第五章 循环程序结构设计

郎大鹏

第五章 循环程序结构设计

第五章 循环程序结构设计

5.1 引言

例子：使用顺序结构程序可以求解一元方程ax+b=0的实根。

```
#include <stdio.h>
void main()
{
      float a,b,x;
      scanf("%f%f",&a,&b);
      x=-b/a;
      printf("x=%f\n",x);
}
```

5.1 引言

例子：使用选择结构程序可以求解一元方程$ax^2+bx+c=0$的实根。

```
#include <stdio.h>
#include <math.h>
void main()
{
      float a,b,c,d,x1,x2;
      scanf("%f%f%f",&a,&b,&c);
      d=b*b-4*a*c;
      if(d>=0)
       { x1=(-b+sqrt(d))/(2*a);
         x2=(-b-sqrt(d))/(2*a);

    printf("x1=%f,x2=%f\n",x1,x2);
        }
      else
       printf("此方程没有实根!\n");
}
```

第五章 循环程序结构设计

5.2.1 while循环

while循环结构

while语句的一般形式为：

while(表达式)

语句

其中：

表达式：循环的条件。该表达式一般是关系表达式或逻辑表达式；但也可以是数值表达式、字符表达式，甚至可以是一个变量或常量。表达式的值为真（非0）时，执行循环体，表达式值为假（0）时，结束循环。

