COMP 110-003
Introduction to Programming
Branding Statements and Boolean Expressions

January 29, 2013

Haohan Li
TR 11:00 – 12:15, SN 011
Spring 2013
Announcements

• Lab 1 grading and comments on Sakai
• Office hour for Wednesday Jan. 30
  – 1:30PM – 3:30PM
Today

• Review worksheet
• Formatting decimals
• If/Else statements
• Boolean Expressions
Review Worksheets

• Print
  – `System.out.println(“COMP110 is my favorite class”);`

• Read input
  – `Scanner keyboard = new Scanner(System.in);`
  – `int myInt = keyboard.nextInt();`
Declare a Variable

- Declare a variable of type float with the identifier, *myFloat*, and initialize the value to 4.6
  - `float myFloat = 4.6;`
  - `float myFloat;`
  - `myFloat = 4.6;`
public class MyProgram
{
    public static void main(String[] args)
    {
        String myString = "This is a string";
        int len = myString.length();
        System.out.println("the length is " + len);
        String shortString = myString.substring(10);
    }
}
Integer Division

- `double myDouble = (1 / 2) * 5.0;`
- It means:
  - `int temp = (1 / 2);`
    - Because 1 and 2 are both integers, the value type of `1/2` is also an integer
    - Its value should be the integer part of 0.5, which is 0
  - `double myDouble = (double) temp * 5.0;`
    - Because 5.0 is a double, then temp is casted to double
    - However, the result will still be 0.0
Floating-Point Division

- `double myDouble = ( 1.0 / 2.0 ) * 5.0;`
- It means:
  - `double temp = ( 1.0 / 2.0 );`
    - Because 1 and 2 are both floating-points, the return type of `1 / 2` is also a floating-point
    - Its value should be 0.5
  - `double myDouble = temp * 5.0;`
    - The result will still be 2.5
char Type

• ‘x’ represents a character in char type
  • char a, b;
  • a = ‘b’; // assign the value ‘b’ to char variable a
  • System.out.println(a);
  • b = ‘c’; // assign the value ‘c’ to char variable b
  • System.out.println(b);
  • a = b; // assign the value of char variable b (which is ‘c’) to
    // the value of char variable a (which was ‘b’)
  • System.out.println(a); // the value of a is ‘c’ now

– Output would be: b, c, c
Suppose that *mary* is an object of class *Person*, and suppose that *increaseAge* is a method of class *Person* that uses one argument, an integer. Write the invocation of the method *increaseAge* for the object *mary* using the argument 5.

- Syntax: `ObjectName.Method(arguments);`
- `mary.increaseAge(5);`
Today

- Review worksheet
- **Formatting decimals**
- If/Else statements
- Boolean Expressions
Formatting Decimals

• Use the class `DecimalFormat`
  – `import java.text.DecimalFormat;`
  – `DecimalFormat df = new DecimalFormat("0.00");`
  – `double d = 12.345678;`
  – `System.out.println("my double with two decimal places: " + df.format(d));`
    • The method is called by `df.format(d)`
    • It will output: my double with two decimal places: 12.35
Today

• Review worksheet
• Formatting decimals
• If/Else statements
• Boolean Expressions
Flow Chart

Student.getUp();
if (time < 7) {
    Student.haveBreakfast();
}
else { // time >= 7
    Student.bringBreakfast();
}
Student.takeBus();
import java.util.*;

public class FlowChart {
    public static void main(String[] args) {
        System.out.println("Give me an integer:");
        Scanner keyboard = new Scanner(System.in);
        int inputInt = keyboard.nextInt();
        if (inputInt > 5) {
            System.out.println("Big number");
        } else {
            System.out.println("Small number");
        }
    }
}

What if your input is 5?
# Java Comparison Operators

## FIGURE 3.4 Java comparison operators

<table>
<thead>
<tr>
<th>Math Notation</th>
<th>Name</th>
<th>Java Notation</th>
<th>Java Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>Equal to</td>
<td>==</td>
<td>balance == 0, answer == 'y'</td>
</tr>
<tr>
<td>≠</td>
<td>Not equal to</td>
<td>!=</td>
<td>income != tax, answer != 'y'</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater than</td>
<td>&gt;</td>
<td>expenses &gt; income</td>
</tr>
<tr>
<td>≥</td>
<td>Greater than or equal to</td>
<td>&gt;=</td>
<td>points &gt;= 60</td>
</tr>
<tr>
<td>&lt;</td>
<td>Less than</td>
<td>&lt;</td>
<td>pressure &lt; max</td>
</tr>
<tr>
<td>≤</td>
<td>Less than or equal to</td>
<td>&lt;=</td>
<td>expenses &lt;= income</td>
</tr>
</tbody>
</table>
Boolean Expressions

• Expression?
  – An **expression** can be a variable, a value, or a combination made up by variables, values and operators
  – An expression **has a value**
  – **Arithmetic expression**: a combination of numbers with a number value
    • $10, \frac{\text{taxRate}}{100}, (\text{cost} + \text{tax}) \times \text{discount}$
  – **String expression**: a combination of Strings with a String value
    • “Hello”, “The total cost is ” + totalCost
Boolean Expressions

• A combination of values and variables by comparison operators. Its value can only be true or false

• Example expressions
  – 5 == 3; // false
  – variable <= 6; // depending on the value of variable
    • What if variable is 5? What if variable is 6?
  – myInt != temp; // depending on both values
    • What if myInt is 0 and temp is 2? Am I lying?

