第3章

选择结构程序设计

CHAPTER 3

脉络导图

案例导读

学习目标

技能目标：

(1) 掌握结构化程序设计的3种基本结构。

(2) 掌握不同的if语句的格式，掌握其实现选择结构的方法。

(3) 掌握使用switch语句的格式及应用方法。

(4) 掌握避免使用选择结构时常见的错误，能熟练编写选择结构程序。

素质目标：

(1) 学会做一个凡事有条理的人，懂得按照事情的计划和顺序来做，学会统筹管理和节约时间，提高学习和办事的效率。

(2) 通过多种选择结构的编程练习，使学生掌握选择结构的概念及多种语句的使用，培养学生将知识应用于解决实际问题的意识。

(3) 通过对易犯错误进行整理和案例分析，培养学生良好的编程习惯，树立问题意识，提升自我的灵活性，从而提升学生自主学习的能力。

技能基础

3.1　选择结构 if 语句

3.1.1　if 语句

if 在英文中的含义是“如果”,也就意味着判断。C 语言用 if 语句可以构成分支结构。它根据给定的条件进行判断,以决定执行某个分支程序段。if 语句的一般格式如下:

```
if(表达式)　语句
```

其中,表达式一般为逻辑表达式或关系表达式。语句可以是一条简单的语句或多条语句,当为多条语句时,需要用“{}”将这些语句括起来,构成复合语句。

if 语句的执行过程:当表达式的值为真(非 0)时,执行语句,否则直接执行 if 语句下面的语句。if 语句的执行流程如图 3-1 所示。

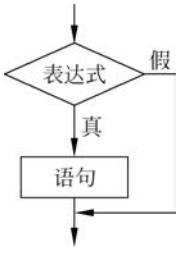

图 3-1　if 语句的执行流程

【例 3-1】　编程实现,输入两个整数,输出这两个数中较大的数。

```
#include <stdio.h>
int main()
{
    int a,b,max;                                    /*定义整型变量 a、b 和 max*/
    printf("请输入两个整数:");                      /*输出屏幕提示*/
    scanf("%d%d",&a,&b);                            /*从键盘输入 a、b 的值*/
    max=a;                                          /*假设 a 是较大的并赋值给 max*/
    if(a<b)                                         /*若 a 比 b 小,则将 b 赋给 max*/
    max=b;
    printf("两数中较大的数为: %d\n",max);           /*输出结果*/
    return 0;                                       /*函数返回值 0*/
}
```

运行结果:

```
请输入两个整数: 5 33
两数中较大的数为: 33
```

程序说明:定义 3 个变量,分别为变量 a、变量 b 和变量 max,用来存放输入的两个整数和较大数。从键盘输入两个整数,首先假设 a 是较大数,将 a 的值赋给 max,然后使用 if 语句进行条件判断,如果 a 小于 b,则 b 为较大数,即将 b 的值赋给 max。

名师点睛

(1) if 后面的表达式必须用“()”括起来。

(2) if 后面的表达式可以是关系表达式、逻辑表达式、算术表达式等。

(3) 表达式中一定要区分赋值运算符“=”和关系运算符“==”。例如,if(x==33)判断x的值是否等于33,而if(x=33)则是把33赋值给x,所以表达式的值为33(非0),即为真。

3.1.2　if-else 语句

if语句只允许在条件为真时指定要执行的语句,而if-else语句还可以在条件为假时指定要执行的语句。if-else语句的一般格式如下:

```
if(表达式)
    语句 1
else
    语句 2
```

