第 3 章　分支程序设计

到目前为止，读者接触到的程序都是从头执行到尾，中间不出现任何特殊情况，这是顺序程序的特点。但是，现实问题并不都是这样。有的问题在处理的过程中，某些步骤可能根据不同情况分别进行不同的处理；某些步骤可能重复处理。这就是本章要讲述的分支程序，和下一章要讲述的循环程序，也就是结构化程序设计中的分支结构和重复性结构。

3.1　判断成绩是否及格——双分支程序设计

【例 3.1】 编程序，输入某学生本学期程序设计课程成绩，判断并输出他是否及格。

解： 该问题用流程图描述成图 3.1，程序代码如下。

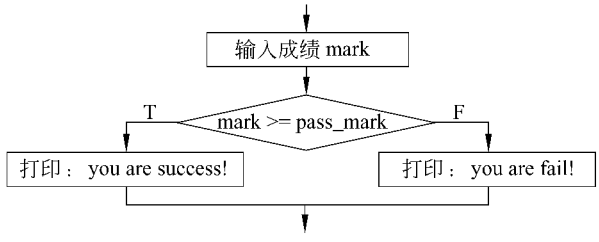

图 3.1　判断成绩是否及格

```
#include <stdio.h>
#define pass_mark  60
void  main(void){
    int  mark;
    printf("please input your mark:");
    scanf("%d",&mark);
    if (mark>=pass_mark)
          printf("you are success!\n");
    else
          printf("you are fail!\n");
}
```

现实程序设计过程中，有很多如例 3.1 的结构，根据不同情况（或根据某条件成立与

否)而进行不同处理并决定具体操作的问题。这种问题用分支程序来描述,分支程序有“单分支”和“双分支”两种。例 3.1 是双分支结构,双分支结构的流程图表示成类似图 3.2 的形式,其 PAD 表示形式如图 3.3 所示。

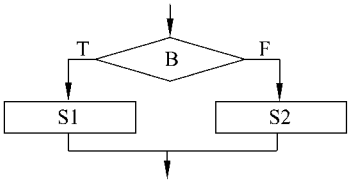

图 3.2　双分支逻辑结构的流程图

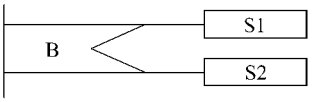

图 3.3　双分支逻辑结构的 PAD 图

图中:

- B 是条件;
- S1、S2 都是具体操作语句。

其含义是:

① 首先计算条件 B;

② 若 B 的值为 true(真),则执行语句 S1 规定的操作,并跳过 S2,执行后继操作;

③ 否则 B 值为 false(假),则跳过 S1,执行语句 S2 规定的操作,然后执行后继操作。

图 3.1 的流程图用 PAD 表示成图 3.4 的形式。

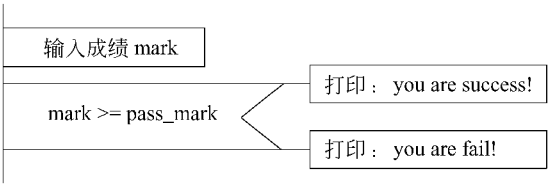

图 3.4　判断成绩是否及格的 PAD 表示

如例 3.1,这种双分支逻辑控制结构在 C 语言中用双分支 if 语句描述。双分支 if 语句形式是:

```
if  (B)
      S1
else
      S2
```

其中,B、S1、S2 的意义,以及该语句的执行过程都如上述图 3.3 的 PAD 的意义。

下述例 3.2、例 3.3 都是双分支逻辑控制结构。

【例 3.2】 计算:

$$MAX(a,b)=\begin{cases}a, & a\geqslant b\\ b, & a<b\end{cases}$$

解:这是求 a、b 极大值问题,用 PAD 描述成图 3.5 的计算过程。写出 C 语言程序片断如下:

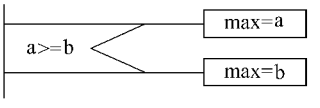

图 3.5　求 A、B 极大值问题的 PAD

```
if  (a>=b)
    max=a;
else
    max=b;
```

【例 3.3】 计算数学中的符号函数

$$\text{sign}(X)=\begin{cases}1, & X>0\\ 0, & X=0\\ -1, & X<0\end{cases}$$

解：该函数用 PAD 可以描述成图 3.6 的计算过程。

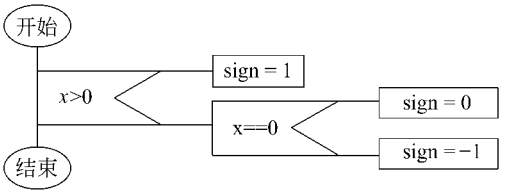

图 3.6　sign 函数的 PAD 图

写出 C 程序片断如下：

```
if (x>0)
    sign=1;
 else  if (x==0)
           sign=0;
      else
           sign=-1;
```