5.2.1 while循环

while循环的执行过程：

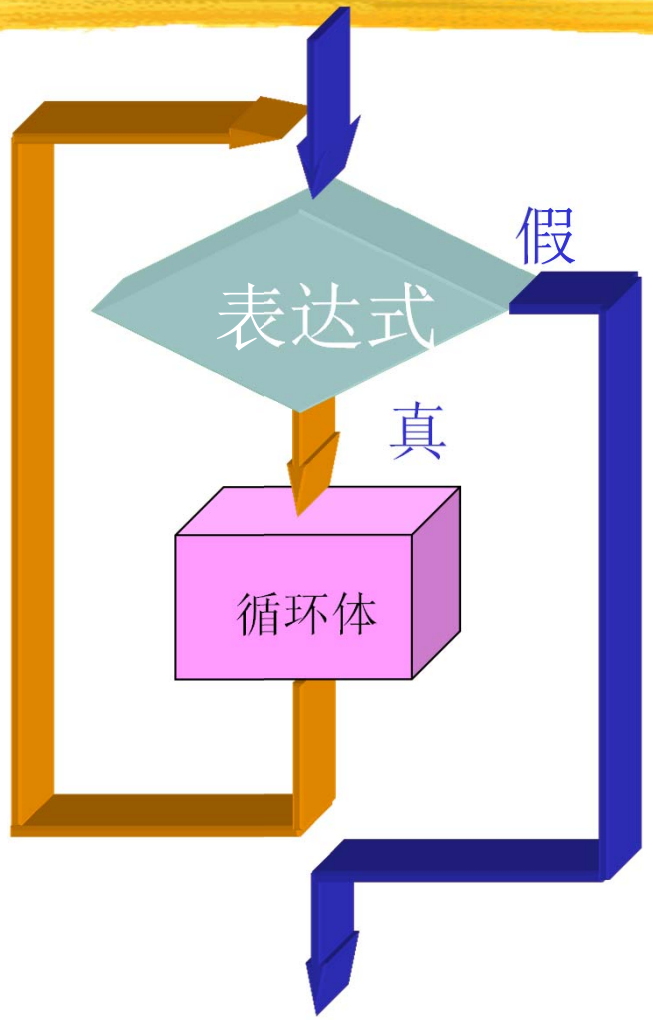
while循环的执行过程如右图所示。

（1）求解“表达式”的值

（2）当表达值为真（非0）时，执行（2）；表达式值为假（0）时，执行（4）；

（3）执行循环体，执行（2）；

（4）执行循环体下面的操作。



5.2.1 while循环

例5.1　从键盘输入一组学生成绩，若输入的成绩大于或等于0，将其累加到总成绩上，若成绩小于0，停止输入，然后计算并输出平均成绩。

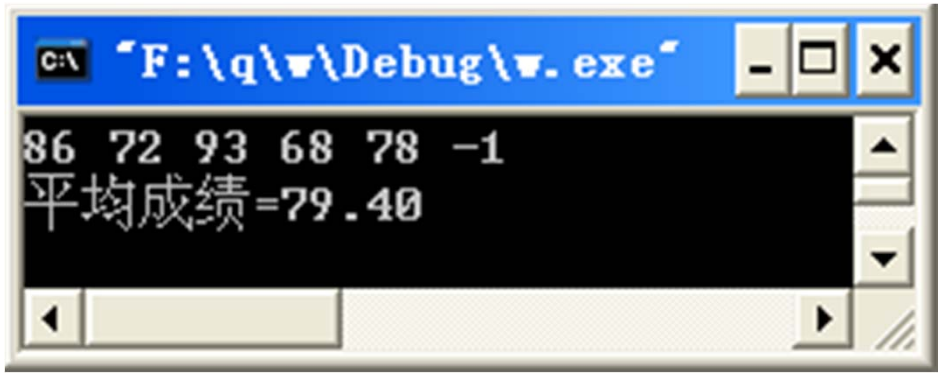
问题分析

这个问题的核心是多次读入数据、多次做加法，所以要循环结构实现。根据题干，循环执行的条件是输入的成绩大于或等于0；循环体包括三个操作：读入成绩、统计总成绩，统计成绩个数。

}

5.2.1 while循环

```
#include <stdio.h>
void main()
{int n=0;
 float m,sum=0,ave;
 scanf("%f",&m);
 while(m>=0)
 {sum=sum+m;
  n++;
  scanf("%f",&m);
 }
 ave=sum/n;
 printf("平均成绩=%0.2f\n",ave);
```



5.2.2 for循环

for循环结构

for语句的一般形式为：

for(表达式1；表达式2；表达式3)

　　语句

5.2.2 for循环

表达式1：设置初始条件，一般情况下为循环控制变量设置初值。该表达式只在进入循环前执行一次。它可以是一个逗号表达式，为多个变量设置初始值；也可以省略。如下

```
#include<stdio.h>
void main()
{int i,sum=0;
i=1;
for( ;i<=100;i++)
sum=sum+i;
printf("1+2+3+4+...+100=%d\n",sum);
}
```

5.2.2 for循环

表达式2：循环条件表达式，用来判断是否继续执行循环。表达式2一般是关系表达式或逻辑表达式；但也可以是数值表达式、字符表达式，甚至可以是一个变量或常量。表达式的值为真（非0）时，执行循环体，表达式值为假（0）时，结束循环。该表达式还可以省略，省略时直接执行循环体。

表达式3：每次执行完循环体后求解该表达式。可以将该表达式理解为循环体的最后一部分。它可以是一个逗号表达式，也可以将其移入循环体，省略该表达式。上例可改写为以下两种形式。

5.2.2 for循环

```
#include <stdio.h>
void main()
{int i,sum=0;
 for(i=1;i<=100; sum+=i, i++)
        ;
 printf("1+2+3+4+...+100=%d\n",sum);
}
```

```
#include <stdio.h>
void main()
{int i,sum=0;
 for(i=1;i<=100;)
        { sum+=i;
i++; }
 printf("1+2+3+4+...+100=%d\n",sum);
}
```

语句：循环体，可以是一条简单的语句，也可是以复合语句。当循环体由多条语句构成时，一定要写成复合语句。

}

5.2.2 for循环

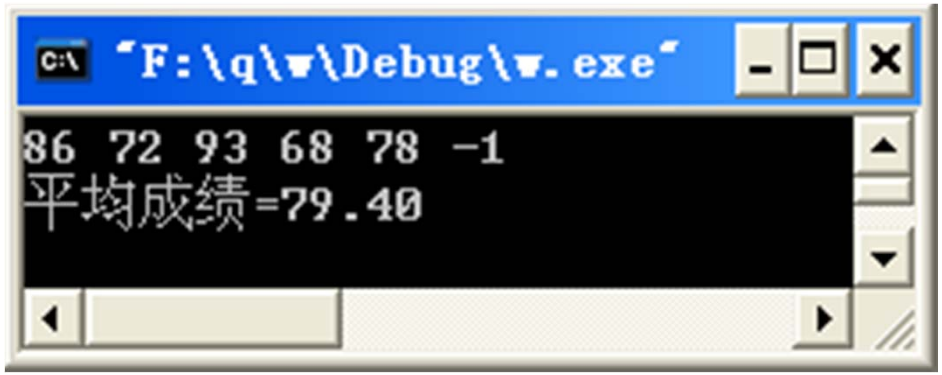
例5.3 从键盘输入10个整型数据，找出其中最大数和最小数。

```
#include "stdio.h"
void main()
{int i,max,min,n;
 for(i=1;i<=10;i++)
 {scanf("%d",&n);
  if(i==1)
  {max=n;
   min=n;}
  else
  {if(n>max)
     max=n;
   if(n<min)
min=n;
  }
 }
 printf("最大数: %d\n最小数: %d\n",max,min);
}
```

