第 3 章 程序控制结构

本章主要介绍实现结构化程序设计的 3 种基本结构以及实现这 3 种结构的相关语句，同时介绍 3 种结构的程序设计方法。

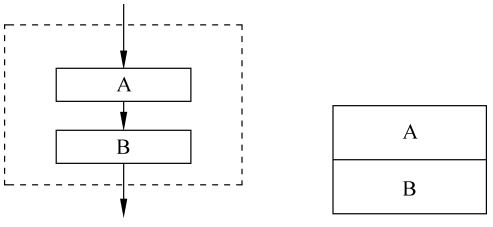
本章学习目标与要求

- 掌握顺序、选择和循环结构程序设计方法。
- 掌握 if 语句及 switch 语句的控制流程。
- 掌握利用 while 语句、do-while 语句及 for 语句进行循环程序设计。
- 理解 break 及 continue 语句对循环控制的影响。
- 熟悉多重循环的嵌套使用。

程序中语句的执行顺序称为“程序结构”。计算机程序是由若干条语句组成的语句序列。如果程序中的语句是按照书写顺序执行的，称为“顺序结构”；如果某些语句是按照当时的某个条件来决定是否执行，称为“选择结构”；如果某些语句要反复执行多次，则称为“循环结构”。

3.1 顺序结构程序设计

顺序结构的程序是自上而下顺序执行的各条语句。顺序结构的传统流程图和 N-S 流程图如图 3-1(a)和图 3-1(b)所示。



(a) 传统流程图　　(b) N-S流程图

图 3-1　顺序结构示意图

下面介绍几个顺序程序设计的例子。

【例 3-1】 输入三角形的三边长，求三角形面积。

已知三角形的三边长 a、b、c，则该三角形的面积公式为：$\text{area}=\sqrt{s(s-a)(s-b)(s-c)}$，其中，$s=(a+b+c)/2$。

源程序如下：

```
#include <stdio.h>
#include <math.h>
void main()
{
  float a,b,c,s,area;
  scanf("%f,%f,%f",&a,&b,&c);
  s=1.0/2*(a+b+c);
  area=sqrt(s*(s-a)*(s-b)*(s-c));
  printf("a=%7.2f,b=%7.2f,c=%7.2f,s=%7.2f\n",a,b,c,s);
  printf("area=%7.2f\n",area);
}
```

【例 3-2】 求 $ax^2+bx+c=0(a\neq 0)$方程的根，a、b、c 由键盘输入，设 $b^2-4ac>0$。

一元二次方程的求根公式为：

$$x_1=\frac{-b+\sqrt{b^2-4ac}}{2a},\quad x_2=\frac{-b-\sqrt{b^2-4ac}}{2a}$$

可以将上面的分式分为两项：

$$p=\frac{-b}{2a},\quad q=\frac{\sqrt{b^2-4ac}}{2a}$$

则 $x_1=p+q$，$x_2=p-q$。

源程序如下：

```
#include <stdio.h>
#include <math.h>
void main()
{
  float a,b,c,disc,x1,x2,p,q;
  scanf("a=%f,b=%f,c=%f",&a,&b,&c);
  disc=b*b-4*a*c;
  p=-b/(2*a);
q=sqrt(disc)/(2*a);
x1=p+q;x2=p-q;
  printf("\nx1=%5.2f\nx2=%5.2f\n",x1,x2);
}
```

3.2 选择结构程序设计

选择结构体现了程序的判断能力。在执行过程中，依据运行时某些变量的值确定某些操作是否执行，或者确定若干个操作中选择哪个操作执行，这种程序结构称为选择结构，又称为分支结构。选择结构有 3 种形式，即单分支结构、双分支结构和多分支结构。C 语言为 3 种选择结构提供了相应的语句，即 if-else 语句、switch 语句和条件运算符，下面介绍选择结构的实现方法。

3.2.1 if 语句的 3 种形式

1. 单分支结构

单分支结构的格式如下：

```
if(表达式)
  语句;
```

功能：计算表达式的值。如果条件为真(非0)则执行语句，否则不执行语句。

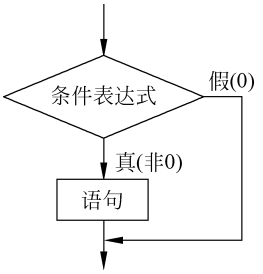


图 3-2　单分支选择结构的执行过程

说明：

(1) 表达式可为任何类型，常用的是关系表达式或逻辑表达式。

(2) 语句可以是任何可执行语句，可以是空语句或复合语句，也可以出现内嵌简单的if语句。

单分支结构的执行过程如图3-2所示。

【例3-3】 将两个整数a,b中的大数存入a中，小数存入b中。

分析：首先将a,b进行比较，如果a已经为大数则无须变动，否则，将两个数对调，即将a存入b中，将b存入a中。对调方法是设一个中间变量temp暂存数据，其操作步骤为：

(1) 将a赋给temp，语句为“temp=a;”。

(2) 将b赋给a，语句为“a=b;”。

(3) 将temp赋给b(原来a的值)，语句为“b=temp;”。

源程序如下：

```
#include <stdio.h>
void main()
{
  int a,b,temp;
  scanf("%d,%d",&a,&b);
  if(a<b)
  {
     temp=a;
     a=b;
     b=temp;
  }
    printf("a=%d,b=%d\n",a,b);
}
```

