第3章　最简单的C程序设计
——顺序程序设计

1. 假如我国国民生产总值的年增长率为7%，计算10年后我国国民生产总值与现在相比增长多少百分比。计算公式为

$$p = (1+r)^n$$

r 为年增长率，n 为年数，p 为与现在相比的倍数。

解：从主教材附录D(库函数)可以查到：可以用pow函数求 y^x 的值，调用pow函数的具体形式是pow(x,y)。在使用pow函数时需要在程序的开头用#include指令将<math.h>头文件包含到本程序模块中。可以用下面的程序求出10年后国民生产总值是现在的多少倍。

```
#include <stdio.h>
#include <math.h>
int main( )
  {float p,r,n;
   r=0.07;
   n=10;
   p=pow(1+r,n);
   printf("p=%f\n",p);
   return 0;
  }
```

运行结果：

```
p=1.967151
```

即10年后国民生产总值是现在的1.967151倍。

2. 存款利息的计算。有1000元，想存5年，可按以下5种办法存：

(1) 一次存5年期。

(2) 先存2年期，到期后将本息再存3年期。

(3) 先存3年期，到期后将本息再存2年期。

(4) 存1年期，到期后将本息存再存1年期，连续存5次。

(5) 存活期存款。活期利息每一季度结算一次。

2017年的银行存款利息如下：

1年期定期存款利息为1.5%；

2年期定期存款利息为2.1%；

3年期定期存款利息为2.75%；

5年期定期存款利息为3%；

活期存款利息为 0.35%(活期存款每一季度结算一次利息)。

如果 r 为年利率,n 为存款年数,则计算本息和的公式为

1 年期本息和：$p=1000\times(1+r)$；

n 年期本息和：$p=1000\times(1+n\times r)$；

存 n 次 1 年期的本息和：$p=1000\times(1+r)^n$；

活期存款本息和：$p=1000\times\left(1+\frac{r}{4}\right)^{4n}$。

说明：$1000\times\left(1+\frac{r}{4}\right)$是一个季度的本息和。

解：设 5 年期存款的年利率为 r5,3 年期存款的年利率为 r3，2 年期存款的年利率为 r2，1 年期存款的年利率为 r1，活期存款的年利率为 r0。

设按第 1 种方案存款 5 年得到的本息和为 p1,按第 2 种方案存款 5 年得到的本息和为 p2,按第 3 种方案存款 5 年得到的本息和为 p3,按第 4 种方案存款 5 年得到的本息和为 p4,按第 5 种方案存款 5 年得到的本息和为 p5。

程序如下：

```
#include <stdio.h>
#include <math.h>
int main( )
  {float r5,r3,r2,r1,r0,p,p1,p2,p3,p4,p5;
   p=1000;
   r5=0.03;
   r3=0.0275;
   r2=0.021;
   r1=0.015;
   r0=0.0035;

   p1=p*(1+r5*5);                    //一次存 5 年期
   p2=p*(1+2*r2)*(1+3*r3);           //先存 2 年期,到期后将本息再存 3 年期
   p3=p*(1+3*r3)*(1+2*r2);           //先存 3 年期,到期后将本息再存 2 年期
   p4=p*pow(1+r1,5);                 //存 1 年期,到期后将本息再存 1 年期,连续存 5 次
   p5=p*pow(1+r0/4,4*5);             //存活期存款,活期利息每一季度结算一次
   printf("p1=%f\n",p1);             //输出按第 1 种方案得到的本息和
   printf("p2=%f\n",p2);             //输出按第 2 种方案得到的本息和
   printf("p3=%f\n",p3);             //输出按第 3 种方案得到的本息和
   printf("p4=%f\n",p4);             //输出按第 4 种方案得到的本息和
   printf("p5=%f\n",p5);             //输出按第 5 种方案得到的本息和
   return 0;
  }
```

运行结果：

```
p1=1150.000000
p2=1127.964966
p3=1127.964966
p4=1077.284058
p5=1017.646240
```

讨论：

(1) 程序在编译时出现警告(warning)，并告知原因是“'='：truncation from 'const double ' to 'float'”(在执行赋值时，出现将双精度常量转换为单精度的情况)。这是由于Visual C++ 6.0在编译时把实常数(如程序中的利率)全部按双精度数处理，因此在向r5，r3等float型变量赋值时，就出现将双精度数赋给单精度变量的情况，这样可能会损失一些精度，故向用户提醒，请用户考虑是否要修改。警告只是提醒，程序可以正常运行，但得到的结果可能会出现一些误差，如果用户认为误差可以容忍，可不理会警告，继续进行连接和运行。

(2) 如果不想出现上面的警告，可以将第4行各变量改为double型，即

```
double r5,r3,r2,r1,r0,p,p1,p2,p3,p4,p5;
```

