第四章　分支结构的C程序设计

4.1 分支结构中的表达式

在其他高级语言中，分支结构中的表达式仅指关系表达式和逻辑表达式，比较简单。C语言中要复杂得多，可以是任何有效的表达式，如算术表达式、赋值表达式、字符表达式、 条件表达式，还可以是任意类型的数据，如整型、实型、字符型、指针类型等。

4.1.1 C语言中的逻辑值

C语言中没有专门定义逻辑类型的变量、常量和输入、输出格式。但对逻辑值作了更宽的规定：表达式的值非0，则表示逻辑真；表达式的值为0，则表示逻辑假。

这就是说，不管什么类型的表达式，只要值不是0就表示真，如1、2、0.5、'a'，都表示真。值只有是0、'\0'(字符'\0'的ASCII值为0)才表示假。

对逻辑真、假值的这种策略，使得所有类型的表达式都能在分支语句中作条件使用，允许我们编制效率极高的程序。

4.1.2 关系表达式

1. 关系表达式的概念

所谓关系表达式是指，用关系运算符将两个表达式连接起来，进行关系运算的式子。

例如，下面的关系表达式都是合法的：

a>b，a+b>c-d，(a=3)<=(b=5)，'a'>='b'，(a>b)= = (b>c)

2. 关系表达式的运算结果——逻辑值（非“真”即“假”）

C语言用整数“1”表示“逻辑真”，用整数“0”表示“逻辑假”。所以，关系表达式的值，还可以参与其它种类的运算，例如算术运算、逻辑运算等。

例如，假设num1=3，num2=4，num3=5，则：

(1) num1>num2的值为0。

(2) (num1>num2)!=num3的值为1。

(3) num1<num2<num3的值为1。

(4) (num1<num2)+num3的值为6，因为num1<num2的值为1，1+5=6。

4.1.3 逻辑表达式

1. 逻辑表达式的概念

关系表达式只能描述单一条件，例如“x>=0”。如果需要描述“x>=0”、同时“x<10”， 就要借助于逻辑表达式了。

所谓逻辑表达式是指，用逻辑运算符将一个或多个表达式连接起来，进行逻辑运算的式子。在C语言中，用逻辑表达式表示多个条件的组合。

例如，(x>=0) && (x<10)就是一个判断数x是否大于等于0且小于10的逻辑表达式。

2. 逻辑表达式的运算结果

逻辑表达式的值也是一个逻辑值（非“真”即“假”）。

例如：

(a>b) && (x>y)　　等效于 a>b && x>y

(a==b) || (x==y)　　等效于 a==b || x==y

(!a) || (a>b)　　等效于 !a || a>b

要根据优先级处理数值运算、关系运算和逻辑运算，想提高某运算的级别或增加运算关系的清晰性，可以加括号。逻辑表达式求解，在值已能确定的情况下不一定求到最后。如：

(1) 表达式a && b && c：

当a=0时，表达式的值为0，不必计算判断b、c；

当a=1、b=0时，表达式的值为0，不必计算判断c；

只有a=1、b=1，才判断c。

(2) 表达式a || b || c:

当a=1（非0）时， 表达式的值为1，不必计算判断b、c；

当a=0时，才判断b，如b=1，则表达式的值为1，不必计算判断c；

只有a=0、b=0，才判断c。

熟练掌握关系运算符和逻辑运算符，可以用逻辑表达式表示一个复杂的条件。例如，判断某年y是否为闰年。y满足二者之一为闰年：（1）y能被4整除，但不能被100整除；（2）y能被400整除。两个条件为或（||）的关系，条件（1）内的两个条件为与（&&）的关系。判断整除用求余运算%，余数为0，则能整除。表达式可写成：

