

实验 5 循环结构程序设计-单重循环

一、实验目的

1. 掌握用 while 语句, do-while 语句和 for 语句实现单重循环的方法。
2. 掌握在程序设计中用循环的方法实现一些常用算法 (求和, 求积, 分类统计等)。
3. 学习调试程序的一些技巧。

二、实验内容和步骤

I. 基础部分: 理解单重循环结构

(1) 给定程序 c5-1-1.c 的功能为: 在屏幕输出如图 5-1 所示图形。请分析程序的运行结果, 回答问题。



图 5-1 程序 c5-1-1.c 的运行结果

```
/* c5-1-1.c */
#include<stdio.h>
int main()
{
    int i=1;
    while(i<=5)
    {
        printf("*****\n");
        i=i+1;
    }
    return 0;
}
```

- ① 画出该程序的流程图;
- ② 编辑并运行该程序, 查看运行结果。
- ③ 将 c5-1-1.c 稍作改动, 去掉程序中的一对大括号, 并另存为 c5-1-2.c。程序如下:

```
/* c5-1-2.c */
#include<stdio.h>
int main()
{
    int i=1;
    while(i<=5)
```

```

        printf("*****\n");
        i=i+1;
        return 0;
    }

```

运行程序，观察程序的运行结果，分析出现这个结果的原因。

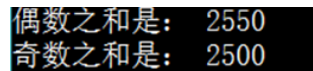
II.提高部分：掌握单重循环结构设计

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 c5-2-1.c 的功能是：计算 1 到 M(M 为偶数) 之间的奇数之和及偶数之和。程序运行结果如图 5-2 所示，请完善程序。



```

偶数之和是: 2550
奇数之和是: 2500

```

图 5-2 程序 c5-2-1.c 的运行结果

```

/*    c5-2-1.c    */
#include <stdio.h>
#define M    100
int main()
{
    int a,b,i;
    a=0; b=0;
    /*****found*****/
    for(i=1;__ (1) __;i+=2)
    {
        a=a+i;
        /*****found*****/
        __ (2) __;
    }
    printf("偶数之和是:  %d\n",b);
    printf("奇数之和是:  %d\n",a);
    return 0;
}

```

- (2) 给定程序 c5-2-2.c 的功能是：从键盘输入整数，分别计算所输入的正整数的和、负整数的和。当输入 0 时，结束输入并输出计算结果。当输入一些整数时，程序运行结果如图 5-3 所示，请完善程序。

```

请输入一些整数（输入0时结束输入）：
12 34 56 -12 -34 -56 0
大于0的整数之和为： 102
小于0的整数之和为： -102

```

图 5-3 程序 c5-2-2.c 的运行结果

```

/*  c5-2-2.c  */
#include <stdio.h>
int main()
{
    int x, sum1, sum2;
    /*****found*****/
    _____ (1) _____
    printf("请输入一些整数(输入 0 时结束输入): \n");
    scanf("%d", &x);
    while ( x != 0 )
    {
        /*****found*****/
        _____ (2) _____
        sum1 = sum1+x;
        else
            sum2 =sum2+ x;
        scanf("%d", &x);
    }
    printf("大于 0 的整数之和为:  %d\n", sum1);
    printf("小于 0 的整数之和为:  %d\n", sum2);
    return 0;
}

```

(3) 给定程序 c5-2-3.c 的功能是：计算正整数 num 的各位上的数字之积。

例如，若从键盘输入 252，则运行结果如图 5-4 所示；若从键盘输入 202，则运行结果如图 4-4-2 所示。请完善程序。

请输入一个数：252 各位数字的积为：20	请输入一个数：202 各位数字的积为：0
--------------------------	-------------------------

图 5-4 程序 c5-2-3.c 的运行结果示例

```

/*  c5-2-3.c  */
#include <stdio.h>
int main()
{
    int num,k;
    /*****found*****/
    _____ (1) _____;
    printf("请输入一个数：");
}

```

```

scanf("%d",&num);
do
{
    k=k*(num%10);
    /*****found*****/
    _____ (2) _____;
} while(num!=0);
printf("\n 各位数字的积为: %d\n",k);
return 0;
}

