# C Statements

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


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: Compiletime error to use a local variable before specifying its value
C: Runtime 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 integral 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 integral 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 integral 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 integral type or a pointer (0 => FALSE, non-0 => TRUE)
Java: Can declare loop control variable in initexpr
C: Cannot declare loop control variable in initexpr

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!)

Copyright © 2003 by Robert M. Dondero, Jr.