实验五　循环结构程序设计

一、实验目的

1. 掌握循环结构的执行过程。
2. 熟练掌握正确使用 while、do…while、for 循环语句和 break、continue 退出循环的方法。
3. 掌握循环结构的嵌套。

二、实验内容

1．针对求解 10 的阶乘，做如下练习。

（1）输入、调试并运行以下程序：

```
#include <stdio.h>
void main()
{   int i,t=1;
for(i=1;i<=10;i++)
     t=t*i;
printf("10!=%d",t);
}
```

（2）对上述程序做如下修改：

```
#include <stdio.h>
void main()
{   int i=1,t=1;
while (i<=10)
{   t*=i;
i++;
     }
printf("10!=%d",t);
}
```

（3）调试、运行该程序，并比较两种循环方法的区别和联系。

2．输出一张乘法口诀表。

要求：

（1）参考下面的程序编程完成其功能。

（2）查看将参考程序中第 5 行 for(y=1;y<=9;y++)改为 for(y=1;y<=x;y++)后再运行结果有什么不同？

【操作提示】参考程序如下：

```
#include <stdio.h>
void main()
{   int x,y;
for(x=1;x<=9;x++)
```

```
    {
for(y=1;y<=9;y++)
printf(" %3d*%d=%2d",x,y,x*y);
printf("\n");
    }
}
```

3．下面程序的功能是：计算 x 以内最大的 5 个能被 12 或 15 整除的自然数之和，x 的值由键盘输入。调试并改正下面程序中的错误。

【操作提示】参考程序如下：

```
#include <stdio.h>
main()
{   int x,m=0,n=0;
printf("x=");
scanf("%d",&x);
while(n<5)
{   if(x%12=0|| x%15=0)
{   m+=x;
n++;
x--;
        }
printf("sum=%d",m);
}
```

4．完善下面的程序。该程序用 $\frac{\pi}{4}\approx 1-\frac{1}{3}+\frac{1}{5}-\frac{1}{7}+\ldots$ 公式求 π 的近似值，直到某一项的绝对值小于 10^{-6} 为止（该项不累加），并计算循环次数。

```
#include <stdio.h>
#include <math.h>
main()
{
    int sign=1,count=0;                  //sign 表示当前项的符号，count 表示循环次数
    double pi=0,n=1.0,term=1.0;          //pi 表示多项式的值，term 表示当前项的值
    while(fabs(term)>=1e-6)
    {
        _____;
        pi=_____;
        _____;
        sign=_____;
        term=_____;
    }
    pi=pi*4;
    printf("pi=%10.6f\n",pi);
```

```
    printf("count=%d\n",count);
}
```

【操作提示】

（1）首先需要仔细观察，找出多项式的规律：①每项的分子都是 1；②后一项的分母是前一项分母加 2；③第一项的符号为正，从第 2 项起，每一项的符号与前一项的符号相反。

（2）在 C 库函数中，有两个求绝对值的函数，一个是 abs(x),求整数 x 的绝对值，结果是整型。另一个是 fabs(x),求双精度数 x 的绝对值，得到的结果是双精度型。本题中求 term 的绝对值，term 是双精度数，因此需要用 fabs(x)函数。在用数学函数时，需要在文件的开头加预处理指令：#include <math.h>。