```

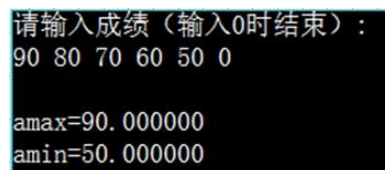
2、程序改错

说明：程序中有错误，错误都在提示行： /*****found*****/的下面一行。请改正程序中的错误，调试运行使它能得出正确的结果。

注意：程序中的其它地方不要随意改动，不得增行或删除行，也不得更改程序的结构！

(1) 给定程序 c5-2-4.c 的功能是：从键盘上输入若干个学生的成绩，统计并输出最高成绩和最低成绩，当输入负数时结束输入。

例如：输入一些成绩后，程序的运行结果如图 5-5 所示，请改正程序中的错误，并运行出正确的结果。



```

请输入成绩（输入0时结束）：
90 80 70 60 50 0

amax=90.000000
amin=50.000000

```

图 5-5 程序 c5-2-4.c 的运行结果示例

```

/*  c5-2-4 .c  */
#include <stdio.h>
int main()
{
    float x,amax,amin;
    printf("请输入成绩（输入 0 时结束）:\n");
    scanf("%f",&x);
    /*****found*****/
    amax=100;amin=0;
    /*****found*****/
    for (    ; x!=0;    )
    {
        if (x>amax) amax=x;
        /*****found*****/
        if ( x<amax) amin=x;
        scanf("%f",&x);
    }
    printf("\namax=%f\namin=%f\n",amax,amin);
}

```

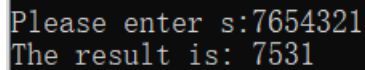
```

    return 0;
}

```

(2) 给定程序 c5-2-5.c 的功能是：从低位开始取出整型变量 s 中奇数位上的数，依次构成一个新数放在 t 中。(注意：整型数据的取值范围是： -2147483648 ~ 2147483647)

例如，当 s=7654321 时，程序运行结果如图 5-6 所示，请改正程序中的错误，使程序运行结果正确。



```

Please enter s:7654321
The result is: 7531

```

图 5-6 程序 c5-2-4.c 的运行结果示例

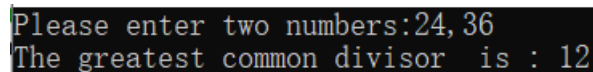
```

/*c5-2-5.c*/
#include <stdio.h>
int main()
{
    int s, t, sl=10;
    printf("\nPlease enter s:");
    scanf("%d", &s);
    /******found*****/
    t = s / 10;
    while ( s > 0)
    {
        s = s/100;
        t = s%10 * sl + t;
        /******found*****/
        sl = sl*100;
    }
    printf("The result is: %d\n", t);
    return 0;
}

```

(3) 给定程序 c5-2-6.c 的功能是：从键盘上输入两个正整数 x, y，求它们的最大公约数。

例如：从键盘上输入 24 和 36，程序运行结果如图 5-7 所示，请改正程序中的错误，使程序运行结果正确。



```

Please enter two numbers:24,36
The greatest common divisor is : 12

```

图 5-7 程序 c5-2-6.c 的运行结果示例

```

/* c5-2-6.c */
#include <math.h>
#include <stdio.h>
int main()
{
    int n, m, t;

```

```

printf("Please enter two numbers:");
scanf("%d,%d", &n, &m);
if(n < m)
{
    t = n;
    n = m;
    m = t;
}
t = n%m;
/*****found*****/
while( t!=0 );
{
    n = m;
    m = t;
    /*****found*****/
    t = n / m;
}
/*****found*****/
printf("The greatest common divisor is : %d\n", n );
return 0;
}

```

3、程序设计

(1) 编写程序 c5-2-7.c, 程序实现的功能是: 求两个正整数[m, n]之间所有既不能被 3 整除也不能被 7 整除的整数之和。

要求:

- ① 画出程序流程图。
- ② 完成程序代码的编写、调试, 最终得到正确的运行结果。

编程提示:

