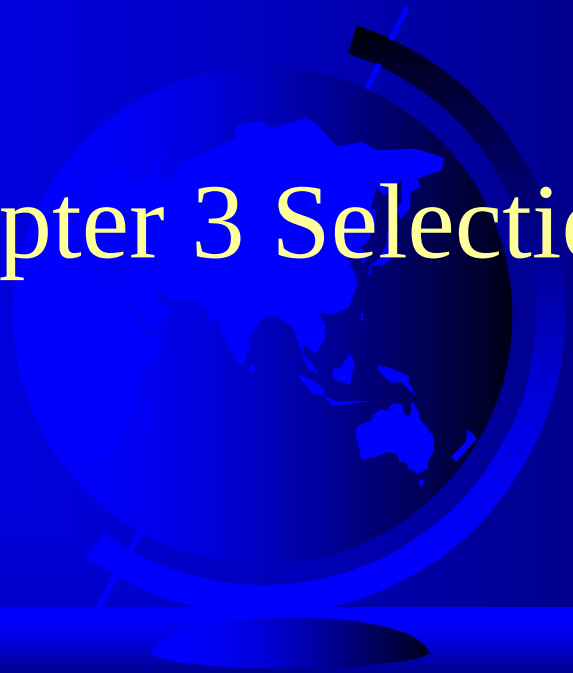
Chapter 3 Selections



布尔类型（boolean）

程序中常常需要做各种比较，例如i是不是大于j。Java提供了6种比较运算符（或者叫关系运算符）。比较运算符的计算结果是一个布尔值：true or false。

```
boolean b = (1 > 2);
```

LISTING 3.1 AdditionQuiz.java

布尔类型可以直接输出

```
1  import java.util.Scanner;
2
3  public class AdditionQuiz {
4    public static void main(String[] args) {
5      int number1 = (int)(System.currentTimeMillis() % 10);
6      int number2 = (int)(System.currentTimeMillis() / 7 % 10);
7
8      // Create a Scanner
9      Scanner input = new Scanner(System.in);
10
11     System.out.print(
12       "What is " + number1 + " + " + number2 + "? ");
13
14     int number = input.nextInt();
15
16     System.out.println(
17       number1 + " + " + number2 + " = " + answer + " is " +
18       (number1 + number2 == answer));
19   }
20 }
```

generate number1

generate number2

show question

display result

```
What is 1 + 7? 8 Enter
1 + 7 = 8 is true
```

```
What is 4 + 8? 9 Enter
4 + 8 = 9 is false
```