}

5.2.2 for循环

```
#include <stdio.h>
void main()
{int n=0;
 float m,sum=0,ave;
 scanf("%f",&m);
 while(m>=0)
 {sum=sum+m;
  n++;
  scanf("%f",&m);
 }
 ave=sum/n;
 printf("平均成绩=%0.2f\n",ave);
```



5.2.2 for循环

```
for (表达式1; 表达式2; 表达式3)
{
        语句;
  }
```

- 首先执行表达式1。如果表达式2的值为非0，就重复执行语句和表达式3，直到表达式2的值为0时止

相当于：

```
表达式1;
 while (表达式2)
   {
        语句;
        表达式3;
    }
```

- 表达式1和表达式3可以没有或者是用逗号分隔的多个表达式的组合。但最好不要有太多的表达式组合

5.2.2 for循环

例如：程序段

```
for(i=1;i<=100;i++)
        sum+=i;
```

```
i=2;
while(i<=100)
{sum+=i;
 i++;}
```

例如：程序段

```
while(m>=0)
 {sum=sum+m;
  n++;
  scanf("%f",&m);
 }
```

```
for( ;m>=0; )
 {sum=sum+m;
  n++;
scanf("%f",&m);
 }
```

5.2.3 do-while循环

Do-while循环结构

Do-while语句的一般形式为：

do

语句

while(表达式)

其中：

语句：循环体，可以是一条简单的语句，也可是以复合语句。当循环体由多条语句构成时，一定要写成复合语句。

表达式：循环的条件。该表达式一般是关系表达式或逻辑表达式；但也可以是数值表达式、字符表达式，甚至可以是一个变量或常量。表达式的值为真（非0）时，执行循环体，表达式值为假（0）时，结束循环。

5.2.3 do-while循环

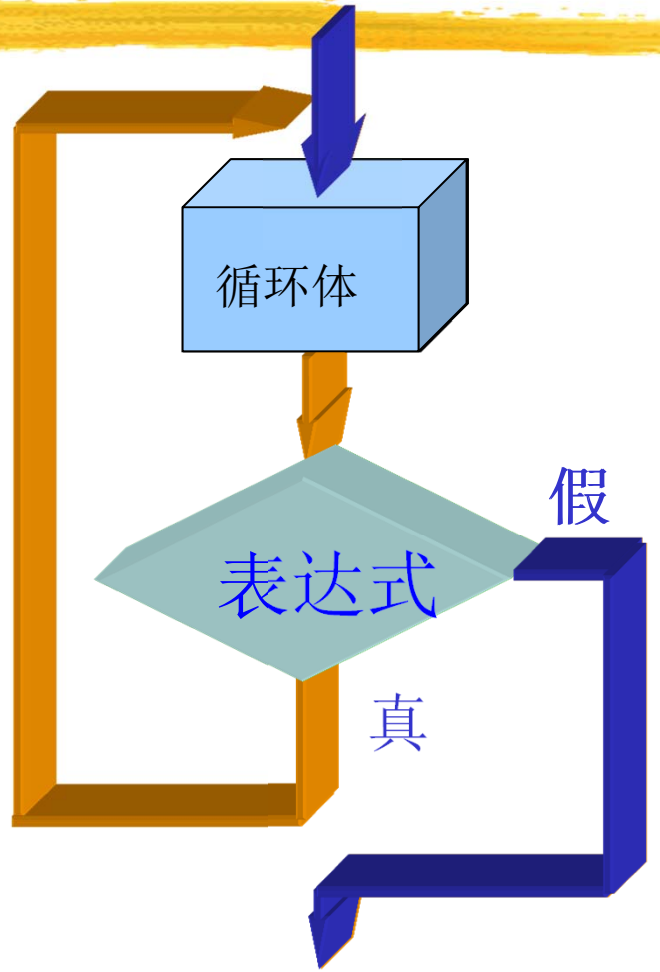
Do-while循环的执行过程：

do-while循环的执行过程如右图所示。

（1）执行循环体；

（2）求解“表达式”的值，若值为真（非0），执行（1）；若值为假（0），执行（3）；

（3）执行while语句下面的语句。



5.2.3 do-while循环

例5.4 计算一个小于30000的自然数各位数字的和值。如，输入1234，输出10。

问题分析

本问题的关键是如何求得一个自然数的每一位数字。一个自然数n与10的余数n%10是该数的个位；表达式n=n/10，可将n值缩小10倍，此时表达式n%10的值为原来n的十位。重复求解表达式n%10、n=n/10，可得到n的各位数字，此过程直到n的值为0时结束。这是一个循环过程，循环的条件是n!=0，循环体有三个操作：求n%10的值，将n%10的值累加到和变量上，n=n/10。

