# 项目 3　“学生选课系统”多信息输入与输出

**学习目标**

1. 掌握循环结构在实际中的应用。
2. 掌握 while 和 do…while 语句的使用规则以及两个语句的使用区别。
3. 掌握 for 语句的使用规则以及 for 语句中各表达式的含义。
4. 掌握 break 和 continue 语句在循环结构中的应用。
5. 能够结合循环结构的用法应用到实际问题中。

**项目内容分析——信息重复输出**

在该系统完成首界面输入后，通过终端输入操作编号 1，进入管理员角色，可以通过管理员角色进行学生选课信息的录入与查询，这一操作可以将涉及多个学生的选课信息输入和输出。因为有多个学生信息，所以会有多次操作。考虑使用相同的处理语句解决多次执行，用循环结构来解决。在本项目中，学生信息的存储、输入和输出涉及多个方面的知识，这里仅以输入输出多个学生信息为简单案例，展开循环结构知识的学习。

## 任务说明

本次任务要为学生选课系统中的信息输出提供思路。为简化程序，这里以学生部分信息的输出举例。输入学生的部分成绩信息，并输出。效果如下所示：

现有学生部分信息列表如下：

| 学号 | 数据结构 | 高等数学 | C 语言 |
|---|---|---|---|
| 1 | 85 | 74 | 96 |
| 2 | 84 | 82 | 91 |
| 3 | 88 | 75 | 95 |

## 知识点巩固

要完成该界面设计，需要学习循环结构相关的知识点。绝大多数的应用程序都包含重复处理，循环结构又称为重复结构。

循环结构和顺序结构、选择结构是结构化程序设计的 3 种基本结构，它们是各种复杂程序的基本组成单元。常用的循环结构有 while、do…while 和 for 三类循环语句。

# 任务 3.1　while 语句的应用


观看视频

while 语句的一般形式：

```
while (表达式) 语句
```

while 循环的特点：先判断条件表达式，后执行循环体语句。

while 语句可简单地记为：只要当循环条件表达式为真(给定的条件成立)，就执行循环体语句。

“语句”就是循环体。循环体可以是一个简单的语句，也可以是复合语句(用花括号括起来的若干语句)。

执行循环体的次数是由循环条件控制的，这个循环条件就是上面一般形式中的“表达式”，它也称为循环条件表达式。当此表达式的值为“真”(以非 0 值表示)时，就执行循环体语句；为“假”(以 0 表示)时，就不执行循环体语句。要构成一个有效的循环，应当指定两个条件：

(1) 需要重复执行的操作，这称为循环体。

(2) 循环结束的条件，即在什么情况下停止重复的操作。

**【课堂案例 3-1】** 用 while 语句统计 n!，n!＝1×2×3×…×n，输入 n 的值，计算结果。

```
#include<stdio.h>
int main()
{
    int n,i=1,f=1;
    printf("请输入 n 的值:\n");
    scanf("%d",&n);
    while(i<=n)
    {
        f=f*i;
        i++;
    }
    printf("%d!=%d\n",n,f);
    return 0;
}
```

运行结果：

```
请输入 n 的值:
5↙
5!=120
```

**【注意】** n 的值不要太大，避免 n! 的值溢出 int 数据类型的表示范围。

**【课堂案例 3-2】** 某学校期末考试成绩统计，要统计每个班中学生 4 门课程的总成绩，并输出。注：本班有 40 名学生。讨论如何统计。

```
#include<stdio.h>
int main()
{
float score1,score2,score3,score4,score5,sum;
int i=1;                                    //设整型变量 i 初值为 1
while(i<=40)                                //当 i 的值小于或等于 40 时执行花括号内的语句
{
printf("请输入第%d 个学生的成绩,中间用逗号分隔:\n",i);
scanf("%f,%f,%f,%f",&score1,&score2,&score3,&score4);
//输入 4 门课程的成绩
sum=score1+score2+score3+score4;      //计算课程的总成绩
printf("该学生的总成绩 sum=%.2f\n",sum);//输出课程的总成绩
i++;                                        //每执行完一次循环使 i 的值加 1
}
return 0;
}
```

运行结果：

```
请输入第 1 个学生的成绩,中间用逗号分隔:
64,84,74,95
该学生的总成绩 sum=317.00
请输入第 2 个学生的成绩,中间用逗号分隔:
96,85,74,93
该学生的总成绩 sum=348.00
请输入第 3 个学生的成绩,中间用逗号分隔:
81,65,74,92
该学生的总成绩 sum=312.00
请输入第 4 个学生的成绩,中间用逗号分隔:
…
```

## 任务 3.2　do…while 语句的应用

do…while 语句的特点：先无条件地执行循环体，然后判断循环条件是否成立。

一般形式为：

```
do
    循环体语句
while (表达式);
```

**【课堂案例 3-3】** 用 do…while 循环语句求 1＋2＋3＋…＋100。

**【解题思路】**

```
#include<stdio.h>
int main()
{   int i,sum=0;
    i=1;
```

```
    do
    {
        sum=sum+i;
        i++;
    }while(i<=100);
    printf("%d\n",sum);
}
```

**【说明】** 通过 do…while 循环语句的应用，对比 while 与 do…while 两种循环，若循环体语句相同，起始条件成立，则二者的结果是等价的。

**【课堂案例 3-4】** 募集慈善基金 10000 元，有若干人捐款，每输入一个人的捐款数后，就计算当时的捐款总和。当某一次输入捐款数后，总和达到或超过 10000 元时，即宣告结束，输出最后的累加捐款总额。

**【解题思路】** 设计一个循环结构，在其中输入捐款数，求出累加值，然后检查此时的累加值是否达到或超过预定值，如果达到了，就结束循环操作。

```
#include<stdio.h>
int main()
{   float amount,sum=0;
    do
    {
      scanf("%f",&amount);
      sum=sum+amount;
    }while(sum<10000);
    printf("捐款结束,共捐款%9.2f元\n",sum);
}
```

**【课堂案例 3-5】** 求 $S_n$＝a＋aa＋aaa＋…＋aa…a(n 个 a)的值，其中 a 是一个数字，n 表示 a 的位数，n 和 a 由键盘输入。例如 2＋22＋222＋2222＋22222(此时 n＝5)。

```
#include<stdio.h>
int main()
{
int  n,t=0,i,sum=0,a;
scanf("%d %d",&n,&a);                    //n 为位数,a 为数字
for(i=1; i<=n; i++) {                    //执行次数为位数
t=t*10+a;
sum=sum+t;
}
printf("%d\n",sum);
return 0;
}
```

注意 while 与 do…while 语句的使用区别：

while 语句是先判断，再执行循环语句，有可能循环语句一次都没有被执行到；do…while 语句是先执行一次循环语句，再进行判断，循环语句至少执行一次。

# 任务 3.3　for 语句的应用



C 语言中还提供了 for 语句用于解决循环问题，for 语句的使用相对灵活、方便，可以代替 while 语句。

观看视频

**1. for 语句的一般形式**

for 语句的一般形式为：

```
for(表达式 1;表达式 2;表达式 3)语句
```

(1) 表达式 1 表示设置初始值，只执行一次。可以对循环变量赋初值，也可以对其他变量赋初值，如果有多条赋初值的语句，中间用逗号隔开。

```
int i,sum;
for(i=1,sum=0;i<=100;i++)
```

(2) 表达式 2 是循环执行的条件，当结果为真时，执行循环体语句。

(3) 表达式 3 为使循环趋于结束的表达式，如循环变量的增值等。

它的执行过程如下：

(1) 求解表达式 1。

(2) 求解表达式 2，若其值为真(非 0)，则执行 for 语句中指定的内嵌语句，然后执行下面第(3)步；若其值为假(0)，则结束循环，转到第(5)步。

(3) 求解表达式 3。

(4) 转回上面第(2)步继续执行。

(5) 循环结束，执行 for 语句下面的一个语句。

其执行过程可用图 3-1 表示。

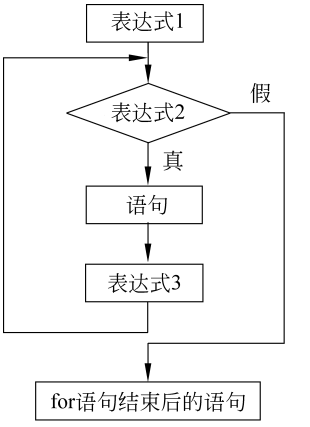


图 3-1　for 语句的执行流程

**【注意】**

(1) for 循环中的“表达式 1”(循环变量赋初值)、“表达式 2”(循环条件)和“表达式 3”(循环变量增量)都是选择项，可以省略，但“;”不能省略。

(2) 表达式 1 省略时，表示在 for 语句中不对变量赋初值，此时如果需要赋初值，则在 for 语句前进行赋值。

```
int i=1,sum=0;
for(;i<=100;i++)
sum+=i;
```

(3) 表达式 2 中的表达式可以是关系表达式、逻辑表达式、数值表达式或字符表达式。

```
char c;
for(;(c=getchar())!='\n';)
putchar(c);
```

表达式 2 省略时，表示无条件运行，此时如果不设置循环结束的条件将成为死循环。

```
int i=2,sum=0;
for(;;i++)
sum+=i;
```

等价于：

```
int i=2,sum=0;
while(1)
{
sum+=i;
i++;
}
```

若无其他结束语句，将成为死循环，可以在循环体中添加 break 语句结束循环。

(4) 表达式 3 可以是一条语句，也可以是多条语句，中间用逗号隔开。

```
int i=1,sum=0;
for(;i<=100;sum+=i,i++)
```

表达式 3 省略时，应在循环体中添加使循环趋于结束的语句。

```
int i=1,sum=0;
for(;i<=100;)
{
sum+=i;
i++;
}
```

以上的语句如果在循环体中没有 i＋＋语句，则没有使循环趋于结束的操作，成为死循环。

**【课堂案例 3-6】** 求解表达式 1－3＋5－7＋9－11＋…－99 的结果。

**【分析】** 规律是每个操作数均为奇数，而操作符＋、－交替。

```
#include<stdio.h>
int main()
{
    int i=1,sum=0,t=-1;
    for(;i<=99;i+=2)
    {
        t=-t;                    //分析:i=1 时,t=1;i=3 时,t=-1;i=5 时,t=1,以此类推
        sum=sum+i*t;
    }
    printf("sum=%d\n",sum);
    return 0;
}
```

运行结果：

```
sum=-50
```



**2. 几种循环的比较**

(1) 对于 while 和 do…while 循环，循环体中应包括使循环趋于结束的语句。for 语句功能最强，一般在表达式 3 中包括使循环趋于结束的语句，也可在循环体中，相对灵活。

(2) 用 while 和 do…while 循环时，循环变量初始化的操作应在 while 和 do…while 语句之前完成，而 for 语句可以在“表达式 1”中实现循环变量的初始化。

**【课堂案例 3-7】** 输入一个整数，将所有的因数输出。

**【分析】** 一个整数中从 1 开始进行判断，只要取余结果为 0，就是该整数的因数。

```
#include<stdio.h>
int main()
{
int x,i;
printf("请输入一个整数:\n");
scanf("%d",&x);
printf("%d的因数是:",x);
for(i=1;i<=x;i++)
if(x%i==0)
printf("%3d",i);
return 0;
}
```

运行结果：

```
请输入一个整数:
18
18的因数是:  1  2  3  6  9 18
```

**3. 多重循环**

循环结构跟分支结构一样，都可以实现嵌套。循环的嵌套即循环中又有循环，执行时，先执行外循环一步，再执行内循环；再执行外循环一步，再执行内循环；以此类推，直到外循环完全结束。

**【课堂案例 3-8】** 输出以下图案。

```
*
**
***
****
```

**【分析】** 一共有 4 行，每行输出不同的 * 个数。

第一行，1 个 * 。

第二行，2 个 * 。

第三行，3 个 * 。

……

得出：第 i 行有 i 个 * 。

```
#include<stdio.h>
int main()
{
  int i,j;
  for(i=1;i<=4;i++)
  {
    for(j=1;j<=i;j++)        //内循环,表示第 i 行有 i 个 *
    printf("*");
    printf("\n");            //每行输出完成后,打印一个换行符,即每个内循环结束后执行
  }
  return 0;
}
```

思考：用“*”输出以下的菱形图案。

```
   *
  ***
 *****
*******
 *****
  ***
   *
```

```
#include<stdio.h>
int main()
{int i,j,k;
for(i=1;i<=4;i++)
{
for(j=1;j<=4-i;j++)
printf(" ");
for(k=1;k<=2*i-1;k++)
printf("*");
printf("\n");}
for(i=1;i<=3;i++)
{
for(j=1;j<=i;j++)
printf(" ");
for(k=1;k<=7-2*i;k++)
printf("*");
printf("\n");
}
}
```

**【课堂案例 3-9】** 打印九九乘法表。

九九乘法表，效果如下：

```
1*1=1
1*2=2  2*2=4
1*3=3  2*3=6  3*3=9
1*4=4  2*4=8  3*4=12  4*4=16
1*5=5  2*5=10  3*5=15  4*5=20  5*5=25
1*6=6  2*6=12  3*6=18  4*6=24  5*6=30  6*6=36
1*7=7  2*7=14  3*7=21  4*7=28  5*7=35  6*7=42  7*7=49
1*8=8  2*8=16  3*8=24  4*8=32  5*8=40  6*8=48  7*8=56  8*8=64
1*9=9  2*9=18  3*9=27  4*9=36  5*9=45  6*9=54  7*9=63  8*9=72  9*9=81
```

可以通过循环嵌套来实现，如果将每个表达式看成 i * j，那么第一行是 i=1、j<=i，第二行是 i=2、j<=i，以此类推。

```
#include<stdio.h>
int main()
{
int i,j;
for (i = 1; i <= 9; i++)
{
    for (j = 1; j <= i; j++)
      printf("%d*%d=%-4d", j, i,i*j);
    printf("\n");
}
return 0;
}
```

其中有两点需要注意，首先是 %-4d，这里的-表示左对齐，因为默认是右对齐，里面的 4 表示占 4 个字符；其次是在每一次循环结束之后会打印一个回车符号以换行。

## 任务 3.4　break 语句与 continue 语句的应用

**1. break 语句**

当 break 关键字用于 while、for 循环时，会终止循环而执行整个循环语句后面的代码。break 关键字和 if 语句一起使用，即满足条件时便跳出循环。

一般形式为：

```
break;
```

**【注意】** break 语句不能用于循环语句和 switch 语句之外的任何其他语句中。

**【课堂案例 3-10】** 输入一个整数，找出该整数中最小的因数(1 以外)，并输出。

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

```
{
    int n,i,k;
    printf("请输入一个整数:\n");
    scanf("%d",&n);
    for(i=2;i<=n;i++)
    if(n%i==0)
    {
        k=i;
        break;
    }
    printf("整数%d的最小因数为(1除外):%d\n",n,k);
    return 0;
}
```

运行结果：

```
请输入一个整数:
9↙
整数9的最小因数为(1除外):3
```

**【课堂案例 3-11】** 使用 while 循环计算 1＋2＋…＋100 的值，利用 break 语句实现。

```
#include<stdio.h>
int main(){
int i=1, sum=0;
while(1){                              //循环条件为死循环
sum+=i;
i++;
if(i>100) break;
}
printf("%d\n", sum);
return 0;
}
```

运行结果：

```
5050
```

while 循环条件为 1，是一个死循环。当执行到第 100 次循环的时候，计算完 i＋＋后 i 的值为 101，此时 if 语句的条件 i＞100 成立，执行 break 语句，结束循环。

在多层循环中，一个 break 语句只向外跳一层。例如，输出一个 4×4 的整数矩阵。

```
#include<stdio.h>
int main(){
    int i=1, j;
    while(1){                          //外层循环
        j=1;
        while(1){                      //内层循环
```

```
            printf("%-4d", i*j);
            j++;
            if(j>4) break;              //跳出内层循环
        }
        printf("\n");
        i++;
        if(i>4) break;                  //跳出外层循环
    }
return 0;
}
```

运行结果：

```
1  2  3   4
2  4  6   8
3  6  9   12
4  8  12  16
```

当 j>4 成立时，执行 break，跳出内层循环；外层循环依然执行，直到 i>4 成立，跳出外层循环。内层循环共执行了 4 次，外层循环共执行了 1 次。

**2. continue 语句**

continue 语句只用在 while、for 循环中，常与 if 条件语句一起使用，判断条件是否成立。continue 语句的作用是跳过循环体中剩余尚未执行的语句而强制进入下一次循环。

一般形式：

```
continue;
```

**【课堂案例 3-12】** 把 100～200 的奇数输出。

```
#include<stdio.h>
int main()
{
int n;
for(n=100;n<=200;n++)
{
if(n%2==0)
continue;
printf("%d",n);
}
}
```

**【说明】** 当 n 能被 2 整除时，执行 continue 语句，结束本次循环（跳过 printf()函数语句），只有 n 不能被 2 整除时才执行 printf()函数。

**【课堂案例 3-13】** 输入一串数字，除 4 和 5 外，其他都原样打印输出。

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

```
char c = 0;
while(c!='\n')                          //按 Enter 键结束循环
c=getchar();
if(c=='4' || c=='5'){                   //按的是数字键 4 或 5
continue;                               //跳过当次循环,进入下一次循环
putchar(c);
}
return 0;
}
```

运行结果：

```
0123456789↙
01236789
```

开始，变量 c 的值为 0，即为'\0'，循环条件 c!＝'\n'成立，开始第一次循环。getchar() 使程序暂停执行，等待用户输入，直到用户按 Enter 键才开始读取字符。

本例我们输入的是 0123456789，当读取到 4 或 5 时，if 的条件 c＝＝'4'||c＝＝'5'成立，就执行 continue 语句，结束当前循环，直接进入下一次循环，也就是说 putchar(c)不会被执行到。而读取到其他数字时，if 的条件不成立，continue 语句不会被执行到，putchar(c)就会输出读取到的字符。

**3. continue 语句和 break 语句的区别**

break 语句与 continue 语句的对比：break 语句用来结束所有循环，循环语句不再有执行的机会；continue 语句用来结束本次循环，直接跳到下一次循环，如果循环条件成立，还会继续循环。

## 任务实现

结合所学的循环知识，完成信息的输出。

| 学号 | 数据结构 | 高等数学 | C 语言 |
|---|---|---|---|
| 1 | 85 | 74 | 96 |
| 2 | 84 | 82 | 91 |
| 3 | 88 | 75 | 95 |

**【分析】** 通过查看输出内容，明确要输入的学生成绩信息及条目个数，确定循环次数及要输入的内容。

实现的功能代码如下。

方法一：不使用循环。

```
#include<stdio.h>
#include<string.h>
int main()
{
    int sno1,score11,score12,score13;
```

```
    int sno2,score21,score22,score23;
    int sno3,score31,score32,score33;
    printf("请输入学生的基本信息:\n 学号\t 数据结构   高等数学   C 语言\n");
    scanf("%d%d%d%d",&sno1,&score11,&score12,&score13);
    scanf("%d%d%d%d",&sno2,&score21,&score22,&score23);
    scanf("%d%d%d%d",&sno3,&score31,&score32,&score33);
    printf("学生的成绩信息如下:\n") ;
    printf("学号\t 数据结构\t 高等数学\tC 语言\n");
    printf("%d\t%d\t\t%d\t\t%d\n",sno1,score11,score12,score13);
    printf("%d\t%d\t\t%d\t\t%d\n",sno2,score21,score22,score23);
    printf("%d\t%d\t\t%d\t\t%d\n",sno3,score31,score32,score33);
    return 0;
}
```

由上可知,不使用循环结构,会导致变量个数增加,且冗余语句较多,相对复杂,不建议使用此种方法。

方法二:使用循环结构(借助数组结构,在项目 4 中将详细讲解)。

```
#include<stdio.h>
#include<string.h>
int main()
{
    int sno[3],score1[3],score2[3],score3[3];
    int i;
    printf("请输入学生的基本信息:\n 学号\t 数据结构   高等数学   C 语言\n");
    for(i=0;i<3;i++)
    scanf("%d%d%d%d",&sno[i],&score1[i],&score2[i],&score3[i]);
    printf("学生的成绩信息如下:\n") ;
    printf("学号\t 数据结构\t 高等数学\tC 语言\n");
    for(i=0;i<3;i++)
    printf("%d\t%d\t\t%d\t\t%d\n",sno[i],score1[i],score2[i],score3[i]);
    return 0;
}
```

**【分析】** 这里将每一列数据通过一个数组进行数据的存储,利用循环依次输出每一个数组元素,简化了方法一中的部分重复语句。

运行结果:

```
请输入学生的基本信息:
学号   数据结构   高等数学   C 语言
 1       85         74       96↙
 2       84         82       91↙
 3       88         75       95↙
学生的成绩信息如下:
学号   数据结构   高等数学   C 语言
 1       85         74        96
 2       84         82        91
 3       88         75        95
```

## 课后实验

# 实验 3　循环结构程序设计

**一、实验目的**

1. 熟练掌握 while、do…while 和 for 循环控制语句的特点及其编程应用。

2. 掌握 break、continue 语句的功能及应用。

3. 掌握嵌套循环结构程序的设计方法。

4. 掌握循环结构程序的调试方法。

**二、实验内容**

1. 输入 10 个实数，输出其和、平均值及最大值(分别用 while、for 语句编程实现)。

2. 数据统计问题：从键盘输入 8 个正整数，统计其中不大于 80 的数值个数。

3. 求斐波那契数列前 40 个数。这个数列有如下特点：第 1 和第 2 个数为 1 和 1。从第 3 个数开始，该数是其前面两个数之和。即：

F(1)＝1(n＝1)

F(2)＝1(n＝2)

F(n)＝F(n－1)＋F(n－2)(n≥3)

4. 输入一个正整数，判断并输出其是否为素数。

5. 输入一串字符，将其中的大写字母转换成对应的小写字母并输出。如输入：EFAB5623AB#$，则输出：efabab。

6. 从键盘输入一串字符(按 Enter 键结束)，统计其中字母、数字及其他字符的个数并输出。

7. 输入一个正整数 n，求 1!＋2!＋…＋n! 并输出。

8. 利用 for 循环，求解 300～500 的所有素数，每行输出 10 个。

9. 找规律：有一个 4 位的正整数，前两位数字相同，后两位数字相同，但前后位的数字不同；该 4 位数各个位的和加起来正好是 12 的整数倍。请根据要求求出这样的 4 位数有哪些，并输出。

10. 有 10 名评委对青年教师教学能力大赛打分，去掉一个最高分，去掉一个最低分，求每个青年教师的最后得分。

11. 假设银行一年整存零取的月息为 1.3%。现在某人手中有一笔钱，他打算在今后 5 年的每年年底取出 1500 元，到第 5 年时刚好取完，请算出他存钱时应存入多少。

**三、实验指导**

1. 输入 10 个实数，输出其和、平均值及最大值(分别用 while、for 语句编程实现)。

**【程序指导】**

(1) while 循环实现：

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

```
{
    float x,sum=0,avg=0,max;
    int i=1;
    printf("请输入 10 个浮点型的数:\n");
    scanf("%f",&x);
    max=x;
    while (i<=9)
    {
        scanf("%f",&x);
        sum=sum+x;
        if (x>max)
            max=x;
        i=i+1;
    }
    avg=sum/10;
    printf("和为:%.2f\n 平均值为:%.2f\n 最大值为:%.0f\n",sum,avg,max);
    return 0;
}
```

(2) for 循环实现:

```
#include<stdio.h>
int main()
{
    float x,sum=0,avg=0,max;
    int i;
    printf("请输入 10 个浮点型的数:\n");
    scanf("%f",&x);
    max=x;
    for(i=1;i<=9;i++)
    {
        scanf("%f",&x);
        sum=sum+x;
        if (x>max)
        max=x;
    }
    avg=sum/10;
    printf("和为:%.2f\n 平均值为:%.2f\n 最大值为:%.0f\n",sum,avg,max);
    return 0;
}
```

运行结果:

```
请输入 10 个浮点型的数:
84.5
65
87
65.5
67.5
```

```
84
72
45
65
85
和为:636.00
平均值为:63.60
最大值为:87
```

2. 数据统计问题：从键盘输入 8 个正整数，统计其中不大于 80 的数值个数。

**【程序指导】**

```
#include<stdio.h>
int main()
{
int i,x,n=0;
printf("请输入 8 个正整数,中间用空格隔开:\n");
for(i=1;i<=8;i++)
{
scanf("%d",&x);
if(x<=80)
{
n++;
if(n==1) printf("不大于 80 的数值有:\n");
printf("%d ",x);
}
}
printf("\n 完成统计,共有%d 个符合条件的整数\n",n);
return 0;
}
```

运行结果：

```
请输入 8 个正整数,中间用空格隔开:
52 74 71 85 96 53 74 84
不大于 80 的数值有:
52 74 71 53 74
完成统计,共有 5 个符合条件的整数
```

3. 求斐波那契数列前 40 个数。这个数列有如下特点：第 1 和第 2 个数为 1 和 1。从第 3 个数开始，该数是其前面两个数之和。即：

$F(1)=1(n=1)$

$F(2)=1(n=2)$

$F(n)=F(n-1)+F(n-2)(n\geqslant 3)$

**【程序指导】**

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

```
{
long int f1,f2;
int i;
f1=1;f2=1;/*初始化前两个数*/
for(i=1;i<=20;i++)
{
printf("%12ld%12ld",f1,f2);
if(i%2==0)printf("\n");
f1=f1+f2;
f2=f2+f1;
}
}
```