• Syntax rule for if statement:
  – if (boolean expression)
    { statements; }
&&: and

• What if you need multiple expressions to be true?
• Syntax rule:
  – `(expression) && (expression) && ...`
    • Expressions go in ( )
  – `(Time < 7) && (I’ve prepared breakfast)`
• Will only be true if **ALL** statements are true
||: or

- What if you need ONE expression to be true out of many expressions
- Syntax rule:
  - `(expression) || (expression) || ...`
    - Again, expressions go in ( )
  - `(I’ve had breakfast) || (Time > 7)`
- Will be true if **ONE** expression is true
!: not

• Syntax rule:
  – !(expression)
    • Again, expressions go in ( )
  – !(I’ve had breakfast)
• Will be true if the expression is false
• ! is not recommended
  – You will get confused. Try to write expressions straightforward
    • Use (cost != 3) instead of !(cost == 3)
    • Use (time <= 7) instead of !(time > 7)
Logical Operators

**FIGURE 3.7** The Effect of the Boolean Operators `&&` (*and*), `||` (*or*), and `!` (*not*) on Boolean Values

<table>
<thead>
<tr>
<th>Value of $A$</th>
<th>Value of $B$</th>
<th>Value of $A &amp;&amp; B$</th>
<th>Value of $A \mid\mid B$</th>
<th>Value of $\neg (A)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>false</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>true</td>
</tr>
</tbody>
</table>
Comparison vs. Logical Operators

• Comparison operators connect values or variables
  – After connection, it’s a boolean expression
  – \( a > b \)
  – \( c == d \)

• Logical operators connect boolean expressions
  – \((a > b) \&\& (c == d)\)
More Complex Boolean Expressions

• Combination of && and ||
  – ( ((3 < 7)|| (2==5) ) && ( (4!=2) && (1 <= 1) ) )
  – ( ( (true)|| (false) ) && ( (true) && (true) )
  – (true) && (true)
  – true

• if ( ( (I’m at Subway) && (You’re at Subway) ) ||
    ( (I’m at Starbucks ) && (You’re at Starbucks) ) )
  {
    I will meet you;
  }
Boolean Variable

- A boolean variable saves a boolean value

```java
boolean systemsAreOK =
    ((temperature <= 100) && (thrust >= 12000) && (cabinPressure > 30));
// You can use "=" to assign a boolean value to a boolean variable
if (systemsAreOK){
    // It's the same as if (systemsAreOK == true)
    System.out.println("Initiate launch sequence.");
}
else{
    System.out.println("Abort launch sequence.");
}
```
Assignment vs. Equal To

- \( if \ ( n1 = n2 ) \)
  - **Error!!!!** It’s an assignment statement!

- \( if \ ( n1 == n2 ) \)
  - Correct. It’s a boolean expression now.
String Comparison

• String comparison
  – `string1 == string2;  //BAD`
  – `string1.equals(string2);  //GOOD`

• Syntax
  – `String.equals(Other_String)`
  – `String.equalsIgnoreCase(Other_String)`
If and Else

• You can use only one if statement
  – *if* (*boolean expression*)
    { *statements*; }
    *other statements*;
  • *Other statements* will always be executed

• You can also use an if-else statement
  – *if* (*boolean expression*)
    { *statements 1*; }
    *else* { *statement 2*; }
  • If the *expression* is true, run *statement 1*, otherwise run *statement 2*
Nested If and Else

```java
if (time < 7){
  if (time < 6){
    cook hams and scramble eggs;
  }
  else{
    grab something from the fridge;
  }
}
else{
  go to school;
}
```

- What’s the logic flow?
  - If the time is smaller than 6, we cook breakfast;
  - If the time is between 6 and 7, we get something cold
  - If the time is greater than 7, we go to school
Nested If and Else

```java
if (time < 6){
    cook hams and scramble eggs;
}
else{
    if (time < 7){
        grab something from the fridge;
    }
    else{
        go to school;
    }
}
```

- What’s the logic flow?
  - If the time is smaller than 6, we cook breakfast;
  - If the time is between 6 and 7, we get something cold
  - If the time is greater than 7, we go to school
Same Logic, Different Code

```c
if (time < 6) {
    cook hams and scramble eggs;
}
else {
    if (time < 7) {
        grab something from the fridge;
    }
    else {
        go to school;
    }
}
```

```c
if (time < 7) {
    if (time < 6) {
        cook hams and scramble eggs;
    }
    else {
        grab something from the fridge;
    }
    else {
        go to school;
    }
}
```
Without Else?

```java
if (time < 6) {
    cook hams and scramble eggs;
}
else {
    if (time < 7) {
        grab something from the fridge;
    }
    else {
        go to school;
    }
}
```

`Exactly the same?`
Without Else?

```java
if (time < 6) {
    cook hams and scramble eggs;
}
else {
    if (time < 7) {
        grab something from the fridge;
    }
    else {
        go to school;
    }
}
```

```java
if (time < 6) {
    cook hams and scramble eggs;
}
else if (time > 6 && time < 7) {
    grab something from the fridge;
}
else if (time > 7) {
    go to school;
}
```

What if time is precisely 7?
if (time < 6){
    cook hams and scramble eggs;
}
else{
    if (time < 7){
        grab something from the fridge;
    }
    else{
        go to school;
    }
}
Using If and Else

• Use if-else statement
• Do not use two if statements
• Always pay attention to boundaries
  – Is it “>” or “>=”?  
  – Is it “<” or “<=”?  
  – Do you need a “==”?
If Thirsty

```plaintext
if ($thirsty==TRUE)
{
}
else
{
}
```