



# **Chapter 5**

## **Control Statements:**

### **Part 2**



# OBJECTIVES



- ❑ The essentials of **counter-controlled repetition**.
- ❑ To use the **for** and **do...while** repetition statements to execute statements in a program repeatedly.
- ❑ To understand multiple selection using the **switch** selection statement.
- ❑ To use the **break** and **continue** program control statements to alter the flow of control.
- ❑ To use the **logical operators**(逻辑运算符) to form complex **conditional expressions**(条件表达式) in control statements.
- ❑ ~~To avoid the consequences of confusing the **equality and assignment operators**.~~



# Topics



- ☐ **5.1 Introduction**
- ☐ **5.2 Essentials of Counter-Controlled Repetition**
- ☐ **5.3 for Repetition Statement**
- ☐ **5.4 Examples Using the for Statement**
- ☐ **5.5 do...while Repetition Statement**
- ☐ **5.6 switch Multiple-Selection Statement**
- ☐ **5.7 break and continue Statements**
- ☐ **5.8 Logical Operators**
- ☐ **5.9 Confusing Equality (==) and Assignment (=)  
Operators**
- ☐ **5.10 Structured Programming Summary**



# 5.1 Introduction



- 选择: **if**, **if...else**, **switch**
- 循环: **while**, **for**, **do...while**
- **Logical operators** to make powerful conditional expressions ( **&&**, **||** )



# Topics

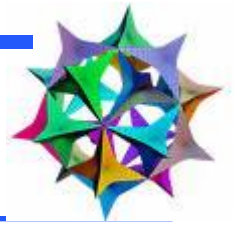
---



- ☐ 5.1 Introduction
- ☐ **5.2 Essentials of Counter-Controlled Repetition**
- ☐ 5.3 for Repetition Statement
- ☐ 5.4 Examples Using the for Statement
- ☐ 5.5 do...while Repetition Statement
- ☐ 5.6 switch Multiple-Selection Statement
- ☐ 5.7 break and continue Statements
- ☐ 5.8 Logical Operators
- ☐ 5.9 Confusing Equality (==) and Assignment (=)  
Operators
- ☐ 5.10 Structured Programming Summary



## 5.2 Essentials of Counter-Controlled Repetition



```
9. int counter = 1; // declare and initialize control variable
10. while ( counter <= 10 ) // loop-continuation condition
11. {
12.     cout << counter << " ";
13.     counter++; // increment control variable by 1
14. } // end while
15. cout << endl; // output a newline
```

1 2 3 4 5 6 7 8 9 10

- Names **counter**(命名计数器变量) 等价 `int counter;  
counter = 1;`
- Declares it to be an integer(声明为整数)
- Reserves space for it in memory(在内存为其保留空间)
- Sets it to an **initial value** of 1(设定一个初始值)



## 5.2 Essentials of Counter-Controlled Repetition



- 计数器变量(控制变量Control Variable)的命名  
counter
- 计数器变量的初始值(initial value)  
counter = 1
- 测试计数器变量终值(final value)的条件  
counter <= 10
- 每次循环时计数器变量修改的增量或减量  
(increment或decrement)  
counter++, ++counter, .....



```
1. int counter = 0 ;  
2. while ( ++counter <= 10 )  
3. {  
4.     cout << counter << " ";  
5. } // end while  
6. cout << endl; // output a newline
```

1 2 3 4 5 6 7 8 9 10

```
1. int counter = 0 ;  
2. while ( counter++ <= 10 )  
3. {  
4.     cout << counter << " ";  
5. } // end while  
6. cout << endl; // output a newline
```





# Topics

---



- ☐ 5.1 Introduction
- ☐ 5.2 Essentials of Counter-Controlled Repetition
- ☐ **5.3 for Repetition Statement**
- ☐ 5.4 Examples Using the for Statement
- ☐ 5.5 do...while Repetition Statement
- ☐ 5.6 switch Multiple-Selection Statement
- ☐ 5.7 break and continue Statements
- ☐ 5.8 Logical Operators
- ☐ 5.9 Confusing Equality (==) and Assignment (=)  
Operators
- ☐ 5.10 Structured Programming Summary



## 5.3 for Repetition Statement



□ 需求：如何使用for语句输出1-10的数字？

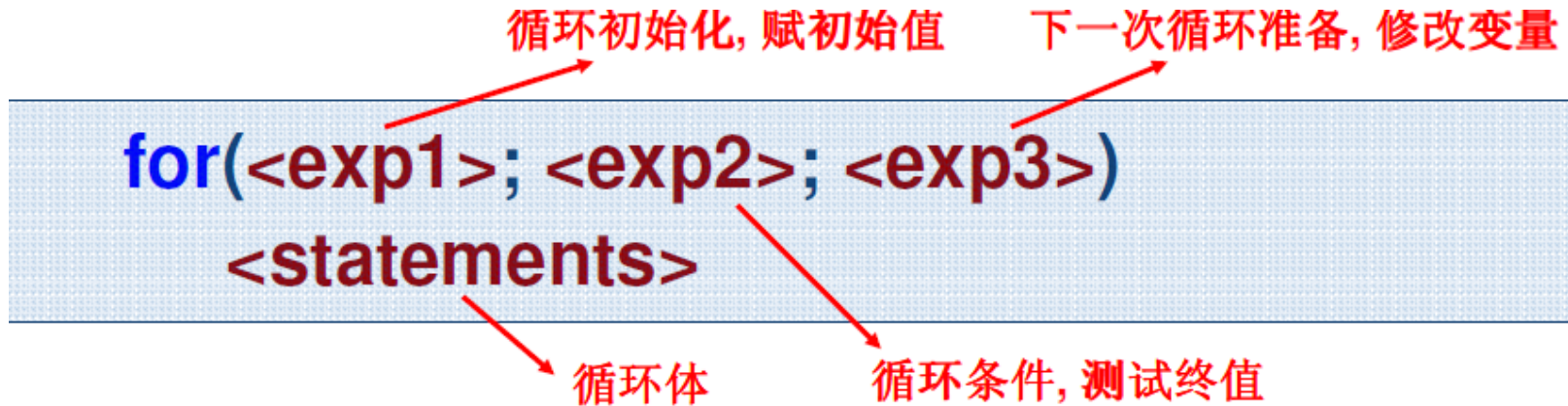
```
9.  int counter = 1; // declare and initialize control variable
10. while ( counter <= 10 ) // loop-continuation condition
11. {
12.     cout << counter << " ";
13.     counter++; // increment control variable by 1
14. } // end while
15. cout << endl; // output a newline
```



## 5.3 for Repetition Statement



### □ 语法规范:



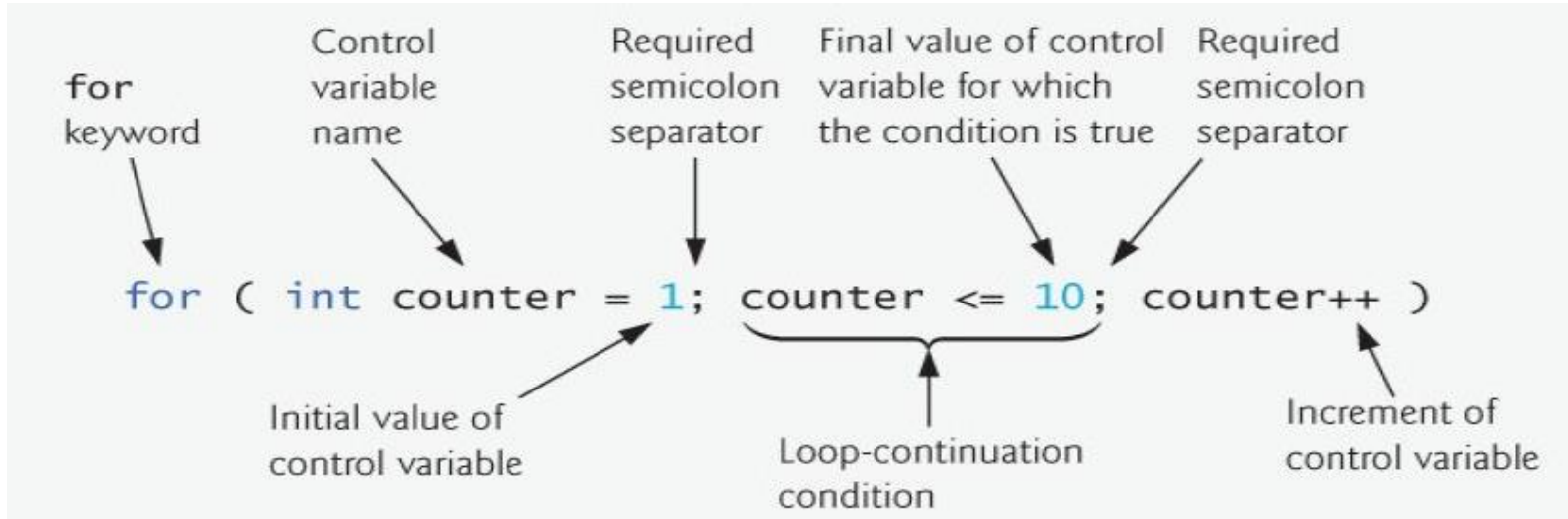
- `<exp1>`, `<exp2>`, `<exp3>`可以是任意表达式, 并且都可以省略
- 通常, `<exp1>`为赋值表达式; `<exp2>`为表示条件的关系或逻辑表达式; `<exp3>`为自增、自减表达式