程序运行情况如下：

```
3,5↙
```

运行结果为：

```
a=5,b=3
```

说明：实现两个数对调，“temp=a; a=b; b=temp;”这3条语句都要执行，构成复合语句，因此要用{ }括起来。

2. 双分支结构

双分支结构的格式如下：

```
if(表达式)
```

```
  语句 1;
else
  语句 2;
```

功能：计算表达式的值，如果为真(非 0)，则执行语句 1，否则执行语句 2。其执行过程如图 3-3 所示。

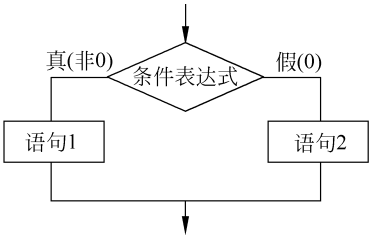


图 3-3　双分支选择结构执行过程

说明：

(1) 语句 1 和语句 2 可以是一条语句、复合语句或是内嵌 if 语句等，也可以是空语句。

(2) 表达式可以是任何类型，常用的是关系表达式或逻辑表达式。

(3) else 语句是 if 的子句，与 if 配对，不能单独出现。

(4) if-else 的配对原则是：else 语句总是与同一层最近的尚未配对的 if 语句配对。

【例 3-4】 输入一个英文字符，若是字母则输出“YES!”，否则输出“NO!”。

源程序如下：

```
#include <stdio.h>
void main()
{
  char c;
  scanf("%c",&c);
  if(c>='a'&&c<='z'||c>='A'&&c<='Z')
    printf("YES!\n");
  else printf("NO!\n");
}
```

程序运行情况如下：

```
X↙
```

输出结果为：

```
YES!
```

再运行一次，输入：

```
3↙
```

输出结果为：

```
NO!
```

请读者自行画出程序流程图。

3. 多分支结构

程序流程多于两个分支称为多分支，多分支程序结构使用嵌套的 if-else 语句实现。

格式如下：

```
if(表达式 1) 语句 1
```

```
else if(表达式 2)语句 2
     else if(表达式 3) 语句 3
        …
        else 语句 n
```

其含义是：依次判断表达式的值，当出现某个值为真时，则执行其对应的语句，然后跳到整个 if 语句之外继续执行后续程序。如果所有的表达式均为假，则执行语句 n。

嵌套的 if-else 语句，是指在语句体中内嵌 if 或 if-else 语句。两个分支都可以内嵌 if 语句，if-else 均内嵌分支结构的一般格式为：

```
if(表达式)
  if(表达式) 语句 1 ⎫
  else 语句 2       ⎭      内嵌 if - else
else                ⎫
  if(表达式) 语句 3 ⎭      内嵌 if - else
else 语句 4
```

【例 3-5】 求如下所示分段函数的 y 值。

$$y=\begin{cases}-1, & x<0\\ 0, & x=0\\ 1, & x>0\end{cases}$$

分析：y 的值存在 3 种可能，若 $x<0$，则 $y=-1$；否则，若 $x=0$，则 $y=0$；否则 $y=1$，内嵌一个双分支结构。

源程序如下：

```
#include <stdio.h>
void main()
{
  int x,y;
  scanf("%d",&x);
  if(x<0)
     y=-1;
  else                    //这个 else 子句是与上面的 if(x<0)配对
      if(x==0) y=0;
      else  y=1;          //这个 else 子句是与最近的 if(x==0)配对
  printf("\nx=%d,y=%d\n",x,y);
}
```

程序运行(3 次)情况如下：

```
5↙
x=5,y=1
0↙
x=0,y=0
-7↙
x=-7,y=-1
```

此程序是在 else 分支内嵌 if-else 语句的，也可以在另一分支嵌套，分支结构改写为如下所示：

```
if(x>=0)
   if(x>0) y=1;
   else y=0;          //与上面的 if(x>0)配对
else  y=-1;           //与最近的未被匹配的 if(x>=0)配对
```

说明：C 语言不限制嵌套层数。习惯在 else 分支语句上嵌套。如果在书写程序时不熟练，则将 if 和 else 的子句设计成复合语句，即用{ }括起来，这可保证 if 与 else 正确配对。

例如：

```
if(a>b)
{
   if(b<c)
     c=a;
}
else
   c=b;
```

3.2.2 条件运算符和条件表达式

条件运算符"?:"是 C 语言中唯一的三目运算符。它的特点是有三个操作数。实际上条件运算符实现了简单的 if-else 语句的功能。表达式形式如下：

(表达式 1)?(表达式 2):(表达式 3)

功能：先计算表达式 1，如果表达式 1 的值是非 0(真)，则取表达式 2 的值，否则，取表达式 3 的值。

例如，当 a=3，b=4 时：

```
max=(a<b)?a:b;
```

变量 max 取变量 a 的值，为 3。

说明：

(1) 条件运算符的优先级高于赋值运算符，但低于关系运算符和算术运算符。其结合方向为自右向左。若有以下表达式：

```
a>b?a:c>d?c:d
```

则等价于

```
a>b?a:(c>d?c:d)
```

(2) 三个表达式类型没有限制，互相可以不相同，此时表达式的值取较高的类型。例如：

```
a>b?2:5.5
```

如果 a<b，则条件表达式的值为 5.5；若 a>b，则条件表达式的值为 2.0 而不是 2。原因是 5.5 为浮点型，条件表达式的值应取较高的类型。

(3) 条件表达式可以作为函数参数，简化程序结构。例如：

```
printf("max of %d,%d is %d\n",a,b,(a>b)?a:b);
```