- 定义三个变量 m, n 和 t, 再定义一个循环控制变量和结果变量 s。
- 从键盘输入两个整数 m 和 n 的值, 判断这两个变量的值, 如果 m>n, 则交换两个变量的值, 使得 m 始终小于 n。
- 用循环依次判断 m 和 n 之间的每一个整数, 在循环体中通过条件语句来判断这个数是否既不能被 3 整除也不能被 7 整除, 如果满足条件, 累加求和, 如果不满足, 则继续循环。
- 程序的基本结构如下:

```

/* c5-2-7.c */
#include <stdio.h>
int main()
{
    定义变量;
    变量赋初值;
    输入 m, n 的值;

```

```

        if( m>n )
            交换 m 和 n 的值;
    for ( _____ )
    {
        if ( 该数不能被 3 整除也不能被 7 整除 )
            变量 s 累加求和 ;
    }
    printf("Sum is : %d \n",s );
    return 0;
}

```

(2) 编写程序 c5-2-8.c, 实现的功能是: 从键盘输入一行字符, 分别统计这行字符中的英文字母、空格、数字和其它字符的个数。

要求:

- ① 画出程序流程图。
- ② 完成程序代码的编写、调试, 最终得到正确的运行结果。

编程提示:

- 先定义一个字符型的变量 (如 ch), 再定义 4 个整型变量作为计数器, 注意: 作为计数器的变量要先赋初值 0。
- 在循环中每次从键盘上读入一个字符, 在循环体中对读入的字符进行判断, 相应的计数器加 1, 当读入的字符为 '\n' 时表示读入结束。
- 编程中可使用如下的循环结构:

```

/* c5-2-8.c */
#include <stdio.h>
int main( )
{
    .....
    ch=getchar();          // (1)
    while(ch!=' \n' )
    {
        if(.....) .....;
        else if(.....) .....;
        .....;
        else .....;
        ch=getchar();      // (2)
    }
    return 0;
}

```

思考: 在以上程序中, (1) 和 (2) 标示的语句 ch=getchar() 的作用在程序中有何不同, 这两个相同的语句可以只写其中一个吗? 为什么?

(3) 编写程序 c5-2-9.c, 实现的功能是: 求 $1! + 2! + 3! + \dots + 20!$ 的值。

要求:

- ① 画出程序流程图。
- ② 完成程序代码的编写、调试, 最终得到正确的运行结果。

编程提示:

- 可以采用如下程序结构:

```
#include <stdio.h>
int main()
{
    定义变量 i 作为循环控制变量;
    定义变量 p 和 sum 分别存放各个整数的阶乘和阶乘之和;
    变量 p 和 sum 赋初值;
    for (i=1; i<=20; i++)
    {
        变量 p 连乘 ;
        变量 sum 累加;
    }
    输出 sum 的值 ;
}
```

(4) 编写程序 c5-2-10.c, 实现的功能是: 按下面公式, 求 π 的近似值 (保留四位小数), 直到最后一项中的分数小于 10^{-6} 为止。

$$\frac{\pi}{2} \left(1 + \frac{1}{1*3}\right) * \left(1 + \frac{1}{3*5}\right) * \left(1 + \frac{1}{5*7}\right) * \left(1 + \frac{1}{7*9}\right) * \dots$$

要求:

- ① 画出程序流程图。
- ② 完成程序代码的编写、调试, 最终得到正确的运行结果。

编程提示:

- 由于程序求和没有给出具体的项数, 要求当最后一项的分数小于 10^{-6} 时循环终止, 因此, 进行累加累乘时要考虑循环的条件。
- 编程时可以考虑用如下结构:

```
#include <stdio.h>
#include <math.h>
int main()
{
    定义变量 (存放和 (积), 以及通项、计数等变量;
    变量赋初值;
    准备第一项的通项 (可以是某项的一部分, 也可以是一项);
    while (fabs (分数通项) > 1.0e-6)
    {
        累加或累乘该通项;
```



```

        下一项计数增值;
        准备计算下一个通项 item;
    }
    打印输出结果;
    return 0;
}

