第3章

最简单的C程序设计——顺序程序设计

3.1 怎样区分表达式和表达式语句？C语言为什么要设表达式语句？什么时候用表达式，什么时候用表达式语句？

解：略。

3.2 C语言为什么要把输入输出的功能作为函数，而不作为语言的基本部分？

解：略。

3.3 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：可按如下方式在键盘上输入：

```
a=3 b=7↙
8.5 71.82A a↙
```

输出为：

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

请注意：在输入完8.5和71.82两个实数给x和y后紧接着输入字符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？

```
a=3 b=7↙
8.5 71.82 A a↙
```

这时 71.82 后面的空格字符就被 c1 读入，c2 读入了字符 A。在输出 c1 时就输出空格。

输出为：

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

如果在输入 8.5 和 71.82 两个实数后输入回车符，会怎么样呢？

```
a=3 b=7↙
8.5 71.82↙
A a↙
```

输出为：

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

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，就输出一个回车符，输出 c2 时就输出字符 A。

在用 scanf 函数输入数据时往往会出现一些想象不到的情况，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。

读者在遇到类似情况时，上机多试验一下就可以找出规律来。

3.4 用下面的 scanf 函数输入数据，使 a＝10，b＝20，c1＝'A'，c2＝'a'，x＝1.5，y＝－3.75，z＝67.8，请问在键盘上如何输入数据？

```
scanf("%5d%5d%c%c%f%f%*f,%f",&a,&b,&c1,&c2,&x,&y,&z);
```

解：

```
#include <stdio.h>
int main()
 {int a,b;
 float x,y,z;
 char c1,c2;
 scanf("%5d%5d%c%c%f%f%*f,%f",&a,&b,&c1,&c2,&x,&y,&z);
 printf("a=%d,b=%d,c1=%c,c2=%c,x=%6.2f,y=%6.2f,z=%6.2f\n",a,b,c1,c2,x,y,z);
 return 0;
}
```

运行情况如下：

```
␣␣␣10␣␣␣20Aa1.5␣-3.75␣2.5,67.8↙          (此行为输入的数据)
a=10,b=20,c1=A,c2=a,x=␣1.50,y=␣-3.75,z=␣67.80   (此行为输出)
```

说明：按%5d式的要求输入a与b时，先输入3个空格，然后再输入10与20。%*f是用来禁止赋值的。在输入时，对应于%*f的地方随意输入了一个实数2.5，该值不会赋给任何变量

3.5 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：可编程序如下：

```
#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;
}
```

运行情况如下：

```
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Turbo C或Turbo C++，则无法使用中文字符串，读者可以改用英文字符串。

3.6 输入一个华氏温度，要求输出摄氏温度。公式为：

$$c=\frac{5}{9}(F-32)$$

输出要有文字说明，取两位小数。

解：相应的程序如下所示。

```
#include <stdio.h>
int main()
{float c,f;
 printf("请输入一个华氏温度：");
 scanf("%f",&f);
 c=(5.0/9.0)*(f-32);          /*注意5和9要用实型表示,否则5/9值为0*/
 printf("摄氏温度为：%5.2f\n",c);
 return 0;
}
```

运行情况如下：

```
请输入一个华氏温度：87↙
得到结果：摄氏温度为：30.56
```

3.7 编程序，用 getchar 函数读入两个字符给变量 c1、c2，然后分别用 putchar 函数和 printf 函数输出这两个字符，并思考以下问题：

(1) 变量 c1、c2 应定义为字符型或整型？还是二者皆可？

(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：

① char c1,c2;

② int c1,c2;

是否无条件等价？

解：可编程序如下：

```
#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
```

请注意：连续用两个 getchar 函数时是怎样输入字符的。如果用以下方法输入：

```
a↙
b↙
```

得到以下运行结果：

```
用 putchar 语句输出结果为: a
      (空一行)
用 printf 语句输出结果为: a
      (空一行)
```

因为第 1 行将 a 和回车符输入到内存的输入缓冲区，因此 c1 得到 a，c2 得到一个回车符。在输出 c2 时就会产生一个回车换行，而不会输出任何可显示的字符。在实际操作时，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。

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 0～255 之间的字符)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。

请分析以下 3 个程序：

程序 1：

```
#include <stdio.h>
int main()
{
  int c1,c2;                          //定义整型变量
  printf("请输入两个整数 c1,c2: ");
  scanf("%d,%d",&c1,&c2);
  printf("按字符输入结果: \n");
  printf("%c,%c\n",c1,c2);
  printf("按 ASCII 码输入出结果为: \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:
97,98
```

程序 2:

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