# 5.3 for Repetition Statement



## □ 1. Header Components



## □ 2. Body({ })

1. <code>for (int counter = 1; counter &lt;= 10; counter++)</code>	Header
2. <code>{</code>	Body
3. <code>statement1; statement2; .....</code>	
4. <code>}</code>	



## 5.3 for Repetition Statement



```
1. for ( int counter = 1; counter <= 10; counter++ )  
2.     cout << counter << " ";  
3.     cout << endl;
```

**1 2 3 4 5 6 7 8 9 10**



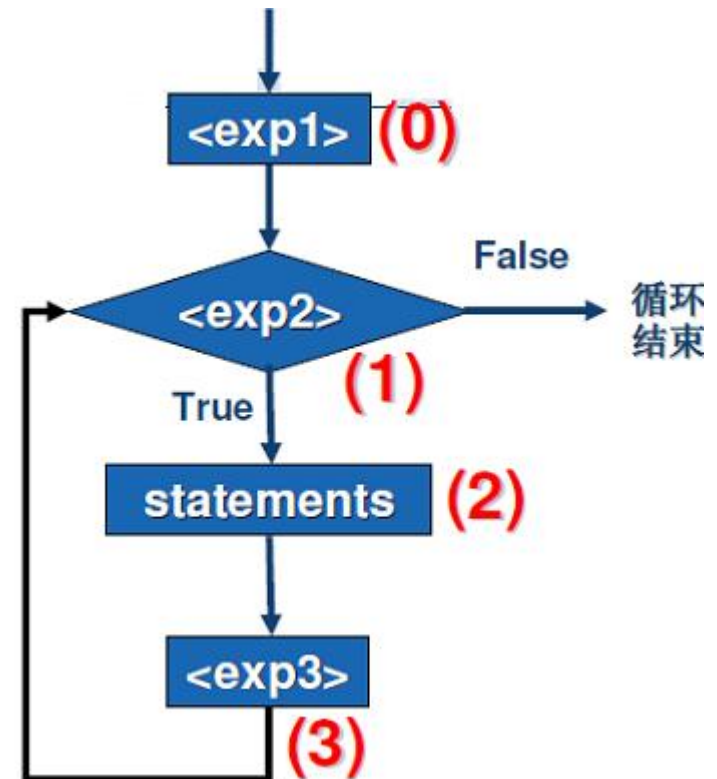
## 5.3 for Repetition Statement



□ 执行顺序:

□ for (**initialization**<sup>(0)</sup>; **loopCondition**<sup>(1)</sup>; **e/increment**<sup>(3)</sup>)  
    **statements1**<sup>(2)</sup>

结论: (0) → (1) ← (3)  
          ↓  
          (2)





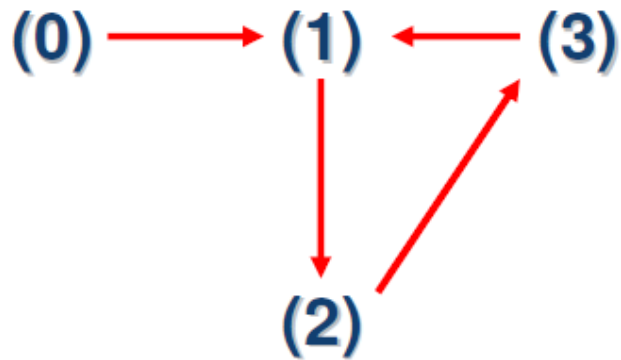


## 5.3 for Repetition Statement



### □ 与while循环的对应关系

```
for(<exp1>; <exp2>; <exp3>)  
    <statements>
```



```
<exp1>;  
while (<exp2>)  
{  
    <statements>;  
    <exp3>;  
}
```



## 5.3 for Repetition Statement



```
1. for ( int counter = 1; counter <= 10; counter++ )  
2.   cout << counter << " ";  
3.   cout << endl;
```

两种循环代码是否完全等价？

```
9.   int counter = 1; // declare and initialize control variable  
10.  while ( counter <= 10 ) // loop-continuation condition  
11.  {  
12.    cout << counter << " ";  
13.    counter++; // increment control variable by 1  
14.  } // end while  
15.  cout << endl; // output a newline
```





## 5.3 for Repetition Statement

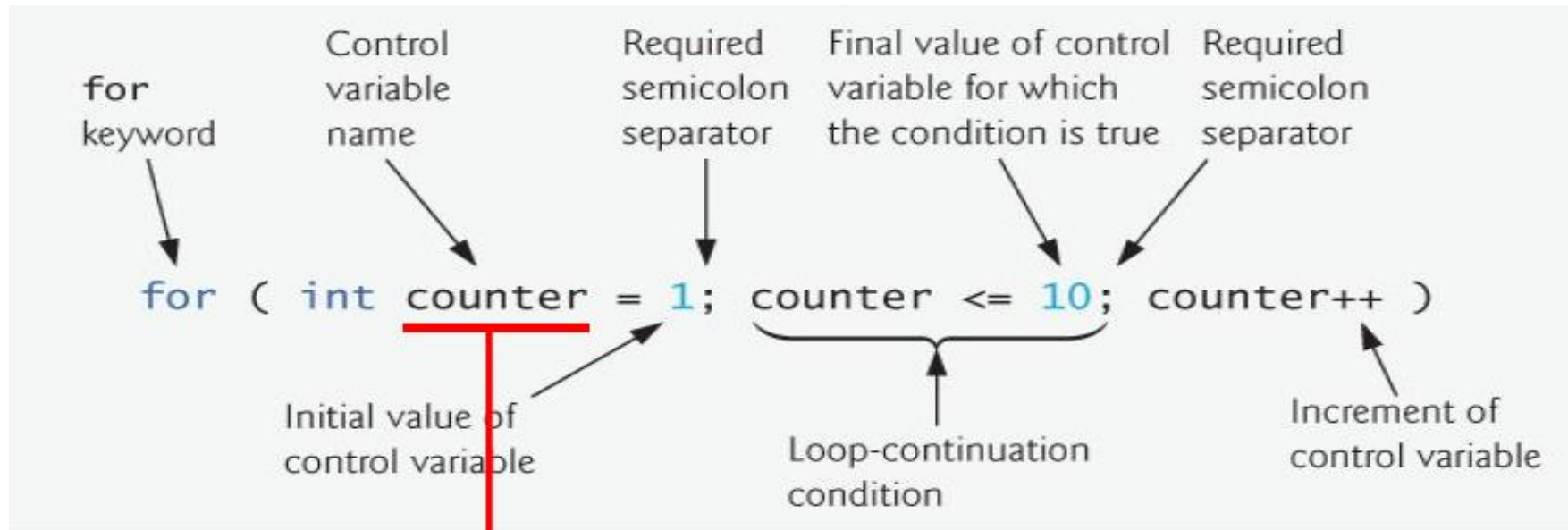


```
1.  for ( int counter = 1; counter <= 10; counter++ )
2.      cout << counter << " ";
3.  cout << counter << endl;

9.  int counter = 1; // declare and initialize control variable
10. while ( counter <= 10 ) // loop-continuation condition
11. {
12.     cout << counter << " ";
13.     counter++; // increment control variable by 1
14. } // end while
15. cout << counter << endl; // output a newline
```



## 5.3 for Repetition Statement



控制变量可以在for结构以外声明，也可以在for结构内部(for头部)声明，但其作用域会有不同



## 5.3 for Repetition Statement



- ❑ 如果在for结构中声明变量，则不要在for语句外使用该变量(避免下面的用法)

```
for ( int i = 0; i <= 10; i++ )  
{ ..... }  
cout << i << endl;
```

- ❑ 如果确实希望在for语句外使用，则不要在for结构中声明该变量

```
int i = 0;  
for ( ; i <= 10; i++ )  
{ ..... }  
cout << i << endl;
```



# Q & A



□ 写出输出结果:

```
1.  int counter = 10;  
2.  while ( --counter )  
3.      cout << counter << " ";  
4.  cout << endl; // output a newline
```

9 8 7 6 5 4 3 2 1

```
1.  int sum = 0, i = 1;  
2.  for ( ; i<=100; )  
3.      sum += i++;  
4.  cout << sum << endl;
```

5050



## 5.3 for Repetition Statement



### Comma expression(逗号表达式)

Operator	Symbol	Form	Operation
Comma	,	x, y	Evaluate x then y, return value of y

- ❑ ① 用逗号隔开的一系列表达式,从左往右依次计算
- ❑ ② 逗号操作符在c++操作符中优先级最低
- ❑ ③ 逗号表达式的值:最右边的表达式的值
- ❑ ④ 主要作用:用于for循环中初始化多个条件,使用多个自增(自减)表达式

```
int x = 0, y = 2;  
int z = ( ++x, x + y );
```