}

5.2.3 do-while循环

```
#include <stdio.h>
void main()
{int n,k,sum=0;
 scanf("%d",&n);
 printf("%d",n);
 do
 {k=n%10;
  sum=sum+k;
  n=n/10;
 }while(n!=0);
 printf(" 的各位数字之和是：%d\n",sum);
}
```

图5.8　例5.4运行结果

5.2.4 循环嵌套

实例解析

例5.5 打印一张如下图所示的“九九表”。

```
1*1=1
1*2=2   2*2=4
1*3=3   2*3=6   3*3=9
1*4=4   2*4=8   3*4=12  4*4=16
1*5=5   2*5=10  3*5=15  4*5=20  5*5=25
1*6=6   2*6=12  3*6=18  4*6=24  5*6=30  6*6=36
1*7=7   2*7=14  3*7=21  4*7=28  5*7=35  6*7=42  7*7=49
1*8=8   2*8=16  3*8=24  4*8=32  5*8=40  6*8=48  7*8=56  8*8=64
1*9=9   2*9=18  3*9=27  4*9=36  5*9=45  6*9=54  7*9=63  8*9=72  9*9=81
```

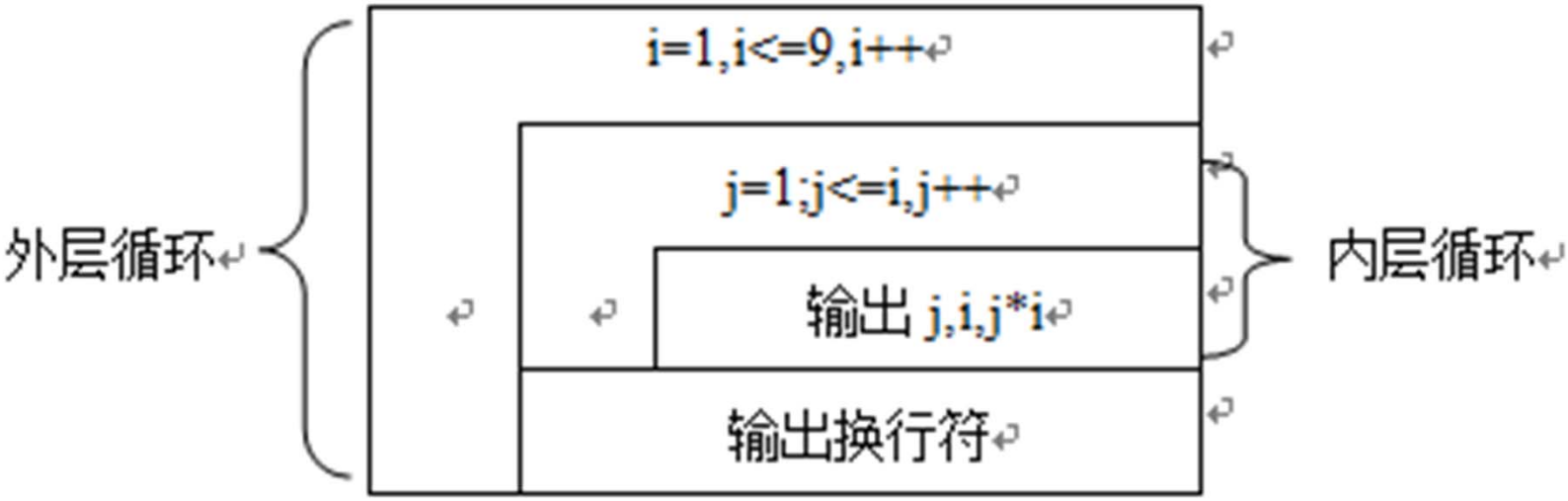
5.2.4 循环嵌套

问题分析

“九九表”的计算特别简单，不需要特别讨论。我们要分析的是“九九表”的形式。“九九表”共输出9行数据，用一个9次的循环控制完成。可以定义变量i来控制循环的执行，变量i的初始值为1，终值为9。每行要输出若干组数据和换行。每行数据的数量与行数有关，即第i行输出i组数据，这i组数据又可用另一个循环结构控制输出，每次输出一组数据。定义变量j控制每行数据的输出，j的初始值为1，终值为i。输出的每组数据又与变量i和j有关，每组数据由变量i、j及i*j构成。综上所述，本程序中用到的数据结构及算法如下。

数据结构：定义2个变量，变量i控制输出9行，变量j控制输出每行内的i个数据。

5.2.4 循环嵌套



5.2.4 循环嵌套

```
#include <stdio.h>
void main()
{int i,j;
 for(i=1;i<=9;i++)
 {for(j=1;j<=i;j++)
   printf("%d*%d=%-4d",j,i,i*j);
  printf("\n");
 }
}
```