5．编程实现以下功能：

（1）计算 s=1-3+5-7+9…-99+101。

（2）计算 s=3！+6！+9！+…+18！。

【操作提示】

①注意表达式求值的取值范围已经超出 int 型的上限，需要定义为 double 型才能得出正确结果。

②本题答案为 6403681859461206。

（3）输出前 30 个素数，每行输出 6 个。

（4）输入两个正整数 m 和 n，求其最大公约数和最小公倍数。

【操作提示】程序流程图见图 5-1。

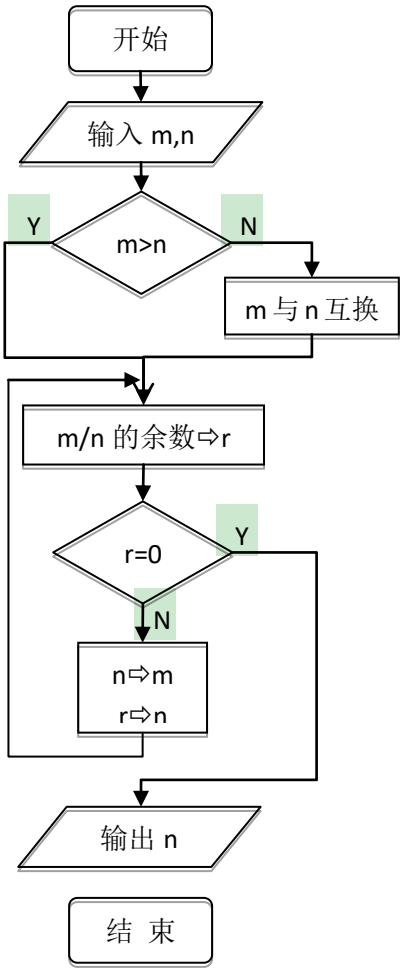

图 5-1

（5）输出以下图案：

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

【操作提示】

根据图案中字符和空格的分布找出规律，可以将图形分成上下两部分打印。图形中每一行字符的起始位置因前面存在空格而不同，因此每一行都要先打印空格再打印字符。

上半部分每行图案的规律是：空格数随着行号 i 的变化分别递减，而字符数随着行号 i 的变化分别递增，找出代数规律，即：第 i 行（i 从 0 开始）的空格数为 3-i 个，字符数为 2i+1 的关系，每行结束还要输出换行符，部分参考代码如下：

```
inti,j,k;
    for(i=0;i<=3;i++)
    {
        for(j=0;j<3-i;j++)
            printf(" ");
```

```
            for(k=0;k<2*i+1;k++)
                printf("*");
            printf("\n");
        }
```

下半部分图案同理。

（6）用二分法求方程 $2x^3-4x^2+3x-6=0$ 在（-10，10）之间的根。

【操作提示】

二分法的思路如下：先指定一个区间 $[x_1,x_2]$，如果函数 $f(x)$ 在此区间是单调变化，可以根据 $f(x_1)$ 和 $f(x_2)$ 是否同符号来确定方程 $f(x)=0$ 在 $[x_1,x_2]$ 区间是否有一个实根。如果 $f(x_1)$ 和 $f(x_2)$ 符号不同，则 $f(x)=0$ 在 $[x_1,x_2]$ 区间必有一个（且只有一个）实根；如果 $f(x_1)$ 和 $f(x_2)$ 符号相同，说明在 $[x_1,x_2]$ 区间无实根，要重新改变 x_1 和 x_2 的值。当确定 $[x_1,x_2]$ 有一个实根后，采取二分法将 $[x_1,x_2]$ 区间一分为二，再判断在哪一个小区间中有实根。如此循环进行下去，直到小区间足够小为止，如图 5-2 所示。

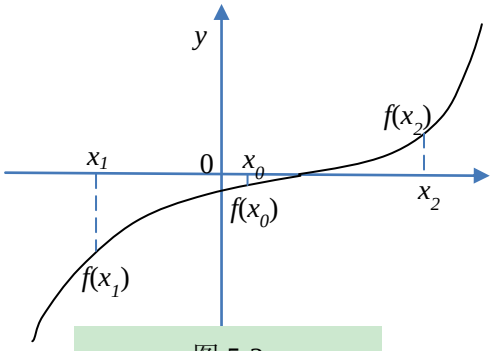

图 5-2

算法步骤如下：

（1）输入 x_1 和 x_2 的值。

（2）求出 $f(x_1)$ 和 $f(x_2)$。

（3）如果 $f(x_1)$ 和 $f(x_2)$ 同符号，说明在 $[x_1,x_2]$ 区间无实根，返回（1），重新输入 x_1 和 x_2 的值；若 $f(x_1)$ 和 $f(x_2)$ 不同符号，则在 $[x_1,x_2]$ 区间必有一实根，执行（4）。

（4）求 x_1 和 x_2 间的中点 $x_0=\dfrac{x_1+x_2}{2}$。

（5）求出 $f(x_0)$。

（6）判断 $f(x_0)$ 和 $f(x_1)$ 是否同符号。

①如果同符号，则应在 $[x_0,x_2]$ 中去找根，此时 x_1 已不起作用，用 x_0 代替 x_1，用 $f(x_0)$ 代替 $f(x_1)$。

②如果不同符号，则应在 $[x_1,x_0]$ 中去找根，此时 x_2 已不起作用，用 x_0 代替 x_2，用 $f(x_0)$

代替 $f(x_2)$。

（7）判断 $f(x_0)$ 的绝对值是否小于某一个指定的值（如 10^{-6}）。若不小于，就返回（4），重复执行（4）~（6）；若小于，则执行（8）。

（8）输出 x_0 的值， x_0 就是所求出的近似根。