Chapter 3 Selections

Liang, Introduction to Java
Programming, Eleventh Edition, (c)
2017 Pearson Education, Inc. All

Relational Operators

Java Operator	Mathematics Symbol	Name	Example (radius is 5)	Result
<	<	less than	radius < 0	false
<=	≤	less than or equal to	radius <= 0	false
>	>	greater than	radius > 0	true
>=	≥	greater than or equal to	radius >= 0	true
==	=	equal to	radius == 0	false
!=	≠	not equal to	radius != 0	true

Uploaded By: Jibreel Bornat

The `boolean` Type and Operators

Often in a program you need to compare two values, such as whether i is greater than j. Java provides six comparison operators (also known as relational operators) that can be used to compare two values. The result of the comparison is a Boolean value: true or false.

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

Uploaded By: Jibreel Bornat   3
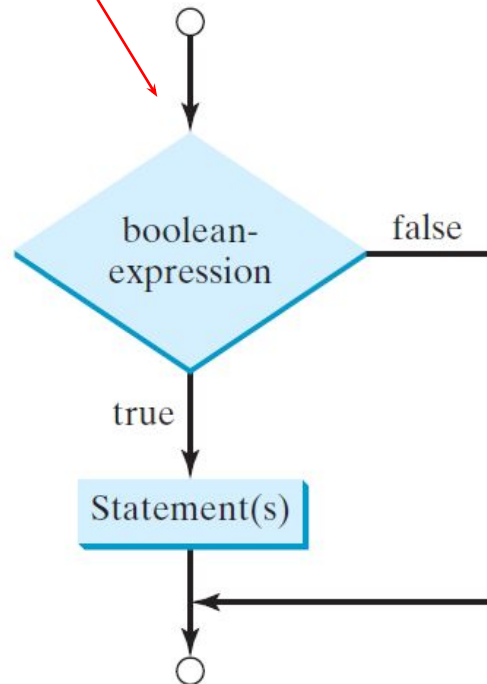
Problem: A Simple Math Learning Tool

This example creates a program to let a first grader practice additions. The program randomly generates two single-digit integers number1 and number2 and displays a question such as "What is 7 + 9?" to the student. After the student types the answer, the program displays a message to indicate whether the answer is true or false.
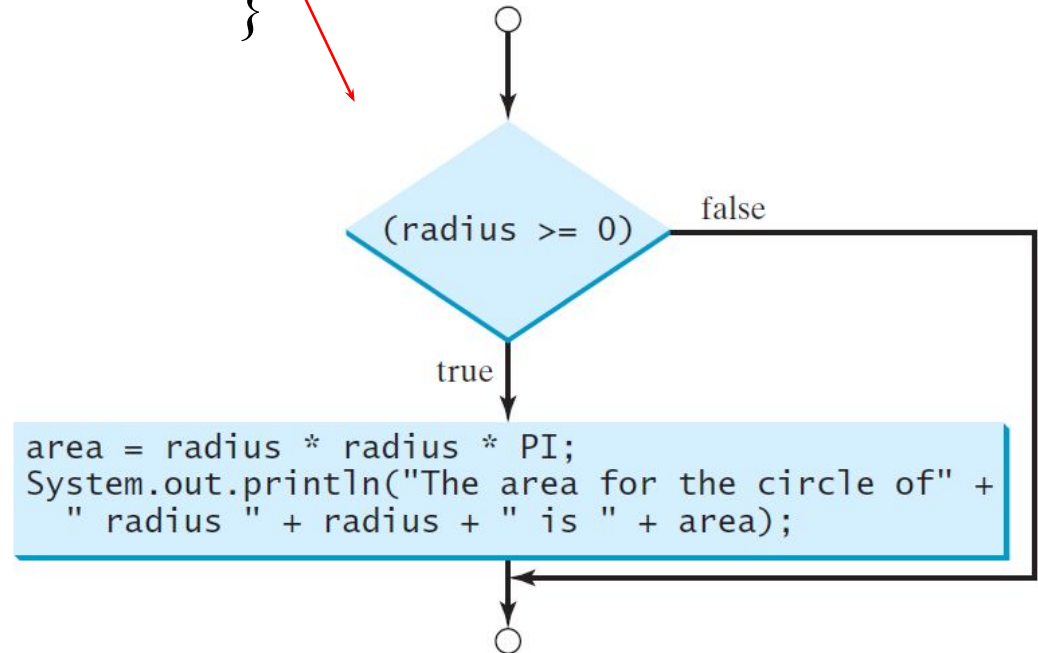
AdditionQuiz | Run

