西北工业大学
NORTHWESTERN POLYTECHNICAL UNIVERSITY

# C程序设计

# Programming in C

**1011014**

主讲：姜学锋，计算机学院

# 编程实现循环

- 3、循环不变式
- 4、do和for循环

## 3.5.1 while语句

- 循环不变式(loop invariant)性质
  - ①循环初始化的时候（即第一次循环之前），A[1..j-1]的“有序性”是成立的；
  - ②在循环的每次迭代过程中，A[1..j-1]的“有序性”仍然保持；
  - ③循环结束的时候，A[1..j-1]的“有序性”仍然成立。

## 3.5.1 while语句

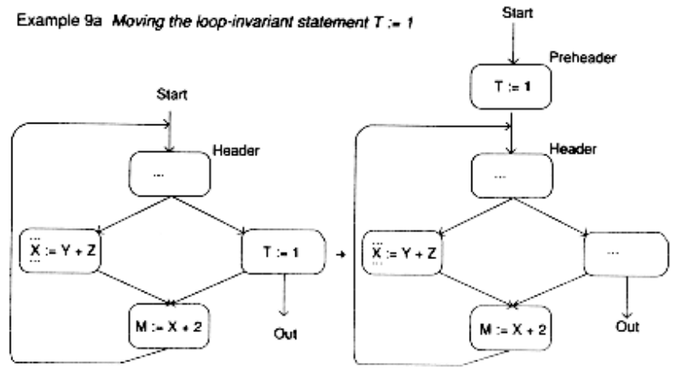
- 循环不变式(loop invariant)
  - 编程中的“数学归纳法”
  - 利用循环不变式来证明循环的正确性与用数学归纳法(induction)证明数学等式的相同点在于：需要验证初值，或初始状态是否满足条件。之后再证明在归纳或递推的过程中仍然满足这种条件。（这个条件在数学归纳中叫做递推关系，在循环中就是循环不变式）。
  - 循环不变式(loop invariant)与数学归纳法(induction)的区别在于：数学归纳可能是无限的，是无限地递推，但循环不变式所要证明的循环是要结束并给出正确结果的。

## 3.5.1 while语句

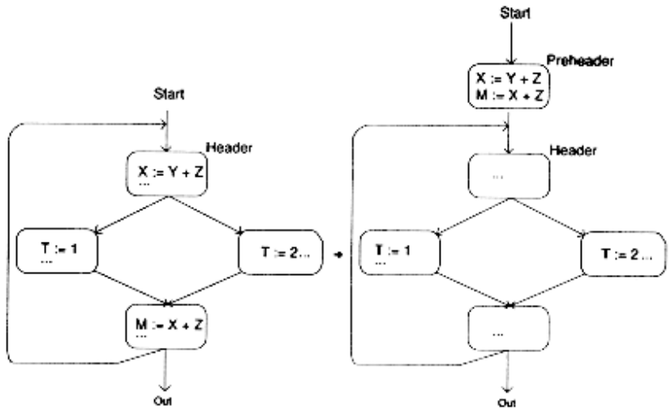
- 循环不变式(loop invariant)
- 编程中的“数学归纳法”

**EXAMPLE 9** Loop-invariant code motion

Example 9a *Moving the loop-invariant statement T := 1*




Example 9b *Moving the loop-invariant statement M := X + Z and X := Y + Z*




This example shows the loop-invariant statements of the preceding examples moved to a preheader in the right graphs.

## 3.5.2 do语句

- do语句
- do语句的作用是先执行语句，然后计算给定的表达式，根据结果判定是否循环执行，语句形式为：

```
do 语句 while (表达式);
```