（ y%4 ==0 && y%100 != 0 ）|| y%400 == 0

判断非闰年，则将上述整个条件取反即可：

！（（ y%4 ==0 && y%100 != 0 ）|| y%400 == 0 ）

4.1.4 其它形式的表达式

C语言分支的表达式比较复杂，因为它可以是任何有效的表达式。常用的还有：

1. 算术表达式

如if(a * b－ 3 * c) {…}，以算术表达式“a*b-3*c”的值是否为真决定程序流向，而不必写成逻辑表达式“a*b-3*c==1”的形式。

if(a) {…}、if(3) {…}、if(0) {…}使用的都是算术表达式。如果写成if(a==1) {…}或if(a==0) {…}，则用的是逻辑表达式，是有冗余且潜在低效的，不是好风格。

2. 赋值表达式

用赋值表达式作表达式，清晰度不高，最容易使人“上当受骗”。

例如：“int a=3， b=5； if(a=b){…}”，请问条件是否成立? 有人说不成立，因为3≠5。错了！ 这里“a=b”是赋值表达式，而不是“a == b”，a为5。表达式取a的值，为真。

再如：“int a=3， b=0； if(a=b){…}”，请问条件是否成立？有人说不成立，因为3≠0。说条件不成立，是正确的，但理由是“a=b”是赋值表达式，赋值后a为0，表达式取a的值，故为假。