Uploaded By: Jibreel Bornat

One-way `if` Statements

if (radius >= 0) {

    area = radius * radius * PI;

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

    System.out.println("The area"

      + " for the circle of radius "

      + radius + " is " + area);

}


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

Uploaded By: Jibreel Bornat

Simple if Demo

Write a program that prompts the user to enter an integer. If the number is a multiple of 5, print HiFive. If the number is divisible by 2, print HiEven.

SimpleIfDemo   Run

Uploaded By: Jibreel Bornat   6

The Two-way `if` Statement

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


 Uploaded By: Jibreel Bornat   7
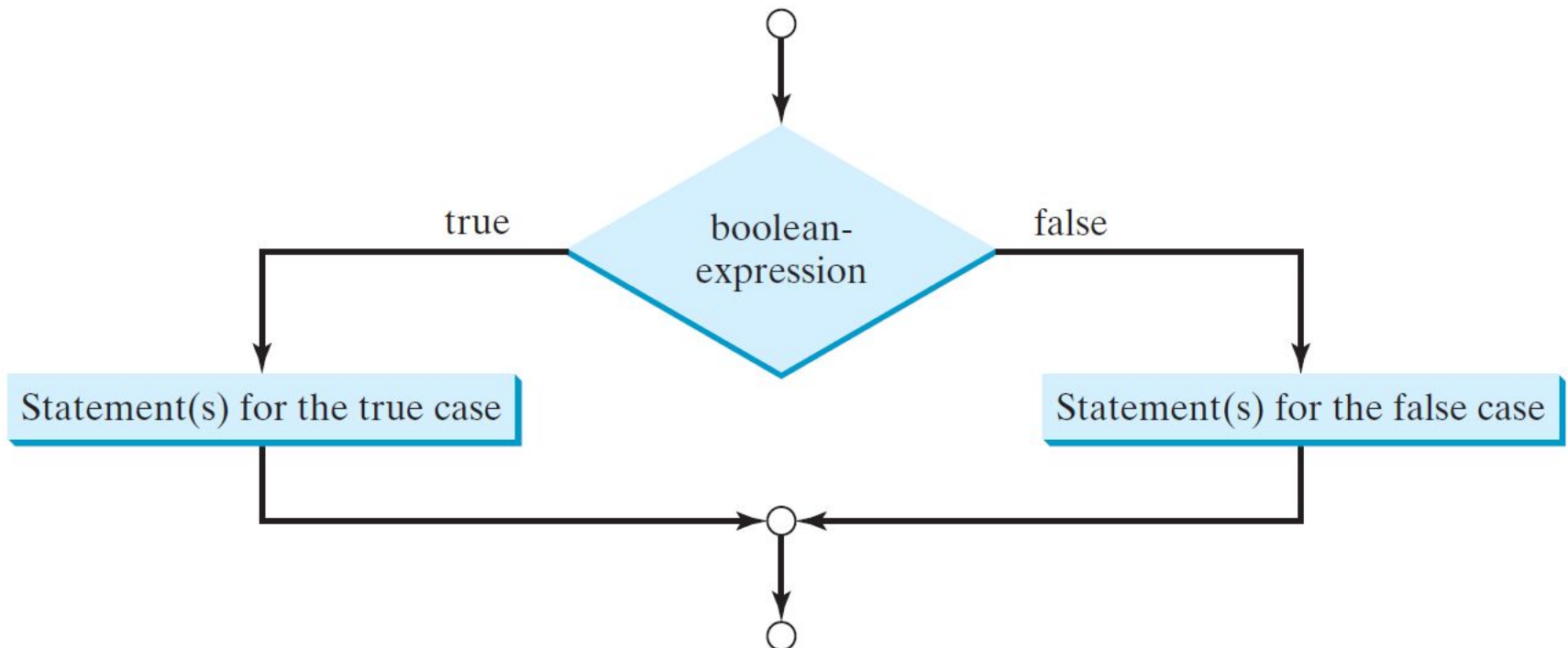
if-else Example

```
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");
}
```

