第 3 章 选择结构程序设计

3.1　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

3.2　有 3 个整数 a,b,c,由键盘输入,输出其中最大的数,请编写程序。

解: 方法一: N-S 图见图 3-1。

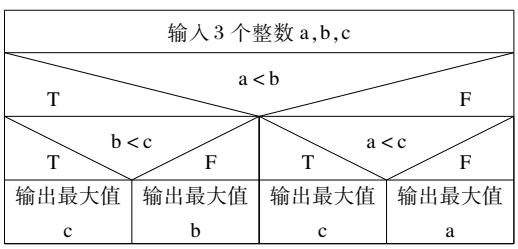

图　3-1

编写程序如下:

```
#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;
}
```

运行结果：

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

3.3　有一个函数：

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

写一段程序，输入 x，输出 y 值。

解：编写程序如下：

```
#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

3.4　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：编写程序如下：

```
#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'。

3.5　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：编写程序如下：

```
#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
```

3.6　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 $< I \leqslant$ 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 $< I \leqslant$ 600 000元时，高于400 000元的部分按3%提成；600 000 $< I \leqslant$ 100 0000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 语句编写程序。
解：编写程序如下：
（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 万～20 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 语句编写程序，N-S 图见图 3-2。

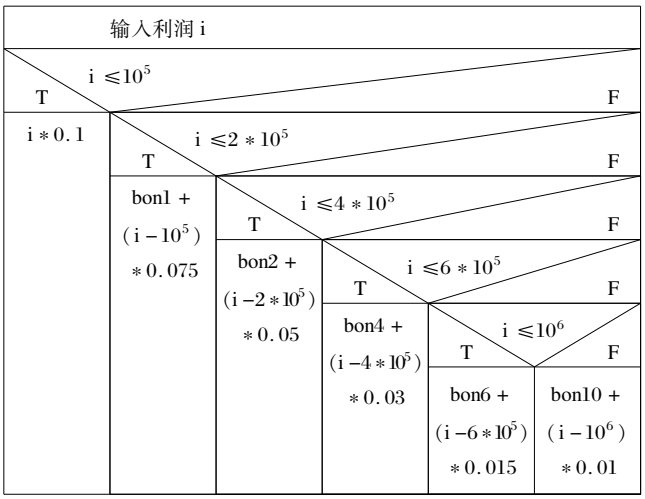

图 3-2

编写程序如下：

```
#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
```

3.7　输入 4 个整数，要求按由小到大的顺序输出。

解：此题采用依次比较的方法排出其大小顺序。在学习了循环和数组以后，可以掌握更多的排序方法。

编写程序如下：

```
#include <stdio.h>
int main()
{ int t,a,b,c,d;
  printf("请输入 4 个数:");
  scanf("%d,%d,%d,%d",&a,&b,&c,&d);
  printf("a = %d,b = %d,c = %d,d = %d\n",a,b,c,d);
  if (a > b)
    { t = a;a = b;b = t;}
  if (a > c)
    { t = a;a = c;c = t;}
  if (a > d)
    { t = a;a = d;d = t;}
  if (b > c)
    { t = b;b = c;c = t;}
  if (b > d)
    { t = b;b = d;d = t;}
  if (c > d)
    { t = c;c = d;d = t;}
  printf("排序结果如下: \n");
  printf("%d  %d  %d  %d  \n"  ,a,b,c,d);
  return 0;
}
```