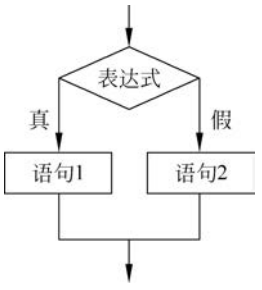

图 3-2　if-else 语句的执行流程

if-else语句的执行过程:当表达式为真(非0)时,执行语句1,否则执行语句2。if-else语句的执行流程如图3-2所示。

名师点睛

(1)“语句1”和“语句2”是“内嵌语句”,它们是if-else语句中的一部分。每个内嵌语句的末尾都应该有分号。

(2) else子句不能作为语句单独使用,它必须是if语句的一部分,与if配对使用。

(3)“语句1”和“语句2”可以是一个简单的语句,也可以是一个包括多个语句的复合语句。

(4) 内嵌语句也可以是一个if语句,这就形成了if嵌套。

【例 3-2】 儿歌“红绿灯,大眼睛,一闪一闪要看清。红灯停,绿灯行,黄灯牢记准备停。”根据输入信号灯s的值,输出车辆通行情况。

```
#include <stdio.h>
int main()
{
    int s;                                  /* 定义整型变量 s 表示交通信号灯 */
    printf("请输入信号灯的值:");            /* 输出屏幕提示 */
    scanf("%d",&s);                         /* 从键盘输入 s 的值 */
    if (s == 1)                             /* s == 1 表示绿灯亮 */
        printf("请车辆有序通行!\n");        /* 输出结果 */
    else                                    /* s 输入其他值,表示红灯亮 */
        printf("请及时停车!\n");            /* 输出结果 */
    return 0;                               /* 函数返回值 0 */
}
```

运行结果:

```
请输入信号灯的值: -3
请及时停车!
```

程序说明:根据输入信号灯的值,输出车辆通行情况。使用if-else语句进行条件判断,如果输入信号灯s的值等于1,条件成立,输出“请车辆有序通行!”。输入其他值else的条

件成立,输出“请及时停车!”。

【例 3-3】 编程实现,输入两个整数,输出这两个数中较大的数(用 if-else 语句实现)。

```
#include <stdio.h>
int main()
{
    int a,b;                                    /*定义整型变量a、b*/
    printf("请输入两个整数:");                  /*输出屏幕提示*/
    scanf("%d%d",&a,&b);                        /*从键盘输入a和b的值*/
    if(a>b)                                     /*若a大于b,则输出a*/
        printf("max=%d\n",a);
    else                                        /*否则输出b*/
        printf("max=%d\n",b);
    return 0;                                   /*函数返回值0*/
}
```

运行结果:

```
请输入两个整数: 5 33
max=33
```

程序说明:使用 if-else 语句进行条件判断,如果 a 大于 b,则 a 为较大数,if 条件成立,输出 a 的值;否则 b 为较大数,else 条件成立,输出 b 的值。

3.1.3 if-else-if 语句

编程时常常需要判定一系列的条件,一旦其中某一个条件为真就立刻停止。这种情况可以采用 if-else-if 语句,其一般格式如下:

```
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 if(表达式n)         语句n
else                     语句n+1
```

if-else-if 语句的执行过程: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 语句之外继续执行程序。如果所有的表达式都为假,则执行最后一个 else 后的语句,然后继续执行后续程序。if-else-if 语句的执行流程如图 3-3 所示。

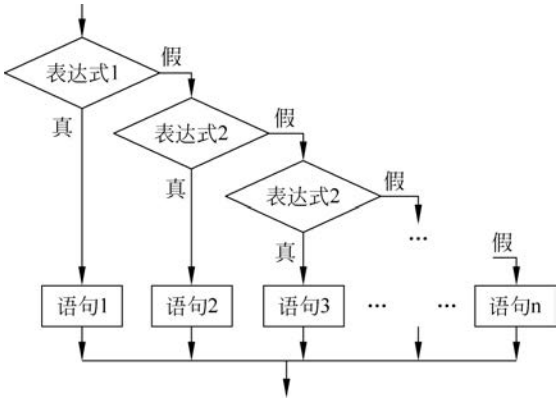

图 3-3 if-else-if 语句的执行流程

【例 3-4】　在例 3-2 的基础上，完善信号灯在红、绿、黄 3 种情况下车辆的通行情况。

```
#include<stdio.h>
int main()
{
    int s;                                  /*定义整型变量 s 表示交通信号灯*/
    printf("请输入信号灯 s 的值:");          /*输出屏幕提示*/
    scanf("%d",&s);                         /*从键盘输入 s 的值*/
    if (s==1)                               /*s==1 表示绿灯亮*/
        printf("请车辆有序通行!\n");         /*输出结果*/
    else if(s==0)                           /*s==0 表示红灯亮*/
        printf("请及时停车!\n");             /*输出结果*/
    else                                    /*s 输入其他值表示黄灯亮*/
        printf("黄灯亮,请准备停车。\n");     /*输出结果*/
    return 0;                               /*函数返回值*/
}
```