除了双分支的逻辑判断程序结构外，还有一种单分支的逻辑判断程序结构。下例就是单分支的逻辑判断程序。

3.2　成绩加上获奖信息——单分支程序设计

【例 3.4】 学校曾经组织了一次“程序设计大奖赛”，规定本学期程序设计课的成绩可以根据是否在大奖赛上获奖而加 5 分。编程序，计算某同学的程序设计课成绩。

解：该问题用流程图描述成图 3.7，编出程序如下：

```
#include  <stdio.h>
char  win;
int  mark;
void  main(void){
    printf("输入你的考试成绩: ");
    scanf("%d",&mark);
    printf("你是否在程序设计大奖赛获奖(Y/N)?\n");
    scanf("\n%c", &win);
```

```
    if ((win=='Y')||(win=='y'))
        mark=mark+5;
    if  (mark>100)
        mark=100;
    printf("你的最后成绩是：%d\n",mark);
}
```

例 3.4 中程序的结构与例 3.1 相比，逻辑判断过程中“否则”部分没有任何操作。这种结构称单分支的逻辑判断结构。单分支的逻辑判断程序结构在流程图中用图 3.8 表示，在 PAD 图中用图 3.9 的形式表示。

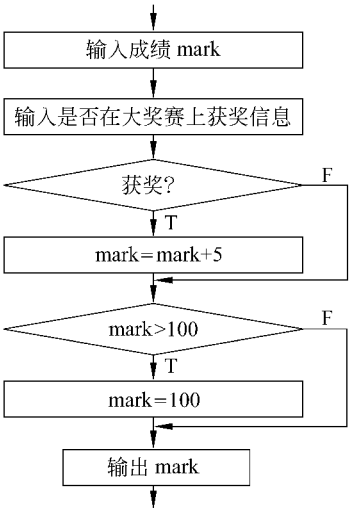

图 3.7　计算成绩

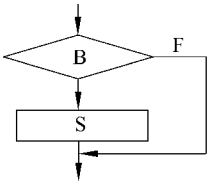

图 3.8　单分支的分支结构的流程图

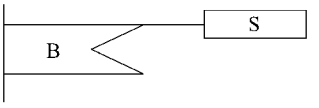

图 3.9　单分支的分支结构的 PAD 图

图中：

- B 是一个条件表达式；
- S 是一个语句。

它的含义是：计算 B 的值，若 B 为 true 则执行语句 S 规定的操作，然后执行后继操作；否则什么也不执行，跳过语句 S，直接去执行后继操作。

图 3.7 的流程图用 PAD 表示成图 3.10 的形式。

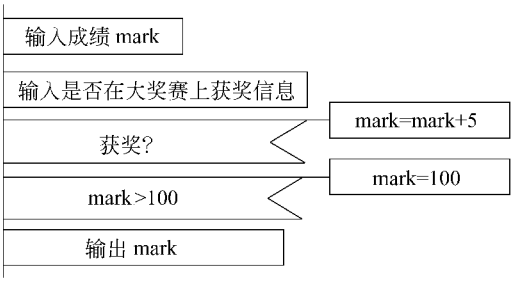

图 3.10　判断成绩是否及格的 PAD 表示

如例 3.4，在 C 中这种单分支的逻辑控制结构用单分支 if 语句表示。单分支 if 语句的形式是：

```
if  (B)
      S
```

其中，B、S 的意义，以及该语句的执行过程都如图 3.9 所示。

【例 3.5】 输入一个年份，判断该年是否闰年。某年是闰年的条件是：能被 4 整除，但不能被 100 整除；或能被 400 整除。

解：该问题的 PAD 如图 3.11 所示，程序如下：

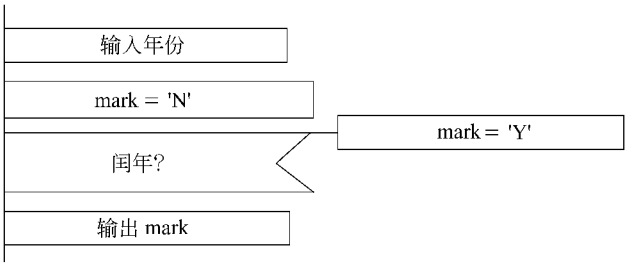

图 3.11 判断闰年

```
#include <stdio.h>
char  mark;
int  year;
void  main(void){
    printf("请输入年份:");
    scanf("%d",&year);
    mark='N';
    if ((year%4==0)&&(year%100!=0)||(year%400==0))
        mark='Y';
    printf("%c\n", mark);
}
```