运行结果：

```
请输入 4 个数: 6,8,1,4 ↙
a = 6,b = 8,c = 1,d = 4
排序结果如下:
4  6  8
```

3.8　有 4 个圆塔，圆心分别为(2,2)、(-2,2)、(-2,-2)、(2,-2)，圆半径为 1m，

见图 3-3。这 4 个塔的高度为 10m，塔以外无建筑物。现输入任一点的坐标，求该点的建筑高度（塔外的高度为零）。

解：N-S 图见图 3-4。

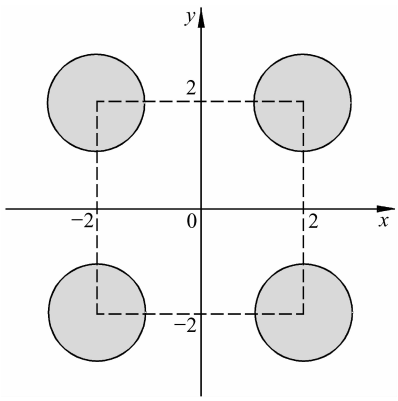

图 3-3

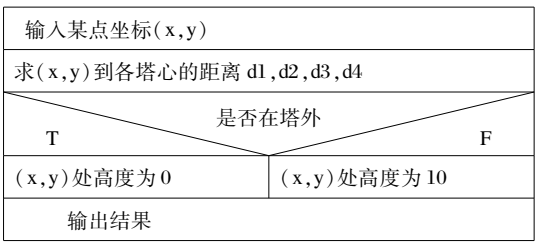

图 3-4

编写程序如下：

```
#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

关于闰年问题的说明

在教材第 3 章中举了计算闰年的例子（例 3.6），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进行说明：

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 1 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。

第4章 循环结构程序设计

4.1　统计全单位人员的平均工资。单位的人数不固定,工资数从键盘先后输入,当输入 -1 时表示输入结束(前面输入的是有效数据)。

解:编写程序如下:

```
#include<stdio.h>
int main()
{ float pay,sum=0,aver;
  int i=0;
  scanf("%f",&pay);                          //输入一位员工的工资
  while (pay!=-1)                            //当输入的工资不等于-1
  { sum=sum+pay;                             //把输入的工资累加到sum中
    i++;                                     //人数加1
    scanf("%f",&pay);                        //再输入一位员工的工资
  }
  aver=sum/i;                                //计算平均工资
  printf("average pay is:%8.2f\n",aver);     //输出平均工资
  return 0;
}
```

运行结果:

```
1234↙
4567.89↙
1456.98↙
-1↙
average pay is:2419.62
```

4.2　一个单位下设3个班组,每个班组人数不固定,需要统计每个班组的平均工资。分别输入3个班组所有职工的工资,当输入 -1 时表示该班组的输入结束。输出班组号和该班组的平均工资。

解:在上题的基础上再加一个外循环,处理3个班组的平均工资。

编写程序如下：

```
#include<stdio.h>
int main()
{ float pay,sum,aver,total=0;
  int i,n;
  for(n=1;n<=3;n++)                    //执行 3 次循环
     {i=0;
      scanf("%f",&pay);
      sum=0;
      while (pay!=-1)
        {sum=sum+pay;
          i++;
          scanf("%f",&pay);
        }
      aver=sum/i;                      //计算序号为 i 的班组的平均工资
      printf("group %d,average pay is:%8.2f\n",n,aver);
                                       //输出此班组的平均工资
      total=total+aver;                //把本组平均工资累加到 totol 中
     }
  printf("The average of all group is %8.2f\n",total/(n-1));//输出总平均工资
  return 0;
}
```

运行结果：

```
1567.87↙
3421.8↙
-1↙
group 1,average pay is: 2494.83
2234.65↙
2346.9↙
-1↙
group 2,average pay is: 2290.77
3563.7↙
5411.76↙
-1↙
group 3,average pay is: 4487.73
The average of all group is 3091.11
```

说明：为了节约篇幅，设每组只有 2 人。注意在执行完 for 循环后，n 的值是 4，因此在最后输出总平均工资时，应将 total 除以(n－1)，而不是 total/4。

4.3　百鸡问题：公元 5 世纪末，我国古代数学家张丘建在他编写的《算经》里提出了“百鸡问题”：“鸡翁一，值钱五；鸡母一，值钱三；鸡雏三，值钱一。百钱买百鸡，问鸡翁、母、雏各几何？”说成白话文是：“公鸡每只值 5 元，母鸡值 3 元，小鸡 3 个值 1 元。用 100 元买 100 只鸡，问公鸡、母鸡、小鸡各应买多少只？”

解：解题思路：根据题意，公鸡最多能买 20 只，母鸡最多 33 只，小鸡最多 100 只，小鸡的数目应是 3 的倍数。可以用穷举法把所有可能的组合进行检测。

方法一：用穷举法把所有可能的组合逐个进行检测，把符合要求的筛选出来。

编写程序如下：

