

# 课程引入

例：1+2+3 该如何实现

定义变量sum=0

sum+=1

sum+=2

sum+=3

输出sum

$$\text{又例： } y = \sum_{n=1}^{100} n$$

定义变量sum=0, i=1

sum+=i

i++

i<=100

Y

N

结束

# 第五章 循环结构程序设计

5.1	while语句
5.2	do-while语句
5.3	for语句
5.4	break语句和continue语句
5.5	程序举例

# 5.1 while 语句

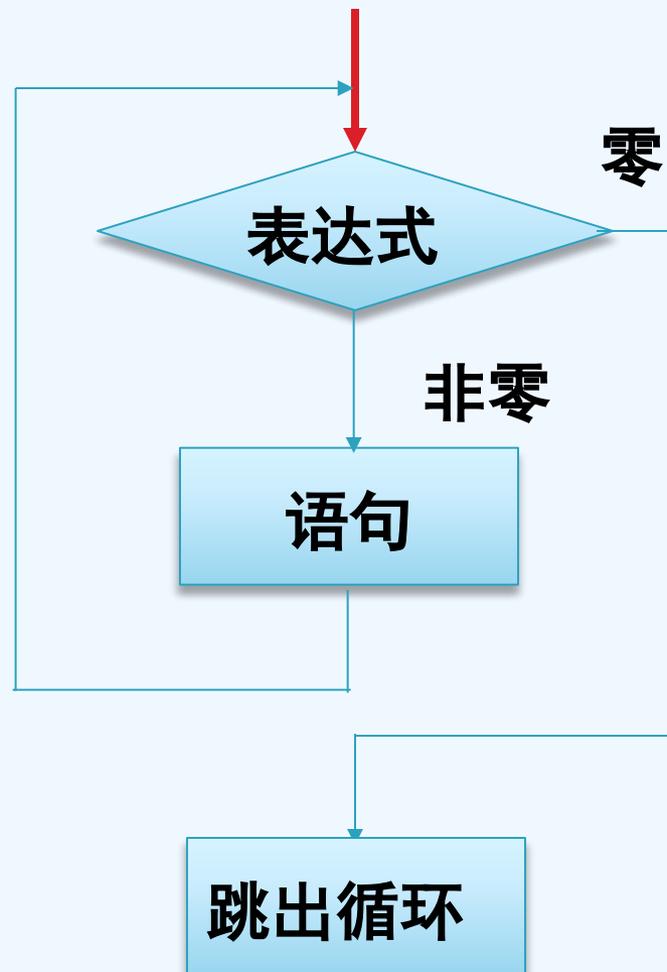
## 1. while 循环语句的形式

**while (表达式)  
循环语句**

此处无 ;

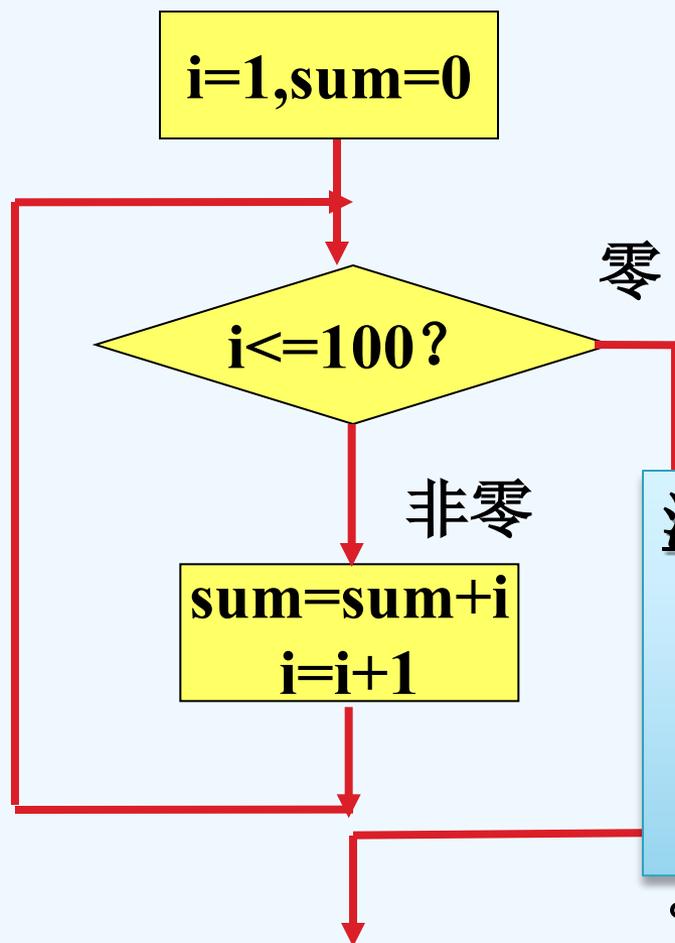
## 2. 执行过程:

先判断条件，后执行语句



# 5.1 while 语句

例1: 求  $\sum_{n=1}^{100} n$



```
main()
{
    int i=1,sum=0;
    while (i<=100)
    {
        sum=sum+i;
        i++;
    }
    printf("%d\n",sum);
}
```

**注意:** (1) 循环体中必须有使循环趋向于结束的语句  
累加和变量需置初值0  
累乘积变量需置初值1  
语句, 复合语句  
有使循环趋向于结束的语句

## 5.2 do-while 语句

### 1. do-while循环语句的形式

do

循环体

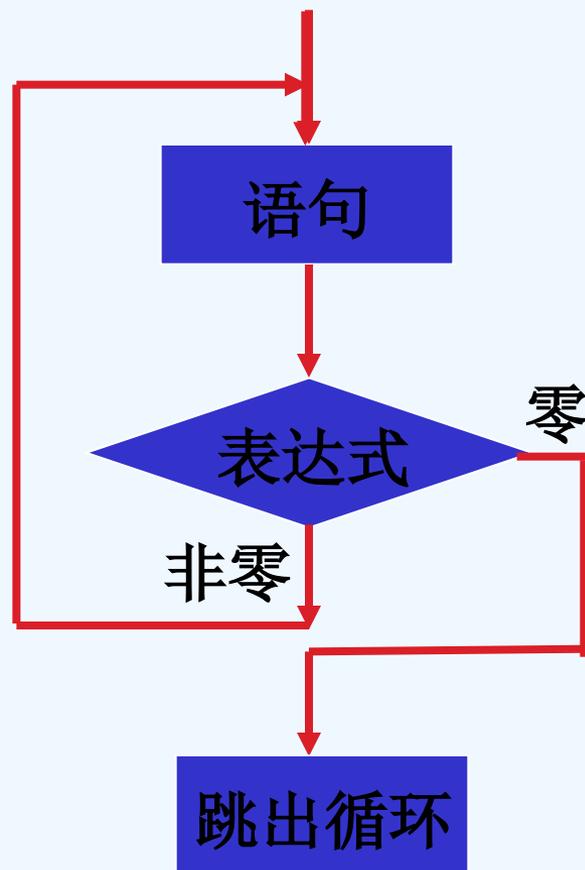
while(条件表达式) ;

此处有

;

先执行语句，后判断条件

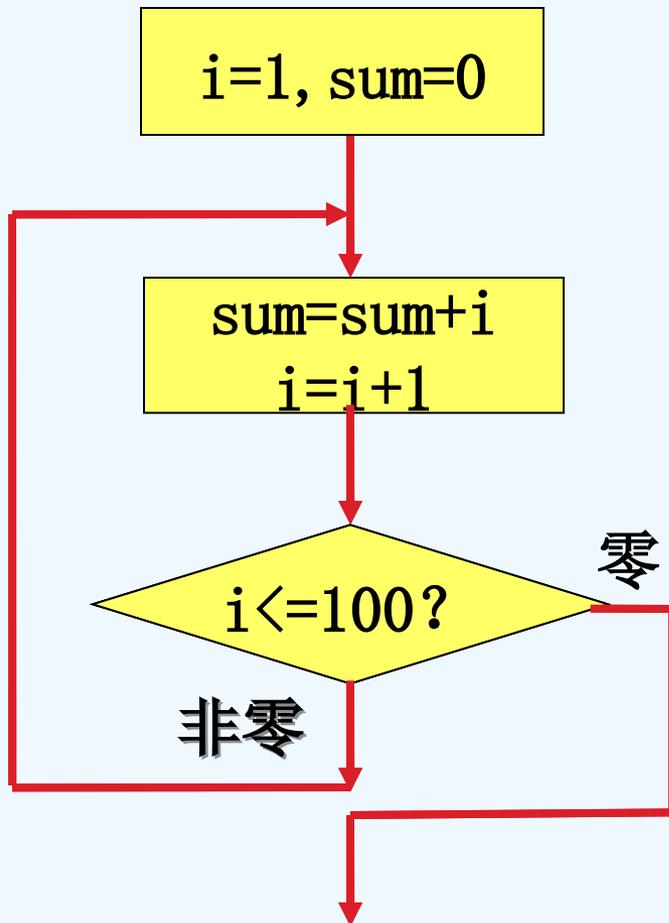
### 2. 执行过程



## 5.2 do-while 语句

例2: 用do-while型循环求:

$$\sum_{n=1}^{100} n$$



```
main()
{
    int i=1, sum=0;
    do
    { sum=sum+i;
      i++;
    }
    while (i<=100);
    printf("%d",sum);
}
```