5.2.4 循环嵌套

嵌套的概念

一个循环体内又包含另一个完整的循环结构，称为循环的嵌套。若内层循环又包含了一个完整的循环结构，构成了多重循环。

5.2.4 循环嵌套

例5.6 输出所有的“水仙花”数。“水仙花”数是一个三位数，它的每一位立方的和与自身相等。如：$153=1^3+3^3+5^3$，153是一个“水仙花”数。

问题分析

三位数的范围是100～999，若定义三个变量i、j、k，分别表示任意一个三位整数的百位、十位和个位，则i、j、k的取值分别是1～9、0～9和0～9。可分别计算由i、j、k构成的三位数和i、j、k的立方和，比较两个数之间的关系，输出“水仙花”数。

数据结构

定义五个变量，变量i、j、k的含义如下所述，变量m存储由i、j、k构成的三位数，变量n存储i、j、k三个数的立方和。

5.2.4 循环嵌套

```
#include <stdio.h>
void main()
{int i,j,k,m,n;
 for(i=1;i<=9;i++)
   for(j=0;j<=9;j++)
     for(k=0;k<=9;k++)
        {m=100*i+10*j+k;
         n=i*i*i+j*j*j+k*k*k;
           if(m==n)
                printf("%d\n",m);
         }
}
```

嵌套循环的执行过程：

（1）求解外层循环条件；

（2）值为真时，执行（3）；值为假时执行（4）

（3）执行一次外层循环体，当执行到内循环时，执行内循环，然后再执行外循环的其它语句；执行（1）；

（4）执行外层循环体后面的语句。

5.2.4 循环嵌套

循环嵌套的注意事项

（1）内、外层循环不能交叉，外层循环必须完整包含内层循环。

（2）若内、外层循环均为for循环，循环控制变量不能同名。

（3）三种循环可以相互嵌套。

第五章 循环程序结构设计

5.3.1 几种循环的比较

（1）三种循环的功能等价，一般情况下可以相互替代。

（2）各种循环都必须包含使循环趋于结束的操作，即有使循环条件在某个时刻变为假（0）的操作。使用while循环和do-while循环时，将这样的操作写在循环体中；使用for循环时，可将这样的操作写为表达式3。

（3）循环准备工作（如控制变量的初始化）的处理。使用while循环和do-while循环时，应将在准备工作在while循环和do-while语句之前完成；使用for循环时，可将准备工作写为表达式1。

5.3.2 提前终止循环break语句

正常情况下，只有当循环条件为假（0）时，循环才能结束，但有时会根据某一条件（不是循环条件）提前结束循环。

例5.7 输入一个大于1的自然数，判断其是否为素数。

问题分析

所谓“素数”（又称“质数”）是指只能被1和自身整除的数，如2、3、5、7、11等。一般根据素数的概念判定一个数是否为素数。对于自然数n，除1和n以外能整除n的数i一定在[2，n/2]中间，所以可用[2，n/2]中间的数依次除n，没有能整除n的，n为素数；若中间某个数能整除n，则足以说明n不是素数，不必再用后面的数除n。

5.3.2 提前终止循环break语句

```
#include <stdio.h>
void main()
{int i,n;
 scanf("%d",&n);
 for(i=2;i<=n/2;i++)
   if(n%i==0) break;
 if(i>n/2)
    printf("%n是素数!\n");
 else
    printf("%n不是素数!\n");
}
```

图5.14　例5.7运行结果

5.3.2 提前终止循环break语句

break语句的一般形式为：

break；

break语句的作用是使程序流程从循环体内跳到循环体外，继续执行循环体下面的语句。通常用来提前结束循环（正常情况下，循环在循环条件表达式的值为假时结束），是循环的非正常出口。

5.3.2 提前终止循环break语句

思考，下面的程序输出什么结果？

```
#include <stdio.h>
#include <string.h>
void main()
{ int i,j;
 for(i=1;i<5;i++)
  { for(j=1;j<5;j++)
   { if(j>i)  break;
    printf("%3d",i*j);
   }
   printf("\n");
  }
}
```

5.3.3 提前结束本次循环continue语句

例5.8 读入10个大于2的整数，统计其中素数的个数。

问题分析

这是一个典型的循环问题，循环次数确定，循环体的操作有两部分：读入数据、判断是否为素数。在自然数中，偶数肯定不是素数，所以当读入的是偶数时，不必判断是否为素数，可直接进入下次循环，读入下一数据。需要解决的新问题是如何只执行循环体的一部分，提前结束本次循环。C语言使用continue；语句实现该功能。

5.3.3 提前结束本次循环continue语句

```
#include <stdio.h>
void main()
{int n,i,k,m=0;
 for(i=1;i<=10;i++)
{    scanf("%d",&n);
     if(n%2==0) continue;
        k=2;
     while(n%k!=0&&k<=n/2)
         k++;
     if(k>n/2)
         m++;
     }
        printf("%d\n",m);
  }
```