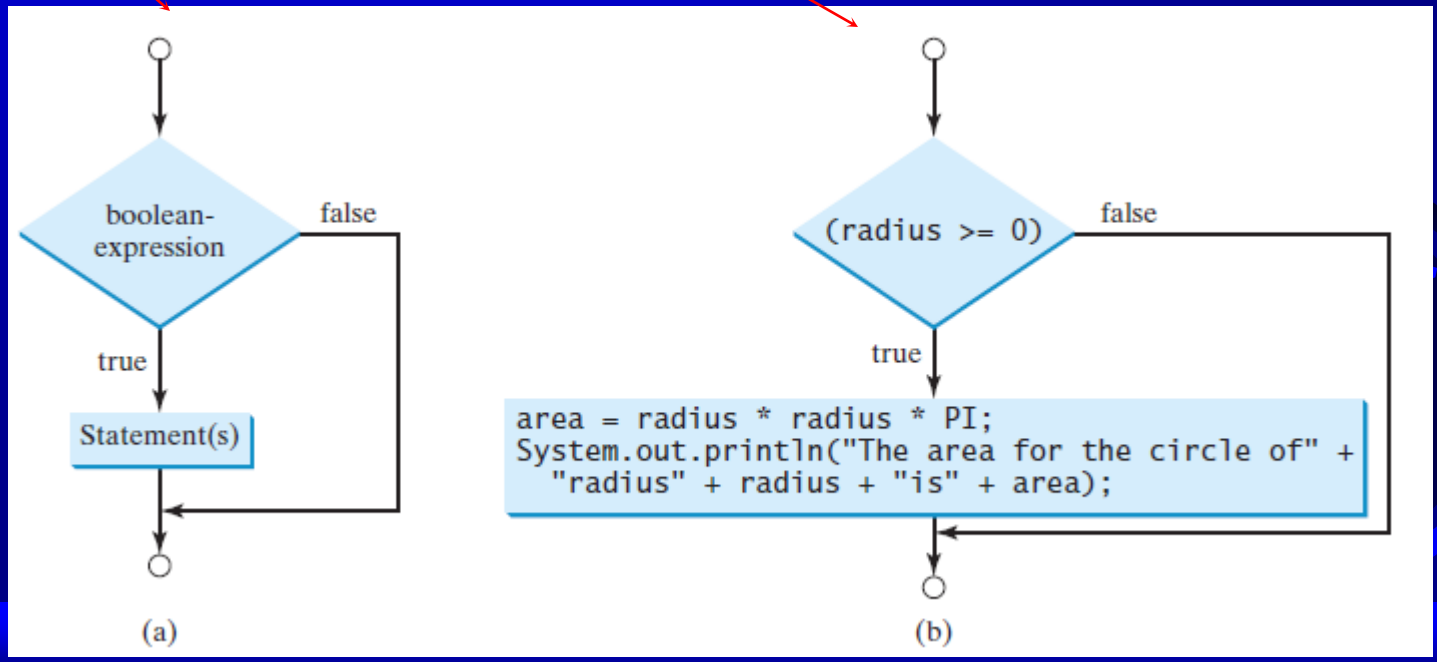
比较运算符

Operator	*Name*
<	小于
<=	小于等于
>	大于
>=	大于等于
==	等于
!=	不等于

单分支if语句

```
if (boolean-expression) {
  statement(s);
}
```

```
if (radius >= 0) {
  area = radius * radius * PI;
  System.out.println("The area"
    + " for the circle of radius "
    + radius + " is " + area);
}
```



注意

- if后面一定要有小括号

```
if i > 0 {
  System.out.println("i is positive");
}
```

(a) Wrong

```
if (i > 0) {
  System.out.println("i is positive");
}
```

(b) Correct

- 如果if后面只有一条语句，可以不用大括号。一个建议是，养成随手加大括号的习惯，哪怕只有一条语句。

```
if (i > 0) {
  System.out.println("i is positive");
}
```

(a)

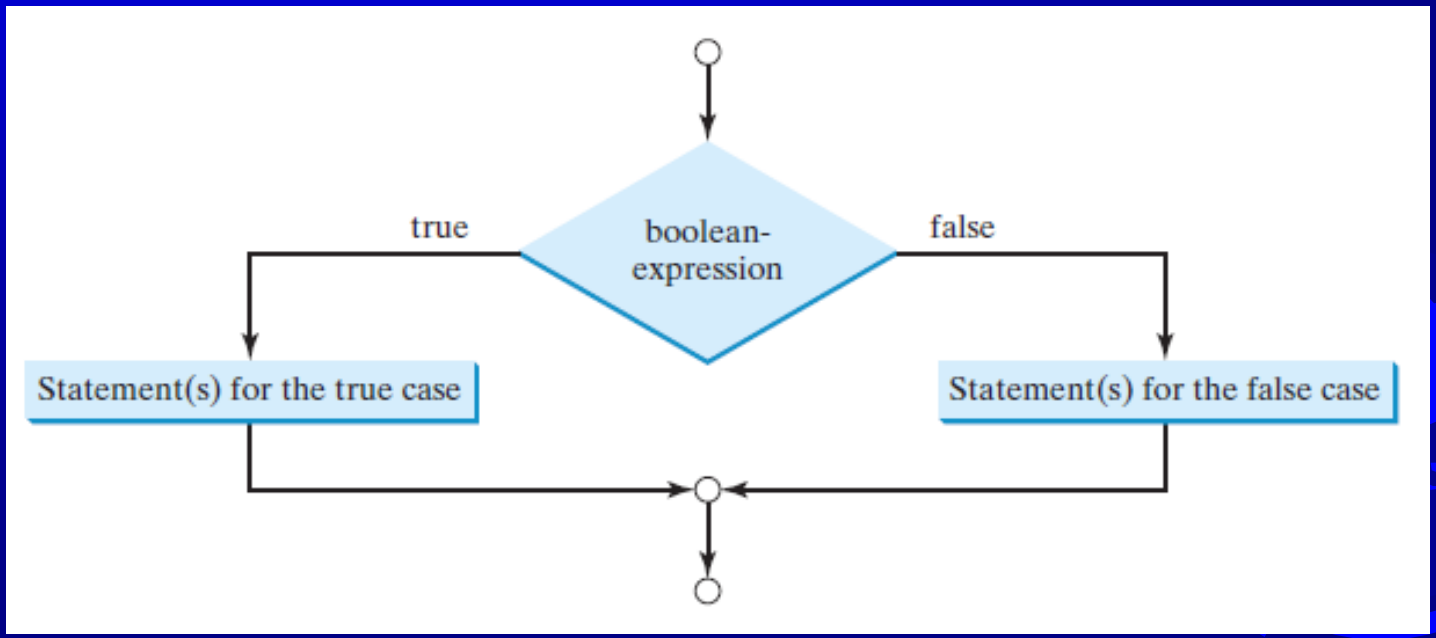
Equivalent

```
if (i > 0)
  System.out.println("i is positive");
```

