Selection Structures

Logical Operators

You can also include logical operators in an If statement's condition. Figure 2-29 lists the logical operators available in Visual Basic and includes examples of using them in the If statement's condition. Logical operators have an order of precedence and are always evaluated after any arithmetic or comparison operators in an expression

Logical Operators

Operator	Operation	Precedence number
Not	reverses the truth-value of the condition; True becomes False, and False becomes True	1
And	all subconditions must be true for the compound condition to evaluate to True	2
Or	only one of the subconditions needs to be true for the compound condition to evaluate to True	3

Example 1

```
If Not Sen Then
```

The condition evaluates to True when the Sen variable contains the Boolean value False; otherwise, it evaluates to False. The clause could also be written more clearly as `If Sen = False Then`.

Example 2

```
If Rate > 0  And Rate < 0.15 Then
```

The compound condition evaluates to True when the value in the Rate variable is greater than 0 and, at the same time, less than 0.15; otherwise, it evaluates to False.

Example 3

```
If Code = "1" And Sales > 4999.99  Then
```

The compound condition evaluates to True when the Code variable contains the string "1" and, at the same time, the value in the Sales variable is greater than 4999.99; otherwise, it evaluates to False.

Example 4

```
If Code = "1" Or Sales > 4999.99  Then
```

The compound condition evaluates to True when the Code variable contains the string "1" or when the value in the Sales variable is greater than 4999.99; otherwise, it evaluates to False.

Figure 2-29 List and examples of logical operators (continues)

Except for the Not operator, all of the logical operators allow you to combine two or more conditions, called subconditions, into one compound condition. The compound condition will always evaluate to either True or False, which is why logical operators are often referred to as Boolean operators. The tables shown in Figure 2-30, called truth tables, summarize how the computer evaluates the logical operators in an expression.

Truth Tables for the Logical Operators

Not operator

value of *condition*	value of Not *condition*
True	False
False	True

And operator

subcondition1	subcondition2	subcondition1 And subcondition2
True	True	True
True	False	False
False	True	False
False	False	False

Or operator

subcondition1	subcondition2	subcondition1 Or subcondition2
True	True	True
True	False	True
False	True	True
False	False	False

Figure 2-30 Truth tables for the logical operators

As the figure indicates, the **Not operator** reverses the truth-value of the *condition*. If the value of the *condition* is True, then the value of Not *condition* is False. Likewise, if the value of the *condition* is False, then the value of Not *condition* is True. When you use the **And operator** to combine two subconditions, the resulting compound condition evaluates to True only when both subconditions are True. When you combine two subconditions using the **Or operator**, the compound condition evaluates to True when either one or both of the subconditions is True. The compound condition evaluates to False only when both subconditions are False.

Logical Operator Example: Gross Pay Calculator Application

The Gross Pay Calculator application calculates and displays an employee's weekly gross pay, given the number of hours worked and the hourly pay rate. The number of hours worked must be greater than 0 but less than or equal to 40. If the number of hours worked is not valid, the application should display N/A (for Not Available).

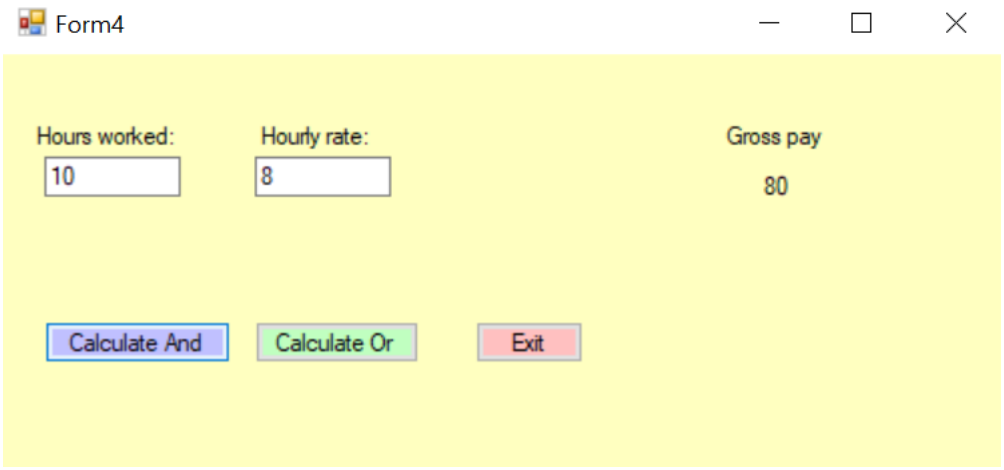

Figure 2-33 User interface for the Gross Pay Calculator Application

```
1   Public Class Form4
2       Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
3           Dim hours As Double = TextBox1.Text
4           Dim rate As Double = TextBox2.Text
5           Dim gross As Double
6           If hours > 0 And hours <= 40 Then
7               gross = hours * rate
8               Label1.Text = gross
9           Else
10              Label1.Text = "N/A"
11          End If
12      End Sub
13      Private Sub Button2_Click(sender As Object, e As EventArgs) Handles Button2.Click
14          Dim hours As Double = TextBox1.Text
15          Dim rate As Double = TextBox2.Text
16          Dim gross As Double
17          If hours <= 0 Or hours > 40 Then
18              Label1.Text = "N/A"
19          Else
20              gross = hours * rate
21              Label1.Text = gross
22          End If
23      End Sub
24      Private Sub Button3_Click(sender As Object, e As EventArgs) Handles Button3.Click
25          Close()
26      End Sub
27  End Class
```

Figure 2-32 shows the code for the Gross Pay Calculator Application