第5章 循环结构程序设计

5.1为什么需要循环控制

- 在日常生活中或是在程序所处理的问题中常常遇到需要重复处理的问题
  - 要向计算机输入全班50个学生的成绩
  - 分别统计全班50个学生的平均成绩
  - 求30个整数之和
  - 教师检查30个学生的成绩是否及格

5.1为什么需要循环控制

- 例如：全班有50个学生，统计各学生三门课的平均成绩。

输入学生1的三门课成绩，并计算平均值后输出

```
scanf("%f,%f,%f",&s1,&s2,&s3);
aver=(s1+s2+s3)/3;
printf("aver=%7.2f",aver);
```

输入学生2的三门课成绩，并计算平均值后输出

```
scanf("%f,%f,%f",&s1,&s2,&s3);
aver=(s1+s2+s3)/3;
printf("aver=%7.2f",aver);
```

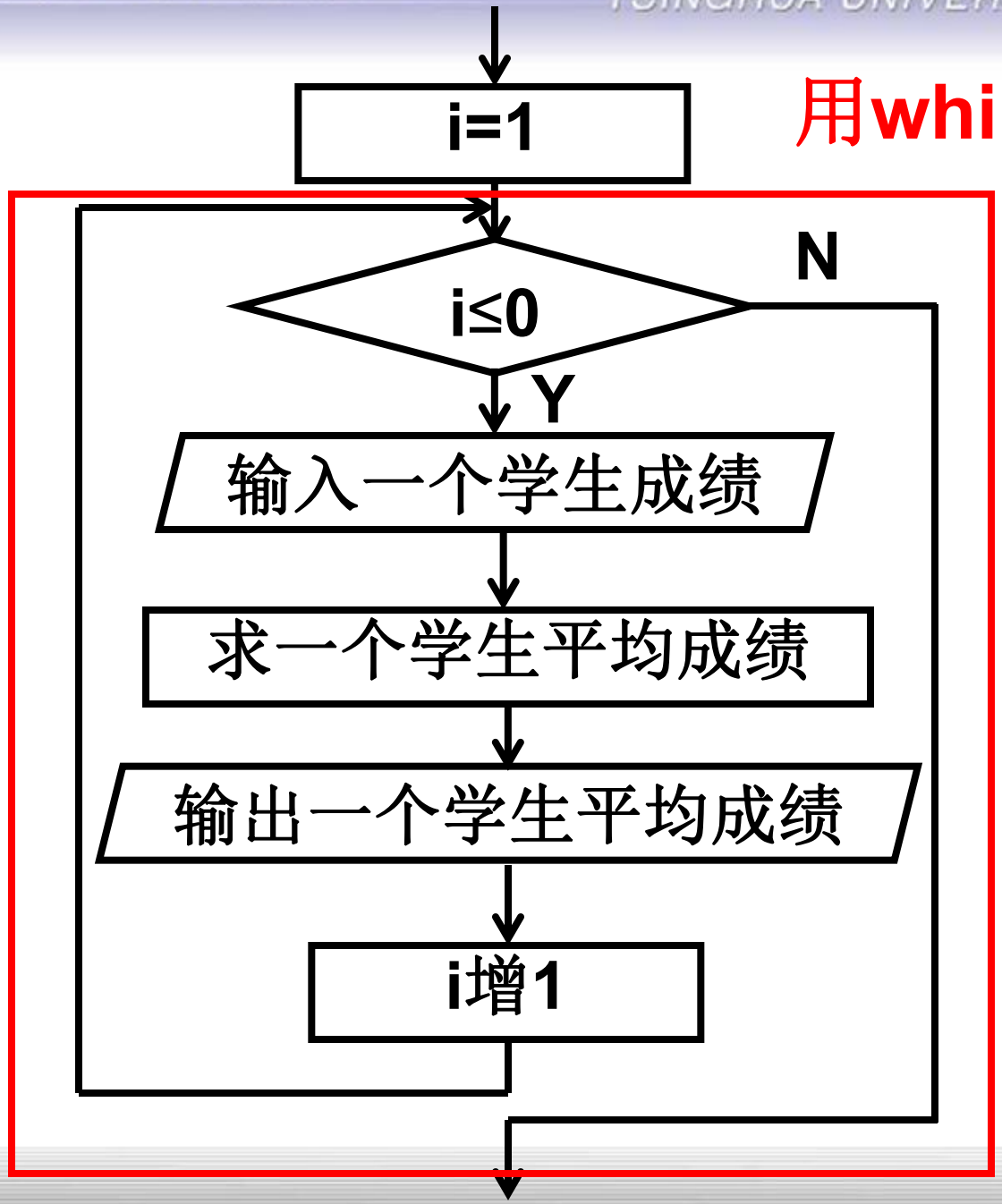
要对50个学生进行相同操作　重复50次

- 大多数的应用程序都会包含循环结构
- 循环结构和顺序结构、选择结构是结构化程序设计的三种基本结构，它们是各种复杂程序的基本构造单元

5.2用while语句实现循环

- 全班有50个学生，统计各学生三门课的平均成绩。

用while循环结构实现




```
while(i<=50)
{ scanf…… ;
  aver=…… ;
  printf…… ;
  i++;
}
```

while语句的一般形式如下：

while (表达式) 语句

循环体

while语句的一般形式如下：

while (表达式) 语句

循环条件表达式

“真”时执行循环体语句
“假”时不执行

while循环的特点是：
先判断条件表达式，后执行循环体语句

例5.1求1+2+3+…+100，即 $\sum_{n=1}^{100} n$

- 解题思路：
  - 这是累加问题，需要先后将100个数相加
  - 要重复100次加法运算，可用循环实现
  - 后一个数是前一个数加1而得
  - 加完上一个数i后，使i加1可得到下一个数

```
#include <stdio.h>
int main()
{
   int i=1,sum=0;
   while (i<=100)
   { sum=sum+i;
     i++;
   }
   printf("sum=%d\n",sum);
   return 0;
}
```

不能少

复合语句

```
#include <stdio.h>
int main()
{
   int i=1,sum=0;
   while (i<=100)
   {  sum=sum+i;
      i++;
   }
   printf("sum=%d\n",sum);
   return 0;
}
```