(b)

双分支if语句

```
if (boolean-expression) {
  statement(s)-for-the-true-case;
}
else {
  statement(s)-for-the-false-case;
}
```



if...else的例子

```
if (radius >= 0) {
  area = radius * radius * 3.14159;

  System.out.println("The area for the "
    + "circle of radius " + radius +
    " is " + area);
}
else {
  System.out.println("Negative input");
}
```

多分支if语句

```
if (score >= 90.0)
  grade = 'A';
else
  if (score >= 80.0)
    grade = 'B';
  else
    if (score >= 70.0)
      grade = 'C';
    else
      if (score >= 60.0)
        grade = 'D';
      else
        grade = 'F';
```

Equivalent

```
if (score >= 90.0)
  grade = 'A';
else if (score >= 80.0)
  grade = 'B';
else if (score >= 70.0)
  grade = 'C';
else if (score >= 60.0)
  grade = 'D';
else
  grade = 'F';
```

单步执行一下

Suppose score is 70.0

The condition is false

```
if (score >= 90.0)
  grade = 'A';
else if (score >= 80.0)
  grade = 'B';
else if (score >= 70.0)
  grade = 'C';
else if (score >= 60.0)
  grade = 'D';
else
  grade = 'F';
```

单步执行一下

Suppose score is 70.0

The condition is false

```
if (score >= 90.0)
  grade = 'A';
else if (score >= 80.0)
  grade = 'B';
else if (score >= 70.0)
  grade = 'C';
else if (score >= 60.0)
  grade = 'D';
else
  grade = 'F';
```

单步执行一下

Suppose score is 70.0

The condition is true

```
if (score >= 90.0)
  grade = 'A';
else if (score >= 80.0)
  grade = 'B';
else if (score >= 70.0)
  grade = 'C';
else if (score >= 60.0)
  grade = 'D';
else
  grade = 'F';
```

单步执行一下

Suppose score is 70.0

grade is C

```
if (score >= 90.0)
  grade = 'A';
else if (score >= 80.0)
  grade = 'B';
else if (score >= 70.0)
  grade = 'C';
else if (score >= 60.0)
  grade = 'D';
else
  grade = 'F';
```

单步执行一下

Suppose score is 70.0

```
if (score >= 90.0)
  grade = 'A';
else if (score >= 80.0)
  grade = 'B';
else if (score >= 70.0)
  grade = 'C';
else if (score >= 60.0)
  grade = 'D';
else
  grade = 'F';
```

Exit the if statement

注意

每一个else一定有一个配对的if，配对的原则是就近匹配。

```
int i = 1;
int j = 2;
int k = 3;

if (i > j)
  if (i > k)
    System.out.println("A");
else
    System.out.println("B");
```

(a)