```

(5) 编写程序 `c5-2-11.c`，实现的功能是：一位百万富翁遇到一位陌生人，陌生人找他谈一个换钱计划：陌生人每天给富翁 10 万元，而富翁第 1 天只需要给陌生人 1 分钱，第 2 天陌生人仍然给富翁 10 万元，而富翁只需要给陌生人 2 分钱，第 3 天陌生人仍然给富翁 10 万元，富翁只需要给陌生人 4 分钱……富翁每天给陌生人的钱是前一天的 2 倍，直到 30 天后，计划终止。请编写一个程序，如果按计划执行，计算 30 天后富翁给了陌生人多少钱，陌生人给了富翁多少钱。

要求：

- ① 画出程序流程图。
- ② 完成程序代码的编写、调试，最终得到正确的运行结果。

III. 拓展实验：通过循环处理文件中的多条记录

1. 编写程序 `c5-3-1.c`，程序所实现的功能是：学生成绩管理系统 (V1.0 版)

文件 `score.txt` 中存放若干同学的学号及高数、英语、C 语言 3 门课的成绩，格式如下：

```

1001  90  80  70
1002  85  78  80
1003  60  70  76
.....

```

要求：

- ① 程序从 `score.txt` 中逐个读入每个同学的成绩；
- ② 计算平均分，在屏幕上输出结果并写入文件 `score_avg.txt` 中。
- ③ 程序运行后，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
1002	85	78	80	81.00
.....				

编程提示：

- 当文件中记录个数未知时，往往用函数 `feof(fp)` 判别是否已经读到文件结尾。当读取还没有到达文件尾部，`feof(fp)` 函数返回值为 0；到达文件尾部 `feof(fp)` 函数返回一个非 0 值。
- 例如：

```

while (!feof(fp))
{
    fscanf(fp, "%d%d%d%d", &num, &math, &eng, &computer);
    .....
}

```

2. 编写程序 c5-3-2.c, 程序所实现的功能是：随机红包发放。

要求：

- ① 键盘输入红包总金额和红包个数，程序产生随机红包；
- ② 将总金额、红包个数、所有的随机红包都保存到文件 red_packet.txt 中。

实验 6 循环结构程序设计—多重循环

一、实验目的

- (1) 掌握用 while 语句，do-while 语句和 for 语句实现多重循环的方法；
- (2) 掌握用循环结构程序设计实现一些常用算法（如穷举、迭代、递推等）。
- (3) 进一步学习调试程序的技巧。

二、实验内容和步骤

I. 基础部分：理解多重循环结构

- (1) 给定程序 c6-1-1.c, 请分析程序的运行结果，回答问题。

```
/*c6-1-1.c*/
#include <stdio.h>
int main()
{
    int i, j;
    i=1;                //i 赋初值 1
    for (j=1; j<=i; j++)
        printf("%2d *%2d =%2d    ", i, j, i*j);
    printf("\n");
    return 0;
}
```

- ① 阅读程序，画出该程序流程图；
- ② 调试运行程序；
- ③ 修改程序中给 i 赋值的语句，分别改为 i=3、i=5 和 i=9, 观察程序的运行结果；

- (2) 给定程序 c6-1-2.c, 请分析程序的运行结果，回答问题。

```
/*c6-1-2.c*/
#include <stdio.h>
int main()
{
    int i, j;
```

```

    for(i=1;i<=9;i++)
    {
        for ( j=1; j<=i; j++)
            printf("%2d *%2d =%2d    ", i, j, i*j );
        printf("\n");
    }
    return 0;
}

```

- ① 画出程序的流程图。
- ② 分析程序的运行结果, 比较该程序与 c6-1-1.c 的运行结果有何不同。
- ③ 程序中, 语句 printf("\n") 属于内循环体还是外循环体?
- ④ 将程序改为 c6-1-3.c, 分析程序的运行结果;

```

/* c6-1-3.c */
#include <stdio.h>
int main()
{
    int i, j;
    for(i=1;i<=9;i++)
    {
        for ( j=1; j<=i; j++)
        {
            printf("%2d *%2d =%2d    ", i, j, i*j );
            printf("\n");
        }
    }
    return 0;
}

```

- ⑤ 如果去掉程序中的一对 “{ }”, 将程序改为 c6-1-4.c, 分析程序的运行结果。

```

/*c6-1-4.c*/
#include <stdio.h>
int main()
{
    int i, j;
    for(i=1;i<=9;i++)
        for ( j=1; j<=i; j++)
            printf("%2d *%2d =%2d    ", i, j, i*j );
    printf("\n");
    return 0;
}