3. 字符表达式

如定义“char c=3;”，则if(c){…}、if('B'){…}使用的都是字符表达式，其值同样是0为假，非0为真。

还有其它形式的表达式，如逗号表达式等，其逻辑值的取法与上述相同。

4.2 if 语 句

4.2.1 if语句的简单形式

if语句的简单形式如下：

if(表达式)语句

其中表达式不限于逻辑表达式或关系表达式，可以是各种表达式，如算术表达式等。当表达式的值为非零时为“真”，当表达式的值为零时为“假”。

if语句的执行过程如图4.1所示。

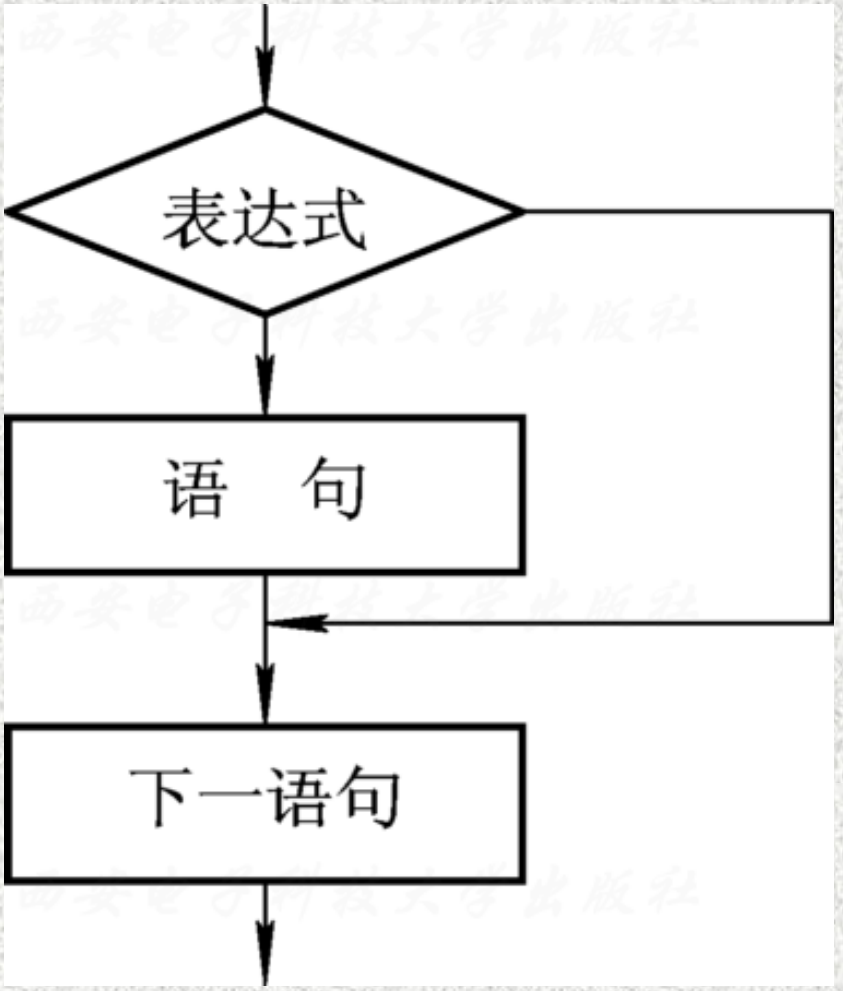

图 4.1 if语句执行过程

当表达式的值为真时，执行后面的语句，接着执行下一语句； 当表达式的值为假时，就直接执行下一语句。

if语句中的“语句”从语法上讲必须也只能是一条语句，它可以是第三章中介绍过的各种语句。很多情况下它不止一句，当超过一句时，必须用花括号括起来，构成一个复合语句。

例 4.1 打印出不及格的成绩。

```
#include <stdio.h>
void main()
{
   float score;
   scanf("%f"，&score);
   if (score<60.0)
      printf("score=%5.1f\n"，score);
}
```

运行：

68 ↙ (输入，不满足条件，无输出)

再次运行：

50 ↙ (输入)

score= ␣ 50.0(输出)

if结构中也可以采用复合语句。

例 4.2输入三角形的三条边长a、 b、 c， 若能构成三角形， 则利用海伦公式求出三角形的面积。(海伦公式： S=sqrt(l*(l－a)*(l－b)*(l－c))， 其中l=(a+b+c)/2。

```
#include <stdio.h>
#include <math.h>
void main()
{  float a， b， c， l， S；
   printf("Input a， b， c： ")；
   scanf("%f %f %f"， &a， &b， &c)；
   if (a<=0 || b<=0 ||c<=0)
   {
```

```
                printf("Illegal input!\n");
                return;
        }
        if (a+b>c && a+c >b && b+c>a)
        {
                l=(a+b+c)/2;
                S=sqrt(l*(l-a)*(l-b)*(l-c)); /* sqrt是库函数，其功能是求平方根 */
                printf("area S=%.2f\n", S);
        }
    }
```