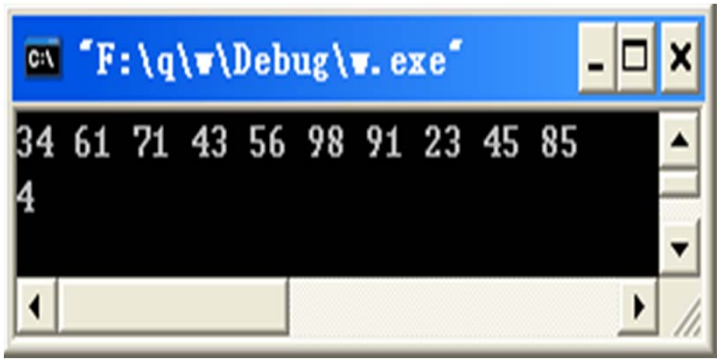


图5.16　例5.8运行结果

5.3.3 提前结束本次循环continue语句

说明：

continue语句使程序执行流程跳过循环体尚未执行的语句，直接执行循环控制语句，进入下一次循环。如果在while或do-while循环中使用continue，执行continue后，转去求解循环条件表达式；在for循环中使用continue，执行continue后，转去求解for语句中的表达式3，再求解表达式2。

5.4 程序举例

例5.9　计算2/1+3/2+5/3+8/5+…前20项和值。

问题分析：

计算多项式值是循环结构的典型应用。解决这类问题的关键是找出多项式的通项表达式。通项表达式一般是与前几项有关的函数，或是与项数有关的函数。

本例中多项式各项是分式，设通项为$t_i=a_i/b_i$。通过分析多项式各项，$a_i=a_{i-1}+b_{i-1}$，$b_i=a_{i-1}$，$t_i=(a_{i-1}+b_{i-1})/a_{i-1}=1+b_{i-1}/a_{i-1}=1+1/t_{i-1}$，找到了通项表达式。

5.4 程序举例

```
#include <stdio.h>
void main()
{int i;
 float t=2,s=0;
 for(i=1;i<=20;i++)
    { s=s+t;
      t=1+1/t;
    }
 printf("%.2f\n",s);
}
```

0=>s,2=>t	
i=1,i<=20,i++	
	s+t=>s
	1+1/t =>t
输出s	

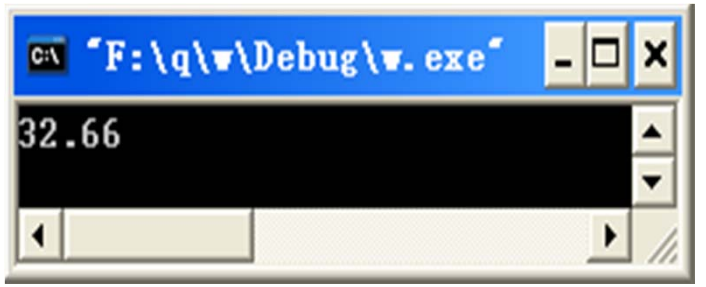


图5.18　例5.9运行结果

5.4 程序举例

例5.10　求两数的最大公约数。

问题分析

通常用欧几里德方法（又称“辗转相除”法）求解两个数的最大公约数。欧几里德方法的基本思想如下。

设两个数分别为a和b。

① 计算a和b的余数r;

② 若r=0，执行④；否则执行③;

③ b=>a, r=>b, 再计算a和b的余数，执行②;

④ b是最大公约数。

求解最大公约数是一个循环过程，循环条件是余数r不为0；循环体包含三个操作：b=>a, r=>b, 计算a和b的余数。

数据结构

定义三个变量。变量a、b分别存储两个原始数据，变量r存储余数。

}

5.4 程序举例

```
#include "stdio.h"
void main()
{int a,b,r;
 scanf("%d%d",&a,&b);
 printf("%d和%d的最大公约是：",a,b);
 r=a%b;
 while(r!=0)
    {a=b;
      b=r;
     r=a%b;
     }
 printf("%d\n",b);
}
```

输入a、b
a%b=>r
当r<>0
b=>a
r=>b
a%b=>r
输出b

图5.20　例5.10运行结果

5.4 程序举例

例5.11 打印输出Fibonacci数列的前20项，每行输出5个数据。Fibonacci数列的通项如下:

$$f_n=\begin{cases}1 & n=1、2\\ f_{n-1}+f_{n-2} & n>=3\end{cases}$$

问题分析：

循环体可分为三部分：计算、输出、为下一次循环做准备。计算主要是根据前两项计算当前项。输出部分首先输出当前项，然后控制换行操作。如果当前计算、输出的是第i项，其前两项分别是第i-1项和i-2项；当进入一下次循环时，计算、输出的是第i+1项，它的前两项分别是第i项和第i-1项，在计算第i+1项前，应先将第i项和第i-1项的值存入表示前两项的变量中，这是计算、输出第i+1项前的准备工作。

