第 3 章

选择(分支)结构程序设计

本章重点

- if 语句的 3 种使用形式;
- if 语句的嵌套;
- switch 语句;
- 选择结构程序设计。

3.1 顺序程序结构

我妹妹的女儿,刚上高中,业余时间跟我学习 C 语言,恰好她的学习进度和我书写本书的进度一致。我给她布置了一个任务,让她输入三角形的三边长,利用下面的公式求出三角形的面积(公式中的 a、b、c 分别代表三角形的三条边),并告诉她 C 系统提供了一个 sqrt()函数,它可以求一个数的算术平方根。sqrt()函数的有关说明在 math.h 头文件中。

$$\text{area} = \sqrt{\frac{a+b+c}{2}\left(\frac{a+b+c}{2}-a\right)\left(\frac{a+b+c}{2}-b\right)\left(\frac{a+b+c}{2}-c\right)}$$

上一个周末,我妹妹的女儿很高兴地来到我家,准备跟我学习本章的内容,顺便她也把我布置的任务代码带给了我,期待我对她的劳动做出评价,下面是她的劳动成果,也同时是本章中的第一个例题。

【例 3-1】 输入三角形的三边,输出其面积。

代码如下:

```
#include <stdio.h>
#include <math.h>
#define S (a+b+c)/2                    //为了书写方便,定义大写的符号常量 S,代表(a+b+c)/2
void main()
{
    float a,b,c,area;                  //定义变量分别代表三角形的三边长和面积
    printf("请输入第一条边的长度=");    //提示输入边长
    scanf("%f",&a);                    //接受一个实数,并送给变量 a
    printf("请输入第二条边的长度=");
    scanf("%f",&b);
    printf("请输入第三条边的长度=");
```

```
    scanf("%f",&c);
    area=sqrt(S*(S-a)*(S-b)*(S-c)); //sqrt()函数能得到一个数的算术平方根,送给变量 area
    printf("三角形的边长分别为%.2f,%.2f,%.2f,面积=%.2f\n",a,b,c,area);  //输出结果
}
```

程序的运行结果如下：

```
请输入第一条边的长度=3
请输入第二条边的长度=4
请输入第三条边的长度=5
三角形的边长分别为 3.00,4.00,5.00,面积=6.00
```

我看了她编写的代码以及运行结果，对她进行了以下几点肯定。

(1) 运行界面很友好，并且很清晰，且不会让使用者产生歧义。

(2) 使用了 float 类型的变量，因为三角形的边长不一定都是整数。

(3) 灵活地使用了符号常量，且符号常量定义的位置很恰当。

我曾经讲过，当某些内容多次出现，且不好记忆时，可以声明一个符号名字来代替这些内容。面积公式中多次出现$(a+b+c)/2$，声明了符号常量 S 来代替这个内容，并且声明的位置在主函数 main()之前，在#include 语句之后。

本程序中，因为符号常量 S 只在 main()函数中使用，不在其他地方使用，如果把符号常量 S 的定义放在最前面，有可能影响 stdio.h 和 math.h 里面的内容。另外也要注意，本任务中也可以不使用符号常量，而改用一个普通变量来代替符号常量的作用。比如在主函数中用“float s;”语句定义一个变量 s，在 a、b、c 都获取值以后，用“s=(a+b+c)/2;”语句，在求面积公式中，用变量 s 代替$(a+b+c)/2$，也可以达到目的。

在例 3-1 的代码中，使用了 3、4、5 这组数据，程序正确地求出了三角形的面积。但再次运行了例 3-1 的代码，并把数据 1、10、5 送给它，看到了如下的结果：

```
请输入第一条边的长度=1
请输入第二条边的长度=10
请输入第三条边的长度=5
三角形的边长分别为 1.00,10.00,5.00,面积=-1.#J
```

“这是为什么呀？我的程序没有错误呀，为什么求出的面积莫名其妙！”我妹妹的女儿大声地嚷嚷着。

也许，有的读者也存在着同样的疑问，请大家不要着急，下面来分析原因并告诉大家解决的方法，也就是本章将要讲述的内容。

在例 3-1 中，程序是一种“顺序结构”，这种结构的特点就是当程序开始执行时，如果没有特殊情况，顺序结构中的每一条语句从头到尾都能被执行一遍，且只能被执行一遍。

对于 3、4、5 这组数据来说，它们恰好构成三角形的三边，所以很顺利地求出面积，而对于 1、10、5 这组数据来说，它们并不构成三角形的三边，但是程序中没有选择地继续用面积公式求面积，使得开平方根时遇到了负数，所以出现一个“莫名其妙”的结果。