## 5.2 do-while 语句

【例5.3】do-while循环的比较。

```
#include <stdio.h>
void main()
{
    int sum = 0;
    scanf("%d",&i);
    while(i <= 10)
    {
        sum = sum + i;
        i++;
    }
    printf("sum=%d\n",sum);
}
```

运行结果:

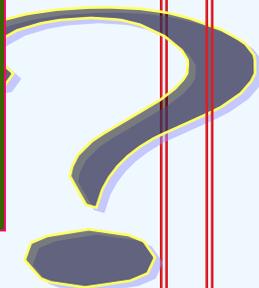
1✓

sum=55

再运行一次:

11✓

sum=0



```
#include <stdio.h>
void main()
{
    int sum = 0, i;
    scanf("%d",&i);
    do
    {
        sum = sum + i;
        i++;
    } while(i <= 10);
    printf("sum=%d\n",sum);
}
```

运行结果:

1✓

sum=55

再运行一次:

11✓

sum=11

结论:

(1) do—while循环语句首先执行循环体，然后计算表达式并检查循环条件，所以循环体至少执行一次。

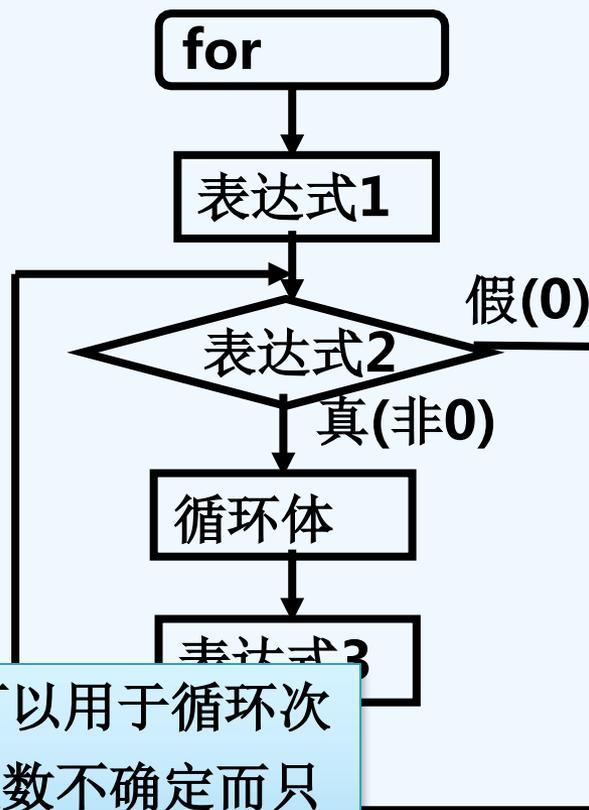
(2) 退出do—while循环的条件与退出while循环的条件相同

## 5.3 for 语句

### ◆ 一般格式:

```
for( 表达式1; 表达式2; 表达式3)  
{  
    循环体语句  
}
```

### ◆ 执行流程:

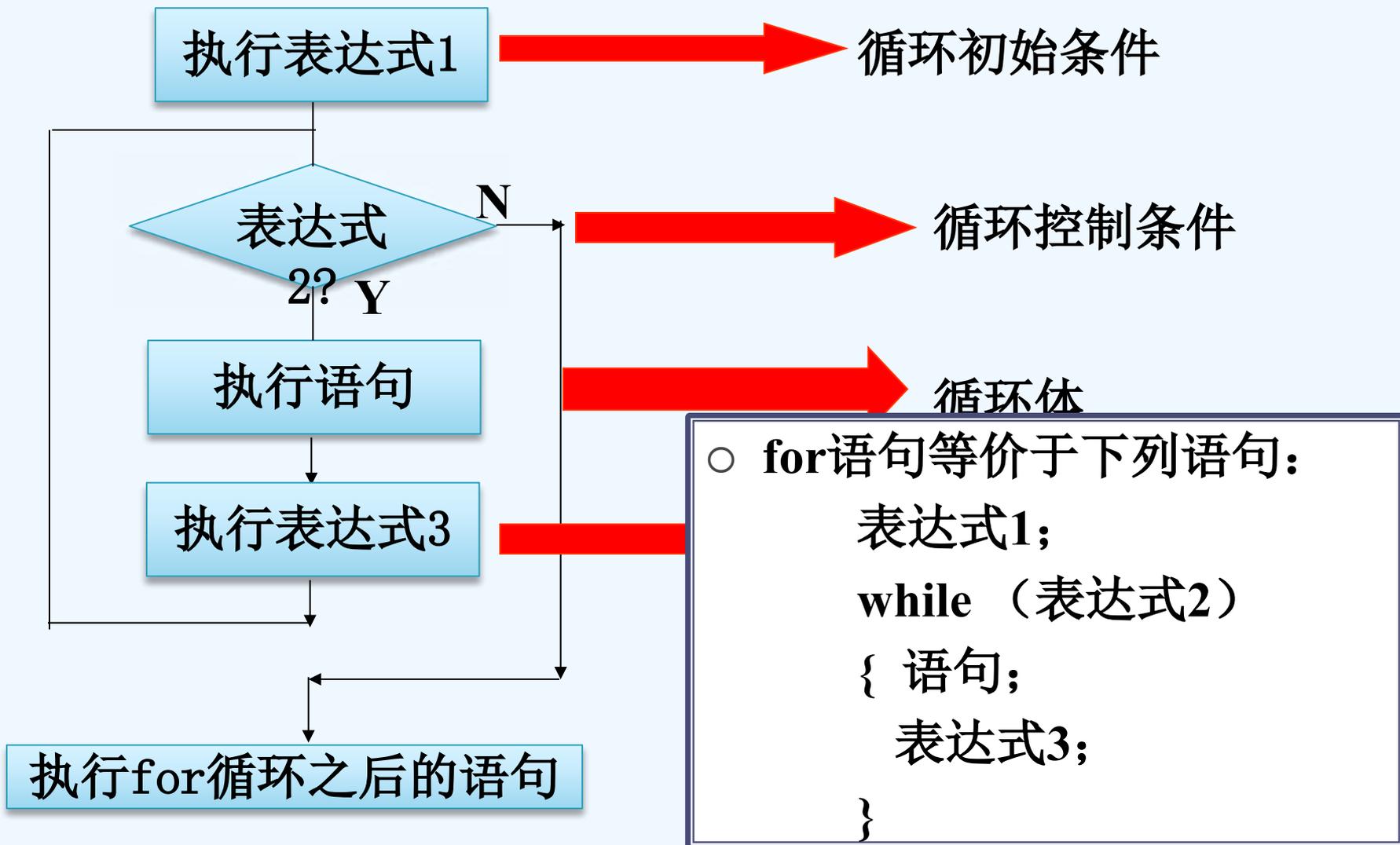


C语言中的for语句使用最为灵活，不仅可以用于循环次数已经确定的情况，而且可以用于循环次数不确定而只给出循环结束条件的情况，它完全可以代替while语句。