【例 3-6】 从键盘上输入一个字符,如果它是大写字母,则把它转换成小写字母输出;否则直接输出。

源程序如下:

```
#include<stdio.h>
void main()
{
   char ch;
   printf("Input a character: ");
   scanf("%c",&ch);
   ch=(ch>='A'&&ch<='Z')?(ch+32):ch;
   printf("%c\n",ch);
}
```

程序运行情况如下:

```
Input a character: D↙
d
```

3.2.3 switch 语句实现多分支选择结构

C 语言提供了 switch 语句直接处理多分支选择。虽然嵌套的 if 语句完全可以实现多分支选择的功能,但是嵌套的层数过多,程序的可读性降低。使用 switch 语句可使程序的结构清晰明了,减少一些嵌套错误。

switch 语句的一般格式如下:

```
switch(表达式)
{
    case  常量表达式1: 语句序列1 [break;]
    case  常量表达式2: 语句序列2 [break;]
      ⋮
    case  常量表达式n: 语句序列n [break;]
    [default : 语句序列n+1 ]
}
```

语句执行过程是:当表达式的值与某个 case 后面的常量表达式的值相等时,执行此 case 分支中的语句序列,如果此语句后有 break 语句,则跳出 switch 语句;如果没有 break 语句,则继续执行下一个 case 分支。若所有的 case 中的常量表达式的值都不能与表达式中的值相匹配,则执行 default 分支中的语句。

说明:

(1) ANSI 标准允许 switch 后的表达式和 case 后的常量表达式可以为整型、字符型和枚举型。但新的 ANSI 标准允许 switch 后面的表达式和 case 后面的常量表达式可以是任何类型的表达式而不再限于整型表达式。

(2) 各 case 后的常量表达式值必须互不相同。

(3) “语句组”可以是一条或多条合法的 C 语句。

【例 3-7】 从键盘上输入一个百分制成绩 score，按下列原则输出其等级：score≥90，等级为 A；80≤score＜90，等级为 B；70≤score＜80，等级为 C；60≤score＜70，等级为 D；score＜60，等级为 E。

源程序如下：

```
#include <stdio.h>
void main( )
{
  int score,grade;
  printf("Input a score(0～100):");
  scanf("%d",&score);
  grade = score/10;     //将成绩整除 10,转化成 switch 语句中的 case 标号
  switch(grade)
       {
           case 10:
           case 9: printf("grade = A\n");break;        //2 个分支同一操作
           case 8: printf("grade = B\n"); break;
           case 7: printf("grade = C\n"); break;
           case 6: printf("grade = D\n"); break;
           case 5:
           case 4:
           case 3:
           case 2:
           case 1:
           case 0:printf("grade = E\n"); break;        //6 个分支同一操作
           default:printf("Input error\n");
       }
}
```

3.3 循环结构程序设计

在给定条件成立时，反复执行某程序段，直到条件不成立为止。给定的条件称为循环条件，反复执行的程序段称为循环体。在 C 语言中可用以下语句实现循环：while 语句、do-while 语句、for 语句、goto 语句和语句标号。

循环三要素：循环变量和其初值；循环条件；循环变量的增值。

3.3.1 当型循环结构

执行特点是：先判断控制条件，如果条件满足则执行语句循环体，直到条件不成立退出循环。如果条件不满足则执行循环后面语句，如图 3-4 所示。

```
while(表达式)
    语句;
```

或

```
while(表达式)
  {
```

```
    语句;
}
```

功能：计算表达式值，其值若为真(非0)则反复执行语句，直到表达式的值为假时为止。

说明：

(1) 表达式可以是任何类型，常用的是关系表达式或逻辑表达式。

(2) 重复执行的操作称为循环体。

(3) 在循环体中还可以包含循环语句，构成多重循环。

【例3-8】 用while语句求sum=1+2+3+…+100。

N-S结构流程图如图3-5所示。

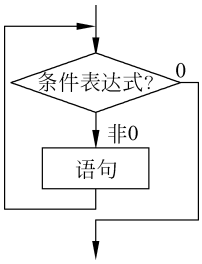


图3-4 当型循环结构

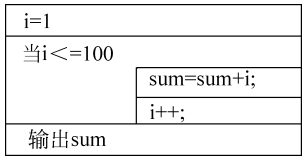


图3-5 例3-8的N-S结构流程图

算法分析：设变量i为加数，i在有规律地变化，自增1，变化的i累加到sum中(称累加器，不断累加加数)。这是一个重复运算问题，构成了循环结构。

其算法如下：

(1) 用while语句设计循环结构，用加数i作为循环控制变量，i的初值为1，终值为100，变化规律为i=i+1(或i++)。变量i满足了3个基本条件，即有一个明确的初值、明确的终值、明确的步长值。

(2) 累加器sum初值为0，sum随加数i而变化，sum=sum+i操作和i=i+1构成了循环体。

源程序如下：

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

运行结果为：

```
5050
```

3.3.2 直到型循环结构

执行特点是：先执行语句循环体，然后判断控制循环的条件。若条件成立，则继续执行循环体，直到条件不成立时，退出循环。

do-while 语句的一般形式为：

```
do{
    语句;
} while(表达式);
```

这个循环语句与 while 循环语句的不同在于：它先执行循环中的语句，然后再判断表达式是否为真，如果为真则继续循环；如果为假则终止循环。因此，do-while 循环语句至少要执行一次循环语句。其执行过程可用图 3-6 表示。