```
    return 0;
}
```

运行结果：

```
请输入两个整数 i1,i2: 298,330↙
按字符输出结果：*,J
按整数输出结果：42,74
```

请注意 c1 和 c2 是字符变量，只占 1 个字节，只能存放 0～255 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，所以只将整型变量 i1 和 i2 在内存存储单元中的最后一个字节(低 8 位)赋给 c1 和 c2。可以看到：298－256＝42，330－256＝74。读者可以写出 298 和 330 的二进制形式，取其低 8 位，即可一目了然。与 ASCII 码 42 和 74 相应的字符为“＊”和“J”，因此。按字符形式输出结果时输出：＊，J。请读者注意分析。

第4章

选择结构程序设计

4.1 什么是算术运算？什么是关系运算？什么是逻辑运算？

解：略。

4.2 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：如果有一个逻辑表达式，若其值为“真”，系统会以1表示，若其值为“假”，会以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。

4.3 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

4.4 编一个程序，当给 x 输入任意的正数时，y 都输出1；当给 x 输入任意的负数时，y 都输出-1；当给 x 输入0时，y 输出0。如果用数学式子表示，就是下面的函数，参见图4.1。

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

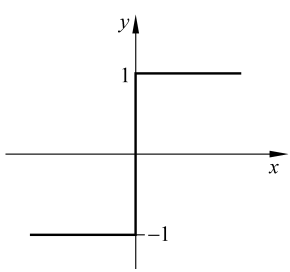

图 4.1

有以下几个程序，请判断哪个(可能不止一个)是正确的？分别画出它们的N-S图。

程序 1：

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

程序 2：将上面程序的 if 语句(第 6～10 行)改为：

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

程序 3：将上述 if 语句改为：

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

程序 4：将上述 if 语句改为：

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

解：先按题目要求写出算法：

```
输入 x
若 x<0,则 y=-1
若 x=0,则 y=0
若 x>0,则 y=1
输出 y
```

或

```
输入 x
若 x<0,则 y=-1
```

否则：

若 x=0,则 y=0

若 x>0,则 y=1

输出 y

也可以用流程图表示，见图 4.2。

分析上面 4 个程序，只有程序 1 和程序 2 是正确的。程序 1 体现了图 4.2 的流程，显然它是正确的；程序 2 的流程图见图 4.3，它也能实现题目的要求。

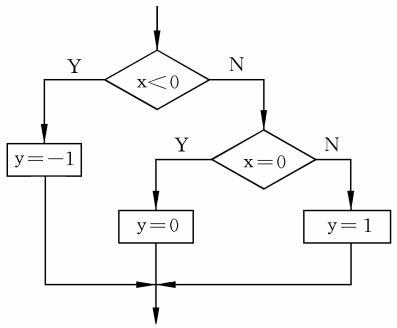

图 4.2

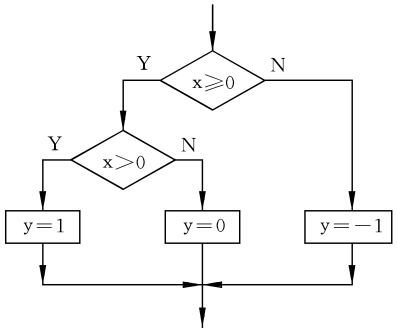

图 4.3

程序 3 的流程图见图 4.4。程序 4 的流程图见图 4.5。它们不能实现题目的要求。请注意程序中的 else 与 if 的配对关系。例如程序 3 中的 else 子句是和它上一行的内嵌的 if 语句配对，而不与第 2 行的 if 语句配对。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，如像程序 3 和程序 4 那样写就很容易出错。

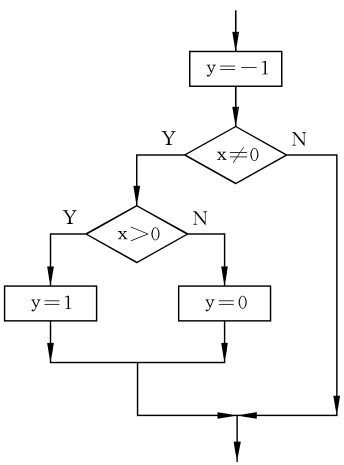

图 4.4

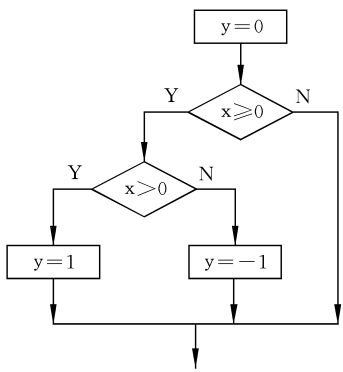

图 4.5

4.5 有 3 个整数 a、b、c，由键盘输入，输出其中最大的数，请编程序。

解：方法一：N-S 图见图 4.6。

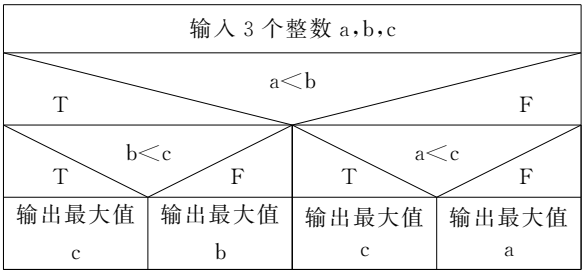

图 4.6

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("3 个整数中的最大数是%d\n",c);
    else
      printf("3 个整数中的最大数是%d\n",b);
  else if (a<c)
      printf("3 个整数中的最大数是%d\n",c);
    else
      printf("3 个整数中的最大数是%d\n",a);
    return 0;
}
```