“哦，我好像明白了……”她若有所思地点着头，然后很兴奋地说“我知道了，今天你要告诉我另一种语句，这种语句能让程序根据条件做出选择，是不是？”

本章将带领大家，去学习如何让程序做出正确的判断，从而有选择地执行某些语句。

3.2　用 if 语句进行选择

每天早上出门之前，我会看看天气，如果外面阴天或者正在下雨，我会带上一把伞，如果阳光普照，那我就不带伞。

现实生活中诸如此类的例子很多，可以说我们每天都会根据具体情况做出不同的选择，想必大家都深有体会。

那么，在编写 C 的程序时，可以用什么样的语句来选择执行某些代码，从而忽略另一些代码呢？

C 系统提供了 if 和 switch 两种语句，它们用来根据条件选择执行某些语句。

3.2.1　if 语句的 3 种使用形式

C 语言中可以用 if 语句结合有关条件，来对程序的走向做出判断和选择，其具体使用形式有以下几种。

1. 用 if 语句实现单分支选择

使用形式：

```
if (条件表达式) {语句体}
```

功能：首先计算条件表达式的值，若值为“真”(非 0)，则先执行语句体，然后到语句体之后去执行，如果条件表达式的值为“假”(0)，则不执行语句体，而是直接转到语句体之后去执行，其流程示意图如图 3-1 所示。

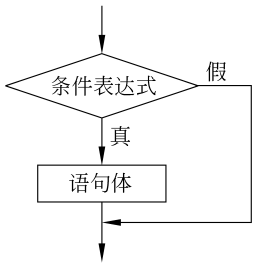

图 3-1　if 语句实现单分支示意图

在使用 if 语句时，要注意下面的问题。

(1) if 语句中的“条件表达式”必须用英文括号“( )”括起来。

(2) 当语句体中只有一条语句构成时，{}可以省略，但规范化编程不提倡省略。

(3) if 和对应的语句体可以不写在一行上，而是写成下面的形式。

```
if(条件表达式)
{
    …                    //此处书写条件满足时要执行的语句
}
```

有了上面的选择结构，就可以把例 3-1 进一步完善了。想必各位读者心中已经有了答案了，下面就是完善后的代码。

```
#include <stdio.h>
#include <math.h>
void main()
{
    float a,b,c,area,s;
```

```
    printf("请输入第一条边的长度=");
    scanf("%f",&a);
    printf("请输入第二条边的长度=");
    scanf("%f",&b);
    printf("请输入第三条边的长度=");
    scanf("%f",&c);
    s=(a+b+c)/2;                          /*用变量 s 来存储三边和的一半*/
    if(a+b>c&&a+c>b&&b+c>a)               /*当满足任意两边之和大于第三边的条件时*/
    {
        area=sqrt(s*(s-a)*(s-b)*(s-c));
        printf("三角形的边长分别为%.2f,%.2f,%.2f,面积=%.2f\n",a,b,c,area);
    }
    printf("谢谢使用本程序\n");
}
```

注意：修改后的代码使用了 if 语句来判断 a、b、c 是否满足构成三角形的条件，如果满足才使用面积公式求出面积并输出。

各位读者现在应该知道，为什么在第 2 章里面那么费力地讲解比较运算符和逻辑运算符了，它们都有可能被用在条件表达式里面作为判断的依据。

“对例 3-1 的运行结果，我还是有一点儿不满意。”在课堂上讲解这个例题的时候，崔佳蓉同学提出了自己的看法。

我心里很清楚崔佳蓉同学为什么对运行结果不满意，因为她有着严密的思维逻辑，总能从多个方面考虑问题。

课堂上，大家观察着程序的运行结果，如果输入的数据构成三角形，程序显示所求的面积并且输出“谢谢使用本程序”，输出结果没有什么不妥呀？结果如下：

```
请输入第一条边的长度=3
请输入第二条边的长度=4
请输入第三条边的长度=5
三角形的边长分别为 3.00,4.00,5.00,面积=6
谢谢使用本程序
```