运行结果：

```
请输入信号灯 s 的值: -3
黄灯亮,请准备停车。
```

程序说明：本例的功能与例 3-2 相同，都是根据输入信号灯的值，输出车辆通行情况。使用 if-else-if 语句进行条件判断，假设 s==1 表示绿灯亮，s==0 表示红灯亮，其他值表示黄灯亮。当输入 s 的值为 1 时，if 语句条件成立，输出“请车辆有序通行!”；当输入 s 的值为 0 时，else if 语句条件成立，输出“请及时停车!”；当输入其他值时，输出“黄灯亮，请准备停车。”

【例 3-5】　学生成绩可分为百分制和五级制，将输入的百分制成绩 score，转换为相应的五级制成绩后输出。百分制与五级制成绩的对应关系如表 3-1 所示。

表 3-1　百分制与五级制成绩的对应表

百　分　制	五级制	百　分　制	五级制
score>100 或 score<0	无意义	70≤score<80	中等
90≤score≤100	优秀	60≤score<70	及格
80≤score<90	良好	0≤score<60	不及格

```
#include<stdio.h>
int main()
{
    int score;                              /*定义表示成绩整型 score*/
    printf("请输入学生成绩:");               /*输出屏幕提示*/
    scanf("%d",&score);                     /*从键盘输入百分制成绩*/
    if(score>100||score<0)                  /*输入分数不合理时提示错误信息*/
        printf("您输入的成绩不正确!\n");
    else if(score>=90)                      /*这里的 else 表示 0=<score&&score<=100*/
        printf("优秀\n");
    else if(score>=80)                      /*这里的 else 表示 0=<score&&score<=90*/
        printf("良好\n");
    else if(score>=70)                      /*这里的 else 表示 0=<score&&score<=80*/
        printf("中等\n");
```

```
    else if(score >= 60)                    /* 这里的 else 表示 0 =< score&&score <= 70 */
          printf("及格\n");
    else                                    /* 这里的 else 表示 0 =< score&&score <= 60 */
          printf("不及格\n");
return 0;                                   /* 函数返回值 */
}
```

运行结果:

```
请输入学生成绩: 95
优秀
```

程序说明:这是一道典型的能够使用 if-else-if 语句的题目,根据对一系列互斥条件的判断来选择执行哪条语句。每个 else 本身都隐含了一个条件,如第 1 个 else 实质上表示条件 score>=0&& score<=100 成立,此隐含条件与对应的 if 所给出的条件完全相反。在编程时要善于利用隐含条件,使程序代码清晰简洁。

3.1.4 if 语句的嵌套

if 语句的嵌套是指在 if 语句中又包括一个或多个 if 语句。内嵌的 if 语句可以嵌套在 if 子句中,也可嵌套在 else 子句中。

(1) 在 if 子句中嵌套具有 else 子句 if 语句。

```
if(表达式 1)
    if(表达式 2)     语句 1
    else             语句 2
else
    语句 3
```

当表达 1 的值为非 0 时,执行内嵌的 if-else 语句;当表达式 1 的值为 0 时,执行语句 3。

(2) 在 if 子句中嵌套不含 else 子句的 if 语句。

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

用“{}”把内层 if 语句括起来,在语法上成为一条独立的语句,使得 else 与外层的 if 配对。

(3) 在 else 子句中嵌套具有 else 子句的 if 语句。

```
if(表达式 1)       语句 1
else if(表达式 2)  语句 2
     else          语句 3
```

第 2 个 if 语句作为第 1 个 if 表达式 1 不成立时的执行语句。当表达式 2 成立时执行语句 2,不成立时执行语句 3。

(4) 在 else 子句中嵌套不含 else 子句的 if 语句。

```
if(表达式 1)       语句 1
else if(表达式 2)  语句 2
```

