Switching branches

The if and else keywords, introduced on the previous page, allow programs to branch in a particular direction according to the result of a test condition, and can be used to repeatedly test a variable to match a value. For example, testing for an integer:

```csharp
if ( num == 1 ) { Console.WriteLine("Monday"); }
else if ( num == 2 ) { Console.WriteLine("Tuesday"); }
else if ( num == 3 ) { Console.WriteLine("Wednesday"); }
else if ( num == 4 ) { Console.WriteLine("Thursday"); }
else if ( num == 5 ) { Console.WriteLine("Friday"); }
```

The program will branch in the direction of the match.

Conditional branching with long if-else statements can often be more efficiently performed using a switch statement instead, especially when the test expression evaluates just one variable.

The switch statement works in an unusual way. It takes a given variable value, or expression, then seeks a matching value among a number of case statements. Statements associated with the matching case statement by a : colon will then be executed.

When no match is found, no case statements will be executed, but you may add a default statement after the final case statement to specify statements to be executed when no match is found. The syntax of a typical switch statement looks like this:

```csharp
switch( variable-name )
{
    case value1 : statement ; break ;
    case value2 : statement ; break ;
    case value3 : statement ; break ;
    default : statement ; break ;
}
```

It is important to follow each case statement with the break keyword. Unlike other programming languages, C# does not allow fall-through from one case statement to another – each case statement must allow control to be handed back in order to exit the switch block.
Start a new **Console Application**, then name the project and **Console.Title** as “Switch”.

Type this statement to create and initialize an integer variable:
```
int num = 3;
```

Next, add a statement to declare a **string** variable:
```
string day;
```

Now, add a statement to initialize the **string** variable according to the value of the integer variable:
```
switch (num)
{
    case 1: day = “Monday”; break;
    case 2: day = “Tuesday”; break;
    case 3: day = “Wednesday”; break;
    case 4: day = “Thursday”; break;
    case 5: day = “Friday”; break;
    // Default statement to be inserted here (Step 5).
}
```

Then, insert a final statement into the **switch** block to initialize the **string** variable when no match is found:
```
default: day = “Weekend Day”; break;
```

Finally, add statements to output the assigned value:
```
Console.WriteLine(“Day “ + num + “ : “ + day);
Console.ReadKey();
```

Press **Start** or **F5** to run the application and see the **string** result of the **switch** block assignment.

---

### C# Switch

A **case** statement can also try to match against a **string** value. For example: **case : “ABC”**.