因为大多数程序的使用者并不是程序的开发者，他们不了解程序的内部结构，他们只关心界面和结果。

如果输入的数据不构成三角形，则输出结果特别不清晰，直接显示“谢谢使用本程序”，运行结果如下。这让使用者有点“丈二和尚摸不着头脑”，程序为什么不给求面积？这是为什么？

```
请输入第一条边的长度=1
请输入第二条边的长度=3
请输入第三条边的长度=5
谢谢使用本程序
```

所以，需要在例 3-1 的代码中，对不合理的数据做出说明，让使用程序的人明白，为什么他输入的那组数据无法给出一个面积。

要完成这个要求，必须学习 if 语句的第二种使用形式。

2. 用 if 语句实现二分支选择——if...else 语句

使用形式：

```
if (条件表达式) {语句体 1} else {语句体 2}
```

功能：先计算条件表达式的值，若条件表达式的值为“真”，则执行语句体 1，条件表达式的值为“假”，则执行语句体 2。

和 if 语句的第一种形式一样，在语句体只有一条语句时，{}同样可省。if 语句的所有内容可以写在一行上，也可以按下面规范化的形式书写。

```
if (条件表达式)
{
    …                                    //此处书写条件成立时要执行的语句
}
else
{
    …                                    //此处书写条件不成立时要执行的语句
}
```

if...else 语句的结构示意图如图 3-2 所示。

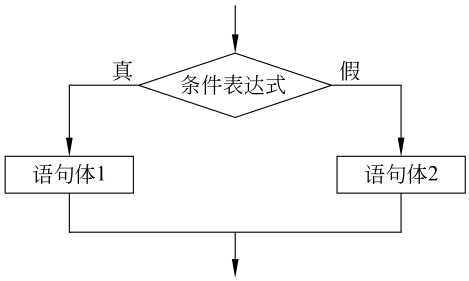

图 3-2　if 语句实现二分支示意图

“这下可好了，可以真正地完善例 3-1 程序了，让使用者明明白白地接受程序的运行结果!”，崔佳蓉高兴地跳了起来。下面是她改造后的部分代码。

```
if (a+b>c&&a+c>b&&b+c>a)
{
    area=sqrt(s*(s-a)*(s-b)*(s-c));
    printf("三角形的边长分别为%.2f,%.2f,%.2f,面积=%.2f\n",a,b,c,area);
}
else                                     //对不满足条件的数据做出处理
{
    printf("你输入的数据%.2f,%.2f,%.2f 不能构成三角形的三边\n",a,b,c);
}
```

【例 3-2】 编程，接受 3 个整数，输出 3 个整数中的最大者。

下面是实现本要求的两种方法，运行结果完全一致，希望读者能从中有所体会。

方法 1：用二分支 if 语句和单分支 if 语句结合实现，代码如下。

```
void main()
{
    int num1,num2,num3,max;              //定义四个变量分别表示三个整数和最大值
    printf("请输入第一个整数=");scanf("%d",&num1);
    printf("请输入第二个整数=");scanf("%d",&num2);
    printf("请输入第三个整数=");scanf("%d",&num3);
    if(num1>num2)                        /*该判断从 num1 和 num2 中找出大值送给 max*/
```

```
    {
        max=num1;
    }
    else
    {
        max=num2;
    }
    if(max<num3) { max=num3;}       /*该判断把 max 和 num3 比较*/
    printf("%d,%d,%d 中的最大值=%d\n",num1,num2,num3,max);
}
```

运行结果如下：

```
请输入第一个整数=23
请输入第二个整数=-87
请输入第三个整数=56
23,-87,56 中的最大值=56
```

方法 2：完全用单分支 if 语句实现，其代码如下。

```
void main()
{
    int num1,num2,num3,max;          //定义四个变量分别表示三个整数和最大值
    printf("请输入第一个整数=");
    scanf("%d",&num1);
    printf("请输入第二个整数=");
    scanf("%d",&num2);
    printf("请输入第三个整数=");
    scanf("%d",&num3);
    max=num1;                        //把第一个数 num1 当最大值送给 max
    if(num2>max)                     //当 num2>max 时
    {
        max=num2;                    //max 接受 num2 的值
    }
    if(max<num3)                     //当 num3>max 时
    {
        max=num3;                    //max 接受 num3 的值
    }
    printf("%d,%d,%d 中的最大值=%d\n",num1,num2,num3,max);
}
```

“老师，你刚才讲的条件判断 if...else 只适合一分为二的情况，也就是非此即彼的情况，可我们往往碰到的问题是多种情况都对应着不同的处理方式，比如我们的成绩等级，90 分以上为 A，80～89 分为 B，70～79 分为 C，60～69 分为 D，0～59 分为 E，如果我们输入一个分数，程序该如何给出我们的等级？”