```

II.提高部分：掌握多重循环结构程序设计

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 c6-2-1.c 的功能是：输出如图 6-1 所示的图形。请完善程序。



图 6-1 程序 c6-2-1.c 的运行结果

```
/* c6-2-1.c */
#include <stdio.h>
int main()
{
    int i,j,k;
    /******found*****/
    for(i=0;i<=____ (1) ____ ;i++)
    {
        for( j=0; j<=10-i ;j++ ) printf(" ");
        /******found*****/
        for( k=0; ____ (2) ____ ;k++ ) printf("*");
        printf("\n");
    }
    return 0;
}
```

思考：如何输出下面的三种图形？

```
*****          *****          *
*****          *****          *****
*****          ***              *****
*****          *                *****
```

(2) 给定程序 c6-2-2.c 的功能是：求出 100~200 间的素数之和。程序运行结果如图 6-2 所示，请完善程序。(注意程序中 flag 变量的作用。)

图 6-2 程序 c6-2-2.c 的运行结果

```
/* c6-2-2.c */
#include <stdio.h>
#include <math.h>
int main()
```

```

{
    int i, j, flag, sum=0;
    for(i=100; i<=200; i++)
    {
        /*****found*****/
        flag=___ (1) ____;
        for( j=2; j<=i-1; j++)
        /*****found*****/
        if( ___ (2) ____ )
        {
            flag=1;
            break;
        }
        /*****found*****/
        if( ___ (3) ____ ) sum+=i;
    }
    printf("The sum is %d\n",sum);
    return 0;
}

```

2、程序改错

说明：程序中有错误，错误都在提示行： /*****found*****/的下面一行。请改正程序中的错误，编译运行使它能得出正确的结果。

注意：程序中的其它地方不要随意改动，不得增行或删行，也不得更改程序的结构！

(1) 给定程序 c6-2-3.c 的功能是： 已知一元以下的硬币中有一角、二角、五角三种面值，列举出将一元兑换成硬币的所有兑换方法。程序的运行结果如图 6-3 所示，请改正程序中的错误，并运行出正确的结果。

五角	二角	一角
0	0	10
0	1	8
0	2	6
0	3	4
0	4	2
0	5	0
1	0	5
1	1	3
1	2	1
2	0	0

图 6-3 程序 c6-2-3.c 的运行结果

```

/* c6-2-3.c */
#include <stdio.h>
#include <math.h>
int main()
{

```

```

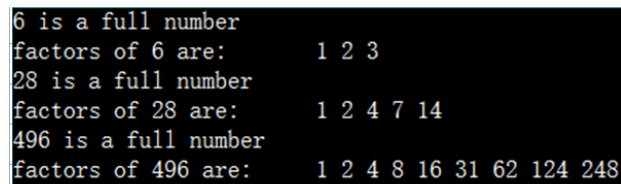
float i,j,k;
printf("\t 五角\t 二角\t 一角\n");
/*****found*****/
for(i=1;i<=2;i++)
    for(j=0;j<=5;j++)
        for(k=0;k<=10;k++)
        {
            /*****found*****/
            if(i*0.5+j*0.2+k*0.1 == 1.0 )
                printf("\t%.0f\t%.0f\t%.0f\n",i,j,k);
        }
return 0;
}

```

(2) 给定程序 c6-2-4.c 的功能是：输出 1~1000 之间的所有完数，并输出每个完数的所有因子。程序的运行结果如图 6-4 所示，请改正程序中的错误，并运行出正确的结果。

提示：所谓完数，就是指一个正整数，它的各因数之和等于其自身。

例如，28 的因数为 1、2、4、7、14， $28=1+2+4+7+14$ ，因此，28 是一个完数。



```

6 is a full number
factors of 6 are:      1 2 3
28 is a full number
factors of 28 are:     1 2 4 7 14
496 is a full number
factors of 496 are:    1 2 4 8 16 31 62 124 248

