Chapter 4

Control Structures I

Chapter Objectives

• Learn about control structures
• Examine relational and logical operators
• Explore how to form and evaluate logical (Boolean) expressions
• Learn how to use the selection control structures if, if…else, and switch in a program

Control Structures

• Three methods of processing a program
  – In sequence
  – Branching
  – Looping
• Branch: Altering the flow of program execution by making a selection or choice
• Loop: Altering the flow of program execution by repetition of statement(s)
Relational Operators

- Relational Operator
  - Allows you to make comparisons in a program
  - Binary operator
- Condition is represented by a logical expression in Java
- Logical expression: expression that has a value of either true or false

Relational Operators in Java

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>==</code></td>
<td>equal to</td>
</tr>
<tr>
<td><code>!=</code></td>
<td>not equal to</td>
</tr>
<tr>
<td><code>&lt;</code></td>
<td>less than</td>
</tr>
<tr>
<td><code>&lt;=</code></td>
<td>less than or equal to</td>
</tr>
<tr>
<td><code>&gt;</code></td>
<td>greater than</td>
</tr>
<tr>
<td><code>&gt;=</code></td>
<td>greater than or equal to</td>
</tr>
</tbody>
</table>
Relational Operators and Primitive Data Types

- Can be used with integral and floating-point data types
- Can be used with the char data type
- Unicode Collating Sequence

Relational Operators and the Unicode Collating Sequence

<table>
<thead>
<tr>
<th>Expression</th>
<th>Value of the Expression</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>'&lt;' &lt; 'a'</td>
<td>true</td>
<td>The Unicode value of 'a' is 97. Because 32 &lt; 97 is true, it follows that '&lt;' &lt; 'a' is true.</td>
</tr>
<tr>
<td>'b' &gt; 'T'</td>
<td>false</td>
<td>The Unicode value of 'b' is 98, and the Unicode value of 'T' is 84. Because 98 &gt; 84 is False, it follows that 'b' &gt; 'T' is false.</td>
</tr>
<tr>
<td>'+' &lt; '+'</td>
<td>false</td>
<td>The Unicode value of '+' is 43, and the Unicode value of '+' is 43. Because 43 &lt; 43 is False, it follows that '+' &lt; '+' is False.</td>
</tr>
<tr>
<td>'G' &lt;= 'G'</td>
<td>true</td>
<td>The Unicode value of 'G' is 71, and the Unicode value of 'G' is 71. Because 71 &lt;= 71 is true, it follows that 'G' &lt;= 'G' is true.</td>
</tr>
</tbody>
</table>

Comparing Strings

- class String
  - Method compareTo
  - Method equals
- Given string str1 and str2
  
  integer < 0 if str1 < str2
  Str1.compareTo(str2) = \{ 0 if str1 = str2
  integer > 0 if str1 > str2
### AND and OR Operators

<table>
<thead>
<tr>
<th>Expression1</th>
<th>Expression2</th>
<th>Expression1 &amp; Expression2</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Expression1</th>
<th>Expression2</th>
<th>Expression1</th>
<th>Expression2</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

### Logical (Boolean) Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>not</td>
</tr>
<tr>
<td>&amp;</td>
<td>and</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The ! (not) Operator

<table>
<thead>
<tr>
<th>Expression</th>
<th>!Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
</tr>
</tbody>
</table>

### Precedence of Operators

<table>
<thead>
<tr>
<th>Operators</th>
<th>Precedence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, +, -, (unary)</td>
<td>first</td>
</tr>
<tr>
<td>*, /, %</td>
<td>second</td>
</tr>
<tr>
<td>+, =</td>
<td>third</td>
</tr>
<tr>
<td>&lt;, &lt;=, &gt;, &gt;=</td>
<td>fourth</td>
</tr>
<tr>
<td>==, !=</td>
<td>fifth</td>
</tr>
<tr>
<td>&amp;&amp;</td>
<td>sixth</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>= (assignment)</td>
<td>last</td>
</tr>
</tbody>
</table>
Short-Circuit Evaluation

- Definition: a process in which the computer evaluates a logical expression from left to right and stops as soon as the value of the expression is known

Selection

- One-Way Selection
- Two-Way Selection
- Compound (Block of) Statements
- Multiple Selections (Nested if)
- Conditional Operator
- switch Structures

One-Way Selection

- Syntax: if(expression) statement
- Expression referred to as decision maker
- Statement referred to as action statement
Two-Way Selection

- Syntax: `if(expression)
    statement1
else
    statement2`
- else statement must be paired with an if

Compound (Block of) Statements

- Syntax
  ```
  { 
    statement1
    statement2
    ...
    statementn
  }
  ```
Multiple Selection: Nested if

- Syntax
  
  ```java
  if(expression1)
    statement1
  else
    if(expression2)
      statement2
    else
      statement3
  ```

- Else associated with most recent incomplete if
- Multiple if statements can be used in place of if...else statements
- May take longer to evaluate

Conditional (? :) Operator

- Ternary operator
- Syntax
  
  ```java
  expression1 ? expression2 : expression3
  ```

- If expression1 = true, then the result of the condition is expression 2
- Otherwise, the result of the condition is expression3

switch Structures

```java
switch(expression)
{
  case value1: statements1
    break;
  case value2: statements2
    break;
  ...
  case valuen: statementsn
    break;
  default: statements
}
```
### Programming Example: Cable Company Billing

- **Input:** customer’s account number, customer code, number of premium channels to which customer subscribes, number of basic service connections (in case of business customers)
- **Output:** customer’s account number and the billing amount

### Programming Example: Cable Company Billing

- **Solution:**
  - Prompt user for information
  - Use switch statements based on customer’s type
  - Use an if statement nested within switch statement to determine amount due by each customer
Chapter Summary

• Control structures are used to process programs
• Logical expressions and order of precedence of operators are used in expressions
• Compare strings
• If statements
• If…else statements
• Switch structures
• Proper syntax for using control statements