大家有疑问的，可以询问和交流

可以互相讨论下，但要小声点

## 5.3 for 语句



## 5.3 for 语句

for语句最简单的形式:

```
for(循环变量赋初值; 循环条件; 循环变量增值)
{   循环体   }
```

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

相当于:

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

## 5.3 循环结构——总结

用上述三种循环语句求  $s = \prod_{n=1}^{100} n$

**while**语句:

```
n = 1; s = 1;
while (n <= 100)
{ s = s * n;
  n++; }
```

**do-while**语句:

```
n = 1; s = 1;
do
{ s = s * n;
  n++; }
while(n <= 100);
```

**for**语句:

```
for (n = 1, s = 1; n <= 100; n++)
  s = s * n;
```

# 逗号运算符

} 逗号可作分隔符使用，将若干变量隔开

如 `int a, b, c;`

} 又可作运算符使用：

其一般形式：

表达式1，表达式2，…，表达式n；

# 逗号运算符

```
【例】 main()  
    { int a,b,x;  
      x=(a=8,b=15,b++,a+b);  
      printf("a=%d,b=%d,x=%d\n",a,b,x);}
```

输出结果: a=8, b=16, x=24

```
【例】 main()  
    { int a,b,x;  
      x=a=8,b=15,b++,a+b;  
      printf("a=%d,b=%d,x=%d\n",a,b,x);}
```

