Visual Basic - Chapter 4

Mohammad Shokoohi

* Adopted from An Introduction to Programming Using Visual Basic 2010, Schneider
Chapter 4 – Decisions

4.1 Relational and Logical Operators
4.2 If Blocks
4.3 Select Case Blocks
4.4 Input via User Selection
4.1 Relational and Logical Operators

- ANSI Values
- Relational Operators
- Logical Operators
- Boolean Data Type
- Two Boolean-Valued Methods
- A Boolean-Valued Function
A **condition** is an expression involving relational and/or logical operators. The value of the condition is Boolean – that is, True or False.
ANSI Character Set

A numeric representation for every key on the keyboard and for other assorted characters.

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>32 (space)</td>
<td>48 0</td>
<td>66 B</td>
<td>122 z</td>
<td></td>
</tr>
<tr>
<td>33 !</td>
<td>49 1</td>
<td>90 Z</td>
<td>123 {</td>
<td></td>
</tr>
<tr>
<td>34 “</td>
<td>57 9</td>
<td>97 a</td>
<td>125 }</td>
<td></td>
</tr>
<tr>
<td>35 #</td>
<td>65 A</td>
<td>98 b</td>
<td>126 ~</td>
<td></td>
</tr>
</tbody>
</table>
A numeric representation for every key on the keyboard and for *other assorted characters*.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>162</td>
<td>©</td>
<td>177</td>
<td>±</td>
</tr>
<tr>
<td>177</td>
<td>±</td>
<td>181</td>
<td>μ</td>
</tr>
<tr>
<td>181</td>
<td>μ</td>
<td>190</td>
<td>¼</td>
</tr>
<tr>
<td>190</td>
<td>¼</td>
<td>247</td>
<td>÷</td>
</tr>
<tr>
<td>247</td>
<td>÷</td>
<td>248</td>
<td>φ</td>
</tr>
</tbody>
</table>

ANSI Character Set (continued)
Chr Function

For \( n \) between 0 and 255,

\[
\text{Chr}(n)
\]

is the string consisting of the character with ANSI value \( n \).

Examples: \( \text{Chr}(65) \) is A

\( \text{Chr}(162) \) is ¢
_asc Function_

For a string \( str \),

\[
\text{Asc}(str)
\]

is ANSI value of the first character of \( str \).

Examples: \( \text{Asc}("A") \) is 65
\[
\text{Asc}("$25") \) is 162
Relational Operators

<   less than
<=  less than or equal to
>   greater than
>=  greater than or equal to
=   equal to
<>  not equal to

ANSI values are used to decide order for strings.
Condition

- A condition is an expression involving relational and/or logical operators.
- Result of the condition is True or False.
Example

When \( a = 3, \ b = 4 \)

\[(a + b) \ < \ 2 \times a\]

\[3 + 4 = 7\]
\[2 \times 3 = 6\]

7 is NOT less than 6 and so the value of the expression is False
Another Example

\[ a = 4 \quad b = 3 \quad c = "hello" \quad d = "bye" \]

\[
( c.\text{Length} - b ) = ( a / 2 )
\]

\[
5 - 3 = 2 \quad 4 / 2 = 2
\]

True because 2 equals 2
Relational Operator Notes

- Relational operators are binary – they require an operand on both sides of the operator
- Value of a relational expression will always be True or False
Logical Operators

Used with Boolean expressions

- **Not** – makes a False expression True and vice versa
- **And** – will yield a True if and only if both expressions are True
- **Or** – will yield a True if at least one of both expressions are True
Example 4.3

\[ n = 4, \text{answ} = \text{“Y”} \] Are the following expressions true or false?

Not (n < 6)

(answ = "Y") Or (answ = "y")

(answ = "Y") And (answ = "y")

Not(answ = "y")
Boolean Expression

• An expression that evaluates to either True or False is said to have Boolean data type.

• Example:
  The statement
  
  \[
  \text{textBox.Text} = \text{CStr}((2 + 3) < 6)
  \]

  displays True in the text box.
Boolean Variable

A variable declared with a statement of the form

```vba
Dim var As Boolean
```

Has Boolean data type. It can assume just the two values True and False.

Example:

```vba
Dim boolVar As Boolean
boolVar = 2 < 6
(txtBox.Text = CStr(boolVar))
displays True in the text box.
```
Syntax Error

The following is NOT a valid way to test whether \( n \) falls between 2 and 5:

\[
2 < n < 5
\]
Correction to Syntax Error

To test if \( n \) falls between 2 and 5 use:

\[
(2 < n) \text{ And } (n < 5)
\]

A complete relational expression must be on either side of the logical operators And and Or.
A common error is to replace the condition `Not ( 2 < 3 )` with the condition `2 > 3`.

