第3章　分支结构

分支结构又称选择结构，是程序设计的基本结构。它通过对给定的条件进行判断，决定执行两个或多个分支中的哪一个。因此，在编写选择结构之前，应该明确判断条件以及当判断结果为“真”或“假”时执行的操作。

通过本章学习，应掌握在C语言中实现选择结构的控制语句的格式、功能和执行过程，熟练使用选择控制语句编写具有分支基本结构的程序。

C语言程序中提供的分支结构有两种：if条件分支结构和switch结构。

3.1　if条件分支结构


3-0.mp4

if语句是二分支选择语句。if语句可以给出两种操作，通过表达式结果(非0或0)选择其中的一种操作。

if语句有以下3种格式。

格式1：单分支条件语句。

```
if (条件表达式)
{
    语句序列
}
```

该语句的功能是对条件表达式进行判断，若结果为真，则执行“{}”中的语句序列，然后执行“}”后面的语句；否则跳过“{}”中的语句序列，直接执行其后的语句。条件表达式可以是关系表达式、逻辑表达式、表达式、常量、变量或函数。流程图描述如图3.1所示。

例3.1　判断用户输入的数据，如果输入的数值大于0，则在屏幕上显示“正数”。

解题思路：键盘输入数据a，利用if语句判断a是否大于0，其流程图如图3.2所示。


3-1.mp4

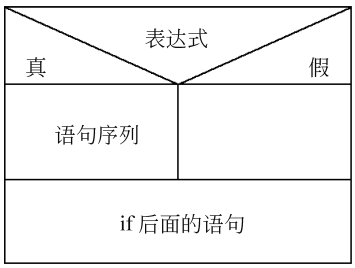

图3.1　单分支条件语句的流程图

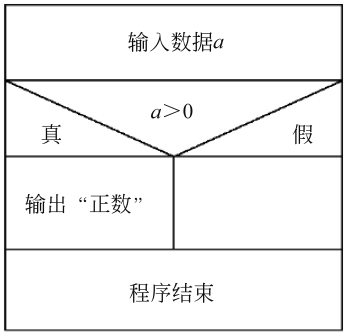

图3.2　例3.1的流程图

程序代码如下：

```
#include <stdio.h>
int main()
{
    int a;
    printf("请输入一个数: ");
    scanf("%d",&a);
    if (a>0)
        printf("正数\n");
    printf("结束\n");
    return 0;
}
```

程序运行结果如下：

第一次运行：

```
5↙
正数
结束
```

第二次运行：

```
-2↙
结束
```

当语句序列只包含一条语句时，包围该语句序列的“{}”可以省略。

格式 2：二分支条件语句。

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

该语句的功能为，如果“条件表达式”的判断结果为真，则执行语句序列 1；如果“条件表达式”的判断结果为假，则执行语句序列 2。流程图描述如图 3.3 所示。

例 3.2 判断用户输入的数据，如果输入的数值大于 0，则在屏幕上显示“正数”；否则在屏幕上显示“不是正数”。


3-2.mp4

解题思路：键盘输入数据 a，利用 if 语句判断 a 是否大于 0，根据判断的结果执行不同的语句，其流程图如图 3.4 所示。

程序代码如下：

```
#include <stdio.h>
int main()
{
```

```
    int a;
    printf("请输入一个数: ");
    scanf("%d",&a);
    if (a>0)
        printf("正数\n");
    else
        printf("不是正数\n");
    return 0;
}
```

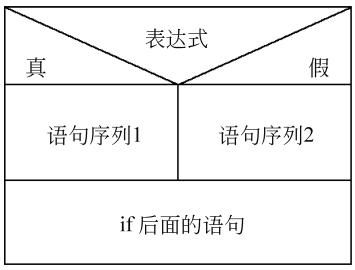

图 3.3　二分支条件语句的流程图

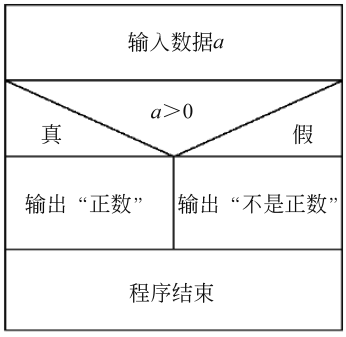

图 3.4　例 3.2 的流程图

程序运行结果如下：

```
-5↙
不是正数
```

格式 3：嵌套条件语句。

```
if (条件表达式 1)
{
    语句序列 1
}
else if (条件表达式 2)
{
    语句序列 2
}
else
{
    语句序列 3
}
```