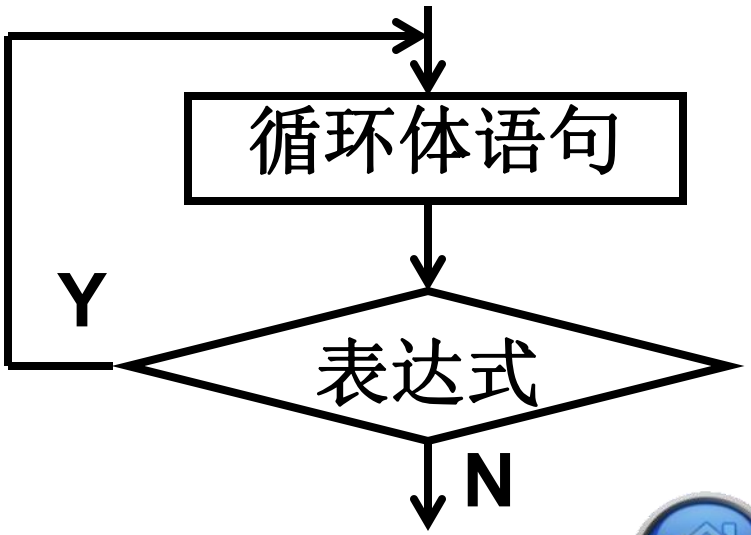
不能丢，否则循环永不结束

```
sum=5050
```

5.3用do---while语句实现循环

- 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
- do---while语句的一般形式为：

```
do
    语句
while (表达式);
```



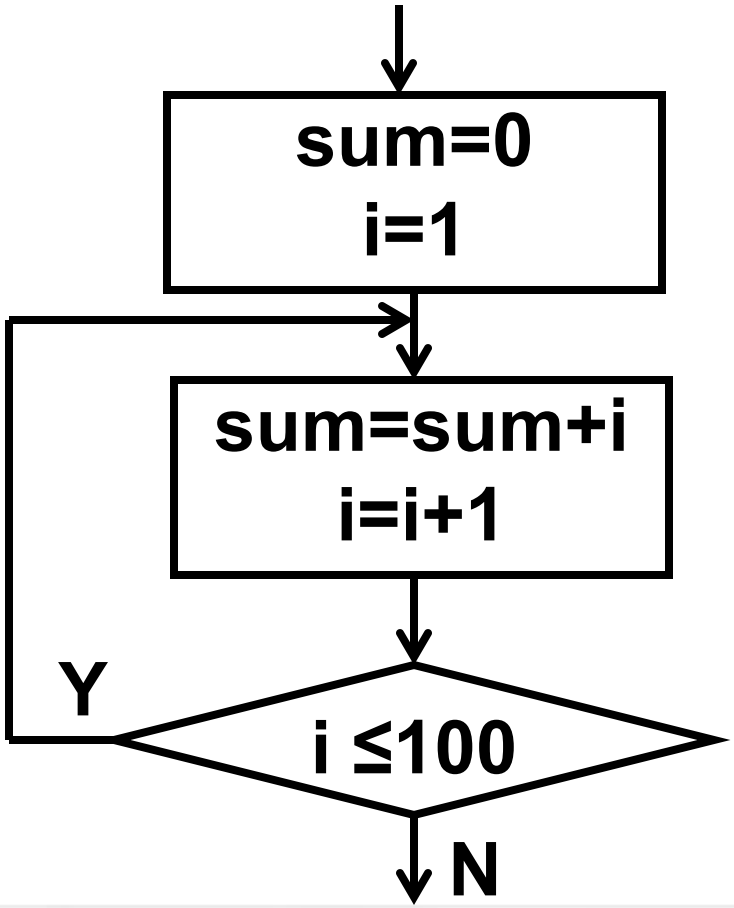
5.3用do---while语句实现循环

例5.2 用do…while语句求：

1+2+3+…+100，即 $\sum_{n=1}^{100} n$

5.3用do---while语句实现循环

- 解题思路：




```
i=1;   sum=0;
do
 {
    sum=sum+i;
    i++;
 }while(i<=100);
```

```
#include <stdio.h>
int main()
{  int i=1,sum=0;
   do
    {
       sum=sum+i;
       i++;
    }while(i<=100);
    printf("sum=%d\n",sum);
    return 0;
}
```

```
sum=5050
```

例5.3 while和do---while循环的比较。

当while后面的表达式的第一次的值为“真”时，两种循环得到的结果相同；否则不相同

```
in
pr
scanf(“%d”,&i);
while(i<=10)
{
  sum=sum+i;
  i++;
}
printf(“sum=%d\n",sum);
```

```
i=?1
sum=55
```

```
i=?11
sum=0
```

```
scanf(“%d”,&i);
do
{
   sum=sum+i;
   i++;
}while(i<=10);
printf(“sum=%d\n",sum);
```

```
i=?1
sum=55
```