输出结果: a=8, b=16, x=8

## 5.3 for 语句——使用格式说明

```
for( 循环变量赋初值; 循环条件; 循环变量增值)  
{ 循环体语句 }
```

### 说明:

for语句中任意一个表达式均可省，但分号“;”不可省

(1)省略“表达式1”：此时在for语句之前给循环变量赋初值。如

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

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

执行时，跳过“求解表达式1”这一步，其他不变。

## 5.3 for 语句——使用格式说明

```
for( 循环变量赋初值; 循环条件; 循环变量增值)  
{ 循环体语句 }
```

### 说明:

(2) 省略表达式2: 即不判断循环条件, 认为表达式2始终为真。

例如: `for(i=1; ;i++) sum=sum+i;`

它相当于:

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

## 5.3 for 语句——使用格式说明

```
for( 循环变量赋初值; 循环条件; 循环变量增值)
{ 循环体语句 }
```

### 说明:

(3)省略表达式3: 此时程序设计者应另外设法保证循环能正常结束。如:

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

## 5.3 for 语句——总结

for语句的使用格式灵活:

- ★ 表达式类型任意, 可省略, 但分号“;”不可省
- ★ 当表达式2被省略时, 需要在循环体中设置循环结束语句, 否则构成死循环

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

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

- ★ 无限循环: `for( ; ; )`
- ★ 括号后边的表达式可以是任意有效的C语言表达式

## 5.3 for 语句——总结

例5

```
main()
{ int i,j,k;
  for(i=0,j=100; i<=j; i++,j--)
  { k=i+j;
    printf("%d+%d=%d\n",i,j,k);
  }
}
```

表达式1、3为  
逗号表达式

### 结论：

表达式2：进行逻辑判断，只要为真就执行循环！

如：for(i=0;i==5;i++) sum+=i;

for(i=0;i=5;i++) sum+=i;

printf("%d\n");

表达式1，3：可以是C语言任意合法表达式

# 真题练习

若输入字符串：abcde<回车>，则以下while循环体将执行（  
）次

```
ch=getchar( );
```

```
while(ch== 'e') printf("*");
```

A) 5

B) 4

C) 1

D) 0

有下列程序：

```
void main( )
```

```
◦ { int k=5;
```

```
◦ while(--k) printf("%d",k-=3);
```

```
◦ printf("\n");
```

```
◦ }执行后的输出结果是（ ）。
```

“ A) 1

B) 2

C) 4

D) 死循环

若变量已正确定义，有下列程序段：

```
i=0;
```

```
do printf("%d, ",i); while(i++);
```

```
printf("%d\n",i);
```

其输出结果是（ ）。

A) 0,0

B) 0,1

C) 1,1

D) 程序进入无限循环

下列程序运行后的输出结果是（ ）。

```
main( )
```

```
{ char c1,c2;
```

```
for(c1='0',c2='9';c1<c2;c1+ +,c2--)
```

```
printf("%c%c",c1,c2);
```

```
printf("\n");}
```

## 5.3 循环嵌套

### 例题1：打印九九乘法表

```
C:\WEXAM\24990001\Debug\progl.exe  
九九乘法表  
-----  
1*1= 1  1*2= 2  1*3= 3  1*4= 4  1*5= 5  1*6= 6  1*7= 7  1*8= 8  1*9= 9  
2*1= 2  2*2= 4  2*3= 6  2*4= 8  2*5=10  2*6=12  2*7=14  2*8=16  2*9=18  
3*1= 3  3*2= 6  3*3= 9  3*4=12  3*5=15  3*6=18  3*7=21  3*8=24  3*9=27  
4*1= 4  4*2= 8  4*3=12  4*4=16  4*5=20  4*6=24  4*7=28  4*8=32  4*9=36  
5*1= 5  5*2=10  5*3=15  5*4=20  5*5=25  5*6=30  5*7=35  5*8=40  5*9=45  
6*1= 6  6*2=12  6*3=18  6*4=24  6*5=30  6*6=36  6*7=42  6*8=48  6*9=54  
7*1= 7  7*2=14  7*3=21  7*4=28  7*5=35  7*6=42  7*7=49  7*8=56  7*9=63  
8*1= 8  8*2=16  8*3=24  8*4=32  8*5=40  8*6=48  8*7=56  8*8=64  8*9=72  
9*1= 9  9*2=18  9*3=27  9*4=36  9*5=45  9*6=54  9*7=63  9*8=72  9*9=81
```

以上内容仅为本文档的试下载部分，为可阅读页数的一半内容。如要下载或阅读全文，请访问：<https://d.book118.com/587062044054010002>