【例 3-9】 用 do-while 语句求 sum＝1＋2＋3＋…＋100。

其 N-S 结构流程图如图 3-7 所示。

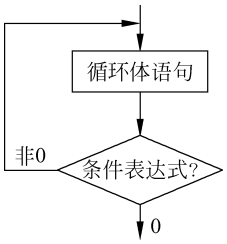


图 3-6 直到型循环结构执行过程

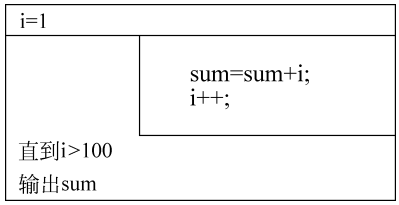


图 3-7 例 3-9 的 N-S 结构流程图

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

3.3.3 次数型循环结构

执行特点是：设计循环时，确定了循环体执行的次数，在执行循环过程中，根据控制变量的变化使程序完成反复操作。

在 C 语言中，for 语句使用最为灵活，它完全可以取代 while 语句。一般形式为：

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

它的执行过程如下：

(1) 求解表达式1。

(2) 求解表达式2,若其值为真(非0),则执行for语句中指定的内嵌语句,然后执行下面第(3)步;若其值为假(0),则结束循环,转到第(5)步。

(3) 求解表达式3。

(4) 转回第(2)步继续执行。

(5) 循环结束,执行for语句下面的一个语句。

其执行过程可用图3-8表示。

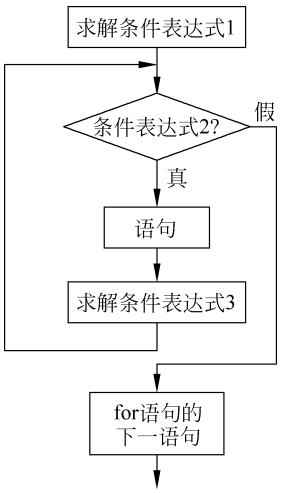


图3-8 次数型循环结构执行过程

for语句最简单的应用形式也是最容易理解的形式如下:

```
for(循环变量赋初值; 循环条件; 循环变量增量)
语句;
```

循环变量赋初值总是一个赋值语句,它用来给循环控制变量赋初值;循环条件是一个关系表达式,它决定什么时候退出循环;循环变量增量定义循环控制变量每循环一次后按什么方式变化。这三个部分之间用“;”分开。

例如:

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

先给i赋初值1,判断i是否小于或等于100,若是则执行语句,之后i值增加1。再重新判断,直到条件为假,即i>100时,结束循环。

对for语句说明如下:

(1) for循环中的“表达式1(循环变量赋初值)”“表达式2(循环条件)”和“表达式3(循环变量增量)”都是可选项,可以省略;但“;”不能省略。

(2) 省略了“表达式1(循环变量赋初值)”,表示不对循环控制变量赋初值。

(3) 省略了“表达式2(循环条件)”,则不做其他处理时便成为死循环。

例如:

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

相当于:

```
i = 1;
while(1)
    {sum = sum + i;
     i ++ ;}
```

(4) 省略“表达式3(循环变量增量)”,可在语句体中加入修改循环控制变量的语句。

例如:

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

(5) 可省略“表达式 1(循环变量赋初值)”和“表达式 3(循环变量增量)”。

例如：

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

相当于：

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

(6) 3 个表达式都可以省略。

例如：

```
for(;;)
语句;
```

相当于：

```
while(1)
语句;
```

(7) 表达式 1 可以是设置循环变量的初值的赋值表达式，也可以是其他表达式。

例如：

```
for(sum=0;i<=100;i++)
sum=sum+i;
```

(8) 表达式 1 和表达式 3 可以是一个简单表达式，也可以是逗号表达式。

```
for(sum=0,i=1;i<=100;i++) sum=sum+i;
```

或

```
for(i=0,j=100;i<=100;i++,j--)k=i+j;
```

(9) 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或字符表达式，只要其值非零，就执行循环体。

例如：

```
for(i=0;(c=getchar())!='\n';i+=c);
```

又如：

```
for(;(c=getchar())!='\n';)
printf("%c",c);
```

3.3.4 循环嵌套与多重循环结构

在一个循环的循环体内又包含另一个循环语句，称为循环嵌套结构。两层循环嵌套结

构称为双层循环结构。两层以上的嵌套结构则称为多重循环结构。在使用循环嵌套时,被嵌套的一定是一个完整的循环结构,即两个循环结构不能相互交叉。

【例 3-10】 在屏幕上打印一个 3 行 7 列的星号矩阵。

源程序如下:

```
#include <stdio.h>
void main()
{
  int i, k;
  for( i = 0; i < 3; i ++)
  {
    for( k = 0; k < 7; k ++)
       printf("*");
       printf("\n");
  }
}
```

运行结果为:

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

3.3.5 几种循环语句的比较

(1) 4 种循环语句都可以用来处理同一个问题,一般可以互相代替。但具体情况下有所侧重。for 语句简洁、清晰,它可将初始条件、判断条件和循环变量放在一行中书写,显得直观、明了,多用于处理初值、终值和步长值都明确的问题。while 语句多用于处理精确计算、利用终止标志控制循环的问题。一般不提倡用 goto 型循环。

(2) 用 while 和 do-while 循环时,循环变量初始化的操作应在 while 和 do-while 语句之前完成,而 for 语句可以在表达式 1 中实现循环变量的初始化。

(3) for 语句与 while 语句执行过程相同,先判断条件后执行循环体; do-while 语句执行循环体后判断循环条件,无论条件是否满足都要执行一次循环体。