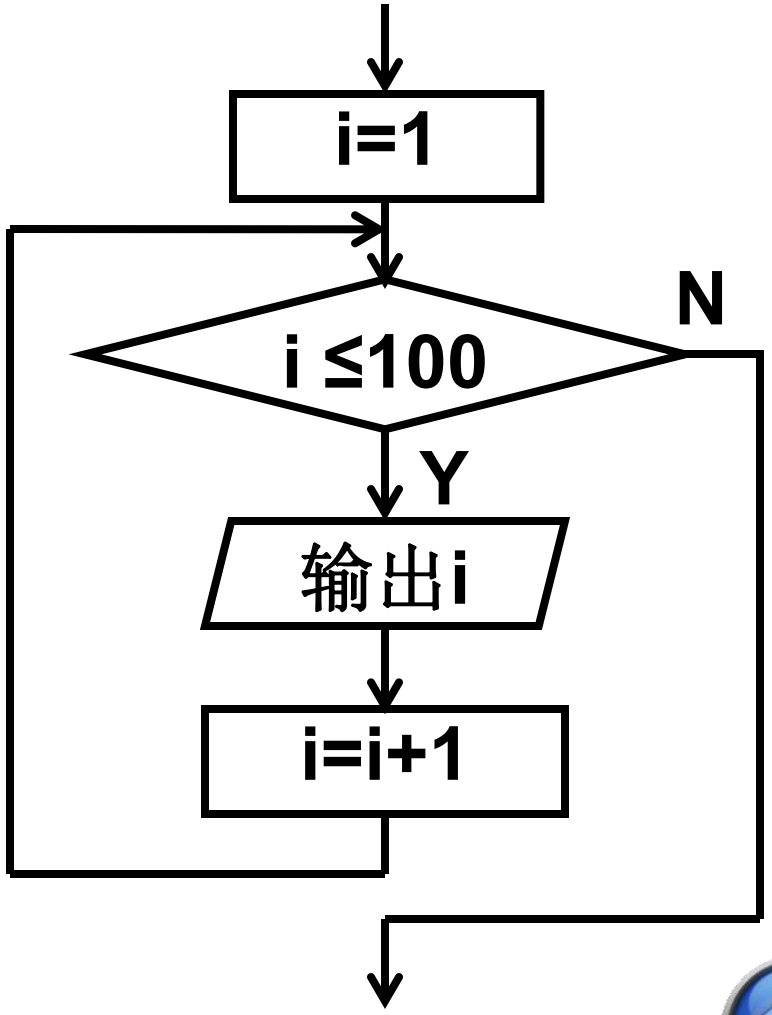
```
i=?11
sum=11
```

5.4用for 语句实现循环

- for语句不仅可以用于循环次数已经确定的情况，还可以用于循环次数不确定而只给出循环结束条件的情况
- for语句完全可以代替while语句

5.4用for 语句实现循环

```
for (i=1;i<=100;i++)
{
    printf("%d ", i );
}
```



5.4用for 语句实现循环

- for语句的一般形式为

  for(表达式1；表达式2；表达式3)

  语句

设置初始条件，只执行一次。可以为零个、一个或多个变量设置初值执行

5.4用for 语句实现循环

- for语句的一般形式为

  for(表达式1；表达式2；表达式3)

  语句

循环条件表达式，用来判定是否继续循环。在每次执行循环体前先执行此表达式，决定是否继续执行循环

5.4用for 语句实现循环

- for语句的一般形式为

  for(表达式1；表达式2；表达式3)

  语句

作为循环的调整器，例如使循环变量增值，它是在执行完循环体后才进行的

5.4用for 语句实现循环

- for语句的执行过程：

(1) 先求解表达式1

(2) 求解表达式2，若其值为真，执行循环体，然后执行下面第(3)步。若为假，则结束循环，转到第(5)步

(3) 求解表达式3

(4) 转回上面步骤(2)继续执行

(5) 循环结束，执行for语句下面的一个语句

5.4用for 语句实现循环

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

等价于

```
i=1;
while(i<=100)
{
    sum=sum+i;
    i++;
}
```

用for语句更简单、方便

5.4用for 语句实现循环

for(表达式1；表达式2；表达式3)

语句

一个或两个或三个表达式均可以省略

5.4用for 语句实现循环

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

与循环变量无关
合法

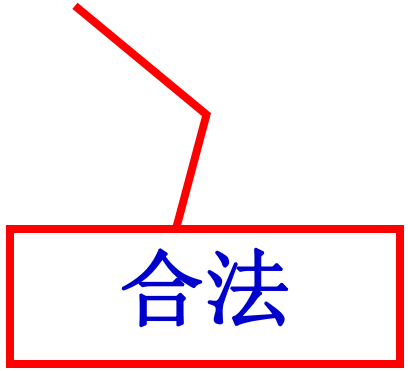
5.4用for 语句实现循环

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

for(i=0,j=100 ; i<=j;  i++,j--  )
      k=i+j;
```

逗号表达式
合法

5.4用for 语句实现循环

```
for(i=0; (c=getchar())!='\n'; i+=c)
        ;

for(     ; (c=getchar())!='\n';    )
        printf("%c", c);
```



5.5循环的嵌套

- 一个循环体内又包含另一个完整的循环结构，称为循环的嵌套
- 内嵌的循环中还可以嵌套循环，这就是多层循环
- 3种循环(while循环、do…while循环和for循环)可以互相嵌套

5.6几种循环的比较

(1)一般情况下，3种循环可以互相代替

(2) 在while和do---while循环中，循环体应包含使循环趋于结束的语句。

(3) 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。

5.7 改变循环执行的状态

5.7.1 用break语句提前终止循环

5.7.2 用continue语句提前结束本次循环

5.7.3 break语句和continue语句的区别

5.7.1 用break语句提前终止循环

- **break**语句可以用来从循环体内跳出循环体，即提前结束循环，接着执行循环下面的语句

5.7.1 用break语句提前终止循环

例5.4 在全系1000学生中，征集慈善募捐，当总数达到10万元时就结束，统计此时捐款的人数，以及平均每人捐款的数目。

5.7.1 用break语句提前终止循环

- 编程思路：
  - 循环次数不确定，但最多循环1000次
    - 在循环体中累计捐款总数
    - 用if语句检查是否达到10万元
    - 如果达到就不再继续执行循环，终止累加
  - 计算人均捐款数

5.7.1 用break语句提前终止循环

- 编程思路：
  - 变量**amount**，用来存放捐款数
  - 变量**total**，用来存放累加后的总捐款数
  - 变量**aver**，用来存放人均捐款数
  - 定义符号常量**SUM**代表**100000**

```
#include <stdio.h>
#define SUM 100000
int main()
{  float amount,aver,total;   int i;
   for (i=1,total=0;i<=1000;i++)
   { printf("please enter amount:");
     scanf("%f",&amount);
     total= total+amount;
     if (total>=SUM) break;
   }
   aver=total / i ;
   printf("num=%d\naver=%10.2f\n"
                                ,i,aver);
   return 0;
}
```

指定符号常量SUM代表100000

```
#include <stdio.h>
#define SUM 100000
int main()
{  float amount,aver,total;   int i;
   for (i=1,total=0;i<=1000;i++)
   { printf("please enter amount:");
     scanf("%f",&amount);
     total= total+amount;
     if (total>=SUM) break;
   }
   aver=total / i ;
   printf("num=%d\naver=%10.2f\n"
                                ,i,aver);
   return 0;
}
```

应该执行1000次

```
#include <stdio.h>
#define SUM 100000
int main()
{  float amount,aver,total;   int i;
   for (i=1,total=0;i<=1000;i++)
   { printf("please enter amount:");
     scanf("%f",&amount);
     total= total+amount;
     if (total>=SUM) break;
   }
  aver=total / i ;
  printf("num=%d\naver=%10.2f\n"
                              ,i,aver);
  return 0;
}
```

达到10万，提前结束循环