该语句的功能为，如果“条件表达式 1”的判断结果为真，则执行语句序列 1；若为假，再判断“条件表达式 2”，如果判断结果为真，则执行语句序列 2；否则，执行语句序列 3。其流程图如图 3.5 所示。


3-3.mp4

例 3.3　找出 a、b、c 这 3 个数中的最大值。

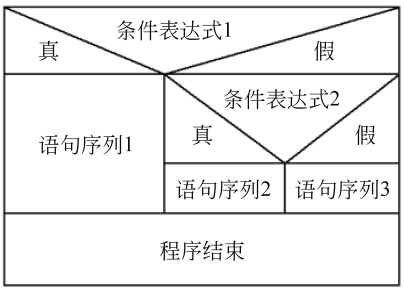

图 3.5　嵌套条件的流程图

解题思路：将 a 与 b 进行比较，如果 a 大，则 a 与 c 进行比较，大的数赋予 max；如果 b 大，则 b 与 c 进行比较，大的数赋予 max。流程图如图 3.6 所示。

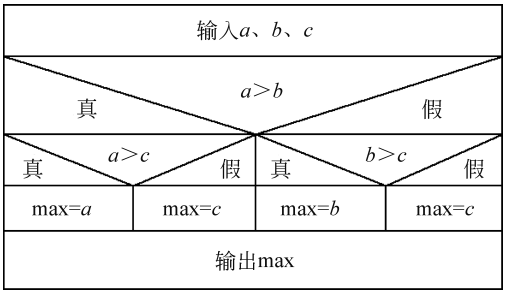

图 3.6　例 3.3 的流程图

程序代码如下：

```
#include <stdio.h>
int main()
{
    int a,b,c,max;
    printf("请输入第 1 个数: ");
    scanf("%d",&a);
    printf("请输入第 2 个数: ");
    scanf("%d",&b);
    printf("请输入第 3 个数: ");
    scanf("%d",&c);
    if (a>b)
        if (a>c)
            max=a;
        else
            max=c;
    else if (b>c)
            max=b;
        else
            max=c;
    printf("最大数为%d\n",max);
```

```
    return 0;
}
```

程序运行结果如下：

```
请输入第 1 个数: 5↙
请输入第 2 个数: 4↙
请输入第 3 个数: 2↙
最大数为 5
```

当多个 if…else 语句嵌套时，为了防止出现二义性，C 语言规定，由后向前使每一个 else 都与其前面的最靠近它的 if 配对。如果一个 else 的上面又有一个未经配对的 else，则先处理上面的(内层的)else 的配对。另外也可以按照语法关系加上“{}”来标识逻辑关系的正确性。如把例 3.3 程序中的判断部分进行如下改写：

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

3.2 switch 开关结构

switch 语句是多分支的选择语句。虽然嵌套的 if 语句可以处理多分支选择，但是用 switch 语句更加直观。

switch 语句格式如下：

```
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>;
}
```

switch 语句的执行顺序是，首先对“表达式”进行计算，得到一个常量结果，然后从上到下寻找与此结果相匹配的常量表达式所在的 case 语句，并以此作为入口，开始顺序执行入口处后面的各语句，直到遇到 break 语句，才结束 switch 语句，转而执行 switch 结构后的其