第 2 个 if 语句作为第 1 个 if 表达式 1 不成立时的执行语句。当表达式 2 成立时执行语句 2，不成立时什么都不执行。

【例 3-6】 编写程序，实现输入 3 个整数，输出最大值。

```
#include <stdio.h>
int main()
{
    int a,b,c,max;                          /* 定义变量 */
    printf("请输入 3 个整数: \n");          /* 输出提示信息 */
    scanf("%d%d%d",&a,&b,&c);               /* 输出提示信息 */
    if(a>b)                                 /* a>b */
        if(a>c)                             /* a>b 并且 a>c,最大值为 a */
            max=a;
        else                                /* a>b 并且 c>a,最大值为 c */
            max=c;
    else                                    /* a<b */
        if(b>c)                             /* b>a 并且 b>c,最大值为 b */
            max=b;
        else                                /* b>a 并且 c>b,最大值为 c */
            max=c;
    printf("max=%d\n",max);                 /* 输出最大值 max */
    return 0;                               /* 函数返回值 */
}
```

运行结果：

```
请输入 3 个整数:
8 33 -15
max=33
```

程序说明：本题可以采用 if 嵌套进行实现，先比较 a 和 b 的大小，如果 a 大于 b，就将 a 与 c 进行比较，如果 a 也大于 c，那么最大值就为 a；否则，最大值为 c。如果 a 小于 b，就将 b 与 c 进行比较，如果 b 大于 c，那么最大值就为 b；否则，最大值为 c。

3.1.5　if 与 else 的配对规则

if 语句在出现嵌套形式时，初学者往往会弄错 if 与 else 的配对关系，特别是当 if 与 else 的数量不对等时。因此，必须掌握 if 与 else 的配对规则。C 语言规定 else 与其上面最接近它，还未与其他 else 语句配对的 if 语句配对。if 与 else 的配对规则如图 3-4 所示。

同时从书写格式上也要注意程序的层次感，优秀的程序员应该养成这种习惯，以便他人阅读和自己修改程序。注意，书写格式不能代替程序逻辑。

如果 if 的个数与 else 的个数相同，则从内层到外层一一对应；而 if 与 else 的数量不一致时，为体现编程者的意图，可在需要时添加“{}”来强制确定配对关系，否则就不能实现编程者的真正意图。

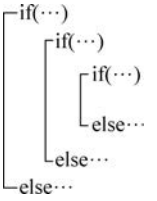

图 3-4　if 与 else 的配对规则

名师点睛

else 总是与它前面最近的且尚未与其他 else 配对的 if 配对。

3.2 选择结构 switch 语句

3.2.1 switch 语句的一般格式

if 语句只能实现两路分支,在两种情况中选择其一执行。虽然嵌套的 if 语句可以实现多路的检验,但有时不够简洁。某些程序运行中多达数个分支,用 if-else 语句可以根据条件沿不同支路向下执行,但程序的层次太多,显得烦琐,在一定程度上影响了可读性。为此 C 语言提供了实现多路选择的另一种语句——switch 语句,称为开关体语句。switch 语句的一般格式如下:

```
switch(表达式)
{
    case 常量表达式 1:     语句 1;
    case 常量表达式 2:     语句 2;
    …
    case 常量表达式 n:     语句 n;
    default:               语句 n+1;
}
```

switch 语句的执行过程:先计算 switch 后面表达式的值,与某个 case 后面常量表达式的值相等时,就执行此 case 后面的所有语句,直到遇到 break 语句或 switch 语句的结束符"}"才结束。如果 case 后无 break 语句,则不再进行判断,继续执行随后所有的 case 后面的语句。如果没有找到与此值相匹配的常量表达式,则执行 default 后的语句 n+1;若无 default 子句,则执行 switch 语句后面的其他语句。

名师点睛

在使用 switch 语句时应注意以下 6 点。

(1) switch 后的表达式和 case 后的常量表达式可以是整型、字符型、枚举型,但不能是实型。

(2) 同一个 switch 语句中,各 case 后的常量表达式的值必须互不相同,否则会出现多种执行方案。

(3) case 后的语句可以是一条语句,也可以是多条语句,此时多条语句不必用大括号"{}"括起来。同一个 switch 语句中,各 case 后的常量表达式的值必须互不相同,否则会出现多种执行方案。

