<table>
<thead>
<tr>
<th>Statement Type</th>
<th>Statement Syntax</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expression Statement</td>
<td>expression;</td>
<td>i = 5; printf(&quot;Hello&quot;); 5; /* valid, but nonsensical */</td>
</tr>
<tr>
<td>Declaration Statement</td>
<td>modifiers datatype variable [= initialvalue]{,variable [= initialvalue]}...;</td>
<td>int i; int i, j; int i = 5, j = 6; const int i; static int i; extern int i;</td>
</tr>
<tr>
<td>Compound Statement</td>
<td>{statement statement ... }</td>
<td>{ int i; i = 5; ... }</td>
</tr>
<tr>
<td>If Statement</td>
<td>if (integerexpr) statement; if (pointerexpr) statement;</td>
<td>if (i == 5) { statement; statement; }</td>
</tr>
<tr>
<td>Switch Statement</td>
<td>switch (integerexpr) { case integerconstant: statements case integerconstant: statements default: statements }</td>
<td>switch (i) { case 1: statement; break; case 2: statement; break; default: statement; }</td>
</tr>
<tr>
<td>While Statement</td>
<td>while (integerexpr) statement while (pointerexpr) statement</td>
<td>while (i &lt; 5) { statement; statement; }</td>
</tr>
<tr>
<td>DoWhile Statement</td>
<td>do statement while (integerexpr); do statement while (pointerexpr);</td>
<td>do { statement; statement; } while (i &lt; 5);</td>
</tr>
<tr>
<td>For Statement</td>
<td>for (initexpr; integerexpr; increxpr) statement for (initexpr; pointerexpr; increxpr) statement</td>
<td>for (i = 0; i &lt; 5; i++) { statement; statement; }</td>
</tr>
<tr>
<td>Return Statement</td>
<td>return; return expr;</td>
<td>return; return i + 5;</td>
</tr>
<tr>
<td>Break Statement</td>
<td>break;</td>
<td>while (i &lt; 5) { statement; if (j == 6) break; statement; }</td>
</tr>
<tr>
<td>Continue Statement</td>
<td>continue;</td>
<td>while (i &lt; 5) { statement; if (j == 6) continue; statement; }</td>
</tr>
<tr>
<td>Goto Statement</td>
<td>goto label;</td>
<td>mylabel: ... goto mylabel; ... }</td>
</tr>
</tbody>
</table>
Differences between C and Java:

Expression Statement:
Java: Only expressions that have a side effect can be made into expression statements
C: Any expression can be made into an expression statement
Java: Has final variables
C: Has const variables

Declaration Statement:
Java: Compile-time error to use a local variable before specifying its value
C: Run-time error to use a local variable before specifying its value

Compound Statement:
Java: Declarations statements can be placed anywhere within compound statement
C: Declaration statements must appear before any other type of statement within compound statement

If Statement
Java: Controlling expr must be of type boolean
C: Controlling expr must be of some integer type or a pointer (0 ⇒ FALSE, non-0 ⇒ TRUE)

While Statement
Java: Controlling expr must be of type boolean
C: Controlling expr must be of some integer type or a pointer (0 ⇒ FALSE, non-0 ⇒ TRUE)

DoWhile Statement
Java: Controlling expr must be of type boolean
C: Controlling expr must be of some integer type or a pointer (0 ⇒ FALSE, non-0 ⇒ TRUE)

For Statement
Java: Controlling expr must be of type boolean
C: Controlling expr must be of some integer type or a pointer (0 ⇒ FALSE, non-0 ⇒ TRUE)
Java: Can declare loop control variable in initeexpr
C: Cannot declare loop control variable in initeexpr

Break Statement
Java: Also has "labeled break" statement
C: Does not have "labeled break" statement

Continue Statement
Java: Also has "labeled continue" statement
C: Does not have "labeled continue" statement

Goto Statement
Java: Not provided
C: Provided (but don’t use it!)