他语句。如果没有找到与此结果相匹配的常量表达式，则从“default：”处开始执行语句序列 $n+1$。其流程图如图 3.7 所示。

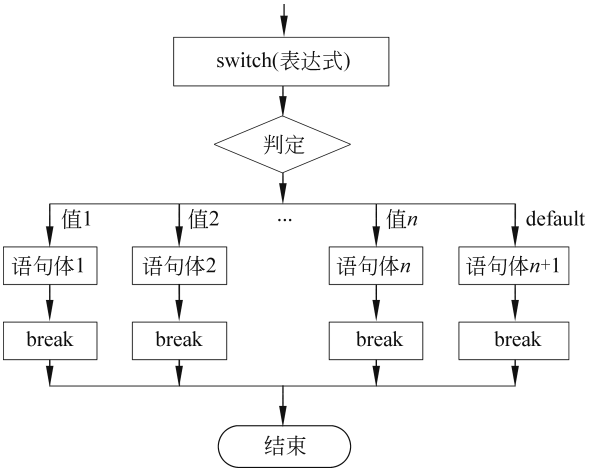

图 3.7　switch 结构的流程图

例 3.4　根据成绩的等级对应输出成绩的分数范围。


3-4.mp4

解题思路：等级为 A，输出 90～100，等级为 B，输出 80～89，等级为 C，输出 70～79，等级为 D，输出 60～69，等级为 E，输出 0～59，若输入不是 A、B、C、D、E 其中之一，显示“数据错误”信息，利用 switch 对其进行分类处理，流程图如图 3.8 所示。

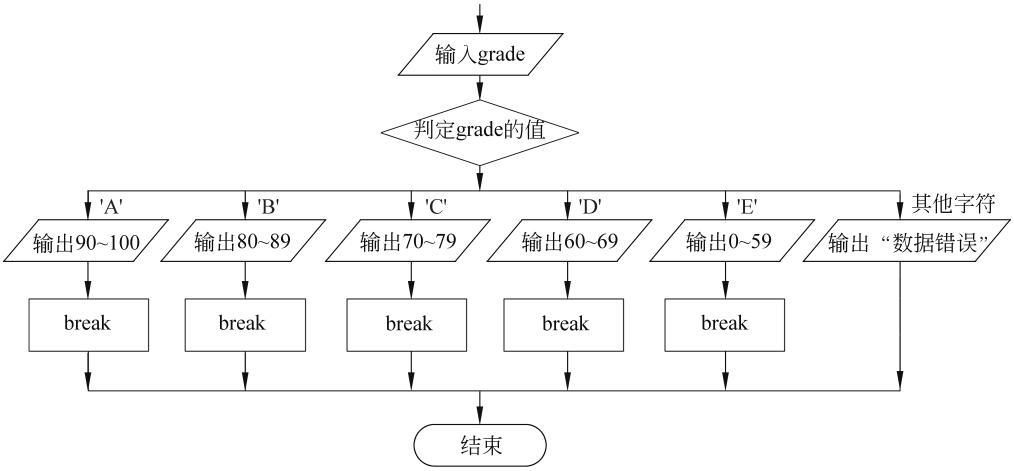

图 3.8　例 3.4 的流程图

程序代码如下：

```
#include <stdio.h>
int main()
{
    char grade;
    printf("输入成绩等级(ABCDE): ");
    scanf("%c",&grade);
```

```
    switch(grade)
    {
        case 'A':printf("90～100\n");break;
        case 'B':printf("80～89\n");break;
        case 'C':printf("70～79\n");break;
        case 'D':printf("60～69\n");break;
        case 'E':printf("0～59\n");break;
        default:printf("数据错误\n");
    }
    return 0;
}
```