3.3.6 循环体内 break 语句和 continue 语句

C 程序的循环体内可以设定循环中断语句提前结束循环,也可以设定结束本次循环体的操作提前进入下一次循环操作,break 语句和 continue 语句就是专门用于循环体中的两条语句,可以实现这个功能。

1. break 语句

break 语句用于强制中断循环的执行,结束循环。break 语句的一般格式为:

```
break;
```

通常 break 语句总是与 if 语句连在一起使用,即满足条件时便跳出循环。

【例 3-11】 计算圆的半径 r 从 1 到 10 时的面积并输出,直到面积大于 100 为止。

源程序如下:

```
#include <stdio.h>
#define  PI  3.1415926
void main() {
   int r;  float area;
   for(r = 1;r < 10;r ++ )
  {  area = PI * r * r;
     if(area > 100)
       break;
     else
          printf("area = 5.2 % f",area);
   }
}
```

运行结果为：

```
area = 3.14
area = 12.57
area = 28.27
area = 50.27
area = 78.54
```

注意：

(1) 在双层循环和多层循环中，break 语句只向外跳一层。

(2) break 语句和 if-else 语句配合使用从而构成第二个结束条件。

2. continue 语句

continue 语句用于中断本次循环，提前进入下次循环。continue 语句的一般格式为：

```
continue;
```

说明：

(1) continue 语句只用在 for、while、do-while 等循环体中，通常与 if 条件语句一起使用，用来加速循环执行。

(2) 循环体中单独使用 continue 语句无意义。

【例 3-12】 输出 100～200 能被 3 整除的自然数。

算法分析：在判断整除操作中，如果不能被 3 整除，就转入执行控制变量的变化语句(n++)，本次循环体中余下的语句不再执行；100～200 的整数要逐个检测，其判断表达式为 n%3!=0。

源程序如下：

```
#include <stdio.h>
void main()
{ int n,i = 0;
  for(n = 100;n <= 200;n ++ )
{  if(n % 3!= 0)
       continue;
       printf(" % d  ",n);
       i ++ ;
       if(i % 10 == 0)              //每行显示 10 个数
```

```
            printf("\n"); }
}
```

运行结果为：

```
102  105  108  111  114  117  120  123  126  129
132  135  138  141  144  147  150  153  156  159
162  165  168  171  174  177  180  183  186  189
192  195  198
```


3.4 程序控制综合举例

【例 3-13】 用$\frac{\pi}{4}=1-\frac{1}{3}+\frac{1}{5}-\frac{1}{7}+\cdots$公式求 π 值，要求精度达到其最后一项的近似值的绝对值小于 10^{-6} 为止。

N-S 流程图如图 3-9 所示。

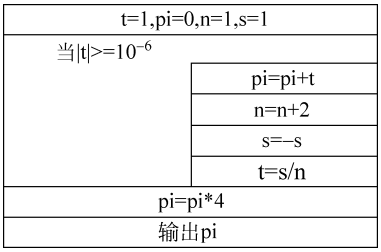


图 3-9 例 3-13 的 N-S 流程图

算法分析：最后一项用变量 t 表示，作为循环结束条件，调用系统函数求绝对值，条件表达式为：fabs(t)＞1e－6；分母 n 作为循环控制变量，步值为 n＋＝2，符号用 s 表示，则 t＝s/n；pi 为累加器，pi＝pi＋t。

源程序如下：

```
#include <stdio.h>
#include <math.h>
void main()
{
  int s;
  float n,t,pi;
  t = 1,pi = 0;n = 1.0;s = 1;
  while(fabs(t)> 1e - 6)
      {pi = pi + t;
       n = n + 2;
       s = - s;
       t = s/n;
      }
  pi = pi * 4;
  printf("pi = %10.6f\n",pi);
  }
```

运行结果为：

```
pi = 3.141594
```

【例 3-14】 求 Fibonacci 数列前 20 个数。这个数列的特点是：第 1、2 项均为 1，从第 3 项开始，该数是前两个数之和，即

$$f_1=1 \qquad (n=1)$$
$$f_2=1 \qquad (n=2)$$
$$f_n=f_{n-1}+f_{n-2} \qquad (n\geqslant 3)$$

算法分析：

(1) 根据题意已知第 1 个数为 $f_1=1$，第 2 个数为 $f_2=1$。通过 f_1 和 f_2 求出下一对数，即新的 f_1 和 f_2；计算公式是：$f_1=f_1+f_2$，$f_2=f_2+f_1$。已给出第 1 对数，只需再求 19 对即可。

(2) 只需定义 f_1，f_2 两个变量，以后求出的新数覆盖旧数。

源程序如下：

```
#include <stdio.h>
void main()
{ long int f1,f2;
   int i;
   f1 = 1,f2 = 1;
   for(i = 1;i <= 10;i ++ )
   {  printf(" %12ld %12ld\n",f1,f2);     //先输出,后求新的 f1,f2,否则会丢掉第 1 对数
      f1 = f1 + f2;
      f2 = f2 + f1;
   }
}
```

运行结果为：

```
   1            1
   2            3
   5            8
  13           21
  34           55
  89          144
 233          377
 610          987
1597         2584
4181         6765
```

【例 3-15】 设计循环嵌套结构，计算 100 元钱买 100 只鸡问题。公鸡 5 元 1 只，母鸡 3 元 1 只，小鸡 1 元 3 只，100 元钱买 100 只鸡，公鸡、母鸡、小鸡各能买多少只？