Uploaded By: Jibreel Bornat   8

Multiple Alternative if Statements

```
if (score >= 90.0)
  System.out.print("A");
else
  if (score >= 80.0)
    System.out.print("B");
  else
    if (score >= 70.0)
      System.out.print("C");
    else
      if (score >= 60.0)
        System.out.print("D");
      else
        System.out.print("F");
```

(a)

Equivalent

This is better

```
if (score >= 90.0)
  System.out.print("A");
else if (score >= 80.0)
  System.out.print("B");
else if (score >= 70.0)
  System.out.print("C");
else if (score >= 60.0)
  System.out.print("D");
else
  System.out.print("F");
```

(b)

TIP

```
if (number % 2 == 0)
   even = true;
else
   even = false;
```
(a)

Equivalent

```
boolean even
   = number % 2 == 0;
```
(b)

 Uploaded By: Jibreel Bornat 10

CAUTION

BIRZEIT UNIVERSITY

```
if (even == true)
   System.out.println(
     "It is even.");
```
(a)

Equivalent

```
if (even)
   System.out.println(
     "It is even.");
```
(b)

Uploaded By: Jibreel Bornat

Problem: An Improved Math Learning Tool

This example creates a program to teach a first grade child how to learn subtractions. The program randomly generates two single-digit integers number1 and number2 with number1 >= number2 and displays a question such as "What is 9 – 2?" to the student. After the student types the answer, the program displays whether the answer is correct.

SubtractionQuiz   Run

STUDENTS-HUB.com

Logical Operators

Operator	Name	Description
!	not	logical negation
&&	and	logical conjunction
||	or	logical disjunction
^	exclusive or	logical exclusion

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

13

Truth Table for Operator !

p	!p	Example (assume age = 24, weight = 140)
true	false	!(age > 18) is false, because (age > 18) is true.
false	true	!(weight == 150) is true, because (weight == 150) is false.

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

14

Truth Table for Operator &&

p_1	p_2	p_1 && p_2	Example (assume age = 24, weight = 140)
false	false	false	(age <= 18) && (weight < 140) is false, because both conditions are both false.
false	true	false	
true	false	false	(age > 18) && (weight > 140) is false, because (weight > 140) is false.
true	true	true	(age > 18) && (weight >= 140) is true, because both (age > 18) and (weight >= 140) are true.

Uploaded By: Jibreel Bornat

Truth Table for Operator ||

p_1	p_2	p_1 || p_2	Example (assume age = 24, weihgt = 140)
false	false	false	
false	true	true	(age > 34) || (weight <= 140) is true, because (age > 34) is false, but (weight <= 140) is true.
true	false	true	(age > 14) || (weight >= 150) is false, because (age > 14) is true.
true	true	true	

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

Truth Table for Operator ^

p_1	p_2	$p_1 \wedge p_2$	Example (assume age = 24, weight = 140)
false	false	false	(age > 34) ^ (weight > 140) is true, because (age > 34) is false and (weight > 140) is false.
false	true	true	(age > 34) ^ (weight >= 140) is true, because (age > 34) is false but (weight >= 140) is true.
true	false	true	(age > 14) ^ (weight > 140) is true, because (age > 14) is true and (weight > 140) is false.
true	true	false	

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

17

Examples

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)));

TestBooleanOperators

Run

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

Problem: Determining Leap Year?