由于采用了双精度变量，得到的运算结果会更精确些，最后几位数字与上面的有些差别。

```
p1=1150.000000
p2=1127.965000
p3=1127.965000
p4=1077.284004
p5=1017.646235
```

(3) 输出运行结果时，得到6位小数，连同整数部分有10位数字，而一个float型变量只能保证6位有效数字，后面几位是无意义的。而且在输出款额时，人们一般只要求精确到两位小数(角、分)，因此可以在printf函数中用%10.2格式符输出。最后5个语句可改为

```
printf("p1=%10.2f\n",p1);          //输出按第1种方案得到的本息和
printf("p2=%10.2f\n",p2);          //输出按第2种方案得到的本息和
printf("p3=%10.2f\n",p3);          //输出按第3种方案得到的本息和
printf("p4=%10.2f\n",p4);          //输出按第4种方案得到的本息和
printf("p5=%10.2f\n",p5);          //输出按第5种方案得到的本息和
```

这时的输出结果如下：

```
p1=1150.00
p2=1127.96
p3=1127.96
p4=1077.28
p5=1017.65
```

3. 购房从银行贷了一笔款d，准备每月还款额为p，月利率为r，计算多少月能还清。设d为300 000元，p为6000元，r为1%。对求得的月份取小数点后一位，对第2位小数按四舍五入处理。

提示：计算还清月数m的公式如下：

$$m=\frac{\lg p-\lg(p-d\times r)}{\lg(1+r)}$$

可以将公式改写为

$$m=\frac{\lg\dfrac{p}{p-d\times r}}{\lg(1+r)}$$

C的库函数中有求对数的函数lg10，是求以10为底的对数，lg(p)表示lg$p$。

解：根据以上公式可以很容易写出以下程序。

```
#include <stdio.h>
#include <math.h>
int main( )
  {float d=300000,p=6000,r=0.01,m;
   m=lg10(p/(p-d*r))/lg10(1+r);
   printf("m=%6.1f\n",m);
   return 0;
  }
```

运行结果：

```
m=  69.7
```

即需要 69.7 个月才能还清。为了验证对第 2 位小数是否已按四舍五入处理，可以将程序第 6 行中的“%6.1f”改为“%6.2f”。此时的输出为

```
m= 69.66
```

可知前面的输出结果是对第 2 位小数按四舍五入处理的。

4. 分析下面的程序：

```
#include <stdio.h>
int main( )
  {char  c1,c2;
   c1=97;
   c2=98;
   printf("c1=%c,c2=%c\n"c1,c2);
   printf("%c1=%d,c2=%d\n",c1,c2);
   return 0;
  }
```

(1) 运行时会输出什么信息？为什么？

解：**运行时输出**

```
c1=a,c2=b
c1=97, c2=98
```

第 1 行是将 c1，c2 按%c 的格式输出，97 是字符 a 的 ASCII 码，98 是字符 b 的 ASCII 码。

第 1 行是将 c1，c2 按%d 的格式输出，所以输出两个十进制整数。

(2) 如果将程序第 4，5 行改为

```
c1=197;
c2=198;
```

运行时会输出什么信息？为什么？

解：由于 Visual C++ 6.0 字符型数据是作为 signed char 类型处理的，它存字符的有效范围为 0～127，超过此范围的处理方法，不同的系统得到的结果不同，因而用%c 格式输出

时,结果是不可预料的。

用%d格式输出时,输出c1=-59,c2=-58。这是按补码形式输出的,内存字节中第1位为1时,作为负数。59与197之和等于256,58与198之和也等于256。对此可暂不深究。

只要知道:用char类型变量时,给它赋的值应在0~127范围内。

(3) 如果将程序第3行改为

```
int c1,c2;
```

运行时会输出什么信息?为什么?

解:如果给c1和c2赋的值是97和98,则输出结果与(1)相同。

如果给c1和c2赋的值是197和198,则用%c输出时是不可预料的字符。用%d输出时,输出整数197和198,因为它们在int类型的有效范围内。

5. 用下面的scanf函数输入数据,使a=3,b=7,x=8.5,y=71.82,c1='A',c2='a'。问在键盘上如何输入。

```
#include <stdio.h>
int main( )
  {int a,b;
   float x,y;
   char c1,c2;
   scanf("a=%d b=%d",&a,&b);
   scanf("%f %e",&x,&y);
   scanf("%c%c",&c1,&c2);
   printf("a=%d,b=%d,x=%f,y=%f,c1=%c,c2=%c\n",a,b,x,y,c1,c2);
   return 0;
  }
```

解:按如下方式在键盘上输入(见下面第1,2两行)。

```
a=3 b=7
8.5 71.82Aa
a=3,b=7,x=8.500000,y=71.820000,c1=A,c2=a
```