Equivalent

```
int i = 1;
int j = 2;
int k = 3;

if (i > j)
  if (i > k)
    System.out.println("A");
  else
    System.out.println("B");
```

(b)

常见的if错误－1

- 忘记加大括号

```
if (radius >= 0)
  area = radius * radius * PI;
  System.out.println("The area "
    + " is " + area);
```

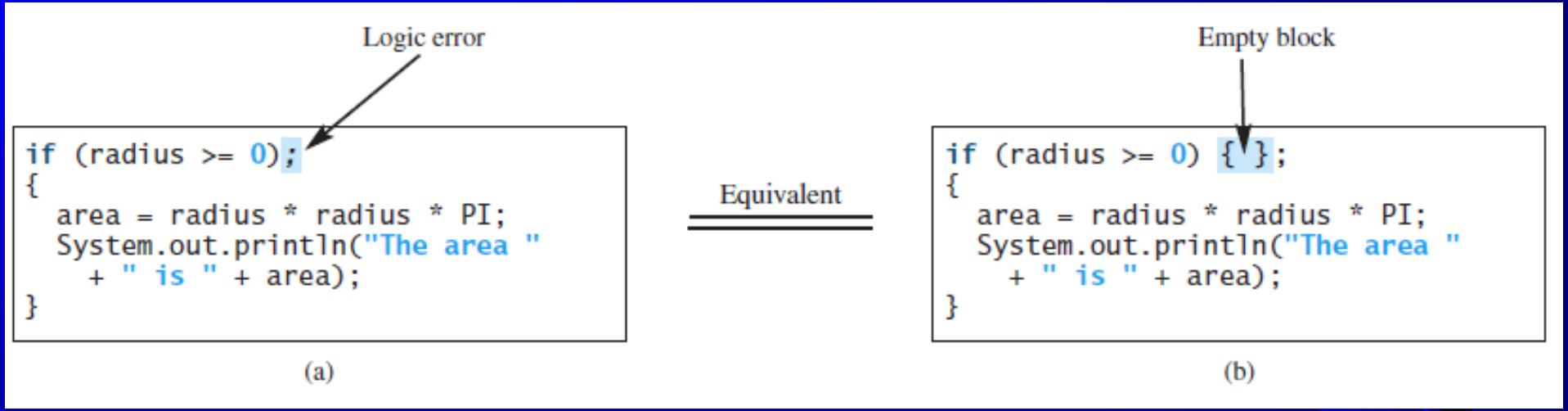
(a) Wrong

```
if (radius >= 0) {
  area = radius * radius * PI;
  System.out.println("The area "
    + " is " + area);
}
```