5.4 程序举例

数据结构：定义四个变量。变量i用于控制循环；变量f表示当前要计算、输出的项；变量f1表示前一项；变量f2表示前两项。

```
#include "stdio.h"
void main()
{int f,f1=1,f2=1,i;
 printf("%6d%6d",f1,f2);
 for(i=3;i<=20;i++)
 {  f=f1+f2;
    printf("%6d",f);
    if(i%5==0) printf("\n");
    f2=f1;
    f1=f;
 }
}
```

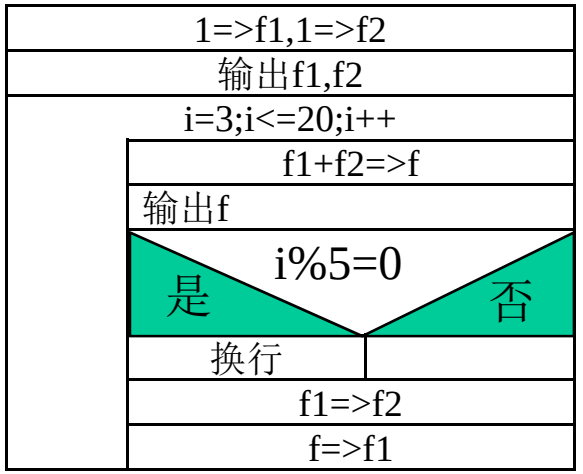


图5.22　例5.11运行结果

5.4 程序举例

例5.12　求解方程$2x^3-4x^2+3x-6=0$在1.5附近的实根

问题分析

一元方程的根的分布是很复杂的，很难找出它们的解析表达式。通常用数值分析方法求解一元方程的近似根。这类方法有迭代法、牛顿迭代法及二分法等。本例主要介绍“牛顿迭代法”。

“牛顿迭代法”又称为“牛顿切线法”，是一种收敛速度比较快的数值计算方法，其原理如右图：

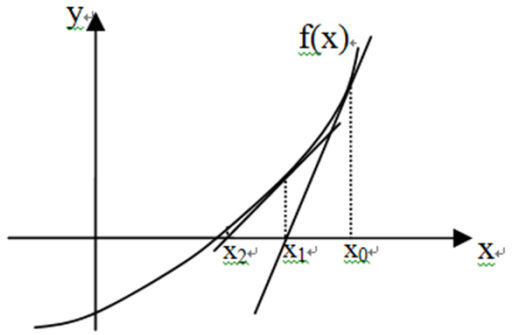


图 5.23　牛顿迭代法

5.4 程序举例

```
#include <stdio.h>
void  main()
{ double x1,x0,f,f1;
  x1=1.5;
  do
   {x0=x1;
    f=((2*x0-4)*x0+3)*x0-6;
    f1=(6*x0-8)*x0+3;
    x1=x0-f/f1;
    }while(fabs(x1-x0)>=1e-5);
   printf("The root of equation
is 5.2f\n",x1);
}
```

1.5=>x1	
	X1=>x0
	f(x0)=>f
	f'(x0)=>f1
	x1=x0-f/f1
当丨x1-x0|<10^{-5}	
输出x1	

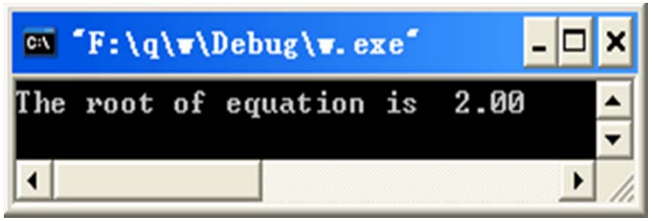


图5.25　例5.12运行结果

5.4 程序举例

例5.13 打印输出下面的图形，行数由键盘输入（行数<=9）。

先分析图形的特点：①图形的行数(n)，②每的字符数，③每字符的起始位置。本例图形有n行，第i行有2*i-1个字符，第i行字符的起始位置是n-i+1（即每行前有n-i个空格）。每行字符的变化有规律：12…i…21。

因此：本例的外层循环体可由4部分构成：①输出空格，②输出字符1、2、…、i，③输出字符i-1、i-2、…、1,④输出换行符。其中前3部分使用循环控制完成。

```
    1
   121
  12321
 1234321
123454321
……
```