第3行是输出的结果。

注意:在输入8.5和71.82两个实数给x和y后,应紧接着输入字符A,中间不要有空格,由于A是字母而不是数字,系统在遇到字母A时就确定输入给y的数值已结束。字符A就送到下一个scanf语句中的字符变量c1。如果在输入8.5和71.82两个实数后输入空格符,会怎么样呢?情况如下:

```
a=3 b=7
8.5 71.82 Aa
a=3,b=7,x=8.500000,y=71.820000,c1= ,c2=A
```

这时71.82后面输入的空格字符就被c1读入,c2读入了字符A。在输出c1时就输出空格,输出c2的值为A。

如果在输入8.5和71.82两个实数后按回车键,会怎么样呢?情况如下:

```
a=3 b=7
8.5 71.82
Aa
a=3,b=7,x=8.500000,y=71.820000,c1=
,c2=A
```

上面 3 行是输入，在输入 71.82 后按回车键。在这时“回车”被作为一个字符送到内存输入缓冲区，被 c1 读入（实际上 c1 读入的是回车符的 ASCII 码），字符 A 被 c2 读取，所以在执行 printf 函数输出 c1 时，就输出一个换行，在下一行输出逗号和 c2 的值 A。

在用 scanf 函数输入数据时往往会出现一些意想不到的情况，例如在连续输入不同类型的数据（特别是数值型数据和字符数据连续输入）的情况。要注意回车符是可能被作为一个字符读入的。

通过此例，可以了解怎样正确进行输入数据。这些知识不能靠枯燥地死记规则，必须善于在实践中注意分析现象，不断总结经验。

6. 请编程序将 China 译成密码，密码规律是：用原来的字母后面第 4 个字母代替原来的字母。例如，字母 A 后面第 4 个字母是 E，用 E 代替 A。因此，China 应译为 Glmre。请编一程序，用赋初值的方法使 c1，c2，c3，c4，c5 这 5 个变量的值分别为'C'，'h'，'i'，'n'，'a'，经过运算，使 c1，c2，c3，c4，c5 分别变为'G'，'l'，'m'，'r'，'e'。分别用 putchar 函数和 printf 函数输出这 5 个字符。

解：

```
#include <stdio.h>
int main( )
  {char c1='C',c2='h',c3='i',c4='n',c5='a';
   c1=c1+4;
   c2=c2+4;
   c3=c3+4;
   c4=c4+4;
   c5=c5+4;
   printf("password is %c%c%c%c%c\n",c1,c2,c3,c4,c5);
   return 0;
  }
```

运行结果：

```
password is Glmre
```

7. 设圆半径 $r=1.5$，圆柱高 $h=3$，求圆周长、圆面积、圆球表面积、圆球体积、圆柱体积。用 scanf 输入数据，输出计算结果，输出时要求有文字说明，取小数点后 2 位数字。请编程序。

解：

```
#include <stdio.h>
int main ( )
  {float h,r,l,s,sq,vq,vz;
   float pi=3.141526;
   printf("请输入圆半径 r,圆柱高 h:");
```

```
    scanf("%f,%f",&r,&h);                    //要求输入圆半径 r 和圆柱高 h
    l=2*pi*r;                                //计算圆周长 l
    s=r*r*pi;                                //计算圆面积 s
    sq=4*pi*r*r;                             //计算圆球表面积 sq
    vq=3.0/4.0*pi*r*r*r;                     //计算圆球体积 vq
    vz=pi*r*r*h;                             //计算圆柱体积 vz
    printf("圆周长为:        l=%6.2f\n",l);
    printf("圆面积为:        s=%6.2f\n",s);
    printf("圆球表面积为:    sq=%6.2f\n",sq);
    printf("圆球体积为:      v=%6.2f\n",vq);
    printf("圆柱体积为:      vz=%6.2f\n",vz);
    return 0;
  }
```

运行结果:

```
请输入圆半径r, 圆柱高h:1.5,3
圆周长为:        l=  9.42
圆面积为:        s=  7.07
圆球表面积为:    sq= 28.27
圆球体积为:      v=  7.95
圆柱体积为:      vz= 21.21
```

说明:如果用 Visual C++ 6.0 中文版对程序进行编译,在程序中可以使用中文字符串,在输出时也能显示汉字。如果用英文的 C 编译系统,则无法使用中文字符串,读者可以改用英文字符串。

8. 编程序,用 getchar 函数读入两个字符给 c1 和 c2,然后分别用 putchar 函数和 printf 函数输出这两个字符。思考以下问题:

(1) 变量 c1 和 c2 应定义为字符型还是整型?或二者皆可?

(2) 要求输出 c1 和 c2 值的 ASCII 码,应如何处理?用 putchar 函数还是 printf 函数?