The correct replacement is `2 >= 3` because `>=` is the opposite of `<`, just as `<=` is the opposite of `>`. 
Two Boolean-Valued Methods

- The expression `strVar1.EndsWith(strVar2)` is true if the value of the first variable ends with the value of the second variable.
- The expression `strVar1.StartsWith(strVar2)` is true if the value of the first variable begins with the value of the second variable.
- **Note:** String literals can be used instead of string variables.
Examples

After the following code is executed each text box will contain the word True.

```vbnet
Dim firstName As String = "William"
textBox1.Text = firstName.EndsWith("am")
textBox2.Text = firstName.StartsWith("Will")
```
A Boolean-Valued Function

• The expression \texttt{IsNumeric(strVar)}
  is true if the value of \textit{strVar} can be
  converted to a number with \texttt{CInt} or \texttt{CDbl}.
  \textbf{Note:} The string variable can be replaced
  with a string literal.

• Examples: \texttt{IsNumeric("123")} is true
  \texttt{IsNumeric("$123")} is true
  \texttt{IsNumeric("3 - 2")} is false
4.2 If Blocks

- If Block
- Nested If Blocks
- ElseIf Clauses
- Input Validation with If Blocks
The following program will take a course of action based on whether a condition is true.

If condition Then
  action 1
Else
  action 2
End If

Will be executed if condition is true
Will be executed if condition is false
Another example If Block

\textbf{If} \textit{condition} \textbf{Then}

\hspace{1cm} \textit{action 1}

\textbf{End If}

\textit{Statement 2}

\textit{Statement 3}

Regardless of whether the condition in the If statement is true or false, these statements will be executed.
Pseudocode and Flowchart for an If Block

If condition is true Then
  Execute action 1
Else
  Execute action 2
End If
Example 1: Form

![Form Diagram]

- `txtFirstNum` for the first number input
- `txtSecondNum` for the second number input
- `txtResult` for displaying the result
Example 1: Code

Private Sub btnFindLarger_Click(...) _
    Handles btnFindLarger.Click

    Dim num1, num2, largerNum As Double
    num1 = CDbl(txtFirstNum.Text)
    num2 = CDbl(txtSecondNum.Text)
    If num1 > num2 Then
        largerNum = num1
    Else
        largerNum = num2
    End If
    txtResult.Text = "Larger number: " & largerNum
End Sub
Example 1: Output

![Image of a maximum function output dialog box with the larger number set to 7.]
Example 2: Form

How many gallons does a ten-gallon hat hold?

Evaluate Answer

txtAnswer

txtSolution
Private Sub btnEvaluate_Click(...) _
Handles btnEvaluate.Click

    Dim answer As Double
    answer = CDb1(txtAnswer.Text)
    If (answer >= 0.5) And (answer <= 1) Then
        txtSolution.Text = "Good, "
    Else
        txtSolution.Text = "No, "
    End If
    txtSolution.Text &= "it holds about 3/4 gals."
End Sub
Example 2: Output

A Quiz

How many gallons does a ten-gallon hat hold? 10

Evaluate Answer

No, it holds about 3/4 of a gallon.
Example 3: Form

![Form Image]

- mtbAnswer
- txtQuote
Example 3

Private Sub btnDisplay_Click(...) _
    Handles btnDisplay.Click
    Dim msg As String
    msg = "Skittles is an old form of bowling " & 
          "in which a wooden disk is used to knock " & 
          "down nine pins arranged in a square. " 
    If txtAnswer.Text.ToUpper = "N" Then 
        MessageBox.Show(msg, "")
    End If
    txtQuote.Text = "Life ain't all beer " & 
    and skittles. - Du Maurier (1894)."
End Sub
Example 3: Output

- **Quotation**
  
  Do you know what the game of skittles is (Y/N)?  N
  
  Display Quotation
  
  Life ain't all beer and skittles. - Du Maurier (1894)

- **Definition**

  Skittles is an old form of bowling in which a wooden disk is used to knock down nine pins arranged in a square.
Example 3: Output (continued)
Nested If Blocks

When one If block is contained inside another If block, the structure is referred to as **nested If blocks**.
Example 5: Form

![Profit/Loss Form](image)

- **Costs:**
- **Revenue:**

[Show Financial Status]
Example 5: Partial Code

If costs = revenue Then
    txtResult.Text = "Break even"
Else
    If costs < revenue Then
        profit = revenue - costs
        txtResult.Text = "Profit is " & FormatCurrency(profit) & "."
    Else
        loss = costs - revenue
        txtResult.Text = "Loss is " & FormatCurrency(loss) & "."
    End If
End If
Example 5: Output
ElseIf Clause