5.4 程序举例

```
#include <stdio.h>
 void  main()
 {   int i,j,n;
     scanf("%d",&n);
   for(i=1;i<=n;i++)
   {for(j=1;j<=n-i;j++)
      printf(" ");
    for(j=1;j<=i;j++)
      printf("%d",j);
    for(j=i-1;j>=1;j--)
      printf("%d",j);
    printf("\n");
   }
}
```

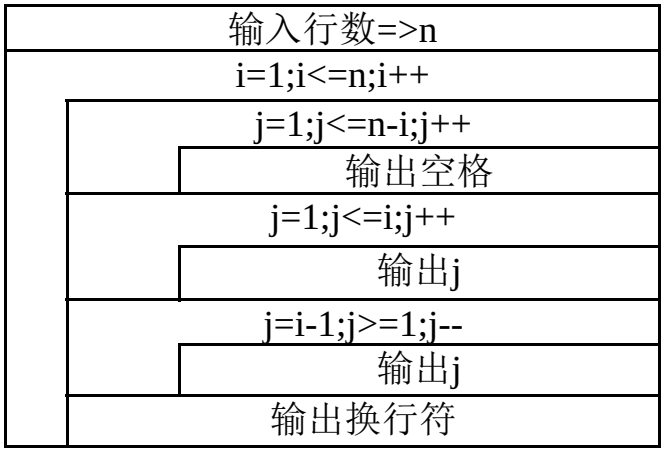


图5.26　例5.13框图

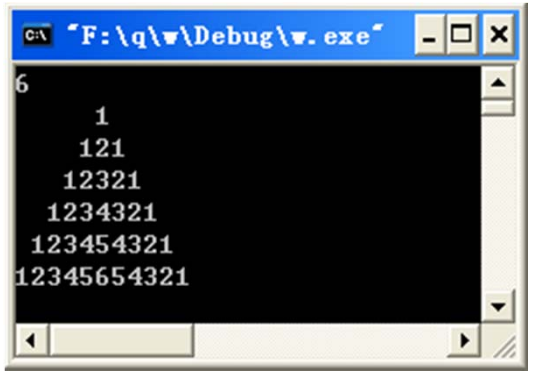


图5.27　例5.13运行结果

第五章 循环程序结构设计

5.5 习题

一、单选题

1．for语句中的表达式可以部分或全部省略，但两个_____不可省略。但当三个表达式均省略后，因缺少条件判断，循环会无限制地执行下去，形成死循环。

A）0　　B）1

C）；　　D），

2．程序段如下int k=-20; while(k=0) k=k+1;

则以下说法中正确的是_____。

A）while循环执行20次　　B）循环是无限循环

C）循环体语句一次也不执行　　D）循环体语句执行一次

3．程序段如下int k=0; while(k++<=2) printf("%d\n",k);

则执行结果是_____。

A）1　　B）2

C）0　　D）无结果

5.5 习题

4．在C语言的循环语句for,while,do-while中，用于直接中断循环的语句是_____。

A）swich　　B）continue

C）break　　D）if

5．循环语句中的for语句,其一般形式如下:for(表达式1;表达式2;表达式3) 语句

其中表示循环条件的是_____。

A）表达式1　　B）表达式2

C）表达式3　　D）语句

6．以下能正确计算1×2×3×…×10的程序段是_____。

A）do {i=1;s=1; s=s*i; i++; } while(i<=10);

B）do {i=1;s=0; s=s*i; i++; } while(i<=10);

C）i=1;s=1; do {s=s*i; i++; } while(i<=10);

D）i=1;s=0; do {s=s*i; i++; } while(i<=10);

5.5 习题

7．不是无限循环的语句为_____。

A）for(y=0,x=1;x>++y;x=i++)i=x;　　B）for(; ;x++=i);

C）while(1){x++;} D）for(i=10; ;i--) sum+=i;

8．有以下程序段int n=0,p; do{scanf(“%d”,&p);n++;}while(p!=12345 && n<3);

此处do—while循环的结束条件是_____。

A）p的值不等于12345并且n的值小于3

B）p的值等于12345并且n的值大于等于3

C）p的值不等于12345或者n的值小于3

D）p的值等于12345或者n的值大于等于3

5.5 习题

9．下面不能连续输出k个星号的循环语句是_____。

A）for (w=k:w!=0;w--)printf("*");

B）w=k;　　while(w--!=0)printf("*"); w++;

C）w=k;　　do{w--;printf("*");}while(w!=0);

D）for (w=k;w;--w)printf("*");

10．以下循环的执行次数是_____。

```
int i,j;
for(i=0,j=1;i<=j+1;i+=2,j--)printf("%d"\n",i);}
```

A）3　B）2

C）1　D）0