```
#include <stdio.h>
#define SUM 100000
int main()
{  float amount,aver,t
   for (i=1,total=0;i<=
   { printf("please ente
     scanf("%f",&amount);
     total= total+amount;
     if (total>=SUM) break;
   }
   aver=total / i ;
   printf("num=%d\naver=%10.2f\n"
                          ,i,aver);
   return 0;
}
```

```
please enter amount:12000
please enter amount:24600
please enter amount:3200
please enter amount:5643
please enter amount:21900
please enter amount:12345
please enter amount:23000
num=7
aver=   14669.71
```

实际捐款人数

```
#include <stdio.h>
#define SUM 100000
int main()
{  float amount,aver,total;   int i;
   for (i=1,total=0;i<=1000;i++)
   { printf("please enter amount:");
     scanf("%f",&amount);
     total= total+amount;
     if (total>=SUM) break;
   }
   aver=total / i ;
   printf("num=%d \naver= %10.2f \n"
                                  ,i,aver);
   return 0;
}
```

只能用于循环语句和switch语句之中，而不能单独使用

5.7.2 用continue语句提前结束本次循环

- 有时并不希望终止整个循环的操作，而只希望提前结束本次循环，而接着执行下次循环。这时可以用**continue**语句

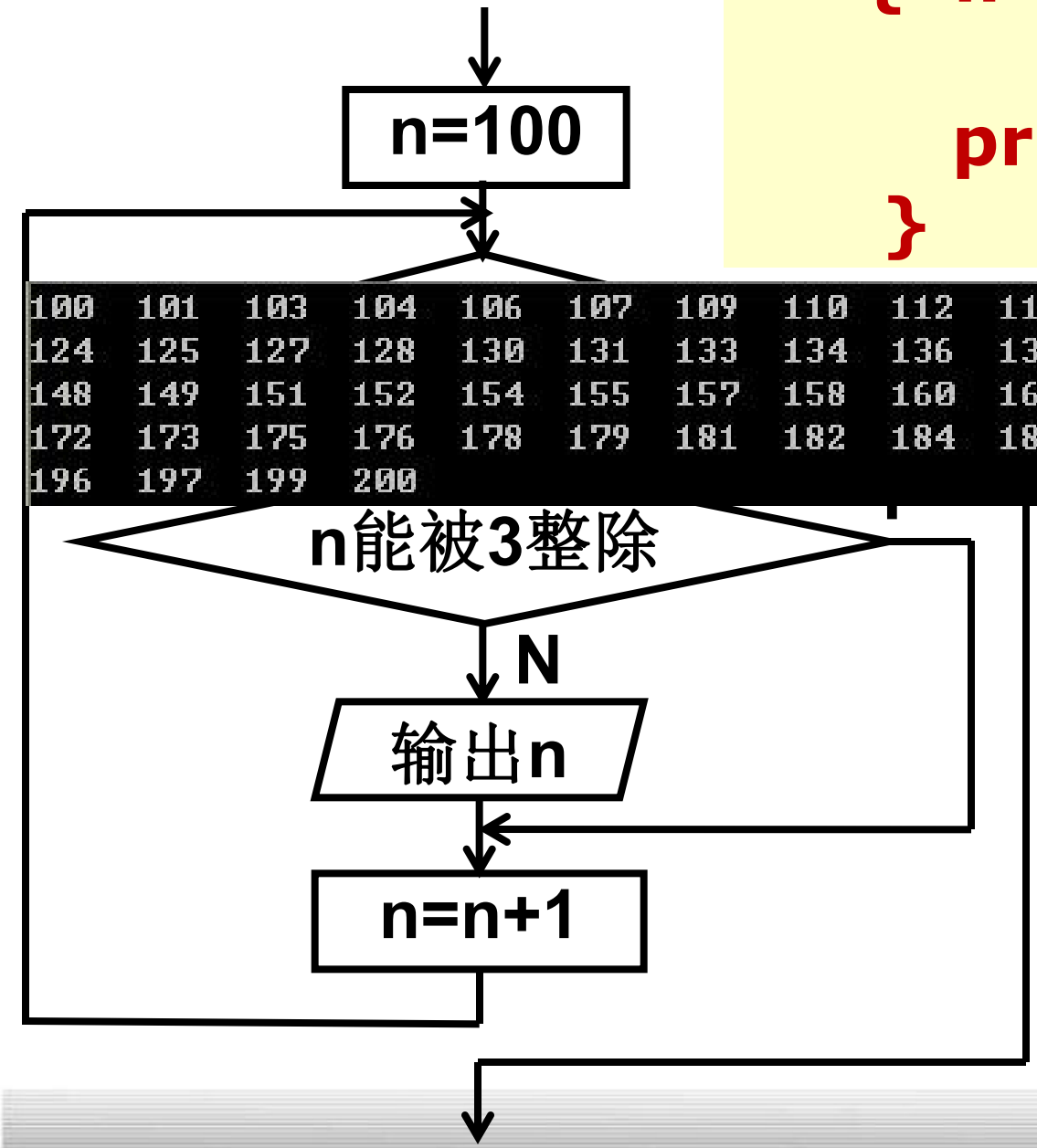
5.7.2 用continue语句提前结束本次循环

例5.5 要求输出100～200之间的不能被3整除的数。

- 编程思路：
  - 对100到200之间的每一个整数进行检查
  - 如果不能被3整除，输出，否则不输出
  - 无论是否输出此数，都要接着检查下一个数(直到200为止)。