(b) Correct

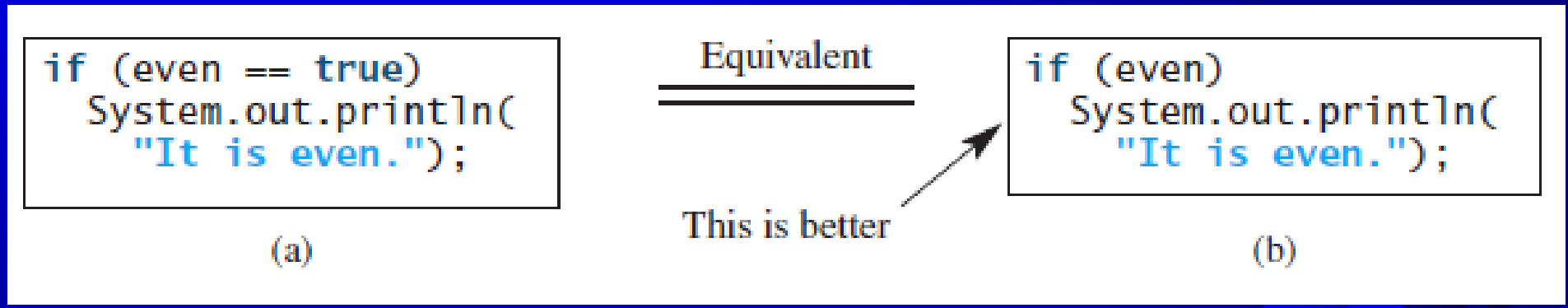
常见的if错误—2

- 乱加分号



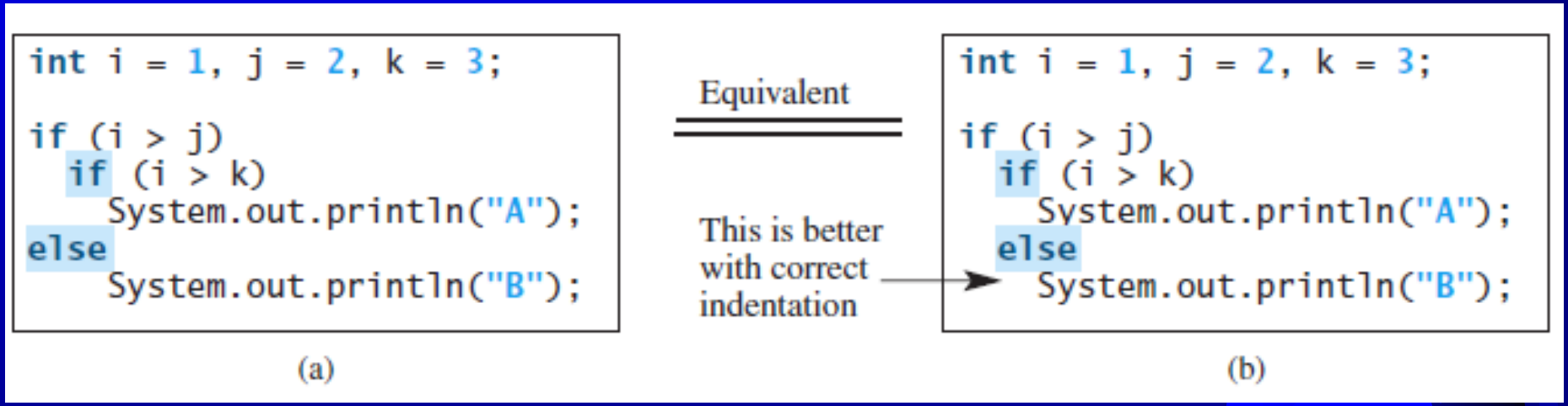
常见的if错误—3

- 冗余的判断



常见的if错误—4

- 受排版误导，搞错了if…else的配对关系



逻辑运算符

Operator	*Name*
!	非
&&	与
||	或
^	异或

非运算!真值表

p	!p	Example (assume age = 24, gender = 'M')
`true`	`false`	!(age > 18) is false, because (age > 18) is true.
`false`	`true`	!(gender != 'F') is true, because (grade != 'F') is false.

与运算&&真值表

p1	p2	p1 && p2	Example (assume age = 24, gender = 'F')
false	false	false	(age > 18) && (gender == 'F') is true, because (age > 18) and (gender == 'F') are both true.
false	true	false	
true	false	false	(age > 18) && (gender != 'F') is false, because (gender != 'F') is false.
true	true	true	

或运算||真值表

p1	p2	p1 || p2	Example (assume age = 24, gender = 'F')
false	false	false	(age > 34) || (gender == 'F') is true, because (gender == 'F') is true.
false	true	true	
true	false	true	(age > 34) || (gender == 'M') is false, because (age > 34) and (gender == 'M') are both false.
true	true	true	

异或运算^真值表

p1	p2	p1 ^ p2	Example (assume age = 24, gender = 'F')
false	false	false	(age > 34) ^ (gender == 'F') is true, because (age > 34) is false but (gender == 'F') is true.
false	true	true	
true	false	true	(age > 34) || (gender == 'M') is false, because (age > 34) and (gender == 'M') are both false.
true	true	false	

例子

```
System.out.println("Is " + number + " divisible by 2 and 3? " +
  ((number % 2 == 0) && (number % 3 == 0)));

System.out.println("Is " + number + " divisible by 2 or 3? " +
  ((number % 2 == 0) || (number % 3 == 0)));

System.out.println("Is " + number +
   " divisible by 2 or 3, but not both? " +
  ((number % 2 == 0) ^ (number % 3 == 0)));
```