程序运行结果如下：

```
B↙
对应的分数范围：80～89
```

程序中 break 语句的作用是结束 switch 语句，转而执行 switch 结构后的其他语句。

说明：

(1) switch 后面“()”中的表达式只能是常量，数据类型包括整型、字符型，所有常量必须互不相同。

(2) 在各个分支中的 break 语句起着退出 switch 语句的作用。若没有 break 语句，程序将从匹配的 case 处开始向后执行所有语句。

(3) 可以使多个 case 语句共用一组语句序列。

(4) 各个 case(包括 default)语句的出现次序可以是任意的。在各个分支中都含有 break 语句时，顺序的改变不影响执行结果。

(5) 每个 case 语句中不必用“{ }”，而整体的 switch 结构一定要写“{ }”。

(6) default 语句是可省略的。

(7) switch 结构也可以嵌套。

例 3.5 编写具有简单计数器功能的程序。

解题思路：程序首先在屏幕上输出加减乘除提示，选择相应的功能后，输入两个操作数，根据功能选择，用 switch 语句对选择结果进行相应的处理。若选择 1、2、3、4，则分别实现对两个操作数进行加、减、乘、除计算；否则直接退出程序。流程图如图 3.9 所示。


3-5.mp4

程序代码如下：

```
#include <stdio.h>
int main()
{
    int ans;
    float x,y;
    printf(" 请选择\n");
    printf(" 1--加        2--减\n");
    printf(" 3--乘        4--除\n");
    printf(" 其他任意键--退出\n");
```

```
    printf(" 请输入选择: ");
    scanf("%d",&ans);
    if (ans==1||ans==2||ans==3||ans==4)
    {
        printf("\n 请输入操作数 1: ");
        scanf("%f",&x);
        printf("请输入操作数 2: ");
        scanf("%f",&y);
        switch(ans)
        {
            case 1:printf("%f +%f =%.2f",x,y,x+y);break;
            case 2:printf("%f -%f =%.2f",x,y,x-y);break;
            case 3:printf("%f * %f =%.2f",x,y,x*y);break;
            case 4:if (y!=0)
            {
                printf("%f/%f =%.2f",x,y,x/y);
                break;
            }
            else
            {
                printf(" 除数不能为 0,程序退出!\n");
                break;
            }
            default :printf("选择错误\n");break;
        }
    }
    return 0;
}
```

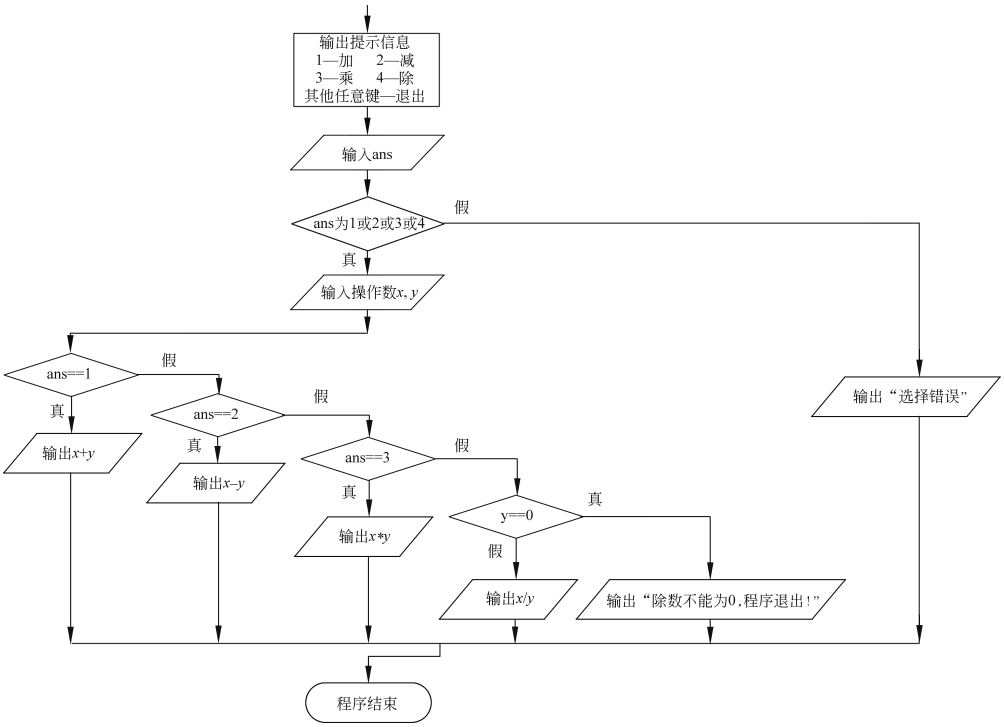

图 3.9　例 3.5 的流程图

程序运行结果如下：

```
  请选择
  1--加          2--减
  3--乘          4--除
  其他任意键--退出
请输入选择：1

请输入操作数 1：23↙
请输入操作数 2：12↙
23.000000 +12.000000 =35.00
```