注意该程序中 mark 值的形成。程序设计中经常使用这种格式，为了形成某变量或某标志单元 v 的值，先给 v 赋值一个初值（该初值是 v 可能取值中的一个）；然后在后续程序中再按不同情况改变 v 的值；最后当这段程序执行结束，便形成了 v 的最终值。

【例 3.6】 求绿化带宽度。如图 3.12 所示，在长 500m、宽 300m 的地域内保护 80000m^2 的地块，求沿四周植树建绿化带的宽度。

解：首先，把问题数学化。若设绿化带的宽度为 x，地块长为 length，宽为 width，保护面积为 area；由数学知识可知：

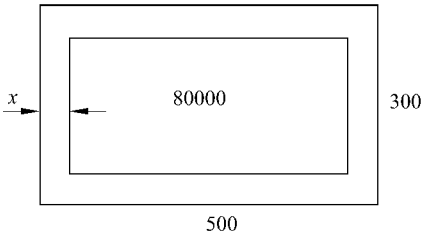

图 3.12 保护地块

```
area=(length-2*x)*(width-2*x)
```

整理后得一元二次方程：

$$4x^2 - 2(\text{length} + \text{width})x + \text{length} * \text{width} - \text{area} = 0$$

然后，找出计算方法。可以使用如下求根公式解该方程。

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

接着，找出算法。该方程求解步骤可用图 3.13 的 PAD 描述，最后，用 C 语言写出程序如下：

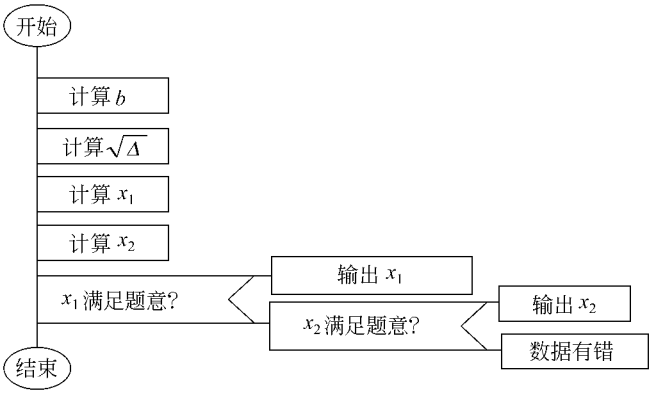

图 3.13　求绿化带宽度

```
#include "stdio.h"
#include "math.h"
void main(){
    float x1,x2,b,d;
    float length, width,area;
    printf("please input length,width,area:\n");
    scanf("%f%f%f",&length,&width,&area);
    b=-2.0*(length+width);                          //计算 b
    d=sqrt(b*b-4.0*4.0*(length*width-area));        //计算√Δ
    x1=(-b+d)/(2*4);                                //求根
    x2=(-b-d)/(2*4);
    if  ((x1>0)&&(2*x1<width)&&(2*x1<length))       //判断根的合理性并输出
        printf("  绿化带宽度为：%.2f\n",x1);
    else  if  ((x2>0)&&(2*x2<width)&&(2*x2<length))
              printf("    绿化带宽度为：%.2f\n",x2);
        else  printf("    原始数据有错误\n");
}
```

该程序运行，若输入数据 500、300、80000，将输出结果：

```
绿化带宽度为：50.00
```

【例 3.7】　求一元二次方程 $ax^2+bx+c=0$ 的根。

解：在例 3.6 中，求解一元二次方程的根直接使用求根公式进行计算，从数学上看该

问题也确实很简单，按公式其解为：

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

只要按公式写出程序即可。但是实际问题并不这样简单，编程序必须考虑各种可能发生的情况，例如 $a=0$ 怎么办？$\Delta<0$ 怎么办？下边按自顶向下逐步求精原则来开发该程序。