运行情况如下：

```
请输入 3 个整数: 72,274,-39↙
3 个整数中的最大数是 274
```

方法二：使用条件表达式，可以使程序更加简明、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;
```

```
}
```

运行情况如下：

```
请输入 3 个整数：89,44,188↙
3 个整数的最大数是 188
```

请注意：在输入 3 个整数时，在两个数之间必须用逗号分隔，这是编程者在 scanf 函数中指定的。如果在运行本程序时，不用逗号分隔，而用空格分隔，如：

```
请输入 3 个整数：89 44 188↙
3 个整数的最大数是 89
```

这个结果显然是错误的。

4.6 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'，70～79 分为'C'，60～69 分为'D'，60 分以下为'E'。

解：程序如下所示。

```
#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：59↙
成绩是 59.0,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'。

4.7 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：程序如下所示。

```
#include <stdio.h>
#include <math.h>
int main()
{
  long int num;
  int 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("%ld",&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
```

4.8 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 10%;利润高于 100 000 元,低于 200 000 元(100 000<I≤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≤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()
{
  long 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("%ld",&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 10 万元至 20 万元时，奖金应由两部分组成：

① 利润为 10 万元时应得的奖金，即 10 万元×0.1。

② 10 万元以上部分应得的奖金，即(num－10 万)×0.075 元。

同理，20 万～40 万元这个区间的奖金也应由两部分组成：

① 利润为 20 万元时应得的奖金，即 10 万元×0.1＋10 万元×0.075。

② 20 万元以上部分应得的奖金，即(num－20 万)×0.05 元。

程序中先把 10 万元、20 万元、40 万元、60 万元、100 万元各关键点的奖金计算出来，即 bon1、bon2、bon4、bon6、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 4.7。

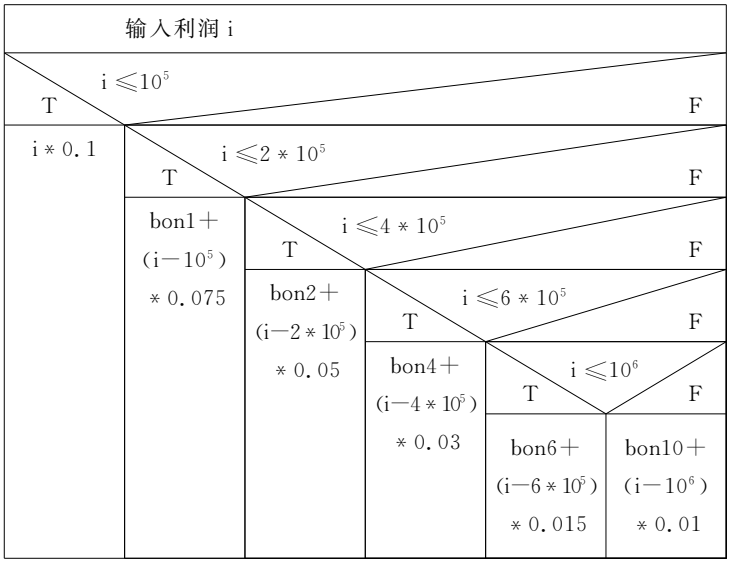

图　4.7

```
#include <stdio.h>
int main()
{
  long 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("%ld",&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;
     case 2:
     case 3: bonus=bon2+(i-200000)*0.05;break;
     case 4:
     case 5: bonus=bon4+(i-400000)*0.03;break;
     case 6:
     case 7:
     case 8:
     case 9: bonus=bon6+(i-600000)*0.015;break;
     case 10: 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: 156890↙
奖金是：14266.75
```