If condition 1 Then
  action 1
ElseIf condition 2 Then
  action 2
ElseIf condition 3 Then
  action 3
Else
  action 4
End If
Example 6: Form

![Form Diagram]

- **txtFirstNum**: Field for the first number.
- **txtSecondNum**: Field for the second number.
- **txtResult**: Field to display the result of finding the larger number.
Example 6: Code

```
Private Sub btnFindLarger_Click(...)(_(' Handles btnFindLarger.Click
    Dim num1, num2 As Double
    num1 = CDbl(txtFirstNum.Text)
    num2 = CDbl(txtSecondNum.Text)
    If (num1 > num2) Then
        txtResult.Text = "Larger number is " & num1
    ElseIf (num2 > num1) Then
        txtResult.Text = "Larger number is " & num2
    Else
        txtResult.Text = "The two are equal."
    End If
End Sub
```
Example 7: Form

![FICA Taxes Form]

Total earnings for this year prior to the current pay period:

Earnings for the current pay period:

Calculate FICA Taxes

FICA taxes for the current pay period:
Example 7: Partial Code

Const WAGE_BASE As Double = 106800
Dim ytdEarnings, curEarnings As Double
Dim socSecBenTax, medicareTax, ficaTaxes As Double
ytdEarnings = CDBl(txtToDate.Text)
curEarnings = CDBl(txtCurrent.Text)
If (ytdEarnings + curEarnings) <= WAGE_BASE Then
    socSecBenTax = 0.062 * curEarnings
ElseIf ytdEarnings < WAGE_BASE Then
    socSecBenTax = 0.062 * (WAGE_BASE - ytdEarnings)
End If
medicareTax = 0.0145 * curEarnings
ficaTaxes = socSecBenTax + medicareTax
Example 7: Output

Total earnings for this year prior to the current pay period: 12345.67

Earnings for the current pay period: 543.21

Calculate FICA Taxes

FICA taxes for the current pay period: $41.56
Input Validation

The statement

\[
\text{If } (\text{IsNumeric}\,(\text{txtBox}\,.\text{Text}) \,= \, \text{True}) \, \text{Then}
\]

is commonly used to validate that input is numeric. It can be condensed to

\[
\text{If IsNumeric\,(\text{txtBox}\,.\text{Text}) \,\text{Then}
\]
Simplified Nested If Statement

Care should be taken to make If blocks easy to understand.

Confusing

Clear

If cond1 Then
    Then
    If cond2 Then
        action
        End If
    End If
End If

If cond1 And cond2
    action
    End If
Comment

Some programs call for selecting among many possibilities. Although such tasks can be accomplished with complicated nested If blocks, the Select Case block (discussed in Section 4.3) is often a better alternative.
4.3 Select Case Block

- A decision-making structure that simplifies choosing among several actions.
- Avoids complex nested If constructs.
- If blocks make decisions based on the truth value of a condition. Select Case choices are determined by the value of an expression called a **selector**.
Select Case Terminology

Each of the possible actions is preceded by a clause of the form

\[ \text{Case } \text{valueList} \]

where \text{valueList} itemizes the values of the \text{selector} for which the action should be taken.
Example 1: Form

![Form Image]

- **txtPosition**
- **txtOutcome**
Example 1: Code

Private Sub btnEvaluate_Click(...) __
    Handles btnEvaluate.Click
    Dim position As Integer = CInt(txtPosition.Text)
    Select Case position
        Case 1
            txtOutcome.Text = "Win"
        Case 2
            txtOutcome.Text = "Place"
        Case 3
            txtOutcome.Text = "Show"
        Case 4, 5
            txtOutcome.Text = "Almost in the money."
        Case Else
            txtBox.Text = "Out of the money."
    End Select
End Sub
Example 1: Output

![Horse Race window with finishing position 2](image)
Example 2: Form

![Horse Race Window](image)

- **txtPosition**: Finishing position (1, 2, 3, ...)
- **txtOutcome**: Evaluate Position
Example 2: Code

Private Sub btnEvaluate_Click(...)_
Handles btnEvaluate.Click
    Dim position As Integer = CInt(txtPosition.Text)
    Select Case position
        Case 1 To 3
            txtOutcome.Text = "In the money. Congratulations"
        Case Is >= 4
            txtOutcome.Text = "Not in the money."
    End Select
End Sub
Example 2: Output
The general form of the Select Case block is

```
Select Case selector
  Case valueList1
    action1
  Case valueList2
    action2
  Case Else
    action of last resort
End Select
```
Rules for Select Case

Each value list contains one or more of the following types of items separated by commas.