例题：闰年的判断

可以简单使用下式判断：

boolean isLeapYear = (year % **4** == **0** && year % **100** != **0**) || (year % **400** == **0**);

Switch语句

```
switch (switch-expression) {
    case value1: statement(s)1;
                 break;
    case value2: statement(s)2;
                 break;
    ...
    case valueN: statement(s)N;
                 break;
    default: statement(s)-for-default;
}
```

Switch语句

表达式的结果必须是char, byte, short, 或int，并且必须放在小括号中。

case的值必须是常量，并且和switch中的表达式类型匹配。

```
switch (switch-expression) {
  case value1:  statement(s)1;
          break;
  case value2: statement(s)2;
          break;
  …
  case valueN: statement(s)N;
          break;
  default: statement(s)-for-default;
}
```

Switch语句

break是可选的，如果没有break，程序将继续执行下一个case的语句，而不会提前跳出switch。

default是可选的，作为所有case都不匹配的时候的分支入口。

```
switch (switch-expression) {
  case value1:  statement(s)1;
          break;
  case value2: statement(s)2;
          break;
  …
  case valueN: statement(s)N;
          break;
  default: statement(s)-for-default;
}
```

单步执行一下

Suppose ch is 'a':

```
switch (ch) {
  case 'a': System.out.println(ch);
  case 'b': System.out.println(ch);
  case 'c': System.out.println(ch);
}
```

单步执行一下

ch is 'a':

```
switch (ch) {
   case 'a': System.out.println(ch);
   case 'b': System.out.println(ch);
   case 'c': System.out.println(ch);
}
```

单步执行一下

Execute this line

```
switch (ch) {
  case 'a': System.out.println(ch);
  case 'b': System.out.println(ch);
  case 'c': System.out.println(ch);
}
```

单步执行一下

Execute this line

```
switch (ch) {
  case 'a': System.out.println(ch);
  case 'b': System.out.println(ch);
  case 'c': System.out.println(ch);
}
```

单步执行一下

Execute this line

```
switch (ch) {
  case 'a': System.out.println(ch);
  case 'b': System.out.println(ch);
  case 'c': System.out.println(ch);
}
```

单步执行一下

Execute next statement

```
switch (ch)
  case 'a': System.out.println(ch);
  case 'b': System.out.println(ch);
  case 'c': System.out.println(ch);
}
```

Next statement;

单步执行一下

Suppose ch is 'a':

```
switch (ch) {
  case 'a': System.out.println(ch);
            break;
  case 'b': System.out.println(ch);
            break;
  case 'c': System.out.println(ch);
}
```

单步执行一下

ch is 'a':

```
switch (ch) {
  case 'a': System.out.println(ch);
            break;
  case 'b': System.out.println(ch);
            break;
  case 'c': System.out.println(ch);
}
```

单步执行一下

Execute this line

```
switch (ch) {
  case 'a': System.out.println(ch);
            break;
  case 'b': System.out.println(ch);
            break;
  case 'c': System.out.println(ch);
}
```

单步执行一下

Execute this line

```
switch (ch) {
  case 'a': System.out.println(ch);
            break;
  case 'b': System.out.println(ch);
            break;
  case 'c': System.out.println(ch);
}
```

单步执行一下

Execute next statement

```
switch (ch)
  case 'a': System.out.println(ch);
            break;
  case 'b': System.out.println(ch);
            break;
  case 'c': System.out.println(ch);
}
```

Next statement;