4.9 有 4 个圆塔，圆心分别为(2,2)、(－2,2)、(－2,－2)、(2,－2)，圆半径为 1，见图 4.8。这 4 个塔的高度为 10m，塔以外无建筑物。今输入任一点的坐标，求该点的建筑高度(塔外的高度为零)。

解：N-S 图见图 4.9。

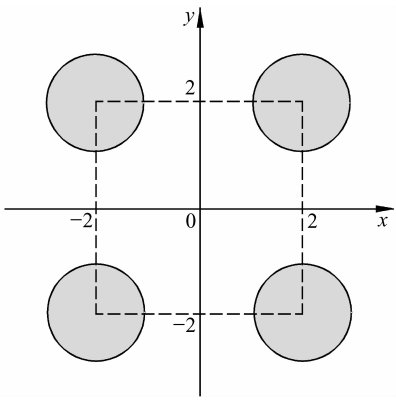

图 4.8

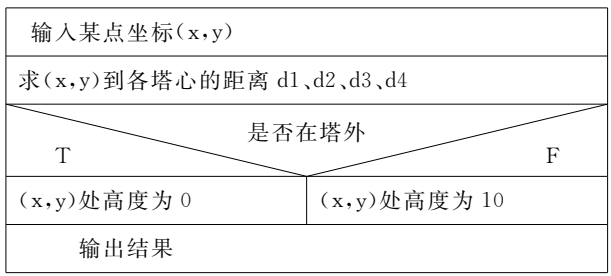

图 4.9

相应的程序如下：

```
#include <stdio.h>
int main()
{ int  h=10;
  float x1=2,y1=2,x2=-2,y2=2,x3=-2,y3=-2,x4=2,y4=-2,x,y,d1,d2,d3,d4;
  printf("请输入一个点(x,y): ");
  scanf("%f,%f",&x,&y);
  d1=(x-x4)*(x-x4)+(y-y4)*(y-y4);              //求该点到各中心点距离
  d2=(x-x1)*(x-x1)+(y-y1)*(y-y1);
  d3=(x-x2)*(x-x2)+(y-y2)*(y-y2);
  d4=(x-x3)*(x-x3)+(y-y3)*(y-y3);
  if (d1>1 && d2>1 && d3>1 && d4>1)  h=0;          //判断该点是否在塔外
  printf("该点高度为 %d\n",h);
  return 0;
}
```