算法分析：设公鸡买 x 只，母鸡买 y 只，小鸡买 z 只，100 元钱最多买 20 只公鸡、33 只母鸡，小鸡：$z=100-x$(公鸡)$-y$(母鸡)。嵌套循环：公鸡 x 从 1～20；母鸡 y 从 1～33。条件：x * 5+y * 3+(100−x−y)/3==100&&(z%3==0)。

源程序如下：

```
#include <stdio.h>
void main()
{
  int x,y,z;
  for(x = 1;x <= 20;x ++ )            //公鸡最大数
     for(y = 1;y <= 33;y ++ )         //母鸡最大数
        {
          z = 100 - x - y;            //求小鸡数
          if((x * 5 + y * 3 + (100 - x - y)/3) == 100)&&(z % 3 == 0)
           printf("x = %d,y = %d,z = %d\n",x,y,z);
```

```
        }
    }
```

运行结果为：

```
x=4,y=18,z=78
x=8,y=11,z=81
x=12,y=4,z=84
```

【例 3-16】 求 100～200 的全部素数。

算法分析：如果 m 为素数，m 不能被 $2\sim\sqrt{m}$ 的任何整数整除。被测数 m 作为循环控制变量；将除数 i 作为内循环的循环控制变量试除 $2\sim k$（为 sqrt(m)）的每一个数；判断一个数是否被另一个数整除，可以使用求余运算符“%”，如果出现整除情况，则使用 break 语句提前结束内循环，说明 m 不是素数，此时 i 值小于 k；若未出现整除情况，循环正常结束，说明 m 为素数，则循环控制变量为 $i>k$。

源程序如下：

```
#include<math.h>
main()
{
  int m,i,k,n=0;
  for(m=101;m<=200;m=m+2)
  {
    k=sqrt(m);
    for(i=2;i<=k;i++)
    if(m%i==0) break;                          //若整除则结束内循环,说明 m 不是素数
    if(i>=k+1)                                 //i 如果超出 k,则为素数
    {printf("%d",m);
     n=n+1;}                                   //统计素数
    if(n%10==0) printf("\n");                  //每行输出 10 个素数
  }
  printf("\n");
}
```

运行结果为：

```
101103107109113127131137139149
151157163167173179181191193197
199
```

【实训 4】 多分支选择结构程序设计

一、实训目的

(1) 掌握多分支选择结构。

(2) 熟悉长整型数据的定义形式。

二、实训任务

企业发放奖金的根据是利润提成。利润(p)低于或等于 10 万元时，奖金可提成 10%；利润高于 10 万元，低于 20 万元时，高于 10 万元的部分，可提成 7.5%；利润在 20 万元到 40

万元之间时，高于 20 万元的部分，可提成 5%；利润在 40 万元到 60 万元之间时高于 40 万元的部分，可提成 3%；利润在 60 万元到 100 万元之间时，高于 60 万元的部分，可提成 1.5%；利润高于 100 万元时，超过 100 万元的部分按 1%提成。从键盘输入当月利润 p，求应发放奖金总数。

三、实训步骤

(1) 从键盘接收利润值 p，将奖金分段列出计算式。

(2) 利用多分支选择结构计算奖金数。

参考源程序 lab3_1.cpp 如下：

```
#include <stdio.h>
void main()
{
  long p;
  long bonus,bonus1,bonus2,bonus4,bonus6,bonus10;
  scanf("%ld",&p);
  bonus1 = 100000 * 0.1;
  bonus2 = bonus1 + 100000 * 0.075;
  bonus4 = bonus2 + 200000 * 0.05;
  bonus6 = bonus4 + 200000 * 0.03;
  bonus10 = bonus6 + 400000 * 0.015;
     if(p <= 100000)
        bonus = p * 0.1;
     else if(p <= 200000)
            bonus = bonus1 + (p - 100000) * 0.075;
          else if(p <= 400000)
                 bonus = bonus2 + (p - 200000) * 0.05;
               else if(p <= 600000)
                      bonus = bonus4 + (p - 400000) * 0.03;
                    else if(p <= 1000000)
                           bonus = bonus6 + (p - 600000) * 0.015;
                         else
                             bonus = bonus10 + (p - 1000000) * 0.01;
  printf("bonus = %ld",bonus);
}
```

运行结果为：

```
100000↙
bonus = 10000
```

思考：利用 switch 语句实现多分支选择结构。

【实训 5】 双重循环结构程序设计

一、实训目的

(1) 掌握利用双重循环结构编程的方法。

(2) 熟悉屏幕上输出格式的控制。

二、实训任务

实现在屏幕上输出下三角九九乘法表。

三、实训步骤

1. 定义控制行、列的变量。

2. 设计双层循环结构。

参考源程序 lab3_2.cpp 如下：

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

运行结果为：

```
1*1=1
2*1=22*2=4
3*1=33*2=63*3=9
4*1=44*2=84*3=124*4=16
5*1=55*2=105*3=155*4=205*5=25
6*1=66*2=126*3=186*4=246*5=306*6=36
7*1=77*2=147*3=217*4=287*5=357*6=427*7=49
8*1=88*2=168*3=248*4=328*5=408*6=488*7=568*8=64
9*1=99*2=189*3=279*4=369*5=459*6=549*7=639*8=729*9=81
```

【实训6】 多重循环结构程序设计

一、实训目的

掌握利用多重循环结构编程的方法。

二、实训任务

两个乒乓球队进行比赛，各出三人。甲队为a,b,c三人，乙队为x,y,z三人。已抽签决定比赛名单。有人向队员打听比赛的名单，a说他不和x比，c说他不和x,z比。请编程序找出两队参赛选手的对阵名单。