```
for(n=100;n<=200;n++)
   {  if (n%3==0)
         continue;
      printf("%d  ",n);
   }
```

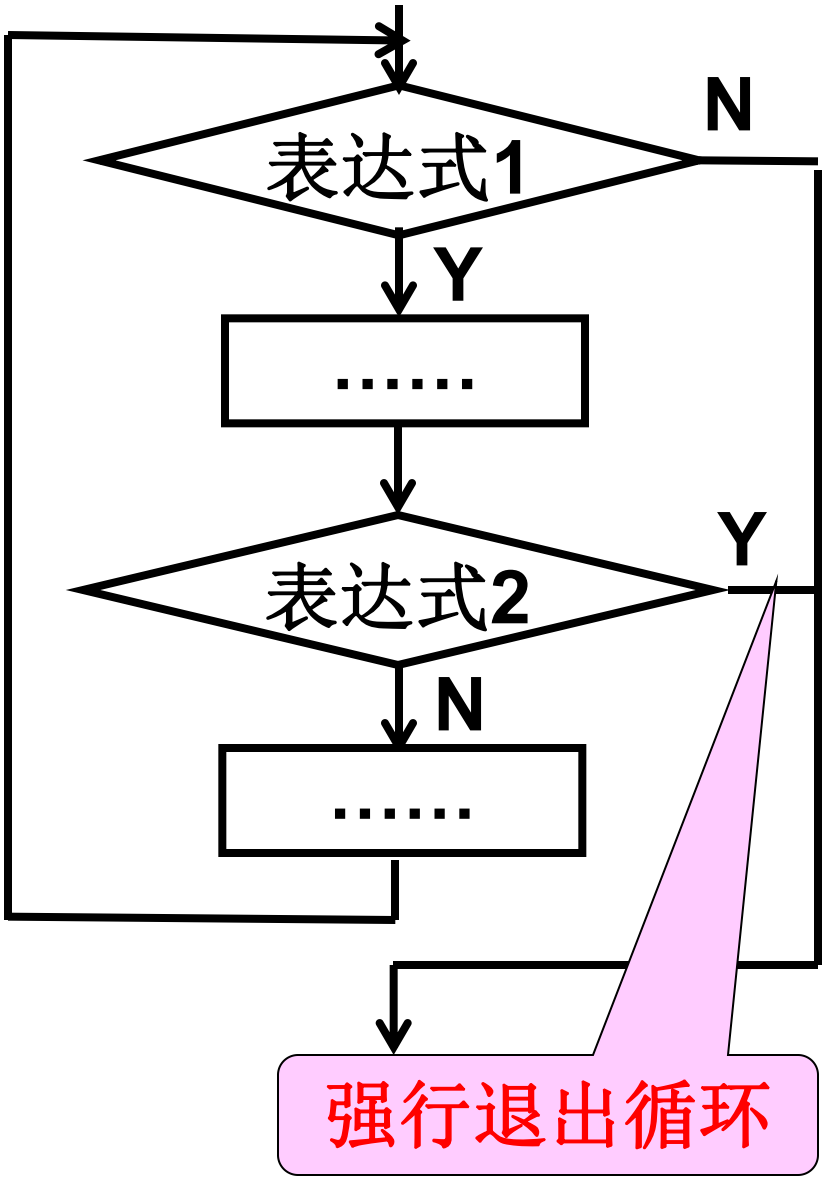
```
100  101  103  104  106  107  109  110  112  113  115  116  118  119  121  122
124  125  127  128  130  131  133  134  136  137  139  140  142  143  145  146
148  149  151  152  154  155  157  158  160  161  163  164  166  167  169  170
172  173  175  176  178  179  181  182  184  185  187  188  190  191  193  194
196  197  199  200
```



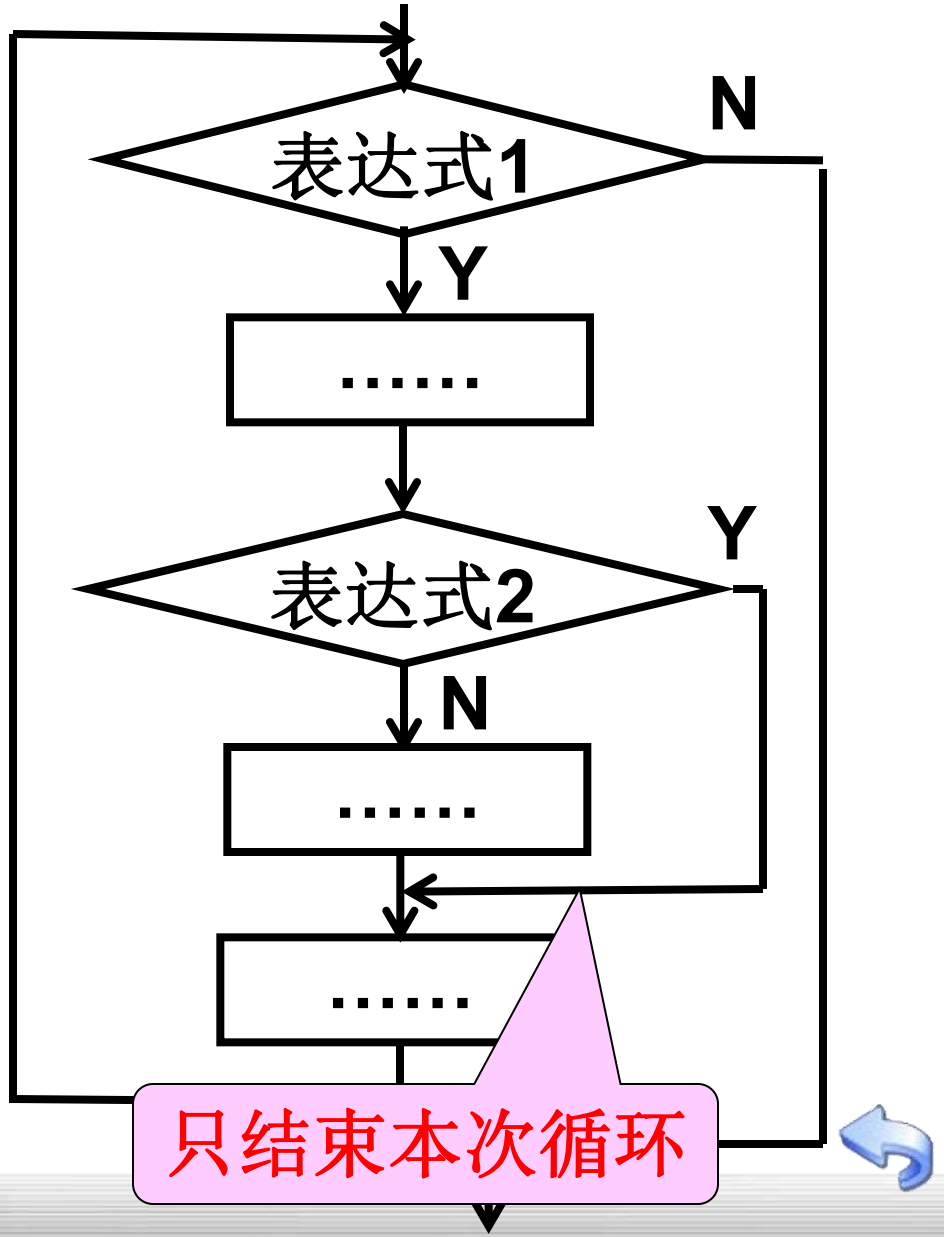
5.7.3 break语句和continue语句的区别

- **continue**语句只结束本次循环，而不是终止整个循环的执行
- **break**语句结束整个循环过程，不再判断执行循环的条件是否成立

break语句



continue语句



例5.6 输出以下4*5的矩阵。

1	2	3	4	5
2	4	6	8	10
3	6	9	12	15
4	8	12	16	20

- 解题思路：
  - 可以用循环的嵌套来处理此问题
  - 用外循环来输出一行数据
  - 用内循环来输出一列数据
  - 按矩阵的格式(每行5个数据)输出