首先解一元二次方程 $ax^2+bx+c=0$ 应该先输入系数 a、b、c；然后求解；最后输出。得到 PAD(见图 3.14)。

在图 3.14 中，输入和输出两步很简单。而求解过程可以分解成 $a=0$ 和 $a\neq0$ 两种情况，得图 3.15 的 PAD。

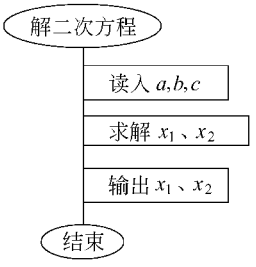

图 3.14 解一元二次方程

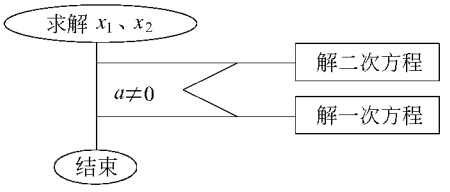

图 3.15 判断 a 是否为 0

下边分别求图 3.15 中的解一次方程和解二次方程。

先解二次方程。解二次方程首先要计算 Δ，然后根据 Δ 值的特性进行计算。包括：

- $\Delta>0$，为两个不等的实数根；
- $\Delta=0$，为两个相等的实数根；
- $\Delta<0$，为两个虚数根。

该计算过程表示成 PAD，如图 3.16 所示。

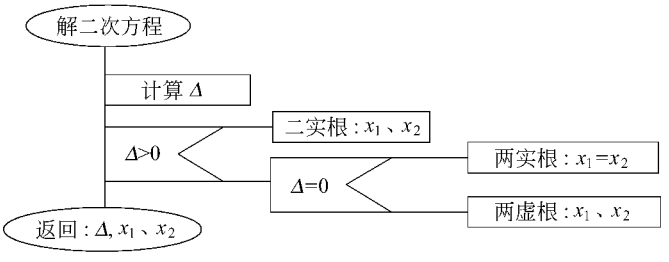

图 3.16 二次方程

下边求解一次方程。解一次方程时同样要考虑 $b=0$ 和 $b\neq0$ 两种情况。当 $b\neq0$ 时，方程的解为 c/b；当 $b=0$ 时，要考察等式 $c=0$。得 PAD，如图 3.17 所示。