运行结果：

① 请输入一个点(x,y)： 0.5,0.7↙
该点高度为 0
② 请输入一个点(x,y)： 2.1,2.3↙
该点高度为 10

4.10 求 $ax^2+bx+c=0$ 方程的解。

根据代数知识，应该有以下几种可能：

① $a=0$，不是二次方程，而是一次方程。

② $b^2-4ac=0$，有两个相等的实根。

③ $b^2-4ac>0$，有两个不等的实根。

④ $b^2-4ac<0$，有两个共轭复根。

请画出 N-S 流程图，并据此编写程序，程序应能处理上面 4 种情况。运行程序时，分别给出不同的 a,b,c 值，相应于上面 4 种情况，分析输出结果。

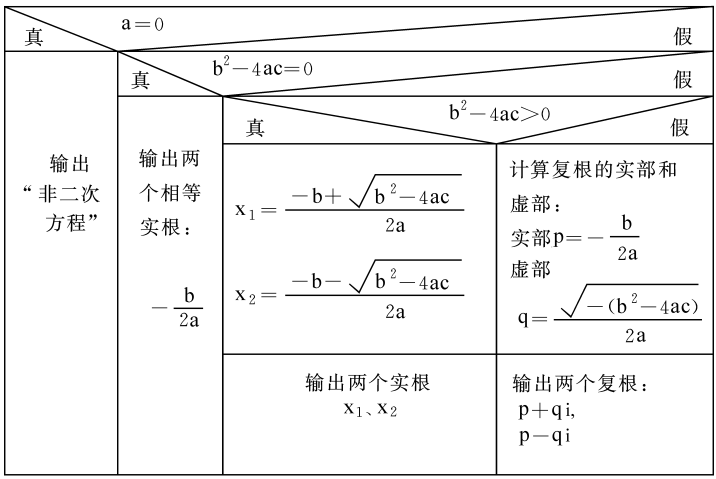

图 4.10

解：画出 N-S 流程图。见图 4.10，据此编写程序如下：

```
#include  <stdio.h>
 #include  <math.h>
 int main ( )
  {float a,b,c,disc,x1,x2,realpart,imagpart;
   printf("please enter a,b,c: ");
   scanf("%f,%f,%f",&a,&b,&c);
   printf("The equation ");
   if(fabs(a)<=1e-6)
     printf("is not a quadratic\\n");
   else
     {
      disc=b*b-4*a*c;
      if(fabs(disc)<=1e-6)
```

```
        printf("has two equal roots: %8.4f\n",-b/(2*a));
      else if(disc>1e-6)
        {x1=(-b+sqrt(disc))/(2*a);
         x2=(-b-sqrt(disc))/(2*a);
         printf("has distinct real roots: %8.4f and %8.4f\n",x1,x2);
        }
      else
        {realpart=-b/(2*a);
         imagpart=sqrt(-disc)/(2*a);

         printf(" has complex roots: \n");
         printf("%8.4f+%8.4fi\n",realpart,imagpart);
         printf("%8.4f-%8.4fi\n",realpart,imagpart);
        }
    }
    return 0;
}
```

程序中用 disc 代表判别式 b^2-4ac，先计算 disc 的值，以减少以后的重复计算。对于判断 b^2-4ac 是否等于 0 时，要注意：由于 disc(即 b^2-4ac)是实数，而实数在计算和存储时会有一些微小的误差，因此不能直接进行如下判断："if(disc==0)…"，因为这样可能会出现本来是零的量，由于上述误差而被判别为不等于零，而导致结果错误。所以采取的办法是判别 disc 的绝对值(fabs(disc))是否小于一个很小的数(例如 10^{-6})，如果小于此数，就认为 disc 等于 0。程序中以变量 realpart 代表实部 p，以 imagpart 代表虚部 q，以增加可读性。

运行结果如下：

```
① please enter a,b,c: " 1,2,1↙
   The equation has two  equal roots:   -1.0000
② please enter a,b,c: " 1,2,2↙
   The equation has complex roots:
   -1.0000+  1.0000i
   -1.0000-  1.0000i
③ please enter a,b,c: " 2,6,1↙
   The equation has distinct real roots:   -0.1771 and  -2.8229
```

在程序中用格式说明%8.4f 指定输出格式，表示输出的数据共占 8 列宽度，其中小数点后有 4 位，因此在输出-1 时，在负号前有一个空格，即␣-1.0000。

关于闰年问题的说明：

在教材第 4 章中举了计算闰年的例子(例 4.5)，有不少读者对闰年规则搞不清楚，纷纷来信询问，他们说，从小就只知道：能被 4 除尽的年份都是闰年。有的读者甚至认为作者弄错了。因此，有必要在此对闰年的规定作一说明：

地球绕太阳转一周的实际时间为 365 天 5 小时 48 分 46 秒。如果一年只有 365 天，

每年就多出 5 个多小时。4 年多出的 23 小时 15 分 4 秒，差不多等于一天。于是决定每 4 年增加一天。但是，它比一天 24 小时又少了约 45 分钟。如果每 100 年有 25 个闰年的话，就少了 18 时 43 分 20 秒，这就差不多等于一天了，这显然是不合适的。

可以算出：每年多出 5 小时 48 分 46 秒，100 年就多出 581 小时 16 分 40 秒。而 25 个闰年需要 25×24＝600 小时。581 小时 16 分 40 秒只够 24 个闰年(24×24＝576 小时)，于是决定每 100 年只安排 24 个闰年(世纪年不作为闰年)。但是这样每 100 年又多出 5 小时 16 分 40 秒(581 小时 16 分 40 秒－576 小时)，于是又决定每 400 年增加一个闰年。这样就比较接近实际情况了。

根据以上情况，决定闰年按以下规则计算：闰年应能被 4 整除(如 2004 年是闰年，而 2001 年不是闰年)，但不是所有能被 4 整除的年份都是闰年。在能被 100 整除的年份中，只有同时能被 400 整除的年份才是闰年(如 2000 年是闰年)，能被 100 整除而不能被 400 整除的年份 (如 1800、1900、2100)不是闰年。

这是国际公认的规则。只说“能被 4 整除的年份是闰年”是不准确的。

教材上介绍的方法和程序是正确的。