```
#include <stdio.h>
int main()
{  int i,j,n=0;
   for (i=1;i<=4;i++)
      for (j=1;j<=5;j++,n++)
      { if (n%5==0) printf ("\n");
        printf ("%d\t",i*j);
      }
   printf("\n");
   return 0;
}
```

累计输出数据的个数

控制一行内输出5个数据

双重循环

```
#include <stdio.h>
int main()
{  int i,j,n=0;
   for (i=1;i<=4;i++)
      for (j=1;j<=5;j++,n++)
      {  if (n%5==0) printf (“\n”);
         printf ("%d\t",i*j);
      }
    printf("\n");
    return 0;
}
```

```
#include <stdio.h>
int main()
{  int i,j,n=0;
   for (i=1;i<=4;i++)
       for (j=1;j<=5;j++,n++)
       {  if (n%5==0) printf (“\n”);
          printf ("%d\t",i*j);
       }
    printf("\n");
    return 0;
}
```

控制输出4行

```
#include <stdio.h>
int main()
{  int i,j,n=0;
   for (i=1;i<=4;i++)
      for (j=1;j<=5;j++,n++)
      {  if (n%5==0) printf (“\n”);
         printf ("%d\t",i*j);
      }
    printf("\n");
    return 0;
}
```

控制每行中输出5个数据

```
#include <stdio.h>
int main()
{  int i,j,n=0;
   for (i=1;i<=4;i++)
      for (j=1;j<=5;j++,n++)
      {  if (n%5==0) printf (“\n”);
         printf ("%d\t",i*j);
      }
    printf("\n");
    return 0;
}
```

i=1时

j由1变到5

i*j的值是1,2,3,4,5

```
1    2    3    4    5
2    4    6    8    10
3    6    9    12   15
4    8    12   16   20
```

如何修改程序，不输出第一行的空行？

```
#includ
int mai
{  int i,j,n=0;
   for (i=1;i<=4;i++
      for (j=1;j<=5;j++,n++)
      {  if (n%5==0) printf ("\n");
         printf ("%d\t",i*j);
      }
   printf("\n");
   return 0;
}
```

j也由1变到5

i*j的值是2,4,6,8,10

```
1    2    3    4    5
2    4    6    8    10

4    8    12   16   20
```

```
#include <
int main()
{  int i,j,n=0;
   for (i=1;i<=4;i++)
      for (j=1;j<=5;j++,n++)
      {  if (n%5==0) printf ("\n");
         if (i==3 && j==1) break;
         printf ("%d\t",i*j);
      }
   printf("\n");
   return 0;
}
```

遇到第3行第1列，终止内循环

```
1    2    3    4    5
2    4    6    8    10
6    9    12   15
4    8    12   16   20
```

```
#include
int mai
    0;
    i<=4;i++)
    for (j=1;j<=5;j++,n++)
    {  if (n%5==0) printf ("\n");
       if (i==3 && j==1) continue;
       printf ("%d\t",i*j);
    }
  printf("\n");
  return 0;
}
```

原来第3行第1个数据3没有输出

5.8循环程序举例

例5.7用 $\frac{\pi}{4} \approx 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \cdots$ 公式求 π 的近似值，直到发现某一项的绝对值小于 10^{-6} 为止(该项不累计加)。

5.8循环程序举例

- 解题思路：
  - 求π近似值的方法很多，本题是一种
  - 其他方法：

$$\pi \approx \frac{22}{7}$$

$$\frac{\pi^2}{6} \approx \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \cdots + \frac{1}{n^2}$$

$$\frac{\pi}{2} = \frac{2\times 2}{1\times 3} \times \frac{4\times 4}{3\times 5} \times \frac{6\times 6}{5\times 7} \times \cdots \times \frac{(n-1)^2}{n\times(n+2)}$$

5.8循环程序举例

$$\frac{\pi}{4} \approx 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \cdots$$

- 每项的分子都是1
- 后一项的分母是前一项的分母加2
- 第1项的符号为正，从第2项起，每一项的符号与前一项的符号相反

$$\frac{1}{n} \Rightarrow -\frac{1}{n+2}$$

5.8循环程序举例

sign=1,pi=0,n=1,term=1
当term ≥10^{-6}
pi=pi+term
n=n+1
sing=-sign
term=sign/n
pi=pi*4
输出pi

```
#include <stdio.h>
#include <math.h>
int main()
{  int sign=1; double pi=0,n=1,term=1;
   while(fabs(term)>=1e-6)
   {  pi=pi+term;
      n=n+2;
      sign=-sign;
      term=sign/n;
    }
   pi=pi*4;
   printf("pi=%10.8f\n",pi);
   return 0;
}
```

求绝对值的函数

```
pi=3.14159065
```

只保证前5位小数是准确的

```
#include <stdio.h>
#include <math.h>
int main()
{  int sign=1; double pi=0,n=1,term=1;
   while(fabs(term)>=1e-6)
   {  pi=pi+term;
      n=n+2;
      sign=-sign;
      term=sign/n;
    }
   pi=pi*4;
   printf("pi=%10.8f\n",pi);
   return 0;
}
```

改为1e-8