继续求考察 $c=0$，应该是：若方程系数 $c=0$ 则为恒等式 $0=0$；若方程系数 $c\neq0$ 则矛盾。得 PAD 如图 3.18 所示。

最后汇总上述分析，并考虑到每步带回信息量太大，不如把打印与求解合并，在求解结束处打印，得到总体解决算法 PAD，如图 3.19 所示。程序如下：

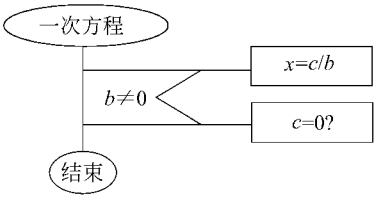

图 3.17　一次方程

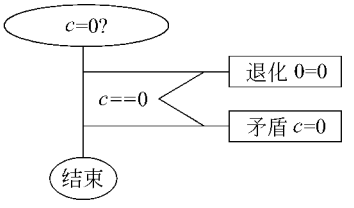

图 3.18　判断 c 是否为 0

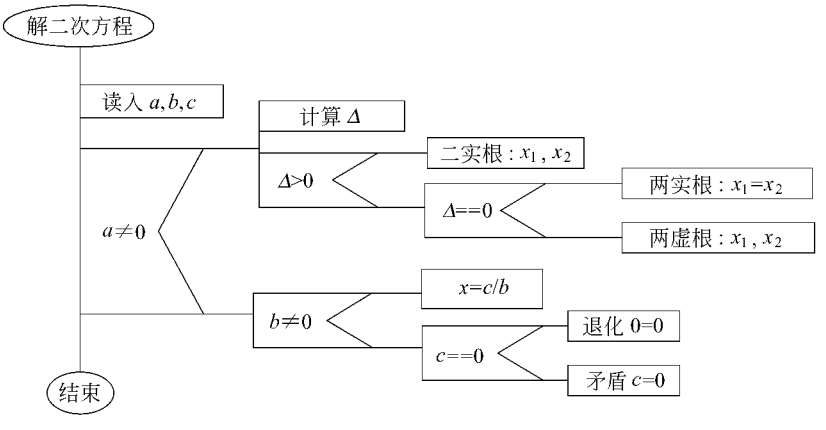

图 3.19　解一元二次方程总体解决算法

```
/* PROGRAM quadratic equation */
#include <stdio.h>
#include <math.h>
void main(){
    float a,b,c,delta;
    /* 读入二次方程的 3 个系数 */
    printf("input the three coefficients of the equation(A,B,C): ");
    scanf("%f%f%f",&a,&b,&c);
    if(a!=0){                                   //二次方程
        delta=b*b-4*a*c;
        if(delta>0)                             //Δ>0 两个不等实数根
            printf("x1=%f,x2=%f\n",
                    (-b+sqrt(delta))/(2*a),(-b-sqrt(delta))/(2*a));
        else
            if(delta==0)                        //Δ=0 两个相等实数根
                printf("x1=x2=%f\n",-b/(2*a));
            else                                //Δ<0 两个虚数根
                printf("x1=%f+%fi, x2=%f-%fi\n",
                        -b/(2*a),sqrt(-delta)/(2*a),
                        -b/(2*a),sqrt(-delta)/(2*a));
    }else
        if(b!=0)                                //一次方程
```

```
            printf("x=%f\n",-c/b);
        else
            if(c==0)                        //0=0
                printf("0=0!\n");
            else                            //矛盾
                printf("%f=0\n",c);
}
```

3.3 逻辑判断——布尔类型

不论双分支的逻辑控制结构，还是单分支的逻辑控制结构，都涉及一个“条件 B”，该条件称为逻辑表达式，也称布尔表达式。该表达式的类型为布尔类型(bool)。

bool 类型仅有两个值：false(假)和 true(真)。在 C 中把 bool 类型也看成整数类型，分别用 0 和 1 表示 false(假)和 true(真)。显然 bool 类型的值域就是由 0 和 1 构成的集合。下边介绍布尔表达式的构成及运算。

3.3.1 关系运算

最基本的布尔表达式由关系运算形成。关系运算比较参与运算的两个量是否满足某一个规定的关系，得到一个 bool 类型值。满足得值 true(真)，不满足得值 false(假)。

前边学习的浮点类型、整数类型、char 类型、正在介绍的 bool 类型以及后续将介绍的枚举类型都属于简单类型(simple-type)，也称标量类型。简单类型有一个共同的特性，即值是可比的(也可以认为是有序的)。并且规定，不论是整数类型(包括了 char 类型、bool 类型、枚举类型)还是浮点类型，都按实数意义上的数值大小进行比较和排序。数值小的排在前边，数值大的排在后边。

1. 关系比较运算

定义了值的顺序之后，就可以对两个值进行大小关系的比较，称为进行关系运算。在 C 中有四个关系运算符可用来对标量类型进行关系运算，它们是：

＜(小于)　＞(大于)　＜＝(小于等于)　＞＝(大于等于)

2. 判等比较运算

还可以对两个值进行相等关系的比较，称为进行判等比较运算。在 C 中有两个判等比较运算符可用来对标量类型进行判等比较运算，它们是：

＝＝(等于)　！＝(不等于)

所有关系运算、判等比较运算都产生 bool 类型结果。这些运算的意义都是明显的，例如：

```
3==3          得  true
3!=3          得  false
5.5<8.1       得  true
'A'>'C'       得  false
```

```
true<false    得  false
true<=true    得  true
```

3.3.2　布尔运算

bool类型仅有两个值：false(假)和true(真)。可施于布尔类型上的运算称“布尔运算”或“逻辑运算”，与数学上逻辑运算相对应，C布尔运算有：

!(非)、&&(与)、||(或)

设b1和b2分别是两个布尔类型量，上述运算的定义如表3.1所示。

表3.1　布尔运算的定义

b1	b2	!b1	b1 && b2	b1 || b2
false	false	true	false	false
false	true		false	true
true	false	false	false	true
true	true		true	true

直观上讲：

- ! 为取反运算。true的反就是false，false的反就是true。
- && 可理解成“并且”。只有两分量都是true时结果才是true，否则结果为false。
- || 可理解成“或者”。两分量只要有一个为true，结果为true，否则结果为false。

总结起来，到目前为止，读者学习了算术运算、关系运算、逻辑运算。按表2.1给出的优先级，在一个混合有各种运算的表达式中，应该先进行算术运算，然后进行关系运算，最后进行逻辑运算。

3.4　获奖分等级——多分支程序设计

例3.4所定义的加分规则过于简单，下边例题重新给出规则。

【例3.8】 学校曾经组织了一次“程序设计大奖赛”，规定本学期程序设计课的成绩可以根据大奖赛的成绩适度加分。加分规则是参赛者加5分，三等奖加15分，二等奖加20分，一等奖加30分，总分不超过100分。编程序，计算某同学的程序设计课成绩。

解：该问题用流程图表示如图3.20所示。C程序代码如下：

```
#include <stdio.h>
int  win;
int  mark;
void  main(void){
    printf("输入你的考试成绩: ");
    scanf("%d",&mark);
    printf("请选择你参加程序设计大奖赛情况\n");
```