(4) default 可以省略,省略时如果没有与 switch 表达式相匹配的 case 常量,则不执行任何语句,程序转到 switch 语句后的下一条语句执行。

(5) 各 case 和 default 子句的先后顺序可以改变,不影响执行结果。

(6) 如果多种情况都执行相同的程序块,则对应的多个 case 可以执行同一语句。

3.2.2 switch 语句的应用

【例 3-7】 春节是我国重要的传统节日之一。春节的饮食有很多讲究,北方民谚曰:

“初一饺子初二面，初三合子往家转，初四烙饼炒鸡蛋，初五初六捏面团，初七、初八炒年糕，初九初十白米饭，十一十二八宝粥，十三十四汆汤丸，正月十五元宵圆。”

编程实现，从键盘上输入 1～15 的数字，显示对应正月初一到正月十五要吃的美食，当输入数字不在 1～15 范围时，输出“年过完了，撸起袖子加油干!”。

```
#include <stdio.h>
int main()
{
    int date;                                   /* 定义表示初几的整型 date */
    printf("今天初几啊?\n");                    /* 输出屏幕提示 */
    scanf("%d",&date);                          /* 从键盘输入日期 */
    switch(date)                                /* switch 语句判断 */
    {
        case 1: printf("吃饺子\n"); break;      /* date 值为 1 时,输出语句并跳出 switch */
        case 2: printf("吃面条\n"); break;      /* date 值为 2 时,输出语句并跳出 switch */
        case 3: printf("吃合子\n"); break;      /* date 值为 3 时,输出语句并跳出 switch */
        case 4: printf("烙饼炒鸡蛋\n"); break; /* date 值为 4 时,输出语句并跳出 switch */
        case 5:                                 /* date 值为 5、6 时,输出语句并跳出 switch */
        case 6: printf("捏面团\n"); break;
        case 7:                                 /* date 值为 7、8 时,输出语句并跳出 switch */
        case 8: printf("吃炒年糕\n"); break;
        case 9:                                 /* date 值为 9、10 时,输出语句并跳出 switch */
        case 10: printf("吃白米饭\n"); break;
        case 11:                                /* date 值为 11、12 时,输出语句并跳出 switch */
        case 12: printf("吃八宝粥\n"); break;
        case 13:                                /* date 值为 13、14 时,输出语句并跳出 switch */
        case 14: printf("汆汤丸\n"); break;
        case 15: printf("吃元宵\n"); break;     /* date 值为 15 时,输出语句并跳出 switch */
        default: printf("年过完了,撸起袖子加油干!\n"); break;
                                                /* date 值为其他时,输出语句并跳出 switch */
    }
}
```

运行结果：

```
今天初几啊?
16
年过完了,撸起袖子加油干!
```

程序说明：这是一道可以利用多分支选择语句的题目，定义整型变量 date，使用 switch 语句判断整型变量 date 的值，利用 case 语句检验 date 值的不同情况；如果 date 的值不是 case 中所检验列出的情况，则输出“年过完了，撸起袖子加油干!”。在 switch 语句中，“case 常量表达式”只相当于一个语句标号，表达式的值和某标号相等则转向该标号执行，但不能在执行完该标号的语句后自动跳出整个 switch 语句，所以出现了继续执行所有后面 case 语句的情况。这与前面介绍的 if 语句是完全不同的，应特别注意。

为了避免上述情况，C 语言还提供了 break 语句，专用于跳出 switch 语句，break 语句只有关键字 break，没有参数。此部分内容将在后面详细介绍。

【例 3-8】 “十二生肖”也称“十二属相”，是我国传统文化中使用最广、影响最深的文化现象之一。所谓十二生肖，是古人将十二地支与十二种动物相配，用于记录历史年份的一种