(3) 整型变量与字符变量是否在任何情况下都可以互相代替?如:

```
char c1,c2;
```

与

```
int c1,c2;
```

是否无条件地等价?

解:

```
#include <stdio.h>
int main( )
  {
    char c1,c2;
    printf("请输入两个字符 c1,c2:");
    c1=getchar( );
    c2=getchar( );
    printf("用 putchar 语句输出结果为:");
```

```
    putchar(c1);
    putchar(c2);
    printf("\n");
    printf("用 printf 语句输出结果为:");
    printf("%c %c\n",c1,c2);
    return 0;
}
```

运行结果：

```
请输入两个字符c1,c2:ab
用putchar语句输出结果为:ab
用printf语句输出结果为:a b
```

注意：若连续用两个 getchar 函数，输入字符时 a 和 b 之间没有空格，连续输入。

如果分两行输入：

a↙
b↙

结果会怎样？

运行结果：

```
请输入两个字符c1,c2:a
用putchar语句输出结果为:a

用printf语句输出结果为:a
```

第 1 行是输入数据，输入 a 后按回车键。结果还未来得及输入 b，程序马上输出了其下 4 行结果(包括 2 个空行)。

因为第 1 行将 a 和换行符输入到内存的输入缓冲区，因此 c1 得到 a(ASCII 码为 97)，c2 得到换行符(ASCII 码为 10)。再用 putchar 函数输出 c1，就输出了字符 a，在输出 c2 时，就把换行符转换为回车和换行两个操作，输出一个换行，后面的 printf("\n")又输出一个换行，所以就相当于输出一个空行，此行不显示任何字符。后面用 printf 函数输出 c1 和 c2，同样也输出了字符 a 和一个空行。

注意：在用连续两个 getchar 输入两个字符时，只要输入了“a↙”，系统就会认为用户已输入了两个字符。所以应当连续输入 ab 两个字符然后再按回车键，这样就保证了 c1 和 c2 分别得到字符 a 和 b。

下面回答思考问题：

(1) c1 和 c2 可以定义为字符型或整型，二者皆可。

(2) 可以用 printf 函数输出，在 printf 函数中用%d 格式符，即

```
printf("%d,%d\n",c1,c2);
```

(3) 字符变量在计算机内占 1 个字节，而整型变量占 2 个或 4 个字节。因此整型变量在可输出字符的范围内(ASCII 码为 0～127 的字符)是可以与字符数据互相转换的。如果整数在此范围外，不能代替。

为了进一步说明 char 型与 int 型数据的关系，请注意分析以下 3 个程序。

程序 1:

```
#include <stdio.h>
int main( )
  {
    int c1,c2;                    //定义整型变量 c1,c2
    printf("请输入两个整数 c1,c2:");
    scanf("%d,%d",&c1,&c2);
    printf("按字符输出结果:\n");
    printf("%c,%c\n",c1,c2);
    printf("按 ASCII 码输出结果为:\n");
    printf("%d,%d\n",c1,c2);
    return 0;
  }
```

运行结果:

```
请输入两个整数c1,c2:97,98
按字符输出结果:
a,b
按ASCII码输出结果为:
97,98
```

程序 2:

```
#include <stdio.h>
int main( )
  {
    char c1,c2;                   //c1,c2 定义为字符型变量
    int i1,i2;                    //定义整型变量
    printf("请输入两个字符 c1,c2:");
    scanf("%c,%c",&c1,&c2);
    i1=c1;                        //赋值给整型变量
    i2=c2;
    printf("按字符输出结果:\n");
    printf("%c,%c\n",i1,i2);
    printf("按整数输出结果:\n");
    printf("%d,%d\n",c1,c2);
    return 0;
  }
```

运行结果:

```
请输入两个字符c1,c2:a,b
按字符输出结果:
a,b
按整数输出结果:
97,98
```

程序 3:

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

```
int main( )
  {
    char c1,c2;                    //c1,c2 定义为字符型
    int i1,i2;                     //i1,i2 定义为整型
    printf("请输入两个整数 i1,i2:");
    scanf("%d,%d",&i1,&i2);
    c1=i1;                         //将整数赋值给字符变量
    c2=i2;
    printf("按字符输出结果:\n");
    printf("%c,%c\n",c1,c2);
    printf("按整数输出结果:\n");
    printf("%d,%d\n",c1,c2);
    return 0;
  }
```