运行结果：

```
Input a, b, c: 3 4 5↙
area S=6.00
```

4.2.2 if~else结构

1. if~else结构的形式

if~else结构是if语句的基本形式，形式如下：

if(表达式) 语句1

else 语句2

其执行过程如图4.2所示。

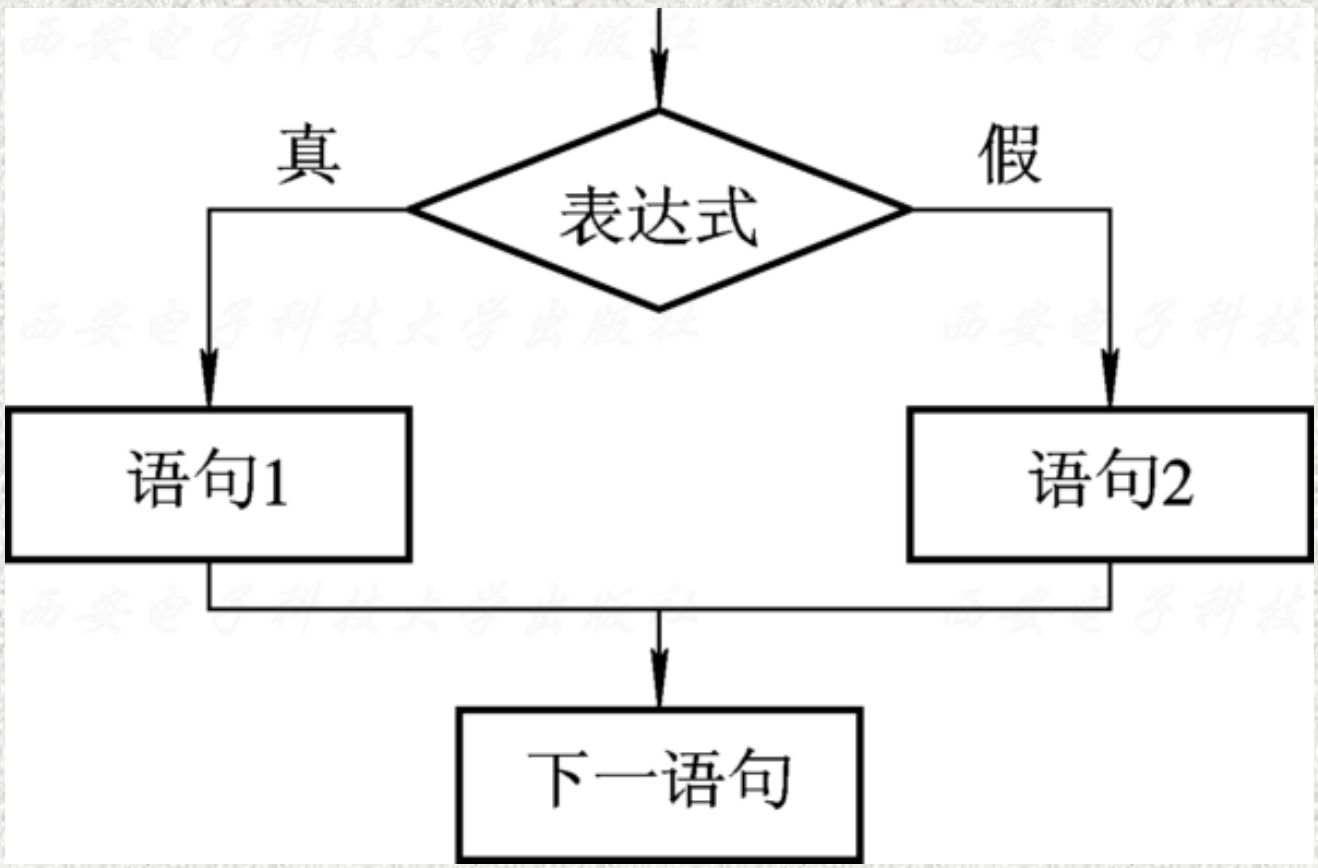

图 4.2 if~else语句执行过程

当表达式的结果为真(非零)时，执行语句1，执行完后跳到下一语句； 当表达式的结果为假(零)时，执行语句2，执行完后顺序执行下一语句。此属二中选一的情况，必选其一，只选其一。

其中语句1和语句2可以是单个语句，也可以是复合语句(即用花括号括起来的一组语句)。复合语句中又可以含有if语句，这就是后面要讲的if嵌套。子句else是任选的，当没有else子句时，就变成第一种形式了。在书写格式上语句1、elsc、语句2都可以另起一行，但仍看做是在同一个if语句内。

例 4.3 打印成绩≥60分为“Pass”，否则为“Fail”。

解 我们可以使用两种方法编程。

方法一：用两个简单的if语句实现。

```
#inchde <stdio.h>
void main()
{
   float score;
  scanf("%f"，&score);
  if(score<60.0) printf("score=%5.1f---Fail＼n"，score);
  if(score>=60.0) printf("score=%5.1f---Pass＼n"，score);
}
```

运行：

50 ↙

score= ␣ 50.0---Fail

再次运行：

80 ↙

score= ␣ 80.0---Pass

在第二个if语句中，if(score>=60.0)是不能缺少的，否则不管score是否小于60，都将打印出第二行的结果。

方法二：用if~else语句实现。

```
#include <stdio.h>
void main()
{
   float score;
   scanf("%f"，&score);
   if(score<60.0) printf("score=%5.1f---Fail＼n"，score);
   else printf("score=%5.1f---Pass＼n"，score);
}
```

