Lecture 7(wed) – Lec 6 (sat) Handouts : Number Systems

topics: review test 1 from last week

discuss Lab 3 due next week: p.188, # 9 Shipping Charges

ch 3 control structures;
logical(boolean) operators and truth tables
comparing strings
the conditional operator
switch (case)
DecimalFormat class

Binary Number System

A. Lab 3
There is no handout for lab 3. It can be found on p. 188, #9, Shipping Charges, with these additions to the program description:
1. The program asks the user for the distance as well as the weight.
2. All input and calculations should be doubles.
3. If the user enters 0 weight and/or 0 distance, the charge should be $0.00.
4. Write a GUI application. There is my demo Applet online to view interaction.

B. Logical (also called Boolean) Operators

relational operators are used to write conditions(boolean expressions)
<  >  <=  >=
==  !=

example: 4 < 5 --> true

Logical operators are used to build complex boolean expressions from simple ones, and are based on the Truth Tables:

<table>
<thead>
<tr>
<th>&amp;&amp; (AND)</th>
<th>RESULT:</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRUE</td>
<td>TRUE</td>
</tr>
<tr>
<td>FALSE</td>
<td>TRUE</td>
</tr>
<tr>
<td>TRUE</td>
<td>FALSE</td>
</tr>
<tr>
<td>FALSE</td>
<td>FALSE</td>
</tr>
</tbody>
</table>
|| (OR) || RESULT: 
| TRUE | TRUE | TRUE |
| TRUE | FALSE | TRUE |
| FALSE | TRUE | TRUE |
| FALSE | FALSE | FALSE |

! (NOT) 
| TRUE | FALSE |
| FALSE | TRUE |

Look at examples.

C. Comparing Strings
Strings are objects in Java, based on the String class (java.lang package)

shorthand syntax to instantiate a String object

// “San Francisco” is called a String literal, a String constant, or an anonymous String object
String firstCity = “San Francisco”;
// create another String object
String secondCity = “Los Angeles”;

The proper way to compare the contents of each String is to use the equals method of the String class. We can call the method on either variable reference:

if ( firstCity.equals(secondCity))

or

if (secondCity.equals(firstCity))

Ignoring case in String comparisons – look at listing 3-9 SecretWord – accepts upper or lower-case passwords.
5. The Conditional Operator

condition? expr1 : expr2;

text examples:

boolean bDone = nCompleted >= 75 ? true : false;
// expression is returned and assigned to bDone

this is equivalent to:

if (nCompleted >= 75)
    bDone = true;
else
    bDone = false;

6. SWITCH - a more elegant alternative to multiple else-ifs

switch only tests equality, not other relations
//case expression – must evaluate to int, short, byte, or char value
switch (case expression) {
    case constant or literal: //can't use variable here
        statement(s);
        break; // required or you drop thru to next case
    default: //optional
        statement(s);
}
How to draw this as a logic diagram:
example:

```java
int nCode = Integer.parseInt(JOptionPane.showInputDialog("Type in code (1 = add, 2 = delete, 3 = find)");
switch (nCode) {
    case 1:
        System.out.println("You entered a 1");
        break;
    case 2:
        System.out.println("You entered a 2");
        break;
    case 3:
        System.out.println("You entered a 3");
        break;
    Default:
        JOptionPane.showMessageDialog(null, "incorrect code");
}
```

example:

```java
int nFirst = 10, nSecond = 3, nThird = 15, nFourth;
switch (nFirst * nSecond / nThird) {
    case 1: // both 1 and 4 give the same output
        case 4:
            nFourth = 5;
            break;
    case 2:
        nFourth = 6;
        break;
    case 3:
        nFourth = 7;
        break;
}
```

7. Decimal Format Class
a. import the class
   import java.text.DecimalFormat;
b. create a DecimalFormat object and give it the desired format as a literal string (a pattern)
DecimalFormat twoPlaces = new DecimalFormat("#0.00"); // # means display only if non-Zero digit
0 means always display a zero if there is no non-Zero digit.
c. call the DecimalFormat method format on the DecimalFormat object, giving it the number to format as an argument.
   Double myNum = 10.0 / 3; // would be 3.3333333333...
   System.out.println(twoPlaces.format(myNum)); // will display 3.33

Here is an alternate way to round down a repeating number:

Math.round( num1 / num2 * 1000 ) / 1000   // for 3 decimal places

Eg. 10.0 / 3 \rightarrow 3.333333333333333 * 1000 \rightarrow 3333.33333333333
Math.round(3333.33333333333) \rightarrow 3333 / 1000 \rightarrow 3.333

8. If time, demo the Binary Number System