3.3 The switch Statement

Another conditional statement in Java is called the switch statement, which directs the executing program to follow one of several paths based on a single value. We also discuss the break statement in this section because it is usually used with a switch statement.

The switch statement evaluates an expression to determine a value, then matches that value with one of several possible cases. Each case has statements associated with it. After evaluating the expression, control jumps to the statement associated with the first case that matches the value. Consider the following example:

```java
switch (idChar)
{
    case 'A':
        aCount = aCount + 1;
        break;
    case 'B':
        bCount = bCount + 1;
        break;
    case 'C':
        cCount = cCount + 1;
        break;
    default:
        System.out.println("Error in Identification Character.");
}
```

First, the expression is evaluated. In this example, the expression is a simple char variable. Execution then transfers to the first statement identified by the case value that matches the result of the expression. Therefore, if idChar contains an 'A', then the variable aCount is incremented. If it contains a 'B', the case for 'A' is skipped and processing continues where bCount is incremented.

If no case value matches that of the expression, then execution continues with the optional default case, indicated by the reserved word default. If no default case exists, then no statements in the switch statement are executed, and processing continues with the statement after the switch. It is often a good idea to include a default case, even if you don't expect it to be executed.

When a break statement is encountered, processing jumps to the statement following the switch statement. A break statement is usually used to break out of each case of a switch statement. Without a break statement, processing continues into the next case of the switch. If the break statement at the end of the 'A' case in the previous example was not there, both the aCount and bCount variables would be incremented when the idChar contains an 'A'. Usually we only want to perform one and only one case, so a break statement is almost always used. Occasionally, though, the "pass through" feature comes in handy.

The expression evaluated at the beginning of a switch statement must be an integral data type, such as an integer or a character. It cannot evaluate to a boolean or floating point value. Furthermore, each case value associated must be a constant; it cannot be a variable or other expression.

Note that the implicit boolean condition of a switch statement is based on equality. The expression at the beginning of the statement is compared to each case value to determine which one it equals. A switch statement cannot be used to determine other relational operations (such as less than), unless some clever processing is done along the way. For example, the GradeReport program in Listing 3.5 prints comments relative to a numeric grade that is entered by the user.

Note that any switch statement could be implemented as a set of nested if statements. However, nested if statements quickly become difficult for a human reader to understand, and are error-prone to implement and debug. But because a switch can only determine equality, sometimes nested if statements are the correct choice. It all depends on the situation.