THE JAVA IF STATEMENT

The `if` statement introduces a detour in the execution path of the program. The flow of control can either take the detour or skip it.

<table>
<thead>
<tr>
<th>Java If Statement Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>if ( truth value )</code></td>
</tr>
<tr>
<td><code>true path</code></td>
</tr>
<tr>
<td><code>statement below</code></td>
</tr>
<tr>
<td><code>if ( truth value )</code></td>
</tr>
<tr>
<td><code>{</code></td>
</tr>
<tr>
<td><code>true path</code></td>
</tr>
<tr>
<td><code>}</code></td>
</tr>
<tr>
<td><code>statement below</code></td>
</tr>
</tbody>
</table>

If you have only one statement in the `true path` then the braces `{ }` are optional.

<table>
<thead>
<tr>
<th>Java If Statement Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>What the computer executes:</td>
</tr>
<tr>
<td>What is the <code>truth value</code>?</td>
</tr>
<tr>
<td><code>true</code></td>
</tr>
<tr>
<td>Execute <code>true path</code></td>
</tr>
<tr>
<td><code>false</code></td>
</tr>
<tr>
<td>Proceed to execute <code>statement below</code></td>
</tr>
</tbody>
</table>

As illustrated in the picture, the computer begins execution of the `if` statement by (1) evaluating the `truth value`, which can be either `true` or `false`. If `true`, the computer (2) executes the `true path` after which it continues with the `statement below`. If the `truth value` is `false`, the computer (3) skips the `true path`, jumping directly to the `statement below`. 
**Example**
This code illustrates the semantics of the `if` statement.

```java
System.out.print( "Enter two numbers: " );
x = scanner.nextDouble();
y = scanner.nextDouble();
// make sure x is the smaller of the two
if ( x > y )  // if x is larger
    // swap values
    double temp;
temp = x;
x = y;
y = temp;
System.out.println( "The smaller is = " + x );
System.out.println( "The larger is = " + y );
```

Lines 7-10 swap the values of `x` and `y`. For those of you who've never seen this done before, here is a picture that shows how it works.

```
temp = x;
```

```
x  28
```

```
y  11
```

```
x  28
```

```
 temp = x;
```

```
x  28
```

```
 temp = x;
```

```
y  11
```

```
x  28
```

```
 x = y;
```

```
x  11
```

```
y  28
```

```
 temp = x;
```

```
x  11
```

```
y  28
```

Line 5 says to do the swap if `x`'s value is larger than `y`'s. So, if `x` has the larger value then the two are swapped, if `x` has the smaller value then the swap is skipped. Thus, at lines 12 and 13, `x` always has the smaller value.
**Line Breaks, Indention and the { } Delimiters**
The Java compiler ignores line breaks and indentation when it interprets the meaning of an `if` statement. It uses the `{ }` delimiters to determine what’s in the true path; if omitted, the true path is the one statement immediately following the truth value.

### Examples
Each of the following code fragments has the exact same behavior.

```java
if ( year % 2 == 0 )
    System.out.println( "Congressional Election Year!" );
```

```java
if ( year % 2 == 0 )
    System.out.println( "Congressional Election Year!" );
```

```java
if ( year % 2 == 0 )
    System.out.println( "Congressional Election Year!" );
```

```java
if ( year % 2 == 0 ) {
    System.out.println( "Congressional Election Year!" );
}
```

```java
if ( year % 2 == 0 ) {
    System.out.println( "Congressional Election Year!" );
}
```

### Beginner Errors on If Syntax
The biggest error Java novices make writing an `if` statement is to put an extraneous semicolon (;) after the truth value, which places only the null statement within the true path.

#### Example
The code fragment below prints the message for all years because line 2 is not in the true path of the `if` statement.

```java
1  if ( year % 2 == 0 ) ;
2      System.out.println( "Congressional Election Year!" );
```
The second biggest error is to omit or misplace the `{ }` delimiters.

**Example**

In the code fragment below, all the customers get free shipping since line 3 is not in the true path of the `if` statement.

```java
if ( totalSales > 100.00 )
    handlingCharge = 0;
shippingCost = 0;
finalCharge = totalSales + handlingCharge + shippingCost;
```

Misplaced `{ }` delimiters is a common mistake among novice programmers. You can help by making an effort to keep your code neatly and consistently indented.

**Exercises**

1. What is the output of this code segment if the user enters 50? If the user enters 51?

```java
Scanner input = new Scanner( System.in );
double x = input.nextDouble( );
if ( x <= 50 )
{    
   x = (x /= 2.0) * 3;
}
System.out.println( "x = " + x );
```

For each of the `if` statements below, circle what’s wrong and explain. None of them is correct.

2. 

```java
Scanner input = new Scanner( System.in );
double x = input.nextDouble( );
If ( x <= 50 )
{    
   x = (x /= 2.0) * 3;
}
System.out.println( "x = " + x );
```

---

1 See Programming 102 – Coding and Testing → Program Readability, p. 7.
For each of the `if` statements below, circle what’s wrong and explain. None of them is correct.

3. ```java
Scanner input = new Scanner(System.in);
double x = input.nextDouble();
if x <= 50
{
    x = (x /= 2.0) * 3;
}
System.out.println("x = " + x);
```

Give the output of this code segment for each of the input values listed below.

```java
Scanner input = new Scanner(System.in);
int x = input.nextInt();
if (x <= 10)
    x = x + 11;
if (x <= 20)
    x = x + 11;
if (x <= 30)
    x = x + 11;
System.out.println(x);
```

4. User enters 5.
5. User enters 10.
The program is supposed to impose a 2% handling charge and 5% shipping charge on sales less than $100. Sales of $100 or more have no extra charges. For each code fragment below, give the output if the user enters 80? If the user enters 100? Which of the three is coded correctly?

10. ```java
Scanner input = new Scanner( System.in );
double sales = input.nextDouble( );
double handling = 0;
double shipping = 0;
if ( sales < 100.00 )
{
    handling = sales * 0.02;
    shipping = sales * 0.05;
    sales += handling + shipping;
}
System.out.println( sales );
```

11. ```java
Scanner input = new Scanner( System.in );
double sales = input.nextDouble( );
double handling = 0;
double shipping = 0;
if ( sales < 100.00 )
{
    handling = sales * 0.02;
    shipping = sales * 0.05;
    sales += handling + shipping;
}
System.out.println( sales );
```

12. ```java
Scanner input = new Scanner( System.in );
double sales = input.nextDouble( );
double handling = 0;
double shipping = 0;
if ( sales < 100.00 )
    handling = sales * 0.02;
    shipping = sales * 0.05;
    sales += handling + shipping;
System.out.println( sales );
```
<table>
<thead>
<tr>
<th></th>
<th><strong>13.</strong> Given variables <code>miles</code>, <code>gallons</code> and <code>mpg</code>, write the Java <code>if</code> statement that calculates your car's gas mileage while preventing any possible division by zero.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>14.</strong> Given variables <code>sales</code> and <code>bonus</code>, write the Java <code>if</code> statement that calculates a salesperson's bonus. His or her employer gives a 5% bonus on the amount of sales that exceed $100,000.</td>
</tr>
<tr>
<td></td>
<td><strong>15.</strong> Given variables <code>quantity</code>, <code>price</code> and <code>extendedPrice</code>, write the Java <code>if</code> statement that calculates extended price (quantity times price). A 10% discount is given when the quantity exceeds 10.</td>
</tr>
<tr>
<td></td>
<td><strong>16.</strong> A mail-order catalog sells cargo pants for $27 a pair or two pairs for $50. For example, it charges $27 for one pair, $50 for two, $77 for three (i.e. $50 plus $27), $100 for four (i.e. $50 twice). Write a complete, correct Java program that inputs the number of pants sold and prints the total charge.</td>
</tr>
<tr>
<td></td>
<td><strong>17.</strong> In the United States we have a presidential election every four years, when the year is evenly divisible by 4 (e.g. 2004, 2008 and 2012). Write a complete, correct Java program that inputs a year and prints <strong>BE SURE TO VOTE</strong> if it is a presidential election year.</td>
</tr>
</tbody>
</table>