```
#include<stdio.h>
int main()
{ int x,y,z,money;
  printf("cocks hens chicks\n");
  for(x=0;x<20;x++)
    for(y=0;y<34;y++)
     for(z=0;z<100;z=z+3)
       {money=5*x+3*y+z/3;
        if(x+y+z==100 && money==100)
          {printf("%9d%9d%9d\n",x,y,z);}
       }
  return 0;
}
```

运行结果：

```
cocks  hens  chichs
    0    25      75
    4    18      78
    8    11      81
   12     8      84
```

说明：一共有 4 种可能方案。经验证，结果是正确的。程序用了 3 个 for 循环，把在允许范围内的每一个 x，y，z 组合都进行测试。

方法二：利用 x+y+z=100 的前提，不必对所有 x，y，z 的组合进行测试，只须测试满足 x+y+z=100 条件的组合是否满足总款为一百元即可。

编写程序如下：

```
#include<stdio.h>
int main()
{ int x,y,z,money;
  printf("cocks hens chicks\n");
  for(x=0;x<=20;x++)
    for(y=0;y<34;y++)
      {z=100-x-y;                          //只对符合此条件的 z 值进行测试
       if(z%3==0)                          //只对能被 3 整除的 z 进行测试
         {money=5*x+3*y+z/3;
          if (money==100)
            printf("%9d%9d%9d\n",x,y,z);
         }
      }
```

```
    return 0;
}
```

运行结果同上。注意程序第 7 行,不是对每一个 z 值都进行测试,只考虑符合 x + y + z = 100 条件的(即 z = 100 - x - y)。此方法比方法一的程序少用了一个 for 循环。穷举的次数少一些。

请注意程序第 8 行 if(z%3 = =0),% 是求余运算符,z%3 的值是 z 被 3 除的余数,如果 z%3 等于 0,表示 z 被 3 整除。请读者考虑为什么要作此项检查? 没有它有何影响? 可上机试验一下。

方法三:可以再减少循环的次数。利用数学知识。根据题意可以列出下面方程式:

$$5x + 3y + \frac{z}{3} = 100 \qquad ①$$

$$x + y + z = 100 \qquad ②$$

由式①和式②可导出

$$7x + 4y = 100 \qquad ③$$

即

$$y = (100 - 7x)/4 \qquad ④$$

利用式④编程。

编写程序如下:

```
#include<stdio.h>
int main()
{ int x,y,z;
  printf("cocks hens chicks\n");
  for(x=0;x<=20;x++)
  {y=(100-7*x)/4;
   if((100-7*x)%4==0 && y>0)
    {z=100-x-y;
     if(z%3==0)
       printf("%9d%9d%9d\n",x,y,z);
    }
  }
  return 0;
}
```

此程序只用了一个 for 循环。只执行 21 次循环就得到结果。

4.4　猴子吃桃问题。猴子第 1 天摘下若干个桃子,当即吃了一半,还不过瘾,又多吃了一个。第 2 天早上又将剩下的桃子吃掉一半,又多吃了一个。以后每天早上都吃了前一天剩下的一半零一个。到第 10 天早上想再吃时,就只剩一个桃子了。求第 1 天共摘多少个桃子。

解:编写程序如下:

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

```
{ int day,x1,x2;
  day=9;
  x2=1;
  while(day>0)
   { x1=(x2+1)*2;          //第 1 天的桃子数是第 2 天桃子数加 1 后的 2 倍
     x2=x1;
     day--;
   }
  printf("total=%d\n",x1);
  return 0;
}
```

运行结果：

```
total=1543
```

4.5　输入两个正整数 m 和 n，求其最大公约数和最小公倍数。

解：编写程序如下：

```
#include<stdio.h>
int main()
{ int  p,r,n,m,temp;
  printf("请输入两个正整数 n,m:");
  scanf("%d,%d,",&n,&m);
  if (n<m)
   {
    temp=n;
    n=m;
    m=temp;
   }
  p=n*m;
  while(m!=0)
   {
    r=n%m;
    n=m;
    m=r;
   }
  printf("它们的最大公约数为:%d\n",n);
  printf("它们的最小公约数为:%d\n",p/n);
  return 0;
}
```

运行结果：

```
请输入两个正整数 n,m: 35,49↙
它们的最大公约数为: 7
它们的最小公约数为: 245
```

4.6　输入一行字符,分别统计出其中英文字母、空格、数字和其他字符的个数。

解：编写程序如下：