思考：在案例 1 的程序中，若把 switch 结构中 case1 后面的 break 语句去掉，然后运行程序并选加法功能，会出现什么情况？若把 break 语句都去掉，又会出现什么情况？

本章小结

本章着重介绍了分支结构中的几个重要概念。

(1) if 分支结构的基本用法。

(2) switch 开关结构的基本用法。

习 题 3

一、单选题

1. x>0 || y==5 的相反表达式为(　　)。

A. x>0 && y==5　　B. x<=0 || y! =5

C. x<=0 && y! =5　　D. x>0 || y! =5

2. 若有定义 float w; int a, b; ,则合法的 switch 语句是(　　)。

A. switch(w) { case 1.0: printf("* *\n"); case 2.0: printf("* *\n"); }

B. switch(a); { case 1 printf("* *\n"); case 2 printf("* *\n"); }

C. switch(b) { case 1: printf("* *\n"); default: printf("\n"); case 1+2: printf("* *\n"); }

D. switch(a+b); { case 1: printf("* *\n"); case 2: printf("* *\n"); default: printf("\n"); }

3. 程序 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;}printf("a=%d, b=%d\n",a,b); }的输出结果是(　　)。

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

4. 若 a=7,b=3,则执行 printf("%d",(a+b)%a||(a-b)%b)的结果是(　　)。

A. 1　　B. 3　　C. 5　　D. 7

5. 若 a=20,b=7,则执行 printf("%d",(a+10)%a&&(a-b)/a)的结果是(　　)。

A. 3　　B. 2　　C. 1　　D. 0

6. 下列关于 switch 语句的描述中，(　　)是正确的。

A. switch 语句中可以有一个 default 子句，也可以没有

B. switch 语句中每个语句序列中必须有 break 语句

C. switch 语句中 default 语句只能放在最后

D. switch 语句中 case 子句后的表达式只能是整型表达式

二、编程题

1. 从键盘输入 x，y 的值，按下列公式求 z 的值。

$$z=\begin{cases}\dfrac{x^2+1}{x^2+2}y, & x\geqslant 0, y>0\\ \dfrac{x-2}{y^2+1}, & x\geqslant 0, y\leqslant 0\\ x+y, & x<0\end{cases}$$

运行过程如下：

```
输入 x,y: -2.5,2↙
z=-0.500000
```

2. 用整数 1～12 依次表示 1 月至 12 月，由键盘输入一个月份数，输出对应的季节中文名称(12 月至第二年 2 月为冬季；3 月至 5 月为春季；6 月至 8 月为夏季；9 月至 11 月为秋季)，分别用 if、switch 实现。运行过程如下：

```
输入月份：5↙
5月是春季
```

3. 城市实行车牌限号，即周一限尾号 1 和 6，周二限 2 和 7，周三限 3 和 8，周四限 4 和 9，周五限 5 和 0，周六日不限号，编程要求：输入周：1～7，显示要限制的尾号。运行过程如下：

```
今天是星期几(1～7)：3↙
星期三限尾号 3 和 8
今天是星期几(1～7)：6↙
周六日不限号
```

4. 判断用户输入的数据，若数值大于 0、等于 0、小于 0，分别在屏幕上显示“正数”“零”“负数”。运行过程如下：

```
输入数据：2↙
正数
```

5. 编程判断某一年是否是闰年。提示：若年份能被 400 整除，或能被 4 整除但不能被 100 整除，则该年份为闰年。运行过程如下：

```
输入年份：2024↙
闰年
输入年份：2025↙
非闰年
```