“很好，崔佳蓉同学总能在关键的时候提出令人兴奋的问题！”我由衷地赞叹着。

对崔佳蓉同学提出的问题，我们完全可以用学到的知识巧妙地组合来完成，不需要我再讲解新的知识点。

但是，我还是想先告诉大家一种比较直观的处理多种分支的 if 语句。

3. 使用 if 语句实现多分支——if...else if 语句

使用形式：

```
if (条件表达式 1) {语句体 1}
else if (条件表达式 2) {语句体 2}
else if (条件表达式 3) {语句体 3}
⋮
else if (条件表达式 n) {语句体 n}
else {语句体 n+1}
```

功能：执行第一个值为“真”的条件表达式所对应的语句体，然后到……处执行。若所有的条件表达式的值都不为“真”，则就执行最后一个 else 所对应的语句体 $n+1$。如果没有 else{语句体 $n+1$}，则直接到……处执行。

在使用该形式的 if 语句时，要注意下面的问题。

(1) else if 的数量没有限制，根据实际情况使用。

(2) 最后的 else{语句体 $n+1$}可以没有。

图 3-3 是 if 语句的 5 分支示意图(5 分支里面有 4 个条件判断)，从中看以看出。

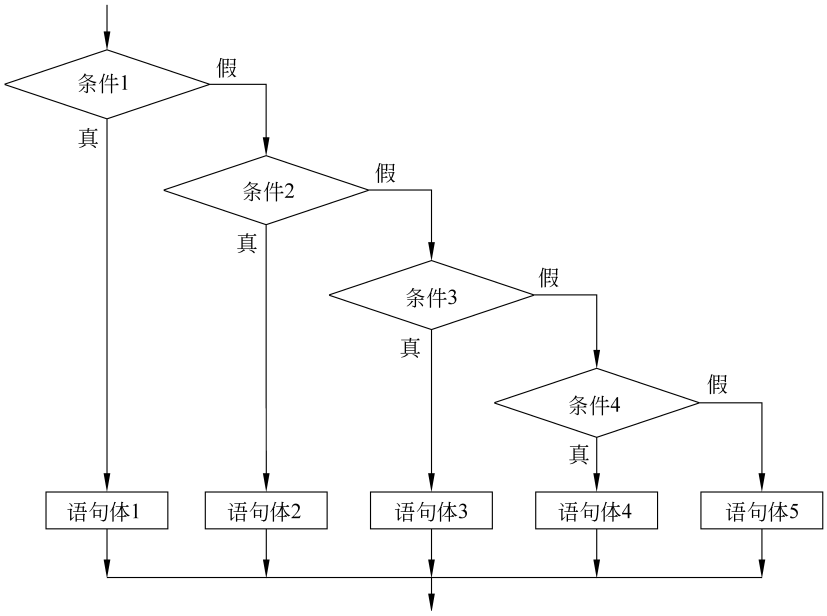

图 3-3　if 语句实现多分支示意图

若第一个条件为“真”，执行完对应的语句体 1 之后，该 if 语句就结束了，不会进行其他判断了。只有当上一个条件不成立时，才会进入下一个条件的判断。

下面用 if 语句的这种使用形式来解决崔佳蓉同学提出的成绩等级问题。

【例 3-3】 编程，输入一个分数，输出该分数对应的等级，等级的原则为：90 分以上为 A，80～89 分为 B，70～79 分为 C，60～69 分为 D，0～59 分为 E。

代码如下：

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

```
void main()
{
    float score;                          //定义变量 score 代表分数
    char grade;                           //定义变量 grade 代表等级字符
    printf("请输入你的成绩:");
    scanf("%f",&score);                   //输入成绩,并送给 score
    if(score<0||score>100)                //当输入的成绩小于 0 或者大于 100
    {
        printf("输入的成绩只能在 0~100 之间\n");
        return;                           //结束本函数的执行,返回调用处
    }
    if(score>=90) { grade='A';}
    else if(score>=80) { grade='B';} /*条件也可以写成 score>=80&&score<90*/
    else if(score>=70) { grade='C';} /*条件也可以写成 score>=70&&score<80*/
    else if(score>=60) { grade='D';} /*条件也可以写成 score>=60&&score<70*/
    else { grade='E';}  /*此处的 else 可换成 if (score<60) 或者 if(score>=0)*/
    printf("你的成绩是%.2f,对应的等级为%c\n",score,grade);
}
```