```
pi=3.14159065
```

```
pi=3.14159263
```

例5.8 求费波那西(Fibonacci)数列的前40个数。这个数列有如下特点：第1、2两个数为1、1。从第3个数开始，该数是其前面两个数之和。即：

$$\begin{cases} F_1 = 1 & (n = 1) \\ F_2 = 1 & (n = 2) \\ F_n = F_{n-1} + F_{n-2} & (n \geq 3) \end{cases}$$

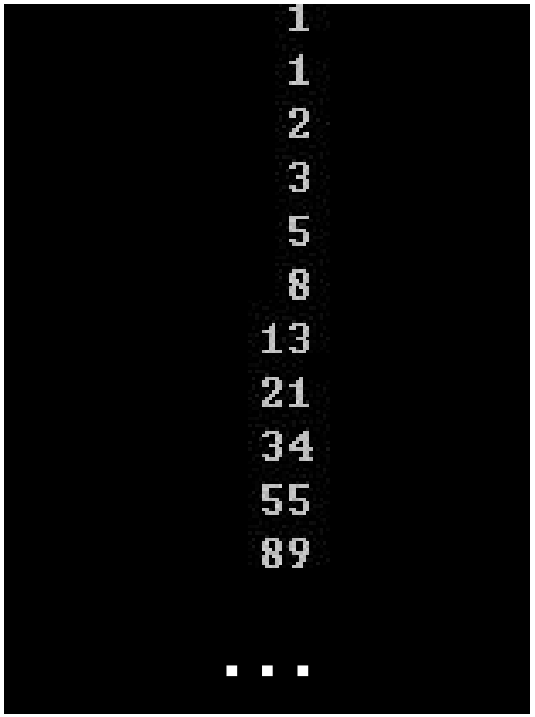
- 这是一个有趣的古典数学问题：
  - 有一对兔子，从出生后第3个月起每个月都生一对兔子。
  - 小兔子长到第3个月后每个月又生一对兔子。
  - 假设所有兔子都不死，问每个月的兔子总数为多少？

第几个月	小兔子对数	中兔子对数	老兔子对数	兔子总数
1	1	0	0	1
2	0	1	0	1
3	1	0	1	2
4	1	1	1	3
5	2	1	2	5
6	3	2	3	8
7	5	3	5	13
⋮	⋮	⋮	⋮	⋮

```
f1=1,f2=1
输出f1,f2
For i=1 to 38
    f3=f1+f2
    输出f3
    f1=f2
    f2=f3
```

```
#include <stdio.h>
int main()
 { int f1=1,f2=1,f3;  int i;
   printf("%12d\n%12d\n",f1,f2);
   for(i=1; i<=38; i++)
   {  f3=f1+f2;
      printf("%12d\n",f3);
      f1=f2;
      f2=f3;
   }
   return 0;
}
```

代码可改进

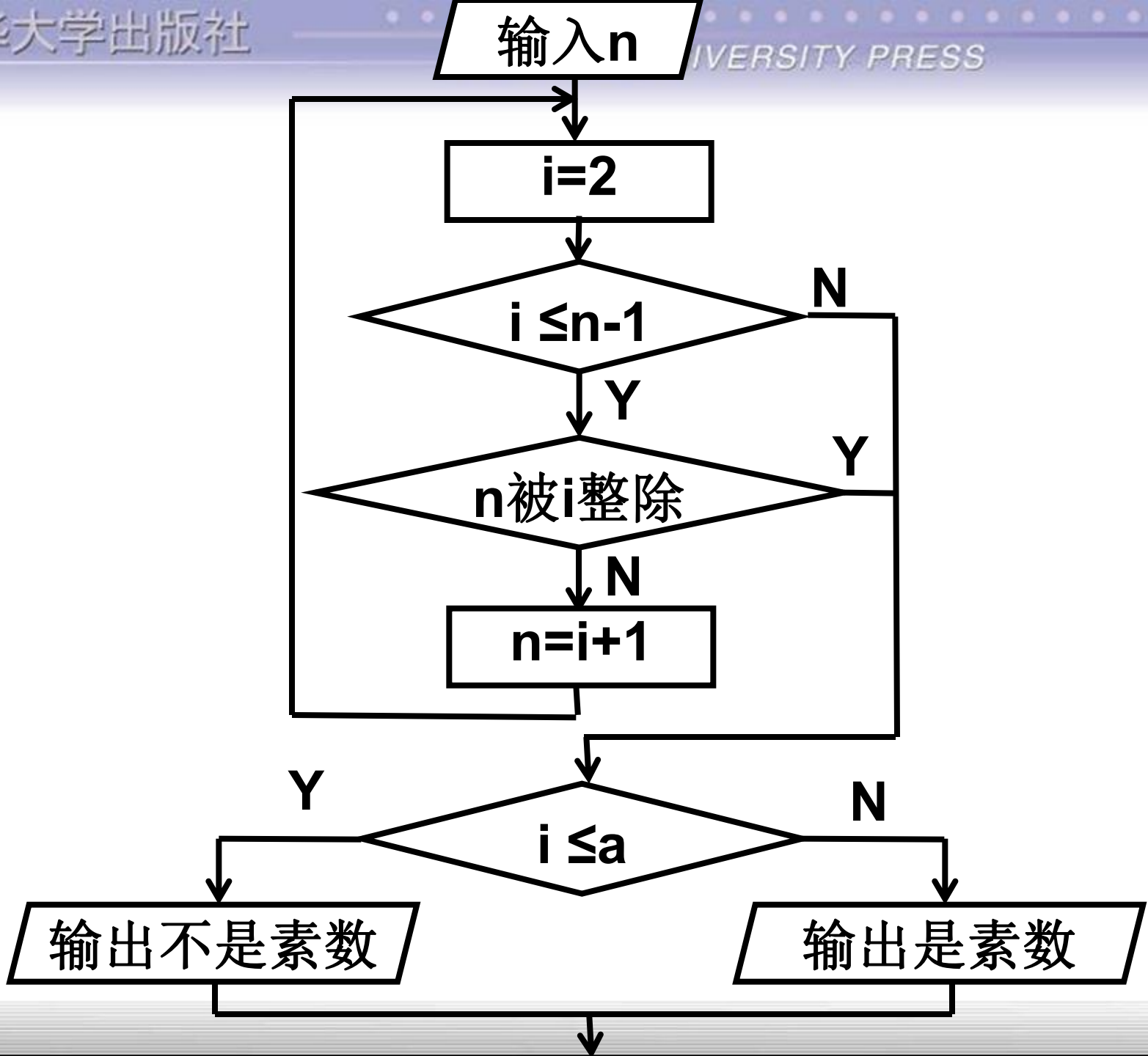