## 5.3 for Repetition Statement



**Comma expression(逗号表达式)**

```
for (int i = 1, j = 10; i <= 10; i++, j--)  
    statement;
```



## 5.3 for Repetition Statement



### Omitted expressions(省略的表达式)

- ① for头部中的三个表达式都可以省略,但分号必须保留
- ② 如果在for语句之前已经对控制变量进行了初始化,那么在for语句中可以省略初始化表达式
- ③ 如果自增(自减)表达式作为语句放在循环体中,则可以省略for语句头结构中的第三个表达式

```
1. int counter = 1;  
2. for ( ; counter < 10; )  
3. {  
4.     cout << counter << " "; // 1 2 3 4 5 6 7 8 9  
5.     counter++;  
6. }
```



## 5.3 for Repetition Statement



Omitted expressions(省略的表达式)

- ❑ ④ 如果循环条件省略，C++认为条件为“真”，将会无限循环

```
for ( ; ; )  
    statement;
```

```
while (true)  
    statement;
```

等价

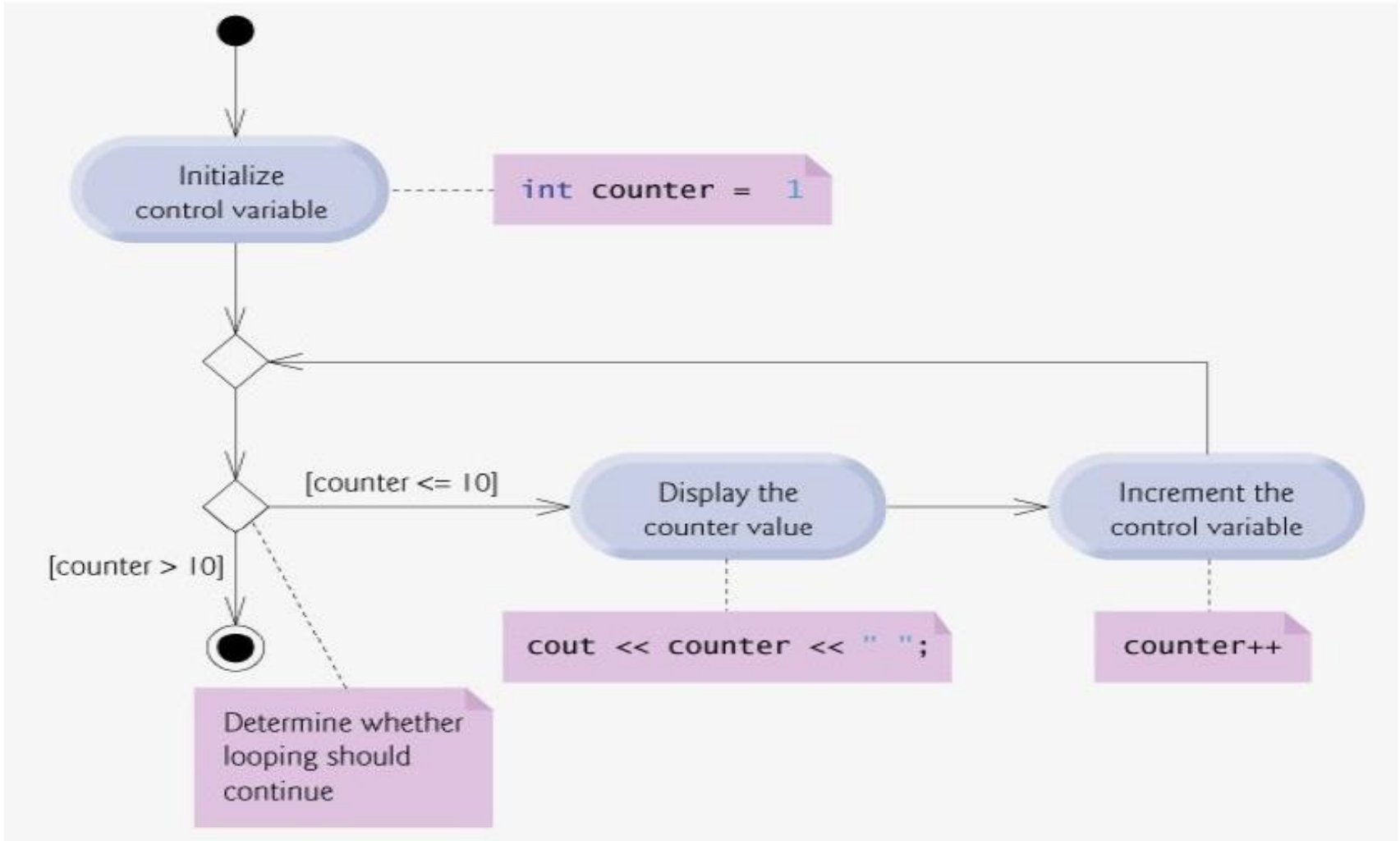
等价

```
for ( ; true; )  
    statement;
```





## 5.3 for Repetition Statement





# Topics

---



- ☐ 5.1 Introduction
- ☐ 5.2 Essentials of Counter-Controlled Repetition
- ☐ 5.3 for Repetition Statement
- ☐ **5.4 Examples Using the for Statement**
- ☐ 5.5 do...while Repetition Statement
- ☐ 5.6 switch Multiple-Selection Statement
- ☐ 5.7 break and continue Statements
- ☐ 5.8 Logical Operators
- ☐ 5.9 Confusing Equality (==) and Assignment (=)  
Operators
- ☐ 5.10 Structured Programming Summary



## 5.4 Examples Using the for Statement



- ❑ `for ( int i = 1; i <= 100; i++ ) cout << i;`  
1, 2, 3, 4, ....., 100
- ❑ `for ( int i = 20; i >= 2; i -= 2 ) cout << i;`  
20, 18, 16, 14, ....., 2
- ❑ `for ( int i = 7; i <= 77; i += 7 ) cout << i;`  
7, 14, 21, 28, 35, ....., 77
- ❑ `for ( int i = 7; i <= 80; i += 7 ) cout << i;`  
7, 14, 21, 28, 35, ....., 77



## 5.4 Examples Using the for Statement



□ 问题1：计算2至20之间偶数之和

1. `int total = 0;`
2. `for ( int number = 2; number <= 20; number += 2 )`
3. `total += number;`

$$2 + 4 + 6 + \dots + 20 = 110$$

4. `for ( int number = 2, total = 0;`
5. `number <= 20;`
6. `total += number, number += 2 ) ;`

$$2 + 4 + 6 + \dots + 20 = 110$$

7. `for ( int number = 2, total = 0;`
8. `number <= 20;`
9. `number += 2, total += number) ;`

$$4 + 6 + 8 + \dots + 22 = 130$$



## 5.4 Examples Using the for Statement



□ 问题2: 计算  $1 + \frac{1}{3} + \frac{1}{5} \dots + \frac{1}{9}$



## 5.4 Examples Using the for Statement



- 问题3：一个人在银行存款1000.00美元，利率为5%，假设所有利息留在账号中，计算并打印出10年内每年年末的金额。用下列公式求出金额：

$$a = p(1 + r)^n$$

- 其中：**p**是原存款额，**r**是年利率，**n**是年数，**a**是年末总存款



## 5.4 Examples Using the for Statement



```
1. // P154. Fig. 5.6
2. // Compound interest calculations with for.
3. #include <iostream>
4. using std::cout;
5. using std::endl;
6. using std::fixed;
7.
8. #include <iomanip>
9. using std::setw; // enables program to set a field width
10. using std::setprecision;
11.
12. #include <cmath> // standard C++ math library
13. using std::pow; // enables program to use function pow
```





```
15 int main()
16 {
17     double amount; // amount on deposit at end of each year
18     double principal = 1000.0; //
19     double rate = .05; // interest
20
21     // display headers
22     cout << "Year" << setw( 21 ) << "Amount on deposit" << endl;
23
24     // set floating-point number format
25     cout << fixed << setprecision( 2 );
26
27     // calculate amount on deposit for
28     for ( int year = 1; year <= 10; year++ )
29     {
30         // calculate new amount for specified year
31         amount = principal * pow( 1.0 + rate, year );
32
33         // display the year and the amount
34         cout << setw( 4 ) << year << setw( 21 ) << amount << endl;
35     } // end for
36
37     return 0; // indicate successful termination
38 }
```

设置域宽，即设置输出数据所占的宽度，如果输出值超出设定域宽，则按照实际宽度输出。仅适用于下一次输出

sticky setting(粘性设置)，此后的输出会按照此处的设定，直到重新设定

$(1.0 + \text{rate})^{\text{year}}$





## 5.4 Examples Using the for Statement



**#include <cmath>**

**□ double pow( double x, double y );**

**□ 计算x 的y 次幂**

**□ pow( 1.0 + rate, year )**

**□ Argument Coercion, 实参的强制类型转换  
(Ch6.5)**



## 5.4 Examples Using the for Statement



- 格式化流算子
- ① 带参数的流操作算子  
    setw(int), setprecision(int)  
    #include<iomanip>
- ② 不带参数的流算子  
    fixed, left, right (缺省)  
    #include<iostream>



## 5.4 Examples Using the for Statement



- 格式化流算子
- ① 粘性设置(sticky setting): 可以设定程序其后所有cout的输出格式, 直到被重置, 如:  
`setprecision(int), fixed, left, right` 等等
- ② 非粘性设置: 只作用在紧接着的cout输出上, 如: `setw(int)`