形式。因其使用的鼠、牛、虎、兔、龙、蛇、马、羊、猴、鸡、狗、猪大部分都是现实中的实生动物,故称“十二生肖”。

编程实现,从键盘上输入年份,输出对应的生肖。

```
#include <stdio.h>
int main()
{
    int year;                                          /*定义表示年份的整型 year*/
     printf("请输入年份:");                             /*输出屏幕提示*/
     scanf("%d",&year);                                /*从键盘输入年份*/
     printf("公元%d年是:",year);
     switch((year+9)%12)                               /*switch 语句判断*/
     {
          case 0: printf("猪年\n"); break;             /*值为 0 时,输出语句并跳出 switch*/
          case 1: printf("鼠年\n"); break;             /*值为 1 时,输出语句并跳出 switch*/
          case 2: printf("牛年\n"); break;             /*值为 2 时,输出语句并跳出 switch*/
          case 3: printf("虎年\n"); break;             /*值为 3 时,输出语句并跳出 switch*/
          case 4: printf("兔年\n"); break;             /*值为 4 时,输出语句并跳出 switch*/
          case 5: printf("龙年\n"); break;             /*值为 5 时,输出语句并跳出 switch*/
          case 6: printf("蛇年\n"); break;             /*值为 6 时,输出语句并跳出 switch*/
          case 7: printf("马年\n"); break;             /*值为 7 时,输出语句并跳出 switch*/
          case 8: printf("羊年\n"); break;             /*值为 8 时,输出语句并跳出 switch*/
          case 9: printf("猴年\n"); break;             /*值为 9 时,输出语句并跳出 switch*/
          case 10: printf("鸡年\n"); break;            /*值为 10 时,输出语句并跳出 switch*/
          case 11: printf("狗年\n"); break;            /*值为 11 时,输出语句并跳出 switch*/
          default: printf("输入错误!\n"); break;      /*其他数值,输出语句并跳出 switch*/
     }
}
```

运行结果:

```
请输入年份: 2022
公元 2022 年是: 虎年
```

程序说明:如果能计算出输入年份在一个生肖周期中的顺序号,那么马上就能知道这一年的生肖。现已知公元 1 年是鸡年,鸡在生肖中的序号是 10,与公元 1 年相差 9,因此先将年份加上 9 再对 12 取余,得到的余数就正好是这一年在生肖周期中的顺序号,余数为 0 时顺序号为 12。

3.3 常见错误分析

3.3.1 误把“=”作为“等于”运算符

很多初学者习惯性地用数学上的等于号“=”用作 C 语言关系运算符“等于”。

【例 3-9】 错误使用“等于”运算符。

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

```
    scanf("%d",&a);
    if(a=1)                /*误把"="用作"等于"运算符*/
        printf("成功!\n");
    else
        printf("失败!\n");
    return 0;
}
```

错误分析：这种写法在程序编译过程中没有任何报错信息，但是实际上无法实现对变量 a 数值的判断功能。此程序无论输入的 a 值是否为 1，都输出“成功!”。

C 语言中“==”是关系运算符，用来判断两个数是否相等，a==1 是判断 a 的值是否为 1；“=”是赋值运算符，a=1 是使 a 的值为 1，这时不管 a 原来是什么值，表达式的值永远为真。因此，该程序需要将 if(a=1)修改为 if(a==1)。

3.3.2　忘记必要的逻辑运算符

在数学领域中，想要判断一个数是否为(3,6)，可以直接用 3<x<6 进行表示。对于初学者来说，很容易将其应用到 C 语言的编程中。

【例 3-10】 错误使用逻辑运算符。

```
#include<stdio.h>
int main()
{
    int x;
    scanf("%d",&x);
    if (3<x<6)                    /*忘记必要的逻辑运算符*/
        printf("成功!\n");
    else
        printf("失败!\n");
    return 0;
}
```

错误分析：该程序在编译时可以顺利通过，但是无法实现对 x 数值的判断功能。例如，输入 x 的值为 7，不满足大于 3 小于 6 的条件，但是输出还是“成功!”。

C 语言中，关系运算符的结合性为从左至右。3<x<6 的求值顺序：先计算 3<x，得到一个逻辑值 0 或 1，再拿这个数与 6 作比较，结果恒为真，失去了比较的意义。对于这种情况，应使用逻辑表达式，写成 if((3<x) && (x<6))。