程序的运行结果如下：

```
请输入你的成绩：250
输入的成绩只能在 0~100 之间

请输入你的成绩：78.5
你的成绩是 78.50,对应的等级为 C
```

本程序用了两个 if 语句。在接受成绩之后，先用第一个 if 语句把不合理的数据排除在外，即当输入的成绩在[0,100]之外时，程序执行了 return 语句。

只要在主函数中执行 return 语句，程序就意味着结束，即 return 之后的代码不会被执行。也就是说只有输入的成绩在[0,100]之间时，第二个 if 语句才能被执行。

对于例 3-3 有的同学提出了以下问题。

“为什么在第二个 if 语句的诸多条件中，比如[80,89]之间的成绩判断，应当写成 score>=80&&score<90，但是程序中却写成 score>=80 呢?”

在这里需提醒大家注意的是，对于例 3-3 的各个成绩区间的判断，比如[80,89]区间的判断条件表达式，可以写成 score>=80&&score<90，也可以写成 score>=80。前者非常明确地指出了成绩的区间，这种情况不论什么时候都不会出错。而后者之所以敢写得如此简单，是因为我们清楚地知道，在 if...else if 格式中，下一个条件都是在上一个条件不成立时才能够得到判断。

另外，当语句体对应的语句数量较多时，可以把语句体以及对应的括号另起一行书写，这种写法更加清晰，也是规范化的书写形式，如下形式所示。

```
if(条件 1)
{
    …                                     //此处书写条件 1 成立时要执行的语句
}
```

```
else if(条件 2)
{
    …                                    //此处书写条件 2 成立时要执行的语句
}
else if(条件 3)
{
    …                                    //此处书写条件 3 成立时要执行的语句
}
⋮
else if(条件 n)
{
    …                                    //此处书写条件 n 成立时要执行的语句
}
else
{
    …                                    //此处书写其他情况要执行的语句
}
```

3.2.2　if 语句的嵌套使用

前面告诉过大家，对于例 3-3 的要求，即使没有学习 if...else if 语句也能完成，这又是什么语句呢？

可以很肯定地告诉大家，这不是什么新的语句，只是对 if 语句的多次使用而已，完全是一种旧知识，只是前面没有使用它而已，这就是 if 语句的嵌套。

那么什么是 if 语句的嵌套呢？

在所学的 if 语句的各种使用形式中，若所对应的语句体内又包含 if 语句，则就形成了 if 语句的嵌套使用。如下面的 if 语句就使用了嵌套形式，其他的 if 嵌套也都类似。

```
if(条件 1)
{
    …
    if (条件 2)
    {...}
    …
}
else
{
    …
    if (条件 n)
    {...}
    else
    {...}
    …
}
```

“那该如何用 if 语句的嵌套来完成学生成绩等级的输出呢？”这个问题下面马上来解决。

说明：同一个问题，切入点不一样，嵌套的形式也不一样，所以不同的程序员会写出

不同的代码。

例3-3处理成绩等级时，可以使用许多不同的嵌套形式，图3-4是例3-3在做成绩判断时使用的一种嵌套形式。

```
if(score<0||score>100)
{
    printf("输入的成绩只能在0~100之间\n");
}
else  //假定输入的成绩在[0,100]之间，对应的语句体内再次使用了if语句
{
    if(score>=90){grade='A';}
    else if(score>=80){grade='B';}
    else if(score>=70){grade='C';}
    else if(score>=60){grade='D';}
    else if{grade='E';}
    printf("你的成绩是%.2f,对应的等级为%c\n",score,grade);
}
```

图3-4　嵌套形式1——在else对应的语句中嵌套

图3-5是例3-3在做成绩判断时可以使用的另外一种嵌套形式。

```
if(score>=60)
{
    if(score>100)
    {
      printf("成绩不能在超过100分\n");
      return;
    }
    else
    {
      if(score>=90) {grade='A';}
      else if(score>=80) {grade='B';}
      else if(score>=70) {grade='C';}
      else {grade='D';}
    }
}
else  //score<60
{
    if(score>=0)
    {
       grade='E';
    }
    else
    {
      printf("成绩不能为负数\n");
      return;
    }
}
printf("你的成绩是%.2f,对应的等级为%c\n",score,grade);
```

图3-5　嵌套形式2——在if和else对应的语句中都使用嵌套