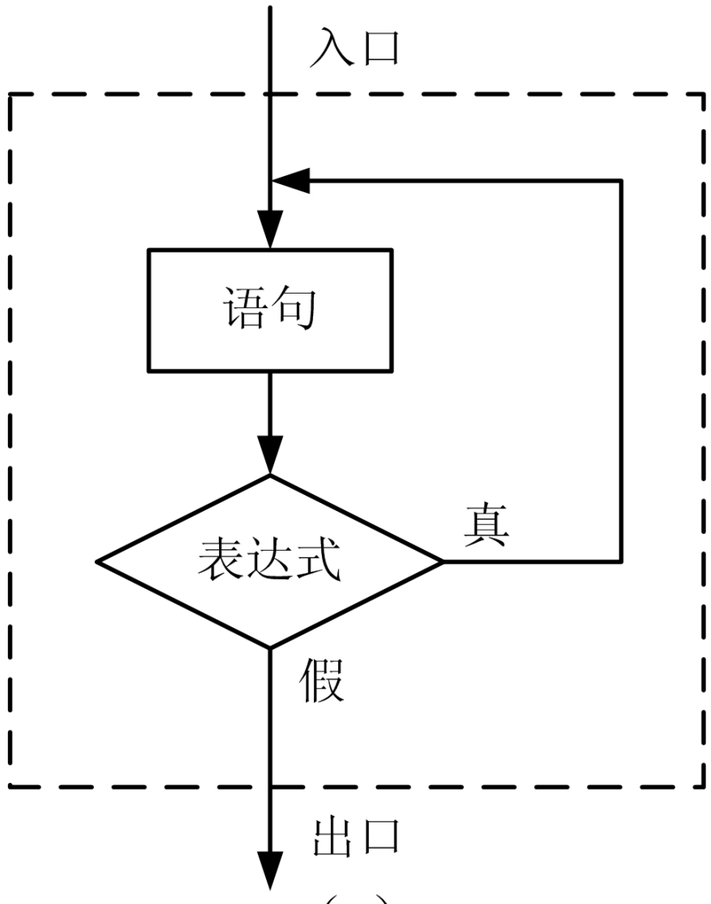
- 其中的语句称为子语句，又称循环体，圆括号内的表达式称为循环条件。

## 3.5.2 do语句

- do语句的执行过程是：
- ①执行子语句；
- ②计算表达式，无论表达式为何种类型均将这个值按逻辑值处理；
- ③如果值为真，则再次执行①；如果值为假，则do语句结束，执行后续语句。

## 3.5.2 do语句

图3.9 do-while语句执行流程




（a）

## 3.5.2 do语句

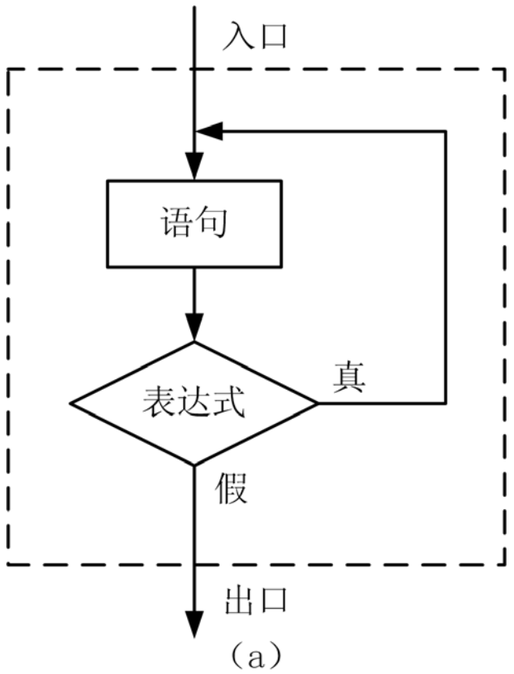
- do语句的说明。
- （1）do语句与while语句的语法和含义类似。
- （2）do语句的最后必须用分号（；）作为语句结束，循环体的复合语句形式为

```
do {
     ..... //复合语句
} while (表达式);
```

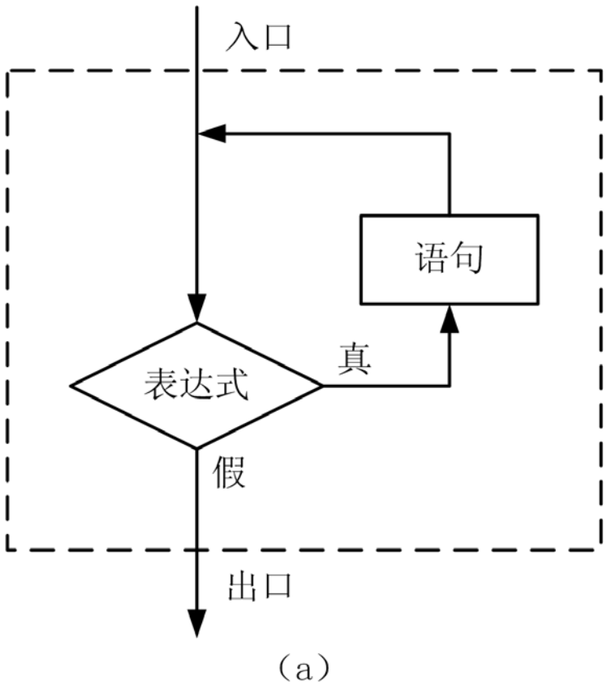
## 3.5.2 do语句

- （3）do语句先执行后判定，while语句则是先判定后执行；do语句至少要执行循环体一次，而while语句可能一次也不执行。

do语句

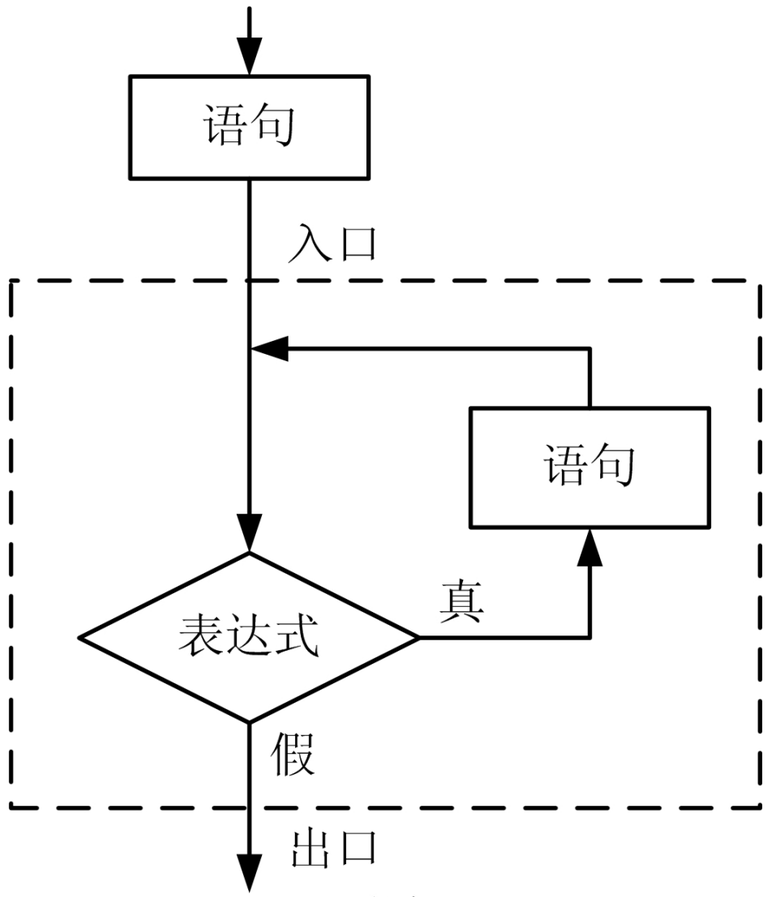



while语句



## 3.5.2 do语句

- （4）do语句结构和while语句结构是可以相互替换的。如图就是用while语句表示do语句的流程图，虚线框内为while语句结构。通常情况下，while语句比do语句用得多，而do语句使用的情形似乎就是如图的while语句结构。



（b）

## 3.5.2 do语句

【例3.9】

连续输入多个数据，计算它们乘积，当输入0时结束。

## 3.5.2 do语句

### 例题分析

问题是要先输入，才判断是否中止循环的执行，使用do循环正好满足这样的需要，它保证循环体至少执行一次。

## 3.5.2 do语句

例3.9

```
1 #include <stdio.h>
2 int main()
3 {
4   int n=1,k=1;
5   do {
6     k=k*n;
7     scanf("%d",&n);
8   } while (n!=0); //输入0时结束循环
9   printf("%d\n",k); //输出乘积
10  return 0;
11 }
```

## 3.5.2 do语句

例3.9

```
1 #include <stdio.h>
2 int main()
3 {
4   int n=1,k=1;
5   do {
6     k=k*n;
7     scanf("%d",&n);
8   } while (n!=0); //输入0时结束循环
9   printf("%d\n",k); //输出乘积
10  return 0;
11 }
```

## 3.5.3 for语句

- for语句
- for语句的作用是计算给定的表达式，根据结果判定循环执行语句，for语句有循环初始和循环控制功能，语句形式为：

```
for (表达式1; 表达式2; 表达式3) 语句;
```