条件表达式

```
if (x > 0)
  y = 1
else
  y = -1;
```

等价于：

```
y = (x > 0) ? 1 : -1;
```

条件表达式的语法（小括号不是必须的，不过为了可读性最好加上）：

(boolean-expression) ? expression1 : expression2

条件运算符可以简化代码

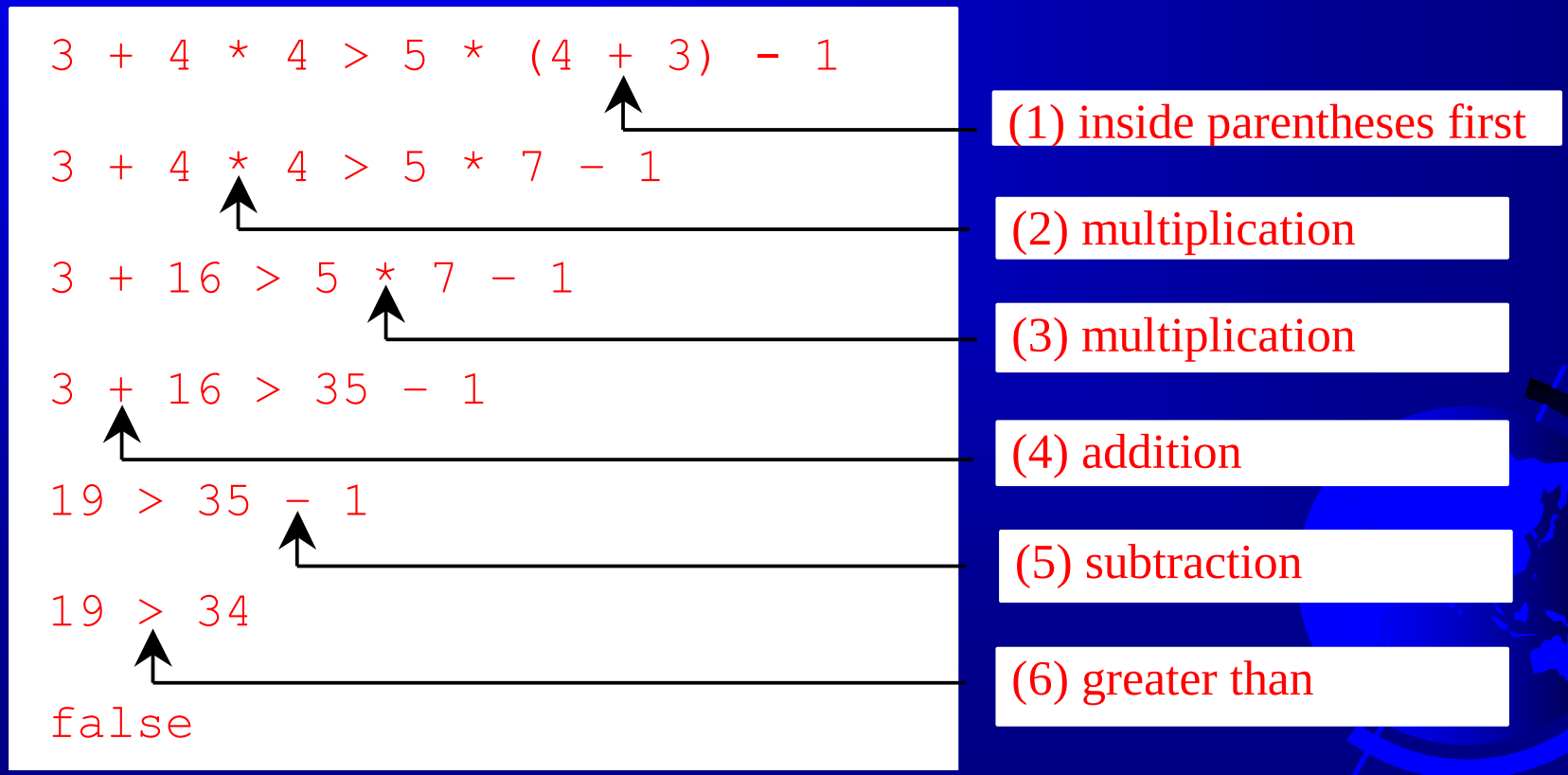
```
if (num % 2 == 0)
   System.out.println(num + "is even");
else
   System.out.println(num + "is odd");


System.out.println(
   (num % 2 == 0)? num + "is even" :
   num + "is odd");
```

运算符优先级（优先级逐行降低）

- var++, var--
- +, -(正负号), ++var,--var
- (type) Casting
- ! (Not)
- *, /, %
- +, - (加减法)
- <, <=, >, >=
- ==, !=
- ^
- &&
- ||
- =, +=, -=, *=, /=, %=

一个例子

分析下面表达式的计算过程：



THE END