**【说明】** 每个循环输出两个数，if(i%2==0)printf("\n")；表示每两个循环输出一个换行，即每4个数一行；数据定义为long类型，是因为后面的斐波那契数列会超出int所表示的范围，因此要用长整型。

4. 输入一个正整数，判断并输出其是否为素数。

**【分析】** 素数是因数只有1和该数本身的整数，判断是否为素数，只需要判定是否存在1和该数本身以外的因数。

**【程序指导】**

```
#include<stdio.h>
int main()
{
    int n,i;
    printf("请输入一个整数:\n");
    scanf("%d",&n);
    for(i=2;i<n;i++)
    if(n%i==0)     //如果n%i==0,则说明i是n的因数,即找到了1和该数本身以外的因数
    break;
    if(i==n)
    printf("%d是素数\n",n);
    else
    printf("%d不是素数\n",n);
    return 0;
}
```

运行结果：

```
请输入一个整数:
13
13是素数
```

5. 输入一串字符，将其中的大写字母转换成对应的小写字母并输出。如输入：EFAB5623AB#$，则输出：efabab。

【程序指导】

```
#include<stdio.h>
int main()
{
char c;
int i;
printf("请输入一串字符:\n");
while((c=getchar())!='\n')
{
if(c>='A' && c<='Z')
c=c+32;
if(c>='a'&&c<='z')
putchar(c);
}
return 0;
}
```

运行结果：

```
请输入一串字符:
EFAB5623AB#$
efabab
```

6. 从键盘输入一串字符(按 Enter 键结束),统计其中字母、数字及其他字符的个数并输出。

【程序指导】

```
#include<stdio.h>
int main()
{
    int letters,space,digit,other;
    char c;
    letters=space=digit=other=0;
    while((c=getchar())!='\n')
    {
        if(c>='a'&&c<='z'||c>='A'&&c<='Z')
        letters ++;
        else if(c>='0'&&c<='9')
            digit++;
        else if(c==' ')
            space++;
        else
            other++;
    }
printf("letters=%d,space=%d,digit=%d,other=%d\n",letters,space,digit,
other);
return 0;
}
```

运行结果：

```
sdjfkjYUYUY8&*&*&   7979
letters=11,space=3,digit=5,other=5
```



7. 输入一个正整数 n,求 1！＋2！＋…＋n！并输出。

**【程序指导】**

方法一：

```
#include<stdio.h>
int main()
{
    int i,n,k=1,s=0;
    scanf("%d",&n);
    for(i=1;i<=n;i++)
    {
        k*=i;
        s+=k;
    }
    printf("s=%d",s);
    return 0;
}
```

方法二：

```
#include<stdio.h>
int main()
{
    int i=1,n,k=1,s=0;
    scanf("%d",&n);
    while(i<=n)
    {
        k*=i;
        s+=k;
        i++;
    }
    printf("s=%d",s);
    return 0;
}
```

方法三：

```
#include<stdio.h>
int main()
{
    int i=1,n,k=1,s=0;
    scanf("%d",&n);
    do
    {
```

```
        k*=i;
        s+=k;
        i++;
    }
    while(i<=n);
    printf("s=%d",s);
}
```

运行结果：

```
5
s=153
```

8. 利用 for 循环，求解 300～500 的所有素数，每行输出 10 个。

**【程序指导】**

```
#include<stdio.h>
#include<math.h>        //用到了 sqrt()函数,需要使用 math 函数库
int main()
{
int m,i,k,n=0;
for(m=300;m<=500;m++)
{
k=sqrt(m);
for(i=2;i<=k;i++)
if(m%i==0)
break;
if(i>=k+1)              //如果 i<=k,说明中间没有执行到 break 语句,即 m%i==0 不成立,
                        //也就是除 1 和它本身外,没有其他因数
{
printf("%4d",m);
n=n+1;
if(n%10==0)
printf("\n");
}
}
return 0;
}
```

运行结果：

```
307 311 313 317 331 337 347 349 353 359
367 373 379 383 389 397 401 409 419 421
431 433 439 443 449 457 461 463 467 479
487 491 499
```

9. 找规律：有一个 4 位的正整数，前两位数字相同，后两位数字相同，但前后位的数字不同；该 4 位数各个位的和加起来正好是 12 的整数倍。请根据要求求出这样的 4 位数有