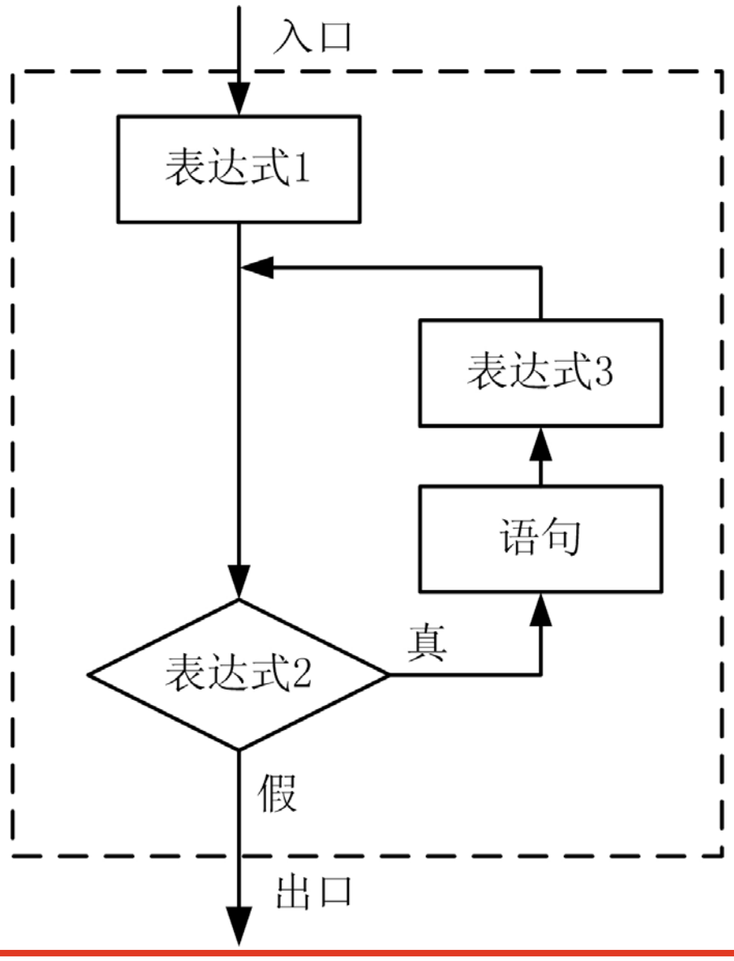
- 其中的语句称为子语句，又称循环体，圆括号内的表达式2称为循环条件。

## 3.5.3 for语句

- for语句的执行过程是：
- ①计算表达式1；
- ②计算表达式2，无论表达式2为何种类型均将这个值按逻辑值处理；
- ③如果值为真，则执行循环体，然后计算表达式3，再重复②；
- ④如果值为假，则for语句结束，执行后续语句。

## 3.5.3 for语句

图3.10 for语句执行流程



## 3.5.3 for语句

- 前面提到，循环结构应有循环初始、循环条件和循环控制三要素，结合for语句的流程图可以得出for语句的应用格式如下：

```
for (循环初始; 循环条件; 循环控制) 循环体;
```

## 3.5.3 for语句

- for语句的表达式1、表达式3允许逗号表达式，除非涉及复杂的循环初始计算和循环控制计算，非得用语句实现不可。循环三要素完全可以构造在一个for语句中。
- 示例

