\*p；
2. 按以下要求对源程序进行优化，提高程序的执行效率。

将交换m和n、判断一个整数是否是偶数、用2约简m和n、求p=2k等操作改成用位运算实现，并且m和n的交换不能够使用中间变量t，需要删除声明t的语句。

1. 将else后的复合语句改用一条表达式语句。
2. 按照要求优化后的程序分别如下：
3. 交换m和n的代码为：

m = m ^ n;

n = n ^ m;

m = m ^ n;

1. 判断一个整数是否是偶数的代码为：

(m&0x0001) == 0 && (n&0x0001) == 0

1. 用2约简m和n的代码为：

m = m >> 1; /\* 用2约简m和n \*/

n = n >> 1;

1. 求p=2k的代码为：

for (p = 1,i = 0; i<k; i++) p = p << 1; /\* 求p=2k \*/

1. else后的复合语句可以修改为：

m = n, m=d;

1. 优化和替换后程序的源代码如下：

#include <stdio.h>

int main(void )

{

int m, n, k, p, i, d;

printf("input m, n \n");

scanf("%d%d", &m, &n);

if (m<n) /\* 交换m和n \*/

{

m = m ^ n;

n = n ^ m;

m = m ^ n;

}

k = 0;

while ((m&0x0001) == 0 && (n&0x0001) == 0) /\* m和n均为偶数 \*/

{

m = m >> 1; /\* 用2约简m和n \*/

n = n >> 1;

k++;

}

for (p = 1,i = 0; i<k; i++) p = p << 1; /\* 求p=2^k \*/

while((d=m-n)!=n)

{

if(d>n)

m = d;

else

m = n, n = d;

}

d \*=p;

printf("the greatest common divisor : %d", d);

return 0;

}

优化和替换后运行结果如图1-2所示。

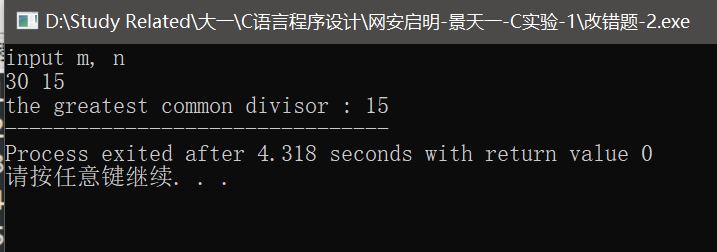


图1-2 实验1-2优化替换后运行结果截图

**1.2.3 程序设计**

（1）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，输入Ctrl+Z程序结束。要求：①用条件表达式；②字符的输入输出用getchar和putchar函数。程序应能循环接受用户的输入，直至输入Ctrl+Z程结束。例如：

A （键盘输入）

a

^Z （键盘输入）

**解答：**

1） 算法流程如图1-3所示。

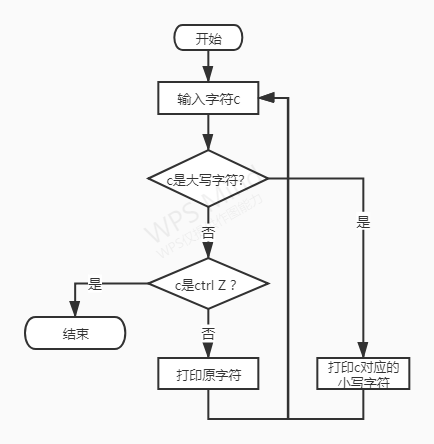


图1-3 程序设计题1的程序流程图

2）源程序清单

#include<stdio.h>

int main(void)

{

while(1)

{

char c;

c=getchar();

if (c>=65 && c<=90)

putchar(c+32);

else if(c==EOF)

break;

else

putchar(c);

}

return 0;

}

3）测试

（a）测试数据：

测试数据如表1-1所示：

表1-1程序题1的测试数据

|  |  |  |
| --- | --- | --- |
| 输入数据 | 预计输出 | 实际输出 |
| B | B | B |
| a | a | a |
| Ctrl Z | 结束 | 结束 |

（b） 对应测试数据的运行结果如图1-4所示。

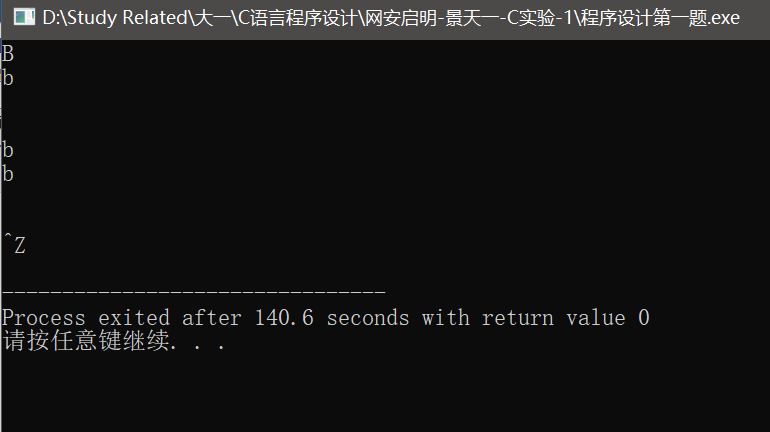


图1-4实验题1-3测试结果如图

输出结果符合预期，说明程序运行正确。

（2）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（x从右至左编号为0～15），并使其向左端（第15位）靠齐。要求：①检查m和n的范围；②x的值以十六进制输入，m和n以十进制输入；③结果以十六进制输出。

**解答：**

1. 解题思路如图1-5所示：

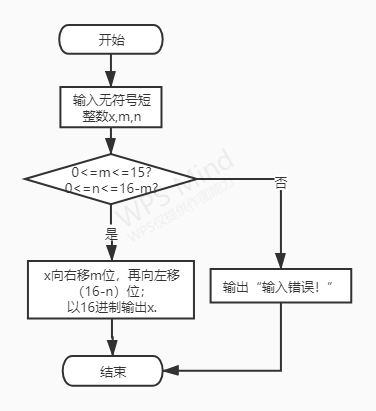


图1-5程序设计题2程序流程图

1. 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 >= 0 && n <= 16-m)

/\*判断m、n的值是否在合理范围内\*/

{

printf("ans=%hx\n",x);

}

else

{

printf("输入错误!\n");

}

return 0;

}

3）测试

（a） 测试数据：

测试数据如表1-3示。

表1-2程序题2的测试数据

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试  用例 | 程 序 输 入 | | | 理 论 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 |
| 用例2 | 1101 0101 1000 0011（D(583） | 16 | 1 | 输入错误（m值超范围） |
| 用例3 | 1101 0101 1000 0011（D583） | 13 | 5 | 输入错误（n值超范围） |

（b） 对应测试测试用例1的运行结果如图1-4所示。

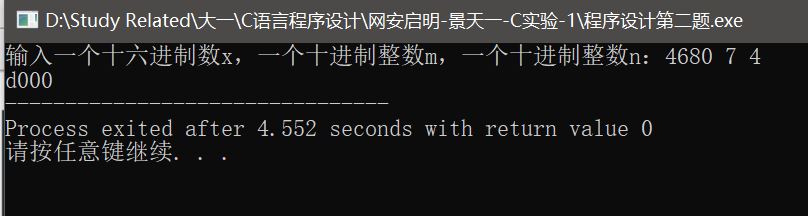


图1-6程序设计题2的测试用例一的运行结果

对应测试测试用例2的运行结果如图1-5所示。

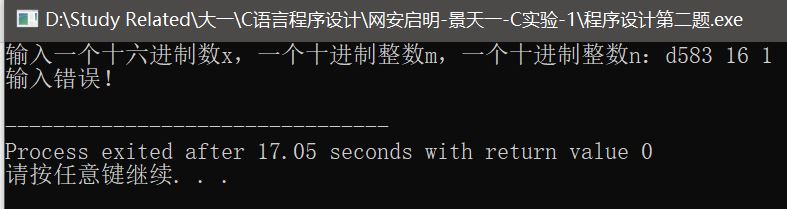


图1-7程序设计题2的测试用例二的运行结果

对应测试测试用例3的运行结果如图1-6所示。

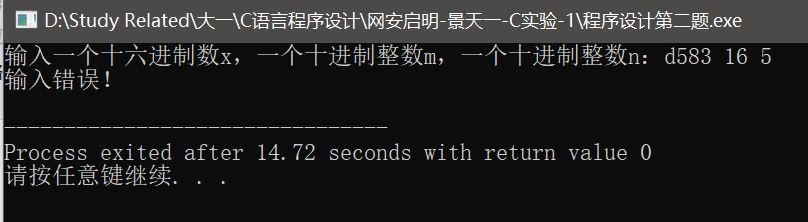


图1-8程序设计题2的测试用例三的运行结果

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。

（3）IP地址通常是4个用句点分隔的小整数（即点分十进制），但这些地址在机器中是用一个无符号长整型数表示的。例如3232235876，其机内二进制表示就是11000000 10101000 00000001 01100100，按照8位一组用点分开，该IP地址就写成192.168.1.100。

读入无符号长整型数表示的互联网IP地址，对其译码，以常见的点分十进制形式输出。要求循环输入和输出，直至输入Ctrl+Z结束。

**解答：**

1. 解题思路：

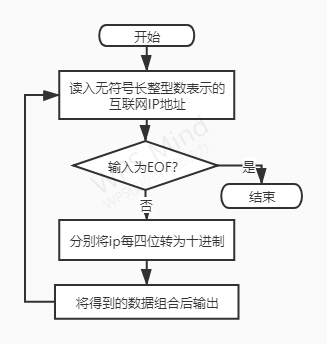
算法流程如图1-9所示。

图1-9程序设计题3程序流程图

1. 程序清单

#include<stdio.h>

int main(void)

{

unsigned long ip,d\_1,d\_2,d\_3,d\_4; //获取数据

while(scanf("%lu",&ip) != EOF) //循环条件

{

d\_1 = (ip & 0xff000000)>>24;

d\_2 = (ip & 0x00ff0000)>>16;

d\_3 = (ip & 0x0000ff00)>>8;

d\_4 = (ip & 0x000000ff);

printf("%d.%d.%d.%d\n",d\_1,d\_2,d\_3,d\_4); //ip地址输出

scanf("%lu",&ip);

}

return 0;

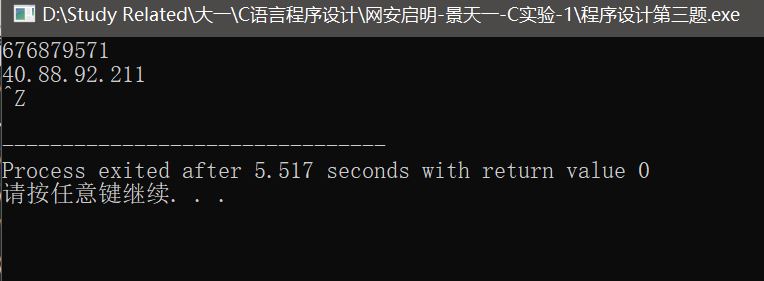
}

1. 测试
   1. 测试数据

测试数据如表1-3所示。

1. 表1-3程序题3的测试数据

|  |  |  |
| --- | --- | --- |
| 测试用例 | 程序输入 | 实际输出 |
| 用例1 | 676879571（00101000 01011000 01011100 11010011） | 211.92.88.40 |
| 用例2 | ^Z | 结束 |

 b) 对应测试测试用例1的运行结果如图1-10所示。

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 实验小结

本次实验主要练习了输入输出函数的用法，在具体编程实验中有重要的意义。遇到的第一个问题是ctrl z的输入，通过查询资料和询问同学解决；第二个问题是在输入ctrl z时终止，通过查阅课本了解到scanf函数的返回值在出错或终止时为EOF，因此设计判断scanf函数返回值的方案来实现。

本次实验最大的体会是，完整实现需求并将其有条理地记录下来是极为重要的，对每一次实验都需要提前预习，完成代码demo，以给后期调试，撰写报告，整理反思留出更多时间。

参考文献

[1] 曹计昌,卢萍,李开. C语言程序设计, 北京：科学出版社, 2013

[2] 卢萍,李开,王多强等. C语言程序设计典型题解与实验指导, 北京：清华大学出版社, 2019