This program first prompts the user to enter a year as an <u>int</u> value and checks if it is a leap year.

A year is a leap year if it is divisible by 4 but not by 100, or it is divisible by 400.

 (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)

LeapYear   Run

Uploaded By: Jibreel Bornat

`switch` Statement Rules

The <u>switch-expression</u> must yield a value of <u>char</u>, <u>byte</u>, <u>short</u>, or <u>int</u> type and must always be enclosed in parentheses.

The <u>value1</u>, ..., and <u>valueN</u> must have the same data type as the value of the <u>switch-expression</u>. The resulting statements in the <u>case</u> statement are executed when the value in the <u>case</u> statement matches the value of the <u>switch-expression</u>. Note that <u>value1</u>, ..., and <u>valueN</u> are constant expressions, meaning that they cannot contain variables in the expression, such as 1 + <u>x</u>.

```
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;
}
```

Uploaded By: Jibreel Bornat

`switch` Statement Rules

The keyword <u>break</u> is optional, but it should be used at the end of each case in order to terminate the remainder of the <u>switch</u> statement. If the <u>break</u> statement is not present, the next <u>case</u> statement will be executed.

The <u>default</u> case, which is optional, can be used to perform actions when none of the specified cases matches the <u>switch-expression</u>.

```
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;
}
```

When the value in a **case** statement matches the value of the **switch-expression**, the statements *starting from this case* are executed until either a **break** statement or the end of the **switch** statement is reached.

animation

Trace switch statement

Suppose day is 2:

```
switch (day) {
  case 1:
  case 2:
  case 3:
  case 4:
  case 5: System.out.println("Weekday"); break;
  case 0:
  case 6: System.out.println("Weekend");
}
```

STUDENTS-HUB.com

Liang, Introduction to Java Programming, Eleventh Edition, (c) 2017 Pearson Education, Inc. All rights reserved.

Uploaded By: Jibreel Bornat

22

Conditional Operators

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

is equivalent to

y = (x > 0) ? 1 : -1;
(boolean-expression) ? expression1 : expression2

Uploaded By: Jibreel Bornat

Conditional Operator

```java
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");
```

 Uploaded By: Jibreel Bornat

Operator Precedence

- `var++, var--`
- `+, - (Unary plus and minus), ++var,--var`
- `(type) Casting`
- `! (Not)`
- `*, /, % (Multiplication, division, and remainder)`
- `+, - (Binary addition and subtraction)`
- `<, <=, >, >= (Relational operators)`
- `==, !=; (Equality)`
- `^ (Exclusive OR)`
- `&& (Conditional AND) Short-circuit AND`
- `|| (Conditional OR) Short-circuit OR`
- `=, +=, -=, *=, /=, %= (Assignment operator)`

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

Operator Precedence and Associativity

The expression in the parentheses is evaluated first. (Parentheses can be nested, in which case the expression in the inner parentheses is executed first.) When evaluating an expression without parentheses, the operators are applied according to the precedence rule and the associativity rule.

If operators with the same precedence are next to each other, their associativity determines the order of evaluation. All binary operators except assignment operators are left-associative.

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

Operator Associativity

When two operators with the same precedence are evaluated, the *associativity* of the operators determines the order of evaluation. All binary operators except assignment operators are *left-associative*.

a – b + c – d is equivalent to ((a – b) + c) – d

Assignment operators are *right-associative*. Therefore, the expression

a = b += c = 5 is equivalent to a = (b += (c = 5))

 Uploaded By: Jibreel Bornat

Example

Applying the operator precedence and associativity rule, the expression 3 + 4 * 4 > 5 * (4 + 3) - 1 is evaluated as follows:

```
3 + 4 * 4 > 5 * (4 + 3) - 1        (1) inside parentheses first

3 + 4 * 4 > 5 * 7 - 1              (2) multiplication

3 + 16 > 5 * 7 - 1                 (3) multiplication

3 + 16 > 35 - 1                    (4) addition

19 > 35 - 1                        (5) subtraction

19 > 34                            (6) greater than

false
```