```

图 6-4 程序 c6-2-4.c 的运行结果

```

/*  c6-2-4.c */
#include <stdio.h>
int main()
{
    int i,j, t;
    for(i=1;i<=1000;i++)
    {
        t=0;
        for(j=1;j<i;j++)
            /*****found*****/
            if(i%j==0) t+=j;
        /*****found*****/
        if(t==i)
        {
            printf("\n%d is a full number\n",i);
            printf("factors of %d are:\t",i);
            /*****found*****/
            for(j=1;j<i;j++)
                if(i%j==0) printf("%d ",j);

```

```

    }
}
printf("\n");
return 0;
}

```

3、程序设计

(1) 编写程序 c6-2-5.c, 实现的功能为: 根据求和公式: $\text{sum} = 1 + \frac{1}{2!} + \frac{1}{3!} + \dots + \frac{1}{n!}$, 用双重循环, 计算 sum 的前 20 项的值。

编程提示:

- 定义一个变量 (假设为 sum) 存放最后的求和结果, sum 的数据类型应为实型, 定义变量 t 存放整数的阶乘。
- 程序的基本结构为:

```

    定义必要的变量;
    for (i=1, sum=0; i<=20; i++)
    {
        t 赋初值 1 ;
        for ( j=1; j<=i; j++ )
            变量 t 连乘求积;
        变量 sum 累加 t 的倒数;
    }

```

思考:

- ① 是否可以将“t 赋初值 1;”放在语句“for (i=1, sum=0; i<=20; i++)”之前?
- ② 内循环控制变量的终值 i, 可不可以改成 20?
- ③ 这个题目用单重循环也能实现。用单重循环和用双重循环的思想有什么区别?

(2) 编写程序 c6-2-6.c, 实现的功能为: 用多重循环编程实现打印出所有的“水仙花数”。所谓水仙花数是指一个 3 位数, 其各位数字的立方和等于该数本身。

如 $153 = 1^3 + 3^3 + 5^3$ 。

编程提示:

- 定义 3 个变量作为循环变量, 这些变量分别存放三位数的每位数字;
- 在循环体中将测试每种组合的值是否为水仙花数, 如果是则输出该数字组合, 否则不输出。
- 程序的基本结构为:

```

#include <stdio.h>
int main( )
{
    定义 4 个整型变量;
    for (a=1; a<=9; a++)
        for (b=0; b<=9; b++)

```

```

        for (c=0; c<=9; c++)
        {
            将这三个数 a, b, c 组成成一个百位数;
            if(满足水仙花数的条件)
                输出 a, b, c ;
        }
    }
}

```

(3) 编写程序 c6-2-7.c, 实现的功能为: 输入一个正整数, 将其逆序输出。

例如, 输入 12345, 输出 54321。

编程提示:

➤ 本题的关键是如何求得每一位数。参考过程如下:

- ① 假设变量 m=12345;
- ② 计算 $m\%10$, 可以得到个位数 5;
- ③ 计算 $m=m/10$, m 的新值为 1234;
- ④ 重复和第②、③步直到 m 的值为 0。

(4) 编写程序 c6-2-8.c, 实现的功能为: 打印数列 $2/1, 3/2, 5/3, 8/5, 13/8, 21/13, \dots$ 的前 20 项之和。

(5) 编写程序 c6-2-9.c, 实现的功能为: 已知在区间 $[0, 3]$ 上, 函数 $f(x)=x^3-x^2-1$ 有一个实根, 试用二分法求函数 $f(x)$ 的根。

编程提示:

➤ 二分法求根的思想是, 设 $f(x)$ 在区间 $[low, high]$ 上, 如果 $f(low)$ 与 $f(high)$ 异号, 那么根据介值定理, $[low, high]$ 之间必有根。参考过程如下:

- ① 取区间中点 $mid = (low+high)/2$;
- ② 如果 $f(mid)$ 与 $f(low)$ 异号, 那么根必存在于 $[low, mid]$ 之间; 反之, 如果 $f(mid)$ 与 $f(high)$ 异号, 那么根必存在于 $[mid, high]$ 之间;
- ③ 用 mid 更新 low 或 $high$, 从而使根所在的区间减半;
- ④ 重复第②、③步, 直至 $|high-low| < \varepsilon$ (ε 可以是 10^{-6}), 这时 mid 即为所求的根。

(6) 编写程序 c6-2-10.c, 实现的功能为: 解决趣味数学问题。有一长阶梯, 若每步跨 2 阶, 最后剩 1 阶; 若每步跨 3 阶, 最后剩 2 阶; 若每步跨 5 阶, 最后剩 4 阶; 若每步跨 6 阶, 最后剩 5 阶; 只有每步跨 7 阶, 最后才正好 1 阶不剩。问这条阶梯共有多少阶?