1. a literal
2. a variable
3. an expression
4. an inequality sign preceded by Is and followed by a literal, variable, or expression
5. a range given in the form $a \text{ To } b$, where $a$ and $b$ are literals, variables, or expressions.
Flowchart for
Select Case

1. Evaluate selector
2. If value list 1?
   Yes → Action 1
   No → In value list 2?
3. If value list 2?
   Yes → Action 2
   No → In value list n?
4. If value list n?
   Yes → Action n
   No → Perform action of last resort
Example 4: Form

What was President Wilson’s first name?

Interpret Answer

txtReply
Example 4: Partial Code

```vbnet
Select Case firstName
    Case "THOMAS"
        txtReply.Text = "Correct."
    Case "WOODROW"
        txtReply.Text = "Sorry, his name" & 
        " was Thomas Woodrow Wilson."
    Case "PRESIDENT"
        txtReply.Text = "Are you for real?"
    Case Else
        txtReply.Text = "Nice try."
End Select
```
Example 4: Output

[Image of a quiz with the question: "What was President Wilson’s first name?" and the answer: "Woodrow."]

Interpret Answer

Sorry, his full name was Thomas Woodrow Wilson.
Example: Form

![Seasons Form Example]
Example: Partial Code

```vbnet
Dim numDays As Integer
Dim season = txtSeason.Text
Select Case season.ToUpper
    Case "WINTER"
        numDays = 87
    Case "SPRING"
        numDays = 92
    Case "SUMMER", "AUTUMN", "FALL"
        numDays = 93
End Select
```
Example: Output

Season: Summer

Number of Days

Summer has 93 days.
Comments

• In a Case clause of the form \( \text{Case } b \text{ To } c \), the value of \( b \) should be less than or equal to the value of \( c \).

• The word Is should precede an inequality sign in a value list.

• If the word Is is accidentally omitted where required, the editor will automatically insert it when checking the line.
The items in the value list must evaluate to a literal of the same data type as the selector.

For instance, if the selector evaluated to a string value, as in

```vbnet
Dim firstName As String = txtBox.Text
Select Case firstName
then the clause
Case firstName.Length
```

would be meaningless.
Block-Level Scope

- A variable declared inside an If or Select Case block has **block-level scope**.
- The variable cannot be referred to outside of the block.
4.4 Input via User Selection

• Using a List Box for Input
• Using Radio Buttons for Input
• Using Check Boxes for Input
• Events Raised by Selections
The Three Types of Controls Used for Selection

- **List Box**: Business, Computer Science, Education, English, Fine Arts, History, Humanities, Mathematics, Physical Sciences, Social Sciences
- **Radio Buttons**: Freshman, Sophomore, Junior, Senior
- **Check Boxes**: Visual Basic, Java, C++, C, C#
Fill a List Box at Design Time via its String Collection Editor

Tasks button

click here to invoke string collection editor
String Collection Editor

Fill by direct typing or by copying and pasting from a text editor or a spreadsheet.
The value of lstMonths.Text is the string consisting of the selected item.
Dim daysInMonth As String
Select Case lstMonths.Text
    Case "September", "April", "June", "November"
        daysInMonth = "30"
    Case "February"
        daysInMonth = "28 or 29"
    Case Else
        daysInMonth = "31"
End Select
txtDays.Text = daysInMonth
The Group Box Control

• Group boxes are passive objects used to group other objects together.

• When you drag a group box, the attached controls follow as a unit.

• To attach controls to a group box, create the group box and then place the controls into the group box.
Group Box Example

Three attached controls:
Button1
Button2
Button3
Radio Button Properties

• To determine if the button is on or off
  
  radioButton.Checked

  has value True if button is on.

• To turn a radio button on
  
  radioButton.Checked = True
Example 3: Form

```
radChild
radMinor
radAdult
radSenior
```
Example 3: Code for Button

If radChild.Checked Then
    txtFee.Text = FormatCurrency(0)
ElseIf radMinor.Checked Then
    txtFee.Text = FormatCurrency(5)
ElseIf radAdult.Checked Then
    txtFee.Text = FormatCurrency(10)
ElseIf radSenior.Checked Then
    txtFee.Text = FormatCurrency(7.5)
Else
    MessageBox.Show("Must make a selection.")
End If
Example 3: Output

![Admission Fee Example](image-url)
The Check Box Control

- Consists of a small square and a caption
- Presents the user with a Yes/No choice
- During run time, clicking on the check box toggles the appearance of a check mark.
- