

## 实验 3 顺序结构程序设计

### 一、实验目的

- (1) 掌握 C 语言中使用最多的一种语句——赋值语句的使用方法;
- (2) 掌握 C 语言的各种数据类型:整型、实型、字符型变量的定义;
- (3) 掌握 C 语言中算术运算符及表达式的使用;
- (4) 掌握 C 语言的顺序结构程序设计方法。

### 二、实验内容和步骤

#### I.基础部分：理解顺序结构程序的执行过程，理解各种数据类型及它们的运算

- (1) 给定程序 c3-1-1.c 的功能为：根据键盘输入的半径，计算圆面积。

① 编辑并运行该程序，观察运行结果。

```
/* c3-1-1.c */
#include <stdio.h>
int main()
{
    float r;
    double area;
    printf("请输入圆的半径:");
    scanf("%f",&r);
    area=3.14159*r*r;
    printf("计算结果如下:\n");
    printf("r=%5.2f, area=%lf\n",r,area);
    return 0;
}
```

运行该程序时，从键盘输入 4.5，程序运行结果如图 3-1 所示。

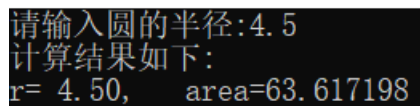


图 3-1 程序 c3-1-1.c 的运行结果

② 程序 c3-1-2.c 将程序 c3-1-1.c 中的语句调换位置。

```
/* c3-1-2.c */
#include <stdio.h>
int main()
{
    float r;
    double area;
    area=3.14159*r*r; //将计算面积的语句提前到输入数据之前
    printf("请输入圆的半径:");
    scanf("%f",&r);
    printf("计算结果如下:\n");
    printf("r=%5.2f, area=%lf\n",r,area);
    return 0;
}
```

```

printf("请输入圆的半径:");
scanf("%f",&r);
printf("计算结果如下:\n");
printf("r=%5.2f, area=%lf\n",r,area);
return 0;
}

```

在某台电脑上编译时程序出现如图 3-2 所示警告信息，程序的运行结果如图 3-3 所示。请分析并回答出现这样结果的原因。

```

-----Configuration: c3-1-2 - Win32 Debug-----
Compiling...
c3-1-2.c
d:\c调试\c3-1-2.c(7) : warning C4700: local variable 'r' used without having been initialized

```

图 3-2 程序 c3-1-2.c 的警告信息

```

请输入圆的半径:4.5
计算结果如下:
r= 4.50, area=36220062378809952.000000

```

图 3-3 程序 c3-1-2.c 的运行结果

(2) 下列程序是浮点类型数据的使用，观察程序的运行结果，分析并回答问题。

```

/* c3-1-3.c */
#include <stdio.h>
int main()
{
    float fa,fb,fc,fsum;
    double d;
    fa=3141.592678;
    fb=6.026e-27;
    fsum=fa+fb;
    d=31415926.78;
    printf("float 类型数据的打印结果: \n");
    printf("fa=%f\t fb=%f\t fsum=%7.2f\n",fa,fb,fsum);//用格式符%f 输出 float 类型变量
    printf("fa=%e\t fb=%e\t fsum=%e\n",fa,fb,fsum); //用格式符%e 输出 float 类型变量
    printf("double 类型数据的打印结果: \n");
    printf("d=%lf, d=%e\n",d,d);
    return 0;
}

```

① 当在某台电脑上编译此程序时，会出现如图 3-4 所示的 warning 信息，为什么？

```

-----Configuration: c3-2-1 - Win32 Debug-----
Compiling...
c3-2-1.c
D:\C调试\c3-2-1.c(7) : warning C4305: '=' : truncation from 'const double' to 'float'
D:\C调试\c3-2-1.c(8) : warning C4305: '=' : truncation from 'const double' to 'float'
D:\C调试\c3-2-1.c(5) : warning C4101: 'fc' : unreferenced local variable

```

图 3-4 程序 c3-2-1.c 的警告信息

② 观察并分析程序中第 2 条和第 3 条输出语句的结果。

(3) 下列程序涉及字符类型和整型类型的互通性, 观察程序的运行结果, 分析并回答问题。

```
/* c3-1-4.c */
#include <stdio.h>
int main()
{
    char c1='a',c2='b',c3='c';
    c1=c1+2;
    c2=c2+8;
    c3=c3-32;
    printf("char 类型数据的打印结果: \n ");
    printf("c1=%c\t c2=%c\t c3=%c\n",c1,c2,c3);
    printf("char 类型数据和 int 类型的关系: \n");
    printf(" c1=%d\t c2=%d\t c3=%d",c1,c2,c3);
    return 0;
}
```

- ① 变量 c1、c2、c3 用格式符%c 和%d 输出时的结果为何不同?
- ② 变量 c1、c2、c3 用格式符%c 输出时为什么输出结果不再是 'a'、'b'、'c',而是变成了 'c'、'j'、'C' ?

(4) 下列程序可以测试整型数据类型变量的取值范围, 观察程序的运行结果, 分析并回答问题。

```
/* c3-1-5.c */
#include <stdio.h>
int main()
{
    int a,b,c,d;
    a = 2147483647; /* 为整型变量 a 赋值最大的正整数 */
    b = a+1;
    c = -2147483648; /* 为整型变量 c 设定最小的负整数 */
    d = c-1;
    printf("正数范围 a=%d , b=%d\n", a, b);
    printf("负数范围 c=%d , d=%d\n", c, d);
    return 0;
}
```

- ① 已知在内存中, 整型数据占 4 个字节空间, 最大值为 2147483647, 程序中变量 b 的值为什么不是 2147483648? 将一个大于 2147483647 的整数赋给一个整型变量, 会得到什么结果?
- ② 占内存 4 个字节的整数的最小值为什么是-2147483648 而不是-2147483647? 程序中变量 d 的值为什么不是-2147483649?

(5) 下列程序为多种数据类型应用，观察程序的运行结果，分析并回答问题。

```
/* c3-1-6.c */
#include <stdio.h>
int main()
{
    int a,b;
    unsigned c,d;
    a = 100;
    b = -100;
    c = a;
    d = b; //将一个负整数赋给一个无符号的变量
    printf("a=%d, b=%d\n", a, b);
    printf("a=%u, b=%u\n", a, b); //按无符号格式输出 a,b
    printf("c=%u, d=%u\n", c, d);
    return 0;
}
```

- ① 观察变量 d 的值，分析将一个负整数赋给一个无符号的变量，会得到什么结果。
- ② 观察变量 c 的值，分析将一个整数赋给无符号变量，会得到什么结果？
- ③ 无符号整数的最大值为 4294967295。改变程序中各变量的值，例如： a = 4294967295, b = -4294967295。重新分析上述问题①和②。

(6) 下列程序为算术运算符“/”、“%”及强制类型转换运算符的使用，观察程序的运行结果，分析并回答问题。

```
/* c3-1-7.c */
#include "stdio.h"
int main( )
{
    int a,b;
    a=2;
    b=1%a;
    printf("%d\n",1/a);
    printf("b=%d\n",b);
    printf("%f %f\n",(float)(1/a), (float)1/a);
    return 0;
}
```

- ① 第一个 printf 语句的输出结果为什么是 0？
- ② 第三个 printf 语句中 2 个输出项的输出结果为什么不同？

(7) 下列程序为不同类型数据的混合运用，观察程序的运行结果，分析并回答问题。

```
/* c3-1-8.c */
#include <stdio.h>
int main()
{
    int a;
```

```

float d;
char c1;
double f;
long m;
unsigned p;
a = 61;
c1 = 'a';
d = 3.56;
f = 3157.890121;
m = -2147483647;
p = -2147483647;
printf(" a=%d \n c1=%c \n d=%6.2f \n", a, c1, d);
printf(" f=%15.12f \n m=%ld \n p=%u \n", f, m, p);
return 0;
}

```

- ① 运行此程序并观察和分析结果(特别注意变量 p 输出的值)。
- ② 改用 scanf 函数输入数据而不用赋值语句, scanf 函数如下:

```
scanf( "%d,%c,%f,%lf,%ld,%u" , &a,&c1,&d,&f,&m,&p);
```

输入的数据如下: **61, a, 3. 563157, 0. 123456789, -2147483648, -2147483647** ✓

请分析运行结果。(说明: 1f 和 1d 格式符分别用于输入 double 型和 long 型数据)

注意: 程序运行时, scanf 输入各类不同数据时采用的分隔符的情况。

- ③ 将 printf 语句改为:

```
printf( "a=%d\nc1=%c\nd=%15.6f\n" , a, c1, d);
printf( "f=%f\nm=%d\np=%d\n" , f, m, p);
```

分析运行程序。

- ④ 将变量 p 改用 %o 格式符输出。
- ⑤ 将 scanf 函数中的 %lf 和 %ld 改为 %f 和 %d, 运行程序并观察分析结果。

(8) 下列程序为不同数据类型的混合运算, 观察程序的运行结果, 分析并回答问题。

```

/* c3-1-9.c */
#include <stdio.h>
int main()
{
    int n=100;
    float f;
    double d;
    f=n+2.5;
    printf("f=%f\n",f);
    n=f/2;
    printf("n=%d\n",n);
    n=(1/2)*f;
    printf("n=%d\n",n);
    d=f*2;
}

```

```

    printf("d=%lf\n",d);
    return 0;
}

```

- ① 在某台电脑上编译此程序时，出现如图 3-5 所示的警告信息，为什么？
- ② 观察并分析程序运行结果。

```

-----Configuration: c3-2-8 - Win32 Debug-----
Compiling...
c3-2-8.c
d:\c调试\c3-2-8.c(8) : warning C4244: '=' : conversion from 'double ' to 'float ', possible loss of data
d:\c调试\c3-2-8.c(10) : warning C4244: '=' : conversion from 'float ' to 'int ', possible loss of data
d:\c调试\c3-2-8.c(12) : warning C4244: '=' : conversion from 'float ' to 'int ', possible loss of data

```

图 3-5 程序 c3-2-8.c 的警告信息

## II. 提高部分：顺序结构程序设计

### 1、程序填空

说明：程序有多个空 (1)、(2) ……需要补充完整。请将程序中的\_\_ (1) \_\_、\_\_ (2) \_\_……删除后，在相应的位置填入正确答案并调试直到得到正确结果为止。注意：不要随意改动程序，不得增行或删行，也不得更改程序的结构！

(1) 给定程序 c3-2-1.c 的功能是：将 a, b 两个变量的值交换并输出。通过赋值语句给变量 a、b 赋值，程序的运行结果如图 3-6 所示。请完善程序，并运行出正确的结果。

**a=1, b=2**

图 3-6 程序 c3-2-1.c 运行结果

```

/*  c3-2-1.c  */
#include <stdio.h>
int  main( )
{
    int  a,b,t;
    /*****found*****/
    ____ (1) ____;    //此处填写通过赋值语句给变量 a 赋值的语句;
    /*****found*****/
    ____ (2) ____;    //此处填写通过赋值语句给变量 b 赋值的语句;
    t=a;
    a=b;
    b=t;              /* 以上三条赋值语句实现 a,b 变量值的交换 */
    /*****found*****/
    ____ (3) ____;    //此处填写输出变量 a、b 的值的语句;
    return 0;
}

```

(2) 给定程序 c3-2-2.c 的功能是：将 a, b 两个变量的值交换并输出。通过 scanf () 函数从键盘输入变量 a、b 的值，程序的运行结果如图 3-7 所示。请完善程序，并运行出正确的结果。

```

请输入变量a、b的值:2 1
交换前: a=2, b=1
交换后: a=1, b=2

```

图 3-7 程序 c3-2-2.c 运行结果

```

/* c3-2-2.c */
#include <stdio.h>
int main( )
{
    int a,b,t;
    printf("请输入变量 a、b 的值:");
    /******found******/
    ____ (1) ____; //此处填写通过 scanf()从键盘输入数据给变量 a、b 的语句;
    /******found******/
    ____ (2) ____//此处填写输出交换前变量 a、b 的值的语句;
    t=a;
    a=b;
    b=t;
    /******found******/
    ____ (3) ____//此处填写输出交换后变量 a、b 的值的语句;
    return 0;
}

```

(3)程序 c3-2-3.c 的功能是：当  $x=2.5$ ,  $a=7$ ,  $y=4.7$  时，计算并输出表达式  $x + a \% 3 * (\text{int})(x + y) \% 2 / 4$  的运算结果。程序的运行结果如图 3-8 所示。请完善程序，运行出正确的结果，并分析得到结果的原因。

```

z=2.500000

```

图 3-8 程序 c3-2-3.c 运行结果

```

/* c3-2-3.c */
#include <stdio.h>
int main( )
{
    /******found******/
    ____ (1) ____; //此处填写对变量 a 的定义和赋值语句
    /******found******/
    ____ (2) ____; //此处填写对变量 x, y, z 的定义和赋值语句
    z= x + a \% 3 * (\text{int})(x + y) \% 2 / 4;
    printf("z=%f\n",z);
    return 0;
}

```

## 2、程序设计

(1) 编写程序 c3-2-9.c, 程序的功能是: 从键盘上输入一个华氏温度, 能够输出相应的摄氏温度, 输出结果保留 2 位小数。华氏温度 F 与摄氏温度 c 的转换关系为:

$$C = \frac{5}{9} (F - 32)$$

提示: 程序框架如下:

```
/*c3-2-4.c*/
#include <stdio.h>
int main( )
{
    //定义需要的变量类型;
    //键盘输入已知华氏温度 F;
    //由赋值语句实现转换为华氏温度 c;
    //输出语句;
}
```

(2) 编写程序 c3-2-5.c, 程序所实现的功能是: 两次调用 getchar() 函数读入两个字符分别赋给 c1 和 c2, 再分别用 putchar( ) 函数和 printf( ) 函数输出这两个字符。

提示: 程序框架如下, 注意输入两个字符时必须是连续的, 中间不能有空格。

```
/* c3-2-5.c */
#include <stdio.h>
int main( )
{
    //定义需要的变量 (注意类型);
    //用 getchar 函数从键盘读入 c1;
    //用 getchar 函数从键盘读入 c2;
    //用 putchar 函数输出变量 c1, c2;
    //用 printf 函数输出变量 c1, c2;
    //return 0;
}
```

(3) 编写程序 c3-2-6.c, 程序所实现的功能是: 设圆半径 r=1.5, 圆柱高 h=3, 求对应的圆周长、圆面积、圆柱体积。要求使用 scanf 函数输入半径和高, 输出计算结果。输出时要有文字说明, 输出结果取小数点后两位数字。

### III. 拓展部分: 文件的使用

(1) 编写程序 c3-3-1.c, 程序实现的功能是: 从键盘输入一个同学的学号及高数、英语、C 语言 3 门课的成绩, 计算平均分, 并写入文件 score\_avg.txt 中。

例如: 从键盘输入数据: 1001 90 80 70, 文件 score\_avg.txt 中的记录应该为:

学号	高数	英语	C 语言	平均分
1001	90	80	70	80.00

(2) 改写 (1) 中的程序为 c3-3-2.c: 学号及 3 门课成绩从文件 score.txt 中读入 (而不是从键盘读入), 写入文件 score\_avg.txt 中。score.txt 文件格式如图 3-6 所示:



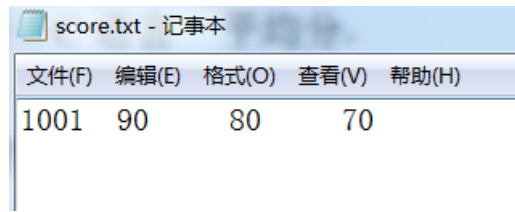


图 3-6 程序运行后文本文件“score.txt”的内容