三、实训步骤

1. 定义a,b,c的各自对手变量i,j,k。

2. 设计三重循环结构。

参考源程序 lab3_3.cpp 如下：

```
#include<stdio.h>
void main()
{
  char i,j,k;/*i是a的对手,j是b的对手,k是c的对手*/
  for(i='x';i<='z';i++)
```

```
    for(j = 'x';j <= 'z';j ++ )
    { if(i!= j)
      for(k = 'x';k <= 'z';k ++ )
      { if(i!= k&&j!= k)
         if(i!= 'x'&&k!= 'x'&&k!= 'z')
        printf("order is a -- %c\tb -- %c\tc -- %c\n",i,j,k);
      }
    }
}
```

运行结果为：

```
order is a -- z        b -- x        c -- y
```

本章小结

本章对实现分支结构化程序设计的 if 语句、switch 语句和实现循环结构的三种语句进行了详细讲解，并通过实际的例子介绍了这些语句的使用方法，从而掌握一些程序设计的方法和常用的算法解决实际问题。

习　题　3

一、选择题

1. C 语言对嵌套 if 语句的规定是：else 总是与(　　)。

A. 其之前最近的 if 配对　　B. 第一个 if 配对

C. 缩进位置相同的 if 配对　　D. 其之前最近的且尚未配对的 if 配对

2. 对以下程序片段，下列说法正确的是(　　)。

```
#include <stdio.h>
void main( )
{   int x = 0,y = 0,z = 0;
    if(x = y + z)
     printf("***");
    else
     printf("###");
}
```

A. 有语法错误，不能通过编译

B. 输出：***

C. 可以编译，但不能通过连接，所以不能运行

D. 输出：###

3. 以下程序的输出结果是(　　)。

```
#include <stdio.h>
void main()
{ int x = 1,y = 0,a = 0,b = 0;
```

```
    switch(x) {
       case 1:switch (y) {
                case 0: a++ ; break;
                case 1: b++ ; break;
                }
       case 2:a++ ; b++ ; break;
       case 3:a++ ; b++ ;
}
  printf("a=%d,b=%d",a,b);
}
```

A. a=1,b=0　　B. a=2,b=1　　C. a=1,b=1　　D. a=2,b=2

4. 在下面的条件语句中(其中S1和S2表示C语言语句),(　　)在功能上与其他3个语句不等价。

A. if (a) S1; else S2;　　B. if (a==0) S2; else S1;

C. if (a!=0) S1; else S2;　　D. if (a==0) S1; else S2;

5. 以下for循环语句的执行次数是(　　)。

```
for(x=0,y=0; (y=123) && (x<4) ; x++ );
```

A. 无限循环　　B. 循环次数不定　　C. 4次　　D. 3次

6. 下面程序段的运行结果是(　　)。

```
x=y=0;
while (x<15) y++ ,x+=++y ;
printf("%d,%d",y,x);
```

A. 20,7　　B. 6,12　　C. 20,8　　D. 8,20

7. 以下是死循环的程序段是(　　)。

A.

```
for(i=1; ; ) {
if(i++ %2==0) continue;
if(i++ %3==0) break;
}
```

B.

```
i=32767;
do { if(i<0) break ; } while ( ++i);
```

C.

```
for(i=1 ; ;)  if( ++i<10) continue;
```

D.

```
i=1; while(i-- );
```

8. 关于以下for循环语句说法正确的是(　　)。

```
int i,k;
```

```
for (i = 0, k = -1; k = 1; i++, k++)
printf("***");
```

A. 判断循环结束的条件非法　　B. 是无限循环

C. 只循环一次　　D. 一次也不循环

9. 若有

```
int k = 2;
while (k = 0) {printf("%d",k);k--;}
```

则下面描述中正确的是(　　)。

A. while 循环执行 10 次　　B. 循环是无限循环

C. 循环题语句一次也不执行　　D. 循环体语句执行一次

10. 下面程序的功能是在输入的一批正数中求最大者,输入 0 结束循环,在横线上填入的正确语句是(　　)。

```
#include <stdio.h>
void main ( )
{ int a,max = 0;
   scanf("%d",&a);
   while(________) {
      if(max < a) max = a;
      scanf("%d",&a);
   }
   printf("%d",max);
}
```

A. a==0　　B. a　　C. !a==1　　D. !a

二、程序阅读题

1. 写出以下程序分别输入 1,2,3,4 后的运行结果。

```
#include <stdio.h>
void main()
{
    int c;
    while((c = getchar())!= '\n')
        switch(c - '2')
    {
        case 0:
        case 1:putchar(c + 4);
        case 2:putchar(c + 4);break;
        case 3:putchar(c + 3);
        default:putchar(c + 2);break;
    }
printf("\n");
}
```

2. 写出下面程序运行的结果。

```
#include <stdio.h>
```

```
void main()
{ int x,i;
   for(i=1; i<=100; i++) {
      x=i;
      if(++x%2==0)
         if(++x%3==0)
            if(++x%7==0)
               printf("%d",x);
   }
}
```

3. 写出下面程序运行的结果。

```
#include<stdio.h>
void main()
{ int i,b,k=0;
   for(i=1; i<=5; i++) {
       b=i%2;
       while(b--==0) k++;
   }
   printf("%d,%d",k,b);
}
```

4. 写出下面程序运行的结果。

```
#include<stdio.h>
void main()
{ int a,b;
   for (a=1,b=1; a<=100; a++) {
      if(b>=20) break;
      if(b%3==1) { b+=3; continue; }
      b-=5;
   }
   printf("%d\n",a);
}
```