运行：

50 ↙

score= ␣ 50.0---Fail

再次运行：

80 ↙

score= ␣ 80.0---Pass

从例4.3中可看出，if~else结构较两个if语句简单、清晰，也不易出错。

上例if语句和else语句后都有一个分号，该分号是C语言语句语法所要求的。if语句后的分号并不是表示if结构已结束，而是其后的printf语句要求的分号； else语句后的分号也是其后的printf语句要求的，同时表示此if语句到此结束，而不必连用两个分号来表示if~else结构的结束。

2. 条件运算符的使用

当if~else结构中的语句是表达式语句时，就可以使用条件运算符“? ：”，即下列if~else语句：

if(表达式1) 表达式2； else 表达式3；

用条件运算符写成通用形式即为:

表达式1? 表达式2：表达式3

说明：

(1) 条件运算符的执行顺序：先解表达式1，若为非0(真)，则求解表达式2，此时表达式2的值就作为整个条件表达式的值。若表达式1的值为0(假)，则求解表达式3，此时表达式3的值就作为整个条件表达式的值。

max=(a>b)? a:b

执行结果就是将条件表达式的值赋给max。也就是将a和b两者中大者赋给max。

(2) 条件运算符优先于赋值运算符，因此上面赋值表达式的求解过程是先求解条件表达式，再将它的值赋给max。

条件运算符的优先级比关系运算符和算术运算符都低。因此，

max=(a>b)? a:b　等效于 max=a>b? a:b

a>b? a:b+1等效于 a>b? a:(b+1)

(3) 条件运算符的结合方向为“自右向左”。如果有以下条件表达式：

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

相当于

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

如果a=1，b=2，c=3，d=4，则条件表达式的值为4。

由于函数调用也返回一个值，因此表达式2，3也可出现函数的调用。如以下形式：

表达式? 函数调用1： 函数调用2

例 4.4 输入x的值，当x>0时调用sqrt(x)，否则调用fabs(x)。

程序：

```
#include <math.h>
#include <stdio.h>
void main()
{
  float x，y;
  scanf("%f"，&x);
  y=x>0? sqrt(x)：fabs(x)；/* fabs是库函数， 其功能是取绝对值*/
  printf("x=%f，y=%f＼n"，x，y);
}
```

运行：

9.0 ↙

x=9.000000，y=3.000000

再次运行：

－9.0 ↙

x=－9.000000，y=9.000000

本例中使用了数学函数sqrt(开平方根)和fabs(求绝对值)，因此在程序头部要加上文件包含命令#include <math.h>。这两个函数的函数值和自变量均为双精度类型，因此x最好定义成双精度型。取整型数的绝对值可使用abs(x)函数。这些可查附录二的库函数表。

例 4.5 打印a，b两个数中较大的一个数。

方法一：用if~else结构实现。

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

方法二：用条件运算符实现。

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

以上两种方法运行结果完全一致，可选用。

3. if语句的嵌套

if语句的嵌套是if~else结构中的语句1或(和)语句2，又是一个if~else结构。

我们直接由例子来说明。

例 4.6 输入一个学生成绩，当成绩≥90时，打印“Very Good”；当80≤成绩＜90时，打印“Good”；当60≤成绩＜80时，打印“Pass”；当成绩＜60时，打印“Fail”。

```
#include <stdio.h>
void main()
{
   float score;
   scanf("%f", &score);
   if(score >= 80)
      if(score>=90) printf("Very Good \n");
      else printf("Good \n");
   else if(score>=60) printf("Pass \n");
         else printf("Fail \n");
}
```

运行：

85 ↙

Good

嵌套的if语句很容易出错，原因在于不知道哪个else与哪个if配对。C语言提供了一个简单规则：从内层开始，else总是与它上面最近的(未曾配对的)if配对。例如语句段：

```
if(x)
   if(y) printf("A");
   else printf("B");
```