# Topics



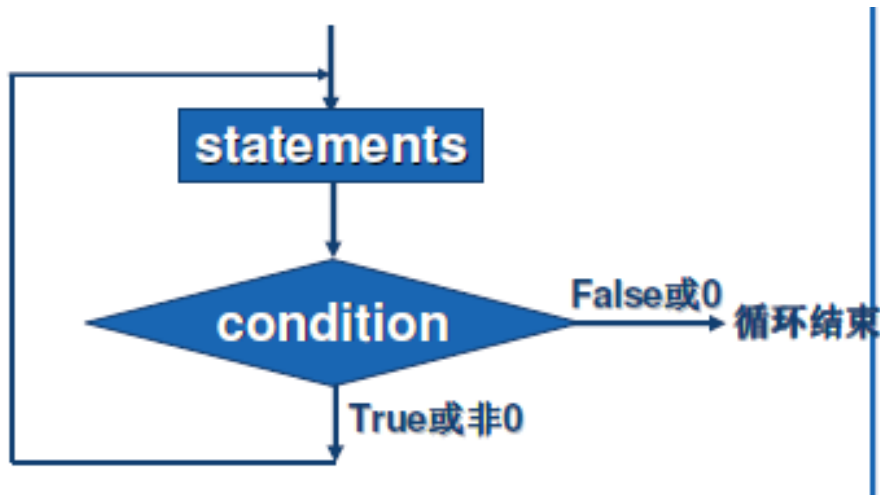
- ☐ 5.1 Introduction
- ☐ 5.2 Essentials of Counter-Controlled Repetition
- ☐ 5.3 for Repetition Statement
- ☐ 5.4 Examples Using the for Statement
- ☐ **5.5 do...while Repetition Statement**
- ☐ 5.6 switch Multiple-Selection Statement
- ☐ 5.7 break and continue Statements
- ☐ 5.8 Logical Operators
- ☐ 5.9 Confusing Equality (==) and Assignment (=)  
Operators
- ☐ 5.10 Structured Programming Summary



# 5.5 do...while Repetition Statement



- ❑ **do{**
- ❑     **statements**
- ❑ **} while ( condition );**





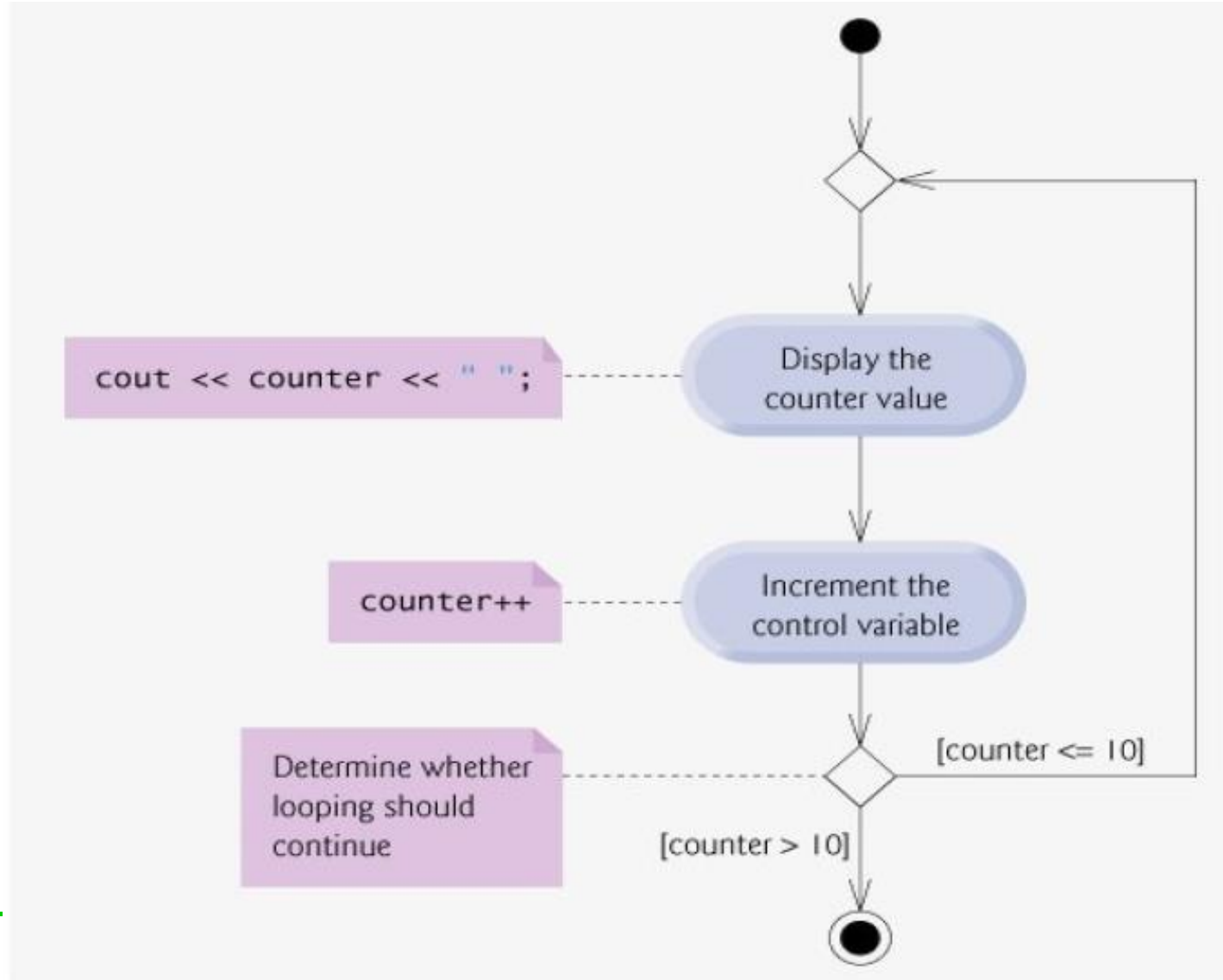
# 5.5 do...while Repetition Statement



```
1. // P157 Fig. 5.7: fig05_07.cpp
2. // do...while repetition statement.
3. #include <iostream>
4. using std::cout;
5. using std::endl;
6.
7. int main()
8. {
9.     int counter = 1; // initialize counter
10.
11.     do
12.     {
13.         cout << counter << " "; // display counter
14.         counter++; // increment counter
15.     } while ( counter <= 10 ); // end do...while
16.
17.     cout << endl; // output a newline
18.     return 0; // indicate successful termination
19. } // end main
```



# 5.5 do...while Repetition Statement





# Topics

---



- ☐ 5.1 Introduction
- ☐ 5.2 Essentials of Counter-Controlled Repetition
- ☐ 5.3 for Repetition Statement
- ☐ 5.4 Examples Using the for Statement
- ☐ 5.5 do...while Repetition Statement
- ☐ **5.6 switch Multiple-Selection Statement**
- ☐ 5.7 break and continue Statements
- ☐ 5.8 Logical Operators
- ☐ 5.9 Confusing Equality (==) and Assignment (=)  
Operators
- ☐ 5.10 Structured Programming Summary





# 5.6 switch Multiple-Selection Statement



- **switch** 多选：根据表达式(expression)的值，执行不同的操作

```
switch ( value ){  
    case 常量表达式1: 语句1  
    case 常量表达式2: 语句2  
    .....  
    case 常量表达式n: 语句n  
    default:           语句n+1  
}
```

值为**整型**、**字符型**或**枚举型**

必须为**常量**，并且不能有多个值相等的**case**分支

- 从**常量表达式1**开始逐个比对**value**的值，从与之相等的常量表达式标签(Label)处**开始执行**！



## 5.6 switch Multiple-Selection Statement



```
1. float f = 0.3;
2. switch ((f)) {
3.     case 0.1: cout << "0.1" << endl;
4.     case 0.2: cout << "0.2" << endl;
5.     default: cout << "default" << endl;
6. }
```

- ❑ error C2450: switch expression of type 'float' is illegal
- ❑ switch()中的value必须为整型、字符型或者枚举型的变量/表达式!



## 5.6 switch Multiple-Selection Statement



```
1. char cc = 'A';  
2. char dd = 'B';  
3. switch( cc ) {  
4.     case 'A': cout << "A is input" << endl;  
5.     case dd) cout << "B is input" << endl;  
6.     default: cout << "default" << endl;  
7. }
```

- ❑ error C2051: case expression not constant
- ❑ case后必须是常量表达式！



## 5.6 switch Multiple-Selection Statement



```
1. char cc = 'A';
2. switch( cc ) {
3.     case 'A': cout << "A is input" << endl;
4.     case 'B': cout << "B is input" << endl;
5.     case 65: cout << "what is input?" << endl;
6.     default: cout << "default" << endl;
7. }
```

- ❑ error C2196: case value '65' already used
- ❑ 各case常量表达式的值应各不相同！
- ❑ ASCII表, P900



## 5.6 switch Multiple-Selection Statement



```
1. char cc = 'B';  
2. switch( cc ) {  
3.     case 'A': cout << "A is input" << endl;  
4.     case 'B': cout << "B is input" << endl;  
5.     case 'C': cout << "C is input" << endl;  
6.     default: cout << "default" << endl;  
7. }
```

□ 从值匹配处开始执行!

```
B is input  
C is input  
default  
Press any key to continue
```



## 5.6 switch Multiple-Selection Statement



```
1. char cc = 'B';  
2. switch (cc) {  
3.     case 'A': cout << "A is input" << endl; break;  
4.     case 'B': cout << "B is input" << endl; break;  
5.     case 'C': cout << "C is input" << endl; break;  
6.     default: cout << "default" << endl;  
7. }
```

- ❑ switch一般需要与break搭配使用!
- ❑ break: 退出switch语句



## 5.6 switch Multiple-Selection Statement



- 需求：从键盘接受成绩输入,统计各个档次成绩的成绩数  
(‘A’, ‘a’), (‘B’, ‘b’), (‘C’, ‘c’),  
(‘D’, ‘d’), (‘F’, ‘f’)
- 输入的成绩数量不确定
  - ❖ 标记值控制的循环
- 成绩档次为5档
  - ❖ switch多选





# 5.6 switch Multiple-Selection Statement



```
54  int grade;
```

P158 Fig. 5.10

```
60  while( grade = cin.get() ) != EOF)
```

```
61  {
```

- ❑ 从标准输入读取一个字符(如 ‘a’)
- ❑ 将读入的字符赋值给整型变量grade，因此取该字符的ASCII码值(97)

```
cout << "The character (" << 'a' << ") has the value "  
      << static_cast<int> ( 'a' ) << endl;
```

```
The character (a) has the value 97
```



# 5.6 switch Multiple-Selection Statement



```
54  int grade;
```

P158 Fig. 5.10

```
60  while( grade = cin.get() ) != EOF)
```

```
61  {
```

- 赋值表达式的值即= 右边的值，EOF作为标记值(Sentinel)来判断输入是否结束。

```
a = b = c = 0;
```



## 5.6 switch Multiple-Selection Statement



```
54  int grade;
```

P158 Fig. 5.10

```
60  while( (grade = cin.get() ) != EOF)
```

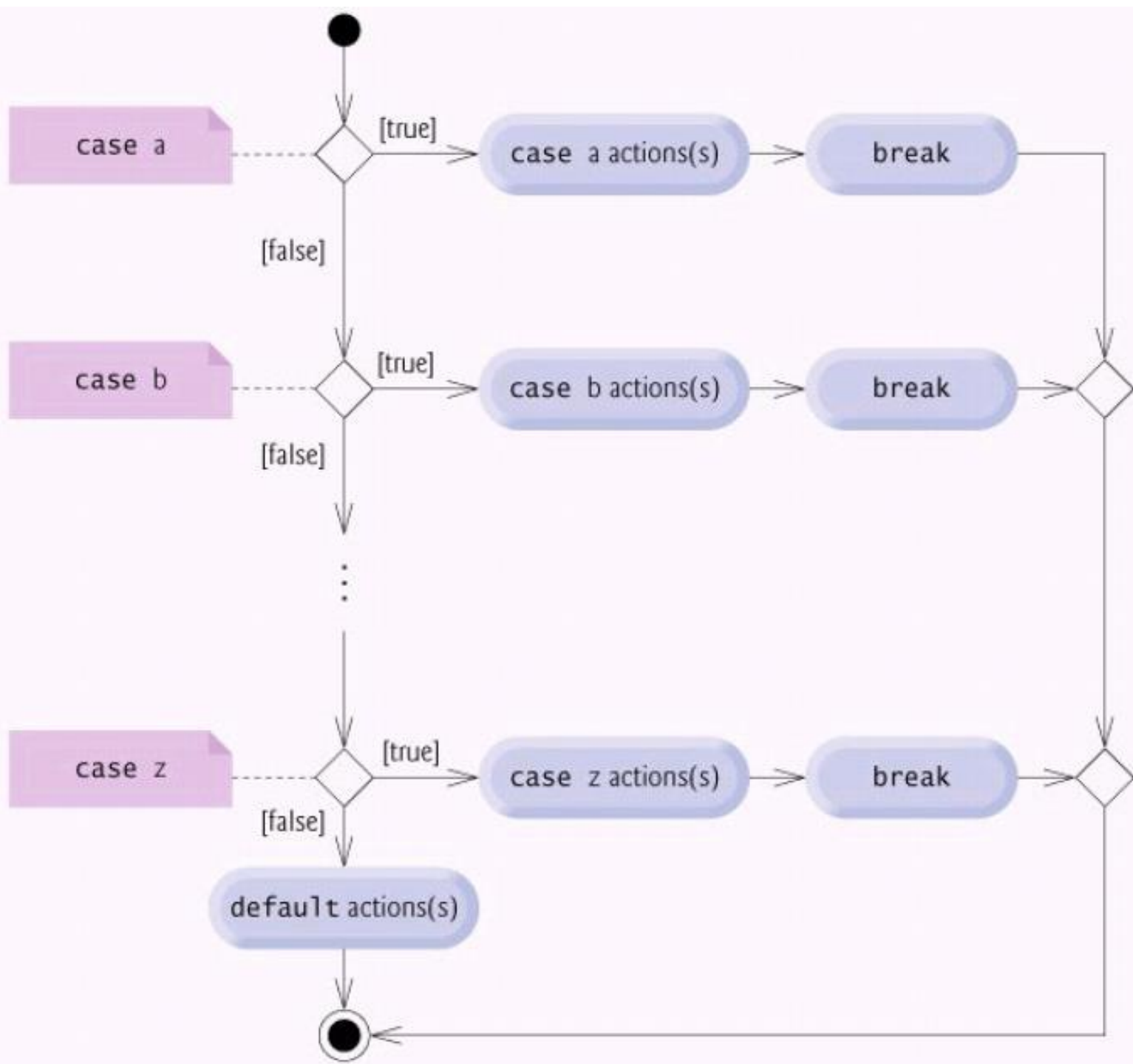
```
61  {
```

□ EOF (End of File)是<iostream>头文件中定义的符号常量(Symbolic integer constant),表示文件结束. 具体值是操作系统相关的,

Win32中为Ctrl+Z

```
#define EOF (-1)
```

程序解读 (P125)





# Topics

---



- ☐ 5.1 Introduction
- ☐ 5.2 Essentials of Counter-Controlled Repetition
- ☐ 5.3 for Repetition Statement
- ☐ 5.4 Examples Using the for Statement
- ☐ 5.5 do...while Repetition Statement
- ☐ 5.6 switch Multiple-Selection Statement
- ☐ **5.7 break and continue Statements**
- ☐ 5.8 Logical Operators
- ☐ 5.9 Confusing Equality (==) and Assignment (=)  
Operators
- ☐ 5.10 Structured Programming Summary



# 5.7 break and continue Statements



□ **break**:立即退出最内层的  
**switch/while/for/do...while**语句，执行后续语句

// P165, Fig. 5.13

```
11 for ( count = 1; count <= 10; count++ ) // loop 10 times
12 {
13     if ( count == 5 )
14         break; // break loop only if count is 5
15
16     cout << count << " ";
17 } // end for
18
19 cout << "\nBroke out of loop at count = " << count << endl;
```

1 2 3 4  
Broke out of loop at count = 5



# 5.7 break and continue Statements



```
1. for ( int i = 1; i <= 5; i++ )
2. {
3.     cout << "i = " << i << ":" << endl;
4.     for ( int count = 1; count <= 10; count++ )
5.     {
6.         if ( count == 5 ) break; // if count is 5
7.         cout << count << " ";
8.     } // end inner for
9.     cout << "\nBroke out of loop at count " << i << endl;
10. } // end outer for
11. cout << "Cannot break to here!" << endl;
```

```
i = 1:
1 2 3 4
Broke out of loop at count = 5
i = 2:
1 2 3 4
Broke out of loop at count = 5
i = 3:
1 2 3 4
Broke out of loop at count = 5
i = 4:
1 2 3 4
Broke out of loop at count = 5
i = 5:
1 2 3 4
Broke out of loop at count = 5
Cannot break to here!
```





# 5.7 break and continue Statements



□ **continue**: 跳过最内层while/for/do...while 结构的剩余语句, 执行下一次循环.

```

      (0)          (1)          (3)
1.  for ( int count = 1; count <= 10; count++ )
2.  {
3.      if ( count == 5 ) // if count is 5,
4.      continue;        // skip remaining code in loop
      (2)
5.      cout << count << " ";
6.  } // end for
```

1 2 3 4 6 7 8 9 10



## 5.7 break and continue Statements



```
1. int count = 0;  
2. while (count <= 9)  
3. {  
4.   
5.     if ( count == 5 ) // if count is 5,  
6.         continue;  
7.     count++; // 死循环, 不应跳过下一次循环准备  
8.     cout << count << " ";  
9. } // end while
```

1 2 3 4 6 7 8 9 10



# 5.7 break and continue Statements



```
1. int count = 0;  
2.  
3. do{  
4.     count++;  
5.     if ( count == 5 ) // if count is 5,  
6.         continue;  
7.  
8.     cout << count << " ";  
9. } while (count <= 9)
```

1 2 3 4 6 7 8 9 10



# 5.7 break and continue Statements



- **break**: 跳出所在的语句体, 执行后续语句
- **continue**: 返回至循环头部, 执行下次循环
- • **for**  
返回至**for**头部的下一次循环准备
- • **while**  
返回至**while**头部的循环条件判断
- • **do...while**  
返回至**while**头部的循环条件判断



# 5.7 break and continue Statements



1. `for ( int i = 1; i <= 5; i++ )`
2. `{`
3. `if (i==2) continue;`
4. `cout << "i = " << i << ":" << endl;`
5. `for ( int count = 1; count <= 10; count++`
6. `{`
7. `if ( count == 5 ) break; // if count is 5`
8. `cout << count << " ";`
9. `} // end inner for`
10. `} // end outer for`
11. `cout << "Cannot break to here!" << endl;`

```
i = 1:
1 2 3 4
Broke out of loop at count = 5

i = 2:
1 2 3 4
Broke out of loop at count = 5

i = 3:
1 2 3 4
Broke out of loop at count = 5

i = 4:
1 2 3 4
Broke out of loop at count = 5

i = 5:
1 2 3 4
Broke out of loop at count = 5
Cannot break to here!
```



# Topics



- ☐ 5.1 Introduction
- ☐ 5.2 Essentials of Counter-Controlled Repetition
- ☐ 5.3 for Repetition Statement
- ☐ 5.4 Examples Using the for Statement
- ☐ 5.5 do...while Repetition Statement
- ☐ 5.6 switch Multiple-Selection Statement
- ☐ 5.7 break and continue Statements
- ☐ **5.8 Logical Operators**
- ☐ 5.9 Confusing Equality (==) and Assignment (=)  
Operators
- ☐ 5.10 Structured Programming Summary



## 5.8 Logical Operators



```
if ( gender == 1 && age >= 65 )  
    seniorFemales++;
```

- 多个逻辑表达式值同时为TRUE,才为TRUE
- 真值表 (Truth Table)

expression1	expression2	expression1 && expression2
false	false	false
false	true	false
true	false	false
<u>true</u>	<u>true</u>	<u>true</u>





## 5.8 Logical Operators



```
1. int a = 2, b = 3, c = 3;  
2. if ( a == 1 && b++ >= 2 && c==3){  
3.     c++;  
4. }  
5. cout << "b = " << b << ", c = " << c << endl;
```

**b = 3, c = 3**

□ 多个逻辑表达式, 从左至右依次求值

□ 短路求值(short-circuit evaluation)

❖ 若a不等于1, 那么整个表达式值肯定为false, 此时后续两个逻辑表达式不做计算

❖ 副作用



## 5.8 Logical Operators



```
if ( average >= 90 || finalExam >= 90 )  
    cout << "Student grade is A" << endl;
```

- ❑ 多个逻辑表达式值只有全部为False，才False
- ❑ 真值表（Truth Table）

expression1	expression2	expression1    expression2
<u>false</u>	<u>false</u>	<u>false</u>
false	true	true
true	false	true
true	true	true



## 5.8 Logical Operators



```
1. int a = 2, b = 3, c = 3;  
2. if ( a == 2 || b++ >= 2 || c==3) {  
3.     c++;  
4. }  
5. cout << "b = " << b << ", c = " << c << endl;
```

**b = 3, c = 4**

- 多个逻辑表达式, 从左至右依次求值, 只要有一个为True, 则True
- 短路求值 (short-circuit evaluation)
  - ❖ 若a等于2, 那么整个表达式值肯定为true, 此时后续两个逻辑表达式不做计算
  - ❖ 副作用



## 5.8 Logical Operators



```
if ( !( grade == sentinelValue ) )  
    cout << "The next grade is " << grade << endl;
```

```
if ( grade != sentinelValue )  
    cout << "The next grade is " << grade << endl;
```

□ 取反, 真值表 (Truth Table)

expression	!expression
false	true
true	false



# 5.8 Logical Operators



Operators						Associativity	Type
()						left to right	parentheses
++	--	static_cast< type >()				left to right	unary (postfix)
++	--	+	-	!		right to left	unary (prefix)
*	/	%				left to right	multiplicative
+	-					left to right	additive
<<	>>					left to right	insertion/extraction
<	<=	>	>=			left to right	relational
==	!=					left to right	equality
&&						left to right	logical AND
						left to right	logical OR
?:						right to left	conditional
=	+=	--	*=	/=	%=	right to left	assignment
,						left to right	comma



## 5.8 Logical Operators



□ 要求  $1 \leq \text{month} \leq 12$ , 否则赋值为1

□ `if( ! (1 <= month && month <= 12) )`  
    `month = 1;`

□ `if( month < 1 || month > 12 )`  
    `month = 1;`



## 5.8 Logical Operators



1. `cout << "True = " << true` → C++关键字
2. `<< ", False = " << false << endl;`
3. `cout << boolalpha`
4. `<< "True = " << true`
5. `<< ", False = " << false << endl;`

□ 没有参数的流操作算子，粘性设置！

```
True = 1, False = 0
True = true, False = false
```





# Topics



- ☐ 5.1 Introduction
- ☐ 5.2 Essentials of Counter-Controlled Repetition
- ☐ 5.3 for Repetition Statement
- ☐ 5.4 Examples Using the for Statement
- ☐ 5.5 do...while Repetition Statement
- ☐ 5.6 switch Multiple-Selection Statement
- ☐ 5.7 break and continue Statements
- ☐ 5.8 Logical Operators
- ☐ **5.9 Confusing Equality (==) and Assignment (=)  
Operators**
- ☐ 5.10 Structured Programming Summary



## 5.9 Confusing Equality (==) and Assignment (=) Operators



```
1. if ( payCode = 4 ) // Logic Error
2.     cout << "You get a bonus!" << endl;
3. if ( payCode == 4 )
4.     cout << "You get a bonus!" << endl;
```

- 易产生混淆的原因:
- 任何具有值的表达式都可以作为判断语句的条件
- 任何赋值语句都具有值,即赋值于该变量的值



## 5.9 Confusing Equality (==) and Assignment (=) Operators



`payCode = 4;`

- 变量Variable: 左值(lvalues)  
一般是指可以搁在赋值表达式的左边
- 常量Constant: 右值(rvalues)  
只能搁在赋值表达式的右边

```
if ( 4 == payCode ) // 建议
```

```
    cout << "You get a bonus!" << endl;
```

```
if ( 4 = payCode ) // 编译错误
```

```
    cout << "You get a bonus!" << endl;
```

**error C2106: '=' : left operand must be l-value**



# Topics

---



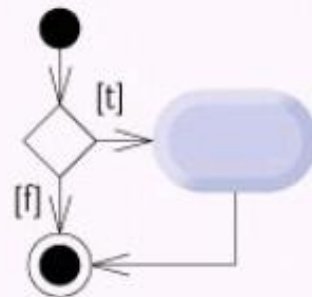
- ☐ 5.1 Introduction
- ☐ 5.2 Essentials of Counter-Controlled Repetition
- ☐ 5.3 for Repetition Statement
- ☐ 5.4 Examples Using the for Statement
- ☐ 5.5 do...while Repetition Statement
- ☐ 5.6 switch Multiple-Selection Statement
- ☐ 5.7 break and continue Statements
- ☐ 5.8 Logical Operators
- ☐ 5.9 Confusing Equality (==) and Assignment (=)  
Operators
- ☐ 5.10 Structured Programming Summary

## Sequence

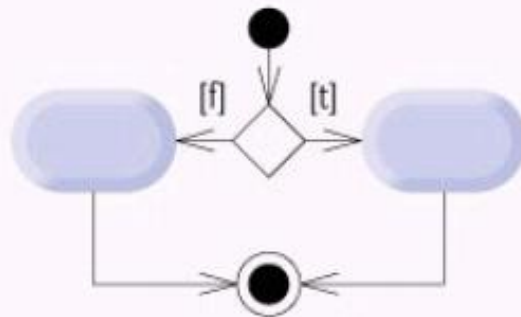


## Selection

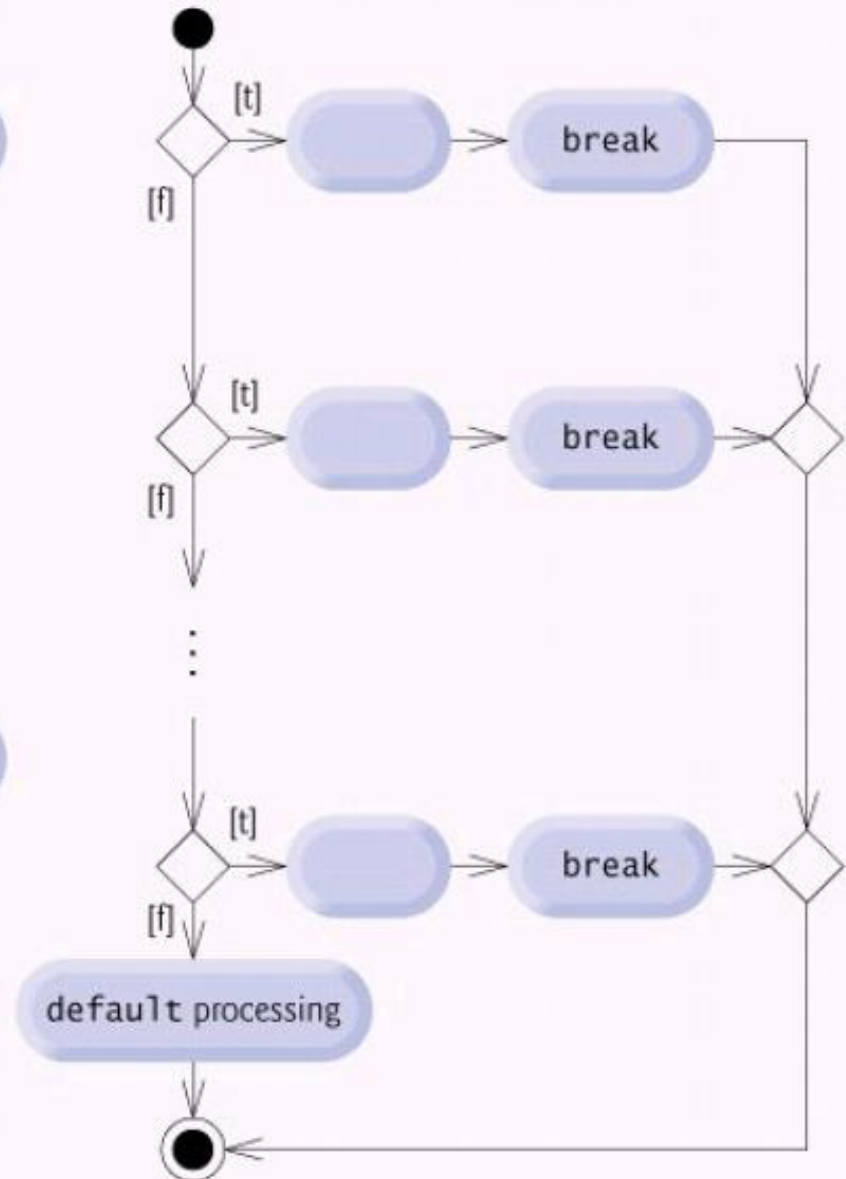
if statement  
(single selection)



if...else statement  
(double selection)



switch statement with breaks  
(multiple selection)



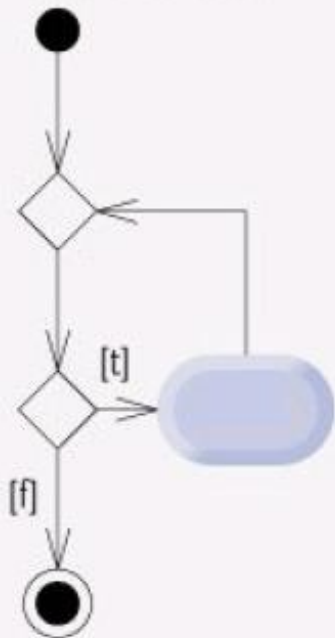


# 5.10 Structured Programming Summary

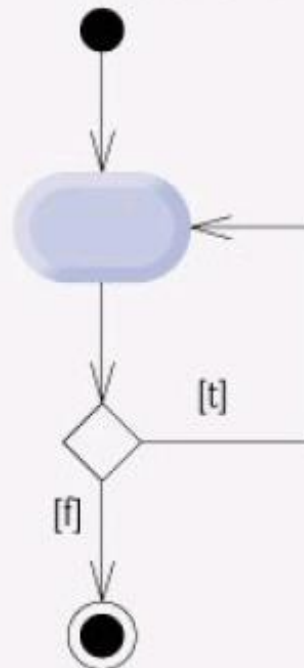


## Repetition

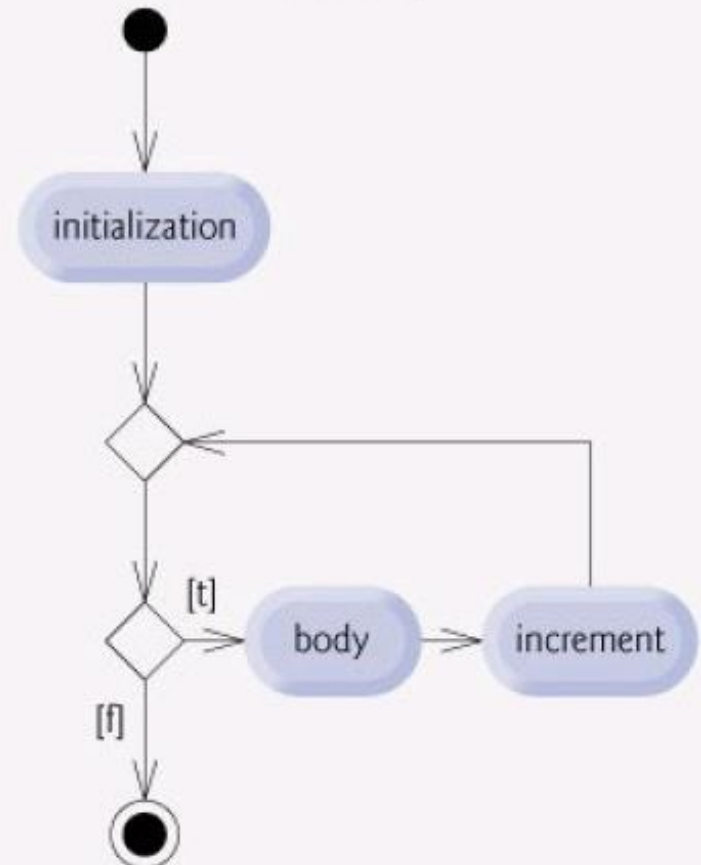
while statement



do...while statement



for statement





## 5.10 Structured Programming Summary



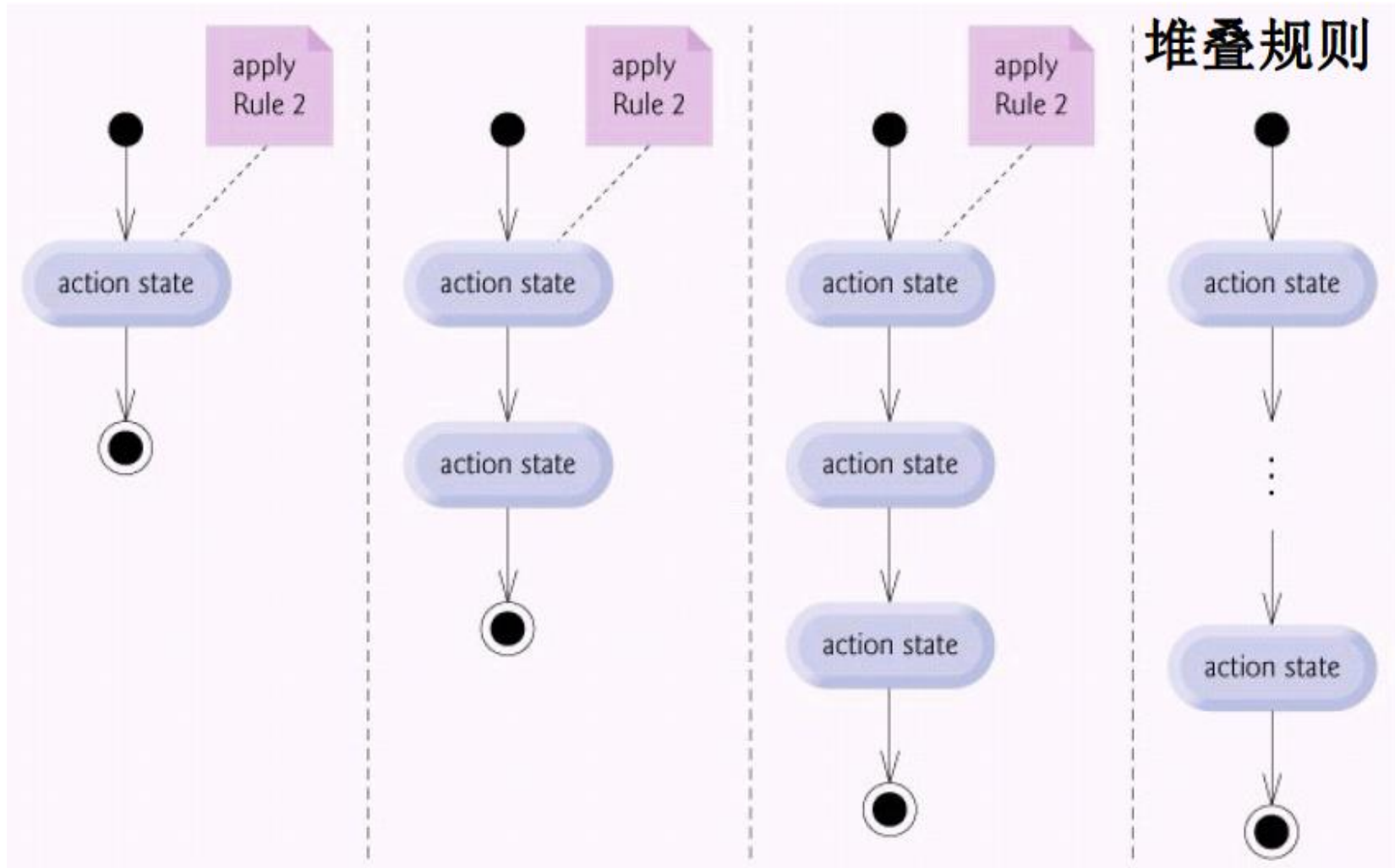
### Rules for Forming Structured Programs

- ① Begin with the "simplest activity diagram".
- ② **堆叠规则**: Any action state can be replaced by **two action states in sequence**.
- ③ **嵌套规则**: Any action state can be replaced by **any control statement** (sequence, if, if...else, switch, while, do...while or for).
- ④ Rules 2 and 3 can be applied **as often as you like** and **in any order**.





# 5.10 Structured Programming Summary



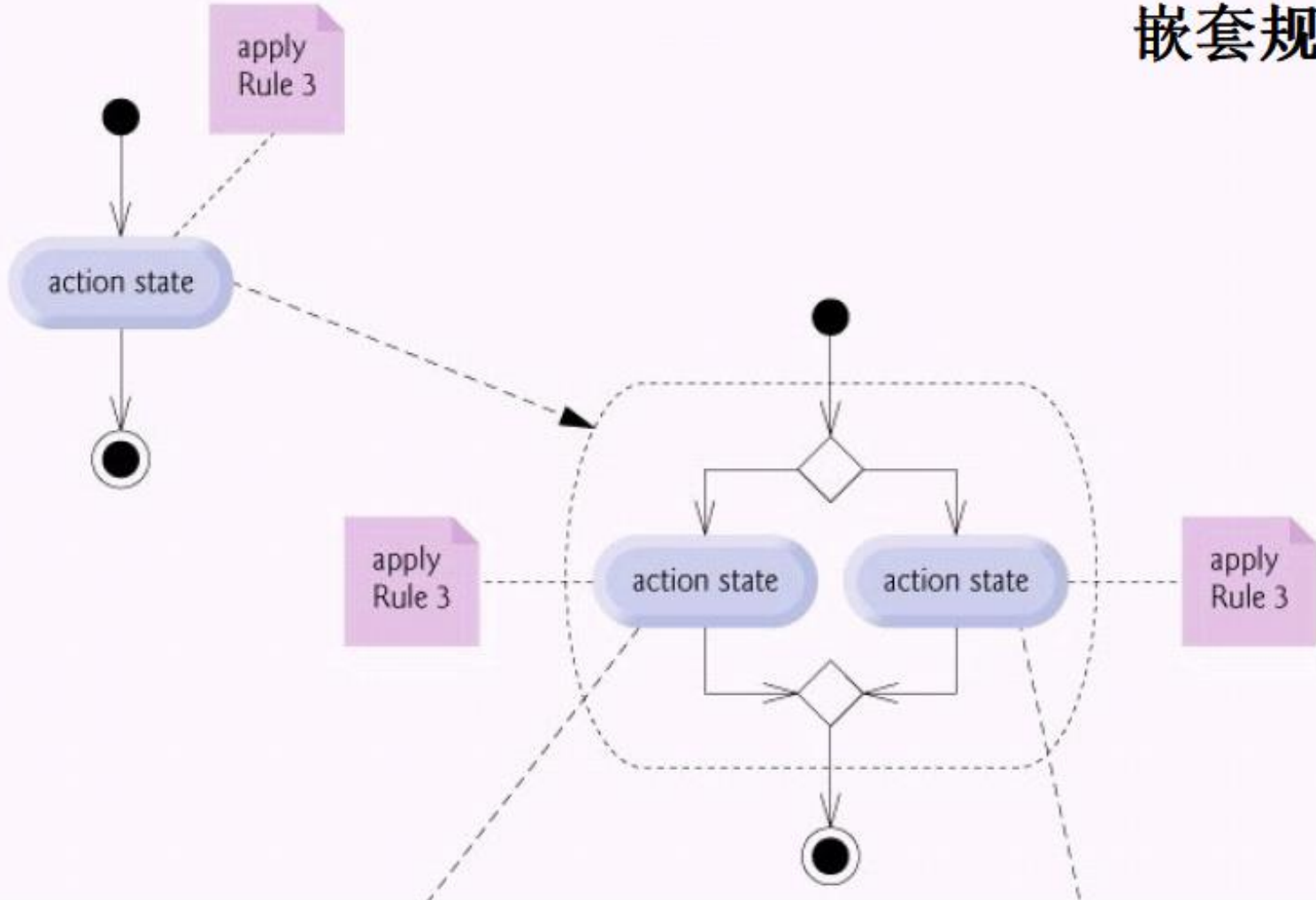


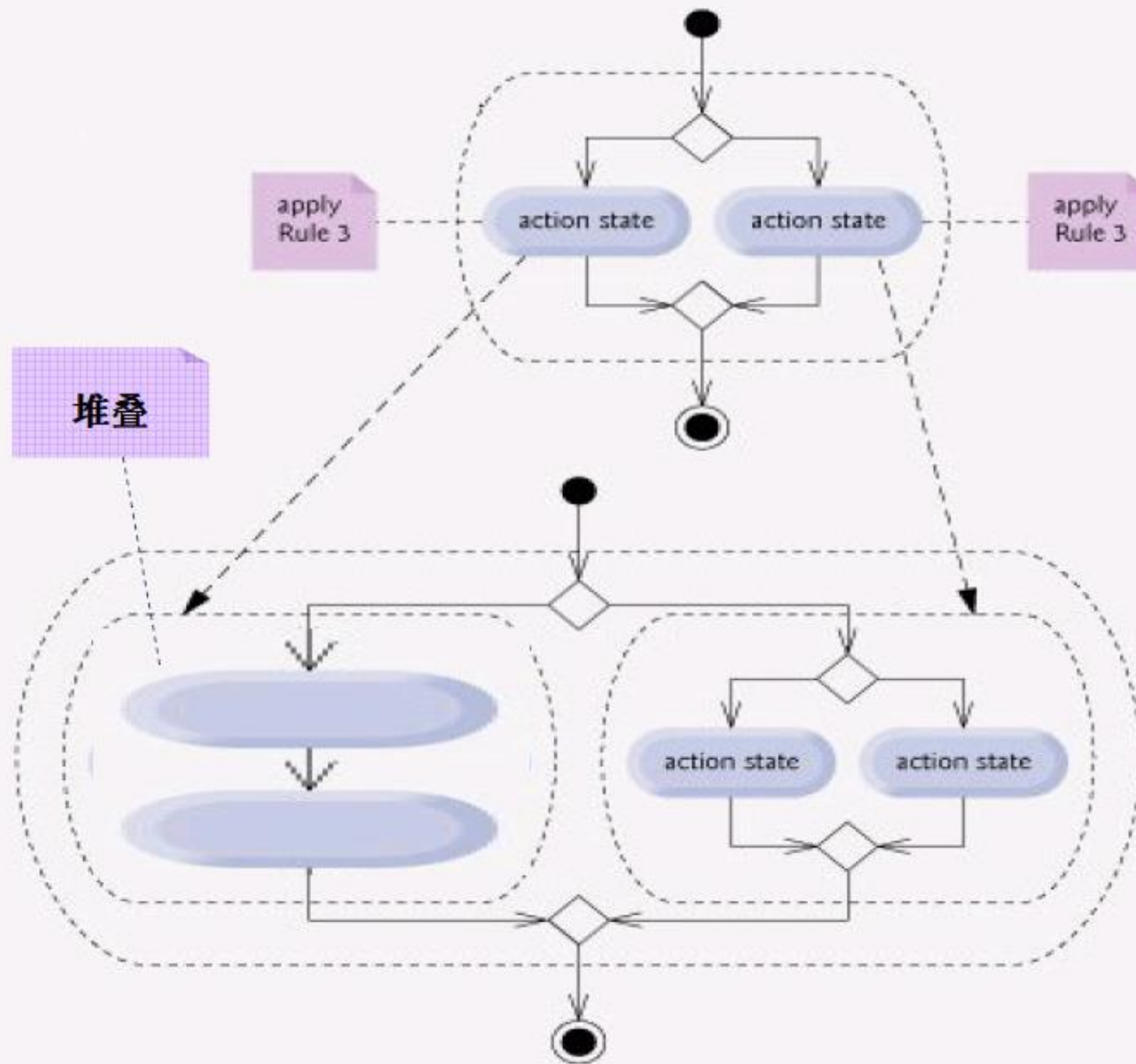


# 5.10 Structured Programming Summary



嵌套规则







# 5.10 Structured Programming Summary



- 结构化编程的优势在于：我们可以仅使用7个单入口-单出口的控制语句，并以堆叠和嵌套两种方式组合，即可设计出任何程序逻辑！
- 进一步，实际上：
  - ❖ if语句可以实现if...else、switch的程序控制功能
  - ❖ while语句可以实现for、do...while的程序控制功能
- 结论：顺序sequence + 选择if + 循环while – 堆叠Stacking+嵌套Nesting 即可完成结构化的程序设计！



# Homework



☐ 实验必选题目:

☐ 5.19, 5.23

☐ 实验任选题目:

☐ 作业题目(Homework):

☐ 5.4, 5.5, 5.26