```
#include <stdio.h>
int main()
 {  int f1=1,f2=1;  int i;
    for(i=1; i<=20; i++)
   { printf("%12d %12d ",f1,f2);
     if(i%2==0) printf("\n");
```

```
           1            1            2            3
           5            8           13           21
          34           55           89          144
         233          377          610          987
        1597         2584         4181         6765
       10946        17711        28657        46368
       75025       121393       196418       317811
      514229       832040      1346269      2178309
     3524578      5702887      9227465     14930352
    24157817     39088169     63245986    102334155
```

例5.9输入一个大于3的整数n，判定它是否素数(prime，又称质数)。

- 解题思路：
  - 让n被i整除(i的值从2变到n-1)
  - 如果n能被2～(n-1)之中任何一个整数整除，则表示n肯定不是素数，不必再继续被后面的整数除，因此，可以提前结束循环
  - 注意：此时i的值必然小于n


输入n
i=2
i ≤n-1
N
Y
n被i整除
Y
N
n=i+1
Y
i ≤a
N
输出不是素数
输出是素数

```
#include <stdio.h>
int main()
 { int n,i;
   printf("n=?");  scanf("%d",&n);
   for (i=2;i<=n-1;i++)
       if(n%i==0) break;
   if(i<n) printf("%d is not\n",n);
   else printf("%d is\n",n);
   return 0;
}
```

```
n=?17
17 is
```

```
n=?327
327 is not
```

```
#include <stdio.h>
int main()
 { int n,i;
   printf("n=?");  scanf("%d",&n);
   for (i=2;i<=n-1;i++)
      if(n%i==0) break;
   if(i<n) printf("%d is not\n",n);
   else printf("%d is\n",n);
   return 0;
}
```

$\sqrt{n}$

k=sqrt(n);

```
#include <stdio.h>
int main()
 { int n,i,k;
   printf("n=?");  scanf("%d",&n);
   for (i=2; i<=k; i++)
      if(n%i==0) break;
   if(i<n) printf("%d is not\n",n);
   else printf("%d is\n",n);
   return 0;
}
```

#include <math.h>

k=sqrt(n);

```
#include <stdio.h>
int main()
 { int n,i,k;
   printf("n=?");  scanf("%d",&n);
   for (i=2; i<=k; i++)
       if(n%i==0) break;
   if(i<=k) printf("%d is not\n",n);
   else printf("%d is\n",n);
   return 0;
}
```

```
#include <math.h>
k=sqrt(n);
```

例5.10 求100～200间的全部素数。

- 解题思路：
  - 使用例5.9的算法
  - 在例5.9程序中只要增加一个外循环，先后对100～200间的全部整数一一进行判定即可

```
……
for(n=101;n<=200;n=n+2)
{ k=sqrt(n);
  for (i=2;i<=k;i++)
      if (n%i==0) break;
  if (i>=k+1)
  {  printf("%d ",n);
     m=m+1;
  }
  if(m%10==0) printf("\n");
}
……
```

只对奇数进行检查

控制每行输出10个数据

例5.11 译密码。为使电文保密，往往按一定规律将其转换成密码，收报人再按约定的规律将其译回原文。

- 非字母字符保持原状不变
- 输入一行字符，要求输出其相应的密码

- 解题思路：问题的关键有两个：

(1) 决定哪些字符不需要改变，哪些字符需要改变，如果需要改变，c=getchar();

◆处理的方法是：输入一个字符给字符变量c，先判定它是否字母(包括大小写)，若不是字母，不改变c的值；若是字母，则还要检查它是否'W'到'Z'的范围内(包括大小写字母)。如不在此范围内，则使变量c的值改变为其后第4个字母。如果在'W'到'Z'的范围内，则应将它转换为A～D(或a～d)之一的字母。

- 解题思路：问题的关键有两个：

(1) 决定哪些字符不需要改变，哪些字符需要改变，如果需要改变，应改为哪个字符

```
if((c>='a' && c<='z') || (c>='A' && c<='Z'))
```

先判定它是否字母(包括大小写)，若不是字母，不改变c的值；若是字母，则还要检查它是否'W'到'Z'的范围内(包括大小写字母)。如不在此范围内，则使变量c的值改变为其后第4个字母。如果在'W'到'Z'的范围内，则应将它转换为A～D(或a～d)之一的字母。

➢解题思路：问题的关键有两个：

（1）决定哪些字符不需要改变，哪些字符需要改变，如果需要改变，应改为哪个字符

```
if(c>='W' && c<='Z' || c>='w' && c<='z')
    c=c+4-26;
else  c=c+4;
```

否'W'到'Z'的范围内(包括大小写字母)。如不在此范围内，则使变量c的值改变为其后第4个字母。如果在'W'到'Z'的范围内，则应将它转换为A～D(或a～d)之一的字母。

- 解题思路：问题的关键有两个：

  (2) 怎样使c改变为所指定的字母？

  - 办法是改变它的ASCII值
  - 例如字符变量c的原值是大写字母'A'，想使c的值改变为'E'，只需执行“c=c+4”即可，因为'A'的ASCII值为65，而'E' 的ASCII值为69，二者相差4

```
char c;
c=getchar();
while(c!='\n')
{ if((c>='a' && c<='z') || (c>='A' &&
                                c<='Z'))
    { if(c>='W' && c<='Z' || c>='w' &&
                                  c<='z')
        c=c-22;
      else  c=c+4;
     }
    printf("%c",c);
    c=getchar();
}
```

可以改进程序

```
China!
Glmre!
```

```
char c;
while((c=getchar())!='\n')
{ if((c>='A' && c<='Z') || (c>='a' &&
                                c<='z'))
   { c=c+4;
     if(c>='Z' && c<='Z'+4 || c>'z')
          c=c-26;
    }
    printf("%c",c);
}
```

不能少