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

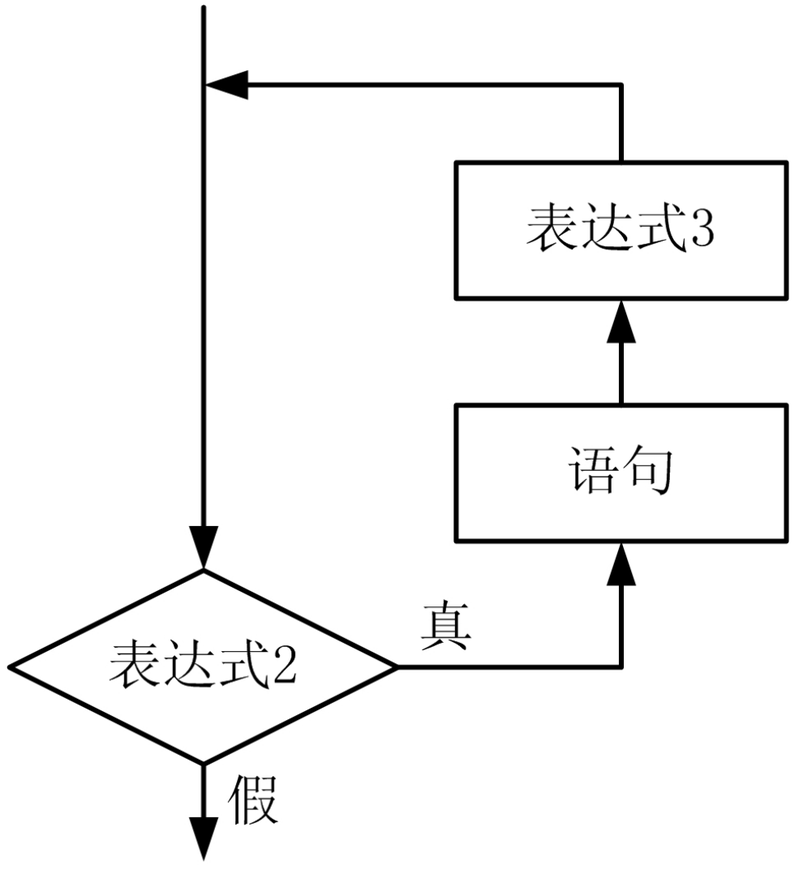
## 3.5.3 for语句

- for语句的说明：
- （1）for语句与while语句的语法和含义类似。

- （2）for语句内中三个表达式，用分号（；）作为间隔，不能把这里的表达式和分号理解为语句。三个表达式均可以省略，但中间的分号不能省略，下面分情形讨论。

## 3.5.3 for语句

- 情形一。省略表达式1。
- 省略表达式1就相当于将循环初始计算省略了，此时应在for语句之前有循环初始，如while语句那样。其执行流程如图（a）所示。




（a）

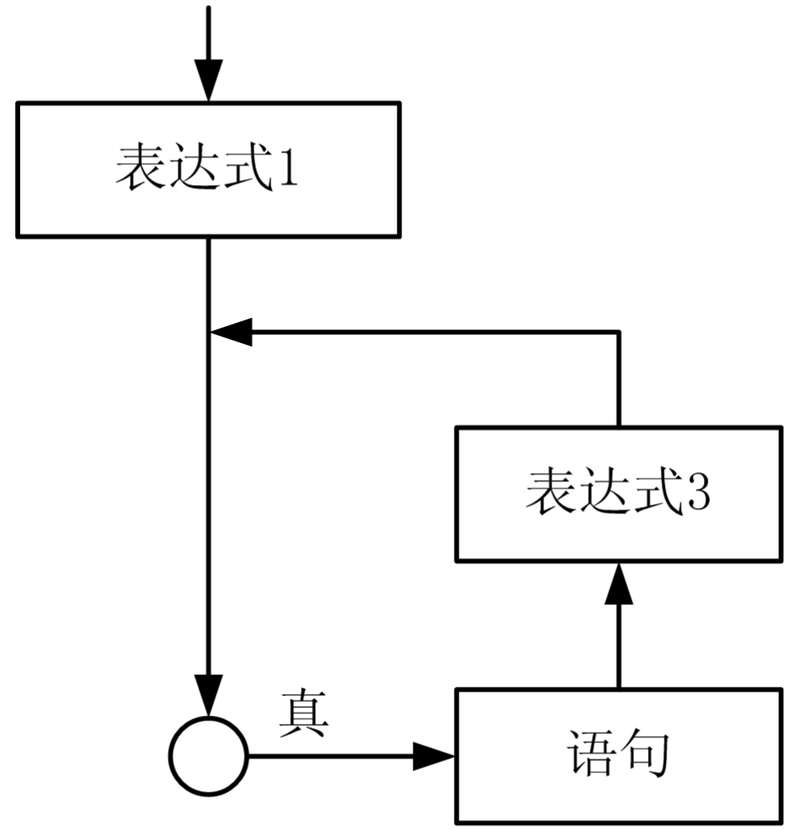
## 3.5.3 for语句

- 示例

```
n=1, sum=0;
for ( ; n<=100 ; n++) sum=sum+n; //累加和
```

## 3.5.3 for语句

- 情形二，省略表达式2。
- C语言规定省略表达式2，则循环条件始终为真，循环永远执行下去。其执行流程如图（b）所示。




（b）

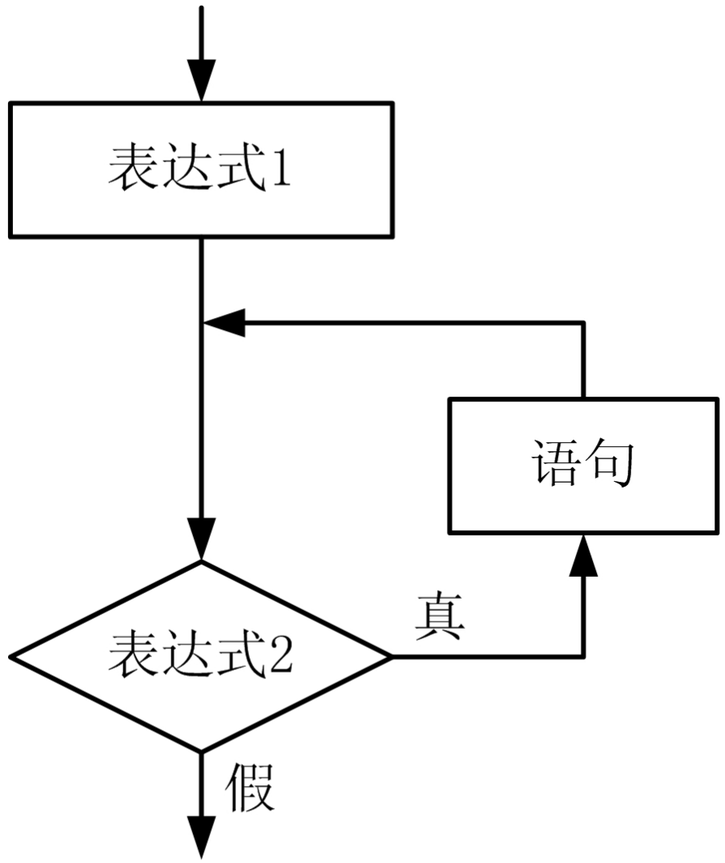
## 3.5.3 for语句

- 示例

```
for (n=1,sum=0 ;  ; n++) sum=sum+n; //无限循环
```

## 3.5.3 for语句

- 情形三，省略表达式3。
- 省略表达式3相当于将循环控制计算省略了，此时应在for语句的循环体里有循环控制，如while语句那样。其执行流程如图（c）所示。




（c）

## 3.5.3 for语句

- 示例

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

## 3.5.3 for语句

- 情形四，省略表达式1和表达式3。
  - 此时，for语句只有充当循环条件的表达式2，完全等同于while语句，由此可见for语句比while语句功能强。
- 示例

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

```
n=1, sum=0;
while (n<=100) { sum=sum+n; n=n+1; }
```

## 3.5.3 for语句

- 情形五，三个表达式全部省略。这是一种在少数环境下应用的极端写法。
- 示例

```
for (   ;   ;   ) 循环体
```

## 3.5.3 for语句

- 几种循环的比较
- 1.共同点
- （1）while、do-while、for循环在逻辑上是相同的，一般情况下可以相互替换。
- （2）作为循环结构，while、do-while、for循环一般地都需要设置循环的初始值、循环结束条件的判定、循环增量（或减量）的计算。
- （3）while、do-while、for循环都可以用break语句来结束循环，用continue语句来结束一次循环。

## 3.5.3 for语句

- 2.不同点
- while、do-while、for循环的不同主要是设置循环的初始值、循环结束条件的判定、循环增量（或减量）的计算的程序顺序不相同。

| | while | do-while | for |
|---|---|---|---|
| 设置循环初始值： | 语句前 | 语句前 | 语句前或表达式1 |
| 结束条件的判定： | 先判后执行 | 先执行后判 | 先判后执行 |
| 循环控制的计算： | 循环体内 | 循环体内 | 循环体内或表达式3 |

## 3.5.3 for语句

- （1）do-while循环至少执行循环体一次。而while循环、for循环可能一次也不执行。
- （2）for循环的循环处理功能最强。while、do-while均可用for循环。


程序设计