```
#include<stdio.h>
int main()
{ char c;
  int letters=0,space=0,digit=0,other=0;
  printf("请输入一行字符:\n");
  while((c=getchar())!='\n')
   {
     if (c>='a' && c<='z'||c>='A' && c<='Z')
        letters++;
     else if (c==' ')
        space++;
     else if (c>='0' && c<='9')
        digit++;
     else
        other++;
    }
   printf ("字母数:%d\n 空格数:%d\n 数字数:%d\n 其他字符数:%d\n",letters,
         space,digit,other);
   return 0;
}
```

运行结果：

```
请输入一行字符:
I am a student.
字母数:11
空格数:3
数字数:0
其他字符数:1
```

4.7　求 $\sum_{n=1}^{20} n!$（即求 $1!+2!+3!+4!+\cdots+20!$）

解：编写程序如下：

```
#include<stdio.h>
int main()
{ double s=0,t=1;
  int n;
  for (n=1;n<=20;n++)
  { t=t*n;
    s=s+t;
  }
```

```
  printf("1! +2! + ... +20! = %22.15e \n",s);
  return 0;
}
```

运行结果：

```
1! +2! +… +20! =2.561327494111820e +018
```

请注意：s 不能定义为 int 型或 long 型，因为用 Visual C ++ 6.0 时，int 型和 long 型数据在内存都占 4 个字节。数据的范围为 -21 亿 ~21 亿。无法容纳求得的结果。今将 s 定义为 double 型，以得到更多的精度。在输出时，用 22.15e 格式，使数据宽度为 22，数字部分中小数位数为 15 位。

4.8　输出所有的“水仙花数”，所谓“水仙花数”是指一个 3 位数，其各位数字立方和等于该数本身。例如，153 是一水仙花数，因为 $153 = 1^3 + 5^3 + 3^3$。

解：编写程序如下：

```
#include < stdio.h >
int main()
{ int i,j,k,n;
  printf("parcissus numbers are ");
  for (n =100;n <1000;n ++)
    { i =n/100;
      j =n/10 - i *10;
      k =n%10;
      if (n == i * i * i + j * j * j + k * k * k)
      printf("%d ",n);
    }
  printf("\n");
  return 0;
}
```

运行结果：

```
narcissus numbers are 153 370 371 407
```

说明：本题用穷举法，对所有 3 位数一一测试，找出其中符合“水仙花数”条件的数。穷举法是最“笨”的方法，也是没有别的方法时用的方法。从 100 到 999，共有 999 个三位数。由于计算机的速度很快，用穷举法处理这些数时间是很快的。

4.9　一个数如果恰好等于它的因子之和，这个数就称为“完数”。例如，6 的因子为 1，2，3，而 6 =1 +2 +3，因此 6 是“完数”。编程序找出 1000 之内的所有完数，并按下面格式输出其因子：

```
6 : its factors are 1,2,3.
```

解：方法一：

编写程序如下：

```
#include<stdio.h>
int main()
{ int k1,k2,k3,k4,k5,k6,k7,k8,k9,k10;
  int i,a,n,s;
  for (a=2;a<=1000;a++)        //a 是 2 ~1000 的整数,检查它是否完数
   {n=0;                       //n 用来累计 a 的因子的个数
    s=a;                       //s 用来存放尚未求出的因子之和,开始时等于 a
    for (i=1;i<a;i++)          //检查 i 是否是 a 的因子
      if (a%i==0)              //如果 i 是 a 的因子
        {n++;                  //n 加 1,表示新找到一个因子
         s=s-i;                //s 减去已找到的因子,s 的新值是尚未求出的因子之和
         switch(n)             //将找到的因子赋给 k1 ~ k9,或 k10
           {case 1:
              k1=i;  break;    //找出的第 1 个因子赋给 k1
            case 2:
              k2=i;  break;    //找出的第 2 个因子赋给 k2
            case 3:
              k3=i;  break;    //找出的第 3 个因子赋给 k3
            case 4:
              k4=i;  break;    //找出的第 4 个因子赋给 k4
            case 5:
              k5=i;  break;    //找出的第 5 个因子赋给 k5
            case 6:
              k6=i;  break;    //找出的第 6 个因子赋给 k6
            case 7:
              k7=i;  break;    //找出的第 7 个因子赋给 k7
            case 8:
              k8=i;  break;    //找出的第 8 个因子赋给 k8
            case 9:
              k9=i;  break;    //找出的第 9 个因子赋给 k9
            case 10:
              k10=i;  break;   //找出的第 10 个因子赋给 k10
           }
        }
    if (s==0)
    {
      printf("%d ,Its factors are ",a);
      if (n>1)  printf("%d,%d",k1,k2);      //n>1 表示 a 至少有 2 个因子
      if (n>2)  printf(",%d",k3);           //n>2 表示至少有 3 个因子,故应再
                                              输出一个因子
      if (n>3)  printf(",%d",k4);           //n>3 表示至少有 4 个因子,故应再
                                              输出一个因子
```