5. 写出下面程序运行的结果。

```
#include<stdio.h>
void main()
{ int k=1,n=263;
   do { k*=n%10; n/=10; } while(n);
   printf("%d\n",k);
}
```

6. 写出下面程序运行的结果。

```
#include<stdio.h>
void main()
{ int i=5;
  do {
     switch(i%2) {
        case 4 : i--; break;
```

```
        case 6 : i--; continue;
      }
      i--; i--;
      printf("%d",i);
  }while(i>0);
}
```

7. 写出下面程序运行的结果。

```
#include<stdio.h>
void main()
{ int i,j;
   for(i=0;i<3;i++,i++) {
      for(j=4; j>=0; j--) {
        if((j+i)%2) {
           j--;
           printf("%d,",j);
           continue;
        }
        --i;
        j--;
        printf("%d,",j);
      }
   }
}
```

8. 写出下面程序运行的结果。

```
#include<stdio.h>
void main()
{ int a=10,y=0;
   do{
      a+=2; y+=a;
      if(y>50) break;
   } while(a=14);
   printf("a=%d y=%d\n",a,y);
}
```

9. 写出下面程序运行的结果。

```
#include<stdio.h>
void main()
{ int i,j,k=19;
   while(i=k-1) {
      k-=3;
      if (k%5==0) { i++; continue; }
      else if(k<5) break;
      i++;
   }
   printf("i=%d,k=%d\n",i,k);
}
```

10. 写出下面程序运行的结果。

```
#include <stdio.h>
void main()
{ int y = 2,a = 1;
   while(y-- !=-1)
      do {
         a* = y;
         a++;
      } while(y--);
   printf("%d,%d\n",a,y);
}
```

三、程序填空题

1. 以下程序输出 x,y,z 三个数中的最小值,请填空使程序完整。

```
#include <stdio.h>
main()
{ int x = 4,y = 5,z = 8;
  int u,v;
  u = x < y ?________;
  v = u < z ?________;
  printf("%d",v);
}
```

2. 下面程序的功能是输出 1～100 每位数的乘积大于每位数的和的数,请填空使程序完整。

```
#include <stdio.h>
void main()
{ int n,k = 1,s = 0,m;
  for (n = 1; n <= 100; n++) {
      k = 1; s = 0;
              ;
      while(________) {
          k* = m%10;
          s+= m%10;
          ________;
      }
      if(k > s) printf("%3d",n);
  }
}
```

3. 下面程序段的功能是计算 1000! 的末尾有多少个零,请填空使程序完整。

```
#include <stdio.h>
void main()
{
     int i,k,m;
     for(k = 0,i = 5; i <= 1000; i+= 5)
     { m = i;
        while(________) { k++; m = m/5; }
```

```
    }
}
```

4. 下面程序接收键盘上的输入,直到按 Enter 键为止,这些字符被原样输出,但若有连续一个以上的空格时只输出一个空格,请填空使程序完整。

```
#include <stdio.h>
void main()
{
 char cx, front = '\0';
 while((cx = getchar())!= '\n')
 {
  if(cx!= ' ') putchar(cx);
  if(cx == ' ')
  if(________)
     putchar(________);
  front = cx;
}
```

5. 要求在运行程序时输入数据 1,输出结果为 55(即 1～10 的和),请填空使程序完整。

```
#include <stdio.h>
void main()
{
int sum = 0,i;
scanf("%d",&i);
do
{ sum += i;
          ;
}while(________);
printf("%d",sum);
}
```

6. 程序的功能是输出 100 以内能被 3 整除的所有整数,请填空使程序完整。

```
#include <stdio.h>
void main()
{ int i;
for(i = 0;           ;i++)
{ if(           ) continue;
printf("%3d",i);
}
}
```

四、编程题

1. 给出一百分制成绩,要求输出成绩等级'A','B','C','D','E'。90 分以上为'A',80～89 分为'B',70～79 分为'C',60～69 分为'D',60 分以下为'E'。

2. 输入两个正整数 m 和 n,求其最大公约数和最小公倍数。

3. 写程序,判断某一年是否是闰年。

4. 求 1!+2!+3!+…+19!+20!。

5. 输入一行字符,分别统计出其中英文字母、空格、数字和其他字符的个数。

6. 打印出所有“水仙花数”,所谓“水仙花数”是指一个三位数,其各位数字立方和等于该本身。例如:153是一个水仙花数,因为$153=1^3+5^3+3^3$。

7. 一个数如果恰好等于它的因子之和,这个数就称为“完数”。例如,6的因子为1、2、3,而6=1+2+3,因此6是“完数”。编程序找出1000之内的所有完数,并按下面格式输出其因子:

```
6  its  factors  are  1、2、3
```

8. 一球从100米高度自由下落,每次落地后返回原高度的一半,再落下。求它在第10次落地时共经过多少米?第10次反弹多高?

9. 猴子吃桃问题。猴子第一天摘下若干个桃子,当即吃了一半,还不过瘾,又多吃了一个。第二天早上又将剩下的桃子吃掉一半,又多吃一个。以后每天早上都吃了前一天剩下的一半零一个。到第10天早上想再吃时,见只剩下一个桃子了。求第一天共摘了多少个桃子。

10. 打印以下图案:

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

11. 用牛顿迭代法求下面方程在1.5附近的根。

$$2x^3-4x^2+3x-6=0$$

12. 给出一个不多于5位的正整数,要求:①求出它是几位数;②分别打印出每一位数字;③按逆序打印出各位数字,例如原数是321,应输出123。