3.3.3　用复合语句时漏掉大括号

【例 3-11】 漏掉大括号。

```
#include<stdio.h>
int main()
{
    int a,b,t;
    scanf("%d,%d",&a,&b);
    if(a>b)                    /*用复合语句时漏掉大括号*/
        t=a;
```

```
            a = b;
            b = t;
        printf("a = %d,b = %d\n",a,b);
        return 0;
}
```

错误分析：这种写法在程序编程过程中，没有任何报错信息，但是运行结果是错误的。由于 if 后面没有大括号，因此，if 只作用于“t=a;”这一条语句，而不管 a>b 是否为真，都将执行后两条语句，正确的写法应为：

```
if(a > b)
{
    t = a;
    a = b;
    b = t;
}
```

3.3.4　在不该加分号的地方加分号

if(表达式)后是没有分号的，如果误加了分号，在程序编译过程中，并不会报错，但是无法实现预定的目标。

【例 3-12】 if 表达式后多加分号。

```
#include < stdio.h >
int main()
{
     int a,b,t = 0;
     scanf("%d,%d",&a,&b);
     if(a == b);                    /* 在不该加分号的地方加分号 */
     t = a + b;
     printf("%d\n,t");
return 0;
}
```

错误分析：程序的本意是如果 a 等于 b，则执行 t=a+b，但由于 if(a+b)后跟有分号，语句“t=a+b;”在任何情况下都执行，即当 a 不等于 b 时程序也会运行语句“t=a+b;”。这是因为 if 后加分号相当于 if 后跟了一个空语句。正确的写法应为“if(a==b) t=a+b;”。


视频讲解

3.4　多分支选择结构程序设计应用实战

3.4.1　实战背景

东汉班固《白虎通义》言：“华山为西岳者，华之为获也。万物生华，故曰华山。”华山之阳，黄河东流；其阴，坐拥秦岭。险峻奇绝，冠绝五岳。华山景区向全国游客实行门票优惠政策：对身高 120cm 及以下儿童实行门票免费；对身高 120～150cm 的儿童实行门票半价优惠。

3.4.2　实战目的

(1) 变量的定义与使用。

(2) 多分支 if-else if-else 语句的应用。

3.4.3　实战内容

编写一个 C 语言程序,输入不同身高,计算输出需要支付的门票价格。

3.4.4　实战过程

```
#include <stdio.h>
int main ()
{
     int height,price;
     printf ("请输入身高(cm):");
     scanf ("%d",&height);
     if (height >= 150)
          price = 40;
     else if (height >= 120&&height <= 150)
          price = 20;
     else
          price = 0;
     printf("您的身高: %dcm,您需要支付: %d 元\n",height,price);
     return 0;
}
```

技能实战运行结果如图 3-5 所示。

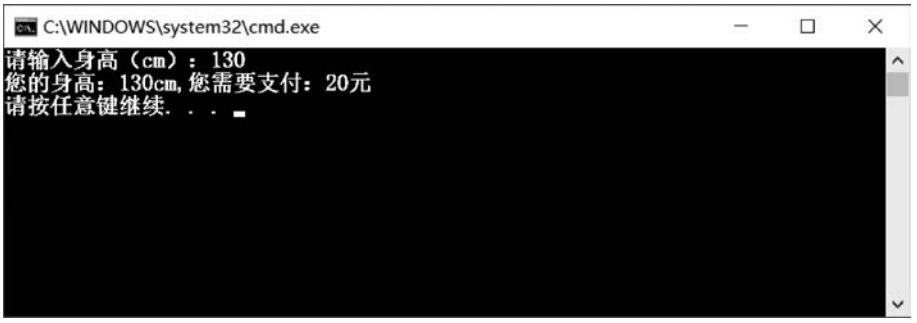

图 3-5　技能实战运行结果

3.4.5　实战意义

通过实战,掌握多分支选择语句的使用方法。

大家应该有敢于突破前人的勇气和智慧,自觉克服安于现状、不思进取的思想观念,坚持用创新的理论成果武装头脑,与时俱进,开拓创新,做出自己应有的贡献。