```
        if (n>4)  printf(",%d",k5);            //以下类似
        if (n>5)  printf(",%d",k6);
        if (n>6)  printf(",%d",k7);
        if (n>7)  printf(",%d",k8);
        if (n>8)  printf(",%d",k9);
        if (n>9)  printf(",%d",k10);
        printf("\n");
      }
    }
    return 0;
}
```

运行结果：

```
6, its factors are 1,2,3
28, its factors are 1,2,4,7,14
496, its factors are 1,2,4,8,16,31,62,124,248
(一共找到 3 个完数)
```

方法二：

```
#include<stdio.h>
int main()
{ int m,s,i;
  for (m=2;m<1000;m++)
    {s=0;
     for (i=1;i<m;i++)
       if ((m%i)==0) s=s+i;
     if(s==m)
      {printf("%d,its factors are ",m);
       for (i=1;i<m;i++)
         if (m%i==0)  printf("%d ",i);
       printf("\n");
      }
    }
  return 0;
}
```

运行结果：

```
6, its factors are 1 2 3
28, its factors are 1 2 4 7 14
496, its factors are 1 2 4 8 16 31 62 124 248
```

4.10　一个球从 100m 高度自由落下，每次落地后反跳回原高度的一半，再落下，再反弹。求它在第 10 次落地时，共经过了多少米？第 10 次反弹多高？

解：编写程序如下：

```
#include<stdio.h>
int main()
{ double sn=100,hn=sn/2;
  int n;
  for (n=2;n<=10;n++)
   {
   sn=sn+2*hn;          //第 n 次落地时共经过的米数
   hn=hn/2;             //第 n 次反跳高度
   }
   printf("第 10 次落地时共经过%f 米\n",sn);
   printf("第 10 次反弹%f 米\n",hn);
   return 0;
}
```

运行结果：

```
第 10 次落地时共经过 299.609375 米
第 10 次反弹 0.097656 米
```

本章习题中 11～13 题，是用迭代方法求方程的数值解，要设计数值算法，需要有高等数学的初步知识。如果读者未学过高等数学，可不必做这几道题。如果读者已学过高等数学，建议尝试做这几道题，至少能看懂题目解释和程序，以了解怎样构造一个数值算法。这方面的知识是很有用的。

4.11　用迭代法求 $x=\sqrt{a}$。求平方根的迭代公式为：

$$x_{n+1}=\frac{1}{2}\left(x_n+\frac{a}{x_n}\right)$$

要求前后两次求出的 x 的差的绝对值小于 10^{-5}。

解：解题思路：用迭代法求平方根的算法如下：

（1）设定一个 x 的初值 x_0；

（2）用以上公式求出 x 的下一个值 x_1；

（3）再将 x_1 代入以上公式的右侧的 x_n，求出 x 的下一个值 x_2；

（4）如此继续下去，直到前后两次求出的 x 值（x_n 和 x_{n+1}）满足以下关系：

$$|x_{n+1}-x_n|<10^{-5}$$

为了便于程序处理，今只用 x_0 和 x_1，先令 x 的初值 $x_0=a/2$（也可以是另外的值），求出 x_1；如果此时 $|x_1-x_0|\geqslant 10^{-5}$，就使 $x_1=>x_0$，然后用这个新的 x_0 求出下一个 x_1；如此反复，直到 $|x_1-x_0|<10^{-5}$ 为止。

编写程序如下：

```
#include<stdio.h>
#include<math.h>
int main()
{ float a,x0,x1;
  printf("enter a positive number:");
```