运行结果：

```
请输入两个整数i1,i2:289,330
按字符输出结果:
!,J
按整数输出结果:
33,74
```

请注意 i,i1 和 i2 占 2 个或 4 个字节(Visual C++ 对它分配 4 个字节)，而 c1 和 c2 是字符变量，只占 1 个字节。如果是 unsigned char 类型，可以存放 0～255 的整数；如果是 signed char 类型，可以存放－128～127 范围内的整数。而现在输入给 i1 和 i2 的值已超过 0～255 的范围，i1 的值为 289，在内存中 i1 的存储情况如图 3.1(a) 所示(为简单起见，用 2 个字节表示)，在赋给字符变量 c1 时，只将其存储单元中最后一个字节(低 8 位)赋给 c1，见图 3.1(b)。而图 3.1(b)中的数据是整数 33，是字符'!'的 ASCII 码，所以用字符形式输出 c1 时，会输出字符'!'。图 3.2 表示 i2 和 c2 的情况，c2 的值为 74，是字符'J'的 ASCII 码，因此，按字符形式输出 c2 时就输出字符'j'。

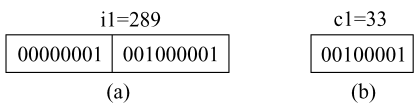

图　3.1

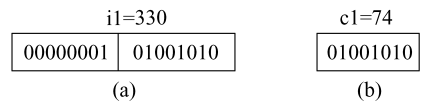

图　3.2

第 4 章　选择结构程序设计

1. 什么是算术运算？什么是关系运算？什么是逻辑运算？

解：略。

2. C 语言中如何表示“真”和“假”？系统如何判断一个量的“真”和“假”？

解：对于逻辑表达式，若其值为“真”，则以 1 表示，若其值为“假”，则以 0 表示。但是在判断一个逻辑量的值时，系统会以 0 作为“假”，以非 0 作为“真”。例如 3 && 5 的值为“真”，系统给出 3 && 5 的值为 1。

3. 写出下面各逻辑表达式的值。设 a=3，b=4，c=5。

(1) a+b>c && b==c

(2) a||b+c && b-c

(3) !(a>b) && !c||1

(4) !(x=a) && (y=b) && 0

(5) !(a+b)+c-1 && b+c/2

解：

(1) 0

(2) 1

(3) 1

(4) 0

(5) 1

4. 有 3 个整数 a，b，c，由键盘输入，输出其中最大的数。

解：

方法一：N-S 图见图 4.1。

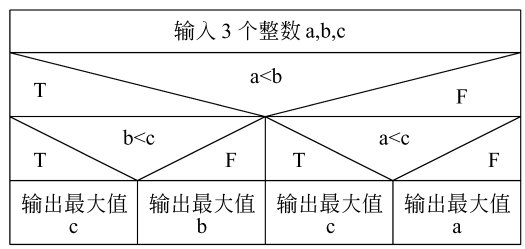

图　4.1

程序如下：

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

```
    int a,b,c;
    printf("请输入 3 个整数:");
    scanf("%d,%d,%d",&a,&b,&c);
    if (a<b)
      if (b<c)
        printf("max=%d\n",c);
      else
        printf("max=%d\n",b);
    else if (a<c)
        printf("max=%d\n",c);
    else
        printf("max=%d\n",a);
    return 0;
  }
```

运行结果：

```
请输入 3 个整数:12,34,9
max=34
```

方法二：使用条件表达式，可以使程序更简明、清晰。

```
#include <stdio.h>
int main( )
  {int a,b,c,temp,max;
   printf("请输入 3 个整数:");
   scanf("%d,%d,%d",&a,&b,&c);
   temp=(a>b)? a:b;          //将 a 和 b 中的大者存入 temp 中
   max=(temp>c)? temp:c;   //将 a 和 b 中的大者与 c 比较,取最大者
   printf("3 个整数的最大数是%d\n",max);
   return 0;
  }
```

运行结果：

```
请输入 3 个整数:12,34,9
3 个整数的最大数是34
```

5. 从键盘输入一个小于 1000 的正数，要求输出它的平方根(如平方根不是整数，则输出其整数部分)。要求在输入数据后先对其进行检查是否为小于 1000 的正数。若不是，则要求重新输入。

解：

```
#include <stdio.h>
#include <math.h>
#define M 1000
int main( )
  {
    int i,k;
```

```
    printf("请输入一个小于%d的整数i:",M);
    scanf("%d",&i);
    if (i>M)
    {printf("输入的数据不符合要求,请重新输入一个小于%d的整数i:",M);
     scanf("%d",&i);
    }
    k=sqrt(i);
    printf("%d的平方根的整数部分是%d\n",i,k);
    return 0;
}
```

运行结果:

① 第一次:输入正确数据。

```
请输入一个小于1000的整数i:345
345的平方根的整数部分是:18
```

② 第二次:输入不正确数据。

```
请输入一个小于1000的整数i:1230
输入的数不符合要求,请重新输入一个小于1000的整数i:130
130的平方根的整数部分是:11
```

讨论:题目要求输入的数小于1000,今为了增加程序的灵活性,定义符号常量M为1000,如果题目要求输入的数小于10000,只须修改define指令即可,不必修改主函数。

用if语句检查输入的数是否符合要求,如果不符合要求应进行相应的处理。从上面的程序看来是很简单的,但在实际应用中是很有用的。因为在程序提供用户使用后,不能保证用户输入的数据都是符合要求的。假若用户输入了不符合要求的数据怎么办?如果没有检查和补救措施,程序是不能供实际使用的。

本程序的处理方法是:提醒用户"输入的数据错了",要求重新输入。但只提醒一次,再错了怎么办?在学习了第5章循环之后,可以将程序改为多次检查,直到正确输入为止。程序如下:

```
#include <stdio.h>
#include <math.h>
#define M 1000
int main( )
  {
    int i,k;
    printf("请输入一个小于%d的整数i:",M);
    scanf("%d",&i);
    while (i>M)
      {printf("输入的数据不符合要求,请重新输入一个小于%d的整数i:",M);
       scanf("%d", &i);
       k=sqrt(i);
      }
```

```
    printf("%d 的平方根的整数部分是%d\n",i,k);
    return 0;
}
```

运行结果：

```
请输入一个小于1000的整数i:1230
输入的数不符合要求，请重新输入一个小于1000的整数i:1245
输入的数不符合要求，请重新输入一个小于1000的整数i:654
654的平方根的整数部分是：25
```

多次输入不符合要求的数据，均通不过，直到输入符合要求的数据为止。

这种检查手段是很重要的，希望读者能真正掌握。本例只是示意性的，程序比较简单。有了此基础，读者根据此思路完全可以做到对任何条件进行检查处理，使程序能正常运行，万无一失。

6. 有一个函数：

$$y=\begin{cases} x & (x<1) \\ 2x-1 & (1\leqslant x<10) \\ 3x-11 & (x\geqslant 10) \end{cases}$$

写程序，输入 x 的值，输出 y 相应的值。

解：

```
#include <stdio.h>
int main( )
  {int x,y;
   printf("输入 x:");
   scanf("%d",&x);
   if(x<1)                                    //x<1
     { y=x;
       printf("x=%3d,    y=x=%d\n" ,x,y);
     }
   else  if(x<10)                             //1=<x<10
         { y=2*x-1;
           printf("x=%d,   y=2*x-1=%d\n",x,y);
         }
       else                                   //x>=10
         { y=3*x-11;
           printf("x=%d,   y=3*x-11=%d\n",x,y);
         }
   return 0;
  }
```

运行结果：

①

```
输入x:4
x=4,   y=2*x-1=7
```

②

```
输入x:-1
x= -1,   y=x=-1
```

③

```
输入x:20
x=20,  y=3*x-11=49
```

7. 有一函数：

$$y=\begin{cases}-1 & (x<0)\\ 0 & (x=0)\\ 1 & (x>0)\end{cases}$$

有人分别编写了以下两个程序，请分析它们是否能实现题目要求。不要急于上机运行程序，先分析上面两个程序的逻辑，画出它们的流程图，分析它们的运行情况。然后上机运行程序，观察并分析结果。

(1)

```
#include <stdio.h>
int main( )
  {
    int x,y;
    printf("enter x:");
    scanf("%d",&x);
    y=-1;
    if(x!=0)
      if(x>0)
        y=1;
    else
      y=0;
    printf("x=%d,y=%d\n",x,y);
    return 0;
  }
```

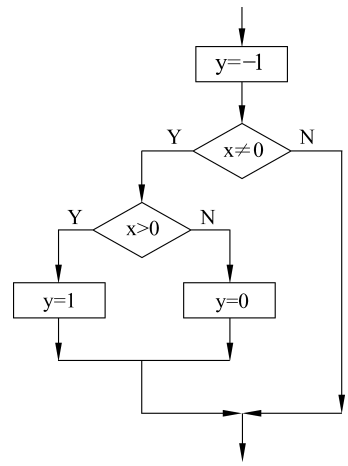

图 4.2

解：程序(1)的流程图见图 4.2。

它不能实现题目的要求。如果输入的 x<0，则输出 y=0。请注意 else 与 if 的配对关系。程序(1)中的 else 子句是和第 9 行的内嵌的 if 语句配对，而不与第 8 行的 if 语句配对。

运行结果：

```
enter x:-6
x=-6,y=0
```

x 的值为－6，输出 y=0，结果显然不对。

(2)

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

```
    int x,y;
    printf("enter x:");
    scanf("%d",&x);
    y=0;
    if(x>=0)
      if(x>0) y=1;
    else   y=-1;
   printf("x=%d,y=%d\n",x,y);
   return 0;
}
```

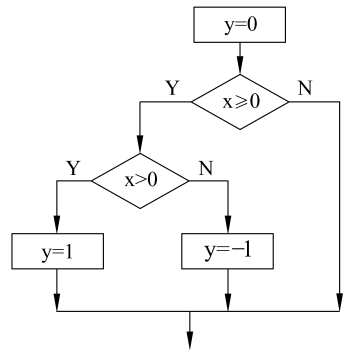

图　4.3

解：程序(2)的流程图见图 4.3。

它也不能实现题目的要求。如果输入的 x<0，则输出 y=0。

运行结果：

```
please enter x:-4
x=-4,y=0
```

x 的值为－4，输出 y=0，结果显然不对。程序(2)中的 else 子句是和第 9 行的内嵌的 if 语句配对，而不与第 8 行的 if 语句配对。

一定要注意 if 与 else 的配对关系。配对关系不随 if 和 else 所出现的列的位置而改变，例如程序(2)中的 else 与第 8 行的 if 写在同一列，但 else 并不因此而与第 8 行的 if 语句配对，它只和在它前面的离它最近的 if 配对。

请和教材第 4 章例 4.5 程序对比分析，进一步理解 if-else 的配对规则。

为了使逻辑关系清晰，避免出错，一般把内嵌的 if 语句放在外层的 else 子句中(如例 4.5 中程序 1 那样)，这样由于有外层的 else 相隔，内嵌的 else 不会被误认为和外层的 if 配对，而只能与内嵌的 if 配对，这样就不会搞混，若像本习题的程序(1)和程序(2)那样写就很容易出错。

可与本章例 4.5 中介绍的程序进行对比分析。

8. 给出一百分制成绩，要求输出成绩等级'A'、'B'、'C'、'D'、'E'。90 分以上为'A'，80～89 分为'B'，70～70 分为'C'，60～69 分为'D'，60 分以下为'E'。

解：

```
#include <stdio.h>
int main( )
  {float score;
   char grade;
   printf("请输入学生成绩:");
   scanf("%f",&score);
   while (score>100||score<0)
    {printf("\n 输入有误,请重输");
     scanf("%f",&score);
    }
   switch((int)(score/10))
    {case 10:
```

```
        case 9: grade='A';break;
        case 8: grade='B';break;
        case 7: grade='C';break;
        case 6: grade='D';break;
        case 5:
        case 4:
        case 3:
        case 2:
        case 1:
        case 0: grade='E';
          }
      printf("成绩是 %5.1f,相应的等级是%c\n",score,grade);
      return 0;
    }
```

运行结果:

①

```
请输入学生成绩:90.5
成绩是  90.5,相应的等级是A
```

②

```
请输入学生成绩:58
成绩是  58.0,相应的等级是E
```

说明:对输入的数据进行检查,如小于0或大于100,要求重新输入。(int)(score/10)的作用是将(score/10)的值进行强制类型转换,得到一个整型值。例如,当score的值为78时,(int)(score/10)的值为7。然后在switch语句中执行case 7中的语句,使grade='C'。

9. 给一个不多于5位的正整数,要求:

① 求出它是几位数;

② 分别输出每一位数字;

③ 按逆序输出各位数字,例如原数为321,应输出123。

解:

```
#include <stdio.h>
#include <math.h>
int main( )
  {
    int num,indiv,ten,hundred,thousand,ten_thousand,place;
                                          //分别代表个位、十位、百位、千位、万位和位数
    printf("请输入一个整数(0~99999):");
    scanf("%d",&num);
    if (num>9999)
          place=5;
    else  if (num>999)
          place=4;
```

```
    else   if (num>99)
           place=3;
    else   if (num>9)
           place=2;
    else place=1;
    printf("位数:%d\n",place);
    printf("每位数字为:");
    ten_thousand=num/10000;
    thousand=(int)(num-ten_thousand * 10000)/1000;
    hundred=(int)(num-ten_thousand * 10000-thousand * 1000)/100;
    ten=(int)(num-ten_thousand * 10000-thousand * 1000-hundred * 100)/10;
    indiv=(int)(num-ten_thousand * 10000-thousand * 1000-hundred * 100-ten * 10);
    switch(place)
      {case 5:printf("%d,%d,%d,%d,%d",ten_thousand,thousand,hundred,ten,indiv);
              printf("\n 反序数字为:");
              printf("%d%d%d%d%d\n",indiv,ten,hundred,thousand,ten_thousand);
              break;
       case 4:printf("%d,%d,%d,%d",thousand,hundred,ten,indiv);
              printf("\n 反序数字为:");
              printf("%d%d%d%d\n",indiv,ten,hundred,thousand);
              break;
       case 3:printf("%d,%d,%d",hundred,ten,indiv);
              printf("\n 反序数字为:");
              printf("%d%d%d\n",indiv,ten,hundred);
              break;
       case 2:printf("%d,%d",ten,indiv);
              printf("\n 反序数字为:");
              printf("%d%d\n",indiv,ten);
              break;
       case 1:printf("%d",indiv);
              printf("\n 反序数字为:");
              printf("%d\n",indiv);
              break;
        }
    return 0;
}
```

运行结果：

```
请输入一个整数(0~99999):98423
位数:5
每位数字为:9,8,4,2,3
反序数字为:32489
```

10. 企业发放的奖金根据利润提成。利润 I 低于或等于 100 000 元的，奖金可提成 10%；利润高于 100 000 元，低于 200 000 元(100 000$<I\leqslant$200 000)时，低于 100 000 元的部分按 10%提成，高于 100 000 元的部分，可提成 7.5%；200 000$<I\leqslant$400 000 时，低于 200 000 元的部分仍

按上述办法提成(下同)。高于200 000元的部分按5%提成;400 000<I≤600 000元时,高于400 000元的部分按3%提成;600 000<I≤1 000 000时,高于600 000元的部分按1.5%提成;I>1 000 000时,超过1 000 000元的部分按1%提成。从键盘输入当月利润I,求应发奖金总数。

要求:

(1) 用if语句编程序。

(2) 用switch语句编程序。

解:

(1) **用if语句编程序。**

```
#include <stdio.h>
int main( )
  {
     int i;
     double bonus,bon1,bon2,bon4,bon6,bon10;
     bon1=100000*0.1;
     bon2=bon1+100000*0.075;
     bon4=bon2+100000*0.05;
     bon6=bon4+100000*0.03;
     bon10=bon6+400000*0.015;
     printf("请输入利润 i:");
     scanf("%d",&i);
     if (i<=100000)
         bonus=i*0.1;
     else if (i<=200000)
         bonus=bon1+(i-100000)*0.075;
     else if (i<=400000)
         bonus=bon2+(i-200000)*0.05;
     else if (i<=600000)
         bonus=bon4+(i-400000)*0.03;
     else if (i<=1000000)
         bonus=bon6+(i-600000)*0.015;
     else
         bonus=bon10+(i-1000000)*0.01;
     printf("奖金是: %10.2f\n",bonus);
     return 0;
  }
```

运行结果:

```
请输入利润i:234000
奖金是:   19200.00
```

此题的关键在于正确写出每一区间的奖金计算公式。例如利润在100 000～200 000元时,奖金应由两部分组成:

① 利润为100 000元时应得的奖金,即100 000元×0.1。

② 100 000 元以上部分应得的奖金，即(num－100 000)×0.075 元。

同理，200 000～400 000 元这个区间的奖金也应由两部分组成：

① 利润为 200 000 元时应得的奖金，即 100 000×0.1＋100 000×0.075。

② 200 000 元以上部分应得的奖金，即(num－200 000)×0.05 元。

程序中先把 100 000 元、200 000 元、400 000 元、600 000 元、1 000 000 元各关键点的奖金计算出来，即 bon1，bon2，bon4，bon6 和 bon10。然后再加上各区间附加部分的奖金即可。

(2) **用 switch 语句编程序**。

N-S 图见图 4.4。

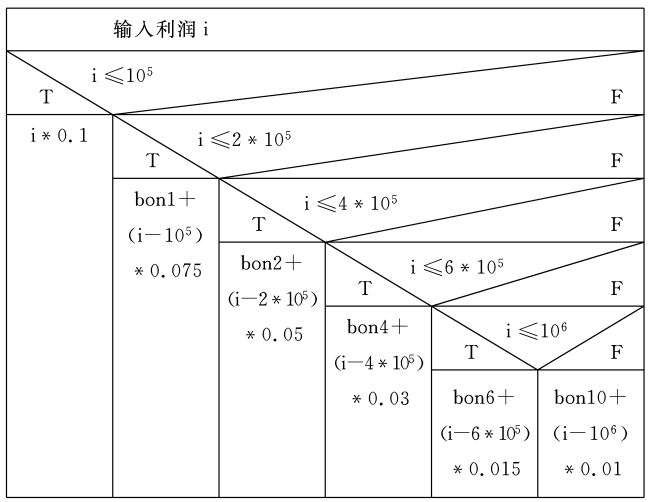

图　4.4

```
#include <stdio.h>
int main( )
  {
    int i;
    double  bonus,bon1,bon2,bon4,bon6,bon10;
    int  branch;
    bon1=100000*0.1;
    bon2=bon1+100000*0.075;
    bon4=bon2+200000*0.05;
    bon6=bon4+200000*0.03;
    bon10=bon6+400000*0.015;
    printf("请输入利润 i:");
    scanf("%d",&i);
    branch=i/100000;
    if (branch>10)  branch=10;
    switch(branch)
      {case 0:bonus=i*0.1;break;
       case 1:bonus=bon1+(i-100000)*0.075;break;
```