(7) 编写程序 c6-2-11.c, 实现的功能为: 输入一个正整数 n, 输出 n 的所有质因子。

例如 $n=13860$, 则输出: 2, 2, 3, 3, 5, 7, 11。

III. 拓展部分: 熟练使用多重循环结构程序设计解决问题

1. 编写程序 c6-3-1.c, 实现的功能是: 小学生 100 以内整数加减乘除运算训练。程序随机产生两个 100 以内的正整数 a 和 b 和一个运算符 (+、-、×、÷), 小学生计算并输入结果。

要求如下:

- ① 程序应该显示运算式子, 并给出小学生运算是否正确的判断信息;
- ② 可以进行多轮运算, 一轮中包括多道题 (例如 10 道题); 每出一题, 学生回答运算结果, 程序给出对错判别; 每轮运算完成后程序给出答题的总结果, 并询问是否进行下一轮的

训练，以决定是否继续；

- ③ 要保证减法运算时不出现负数，除法运算时被除数能整除除数；
- ④ 操作界面友好。

2. 编写程序 c6-2-2.c，用蒙特卡洛法求圆周率。

(1) 蒙特卡洛基本思想

蒙特卡罗 (Monte Carlo) 方法，又称随机抽样或统计试验方法，属于计算数学的一个分支，它是在上世纪四十年代中期为了适应当时原子能事业的发展而发展起来的。传统的经验方法由于不能逼近真实的物理过程，很难得到满意的结果，而蒙特卡罗方法由于能够真实地模拟实际物理过程，故解决问题与实际非常符合，可以得到很圆满的结果。这也是以概率和统计理论方法为基础的一种计算方法，是使用随机数（或更常见的伪随机数）来解决很多计算问题的方法。将所求解的问题同一定的概率模型相联系，用电子计算机实现统计模拟或抽样，以获得问题的近似解。为象征性地表明这一方法的概率统计特征，故借用赌城蒙特卡罗命名。

蒙特卡洛方法可以人为地构造出一个合适的概率模型，依照对该模型进行大量的统计实验，使它的某些统计参量正好是待求问题的解。

(2) 用蒙特卡洛方法求解圆周率

假设正方形内部有一个相切的圆，如图6-5：

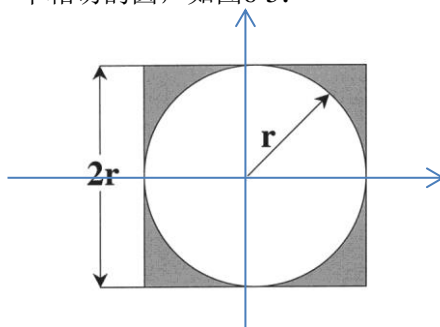


图 6-5 正方形及相切的圆

那么，圆面积和正方形面积的比值是 $\pi/4$ ：

$$\frac{\text{Area of Circle}}{\text{Area of Square}} = \frac{\pi r^2}{(2r)^2} = \frac{\pi}{4}$$

现在，在这个正方形内部，随机产生 n 个点（即 n 个坐标对 (x, y) ），当产生的点的次数足够多（也就是 n 足够大）以后，“落在圆内的点的次数/落在正方形内的点的次数”这个比值会无限接近“圆的面积/正方形的面积”这个比值，也就是圆周率的四分之一。模拟产生的点的次数越多，圆周率的近似结果越精确。

编程提示：

- 如何判断点是否落在圆内呢？只要计算它们与中心点的距离，就可以判断是否落在圆的内部。如果这些点均匀分布，那么圆内的点应该占到所有点的 $\pi/4$ ，因此将这个比值乘以4，就是 π 的值。
- 为了编程方便，将圆的中心点的坐标设为 $(0, 0)$ ，将半径 r 设为 1。

- 只要计算第一象限中的点的分布概率即可。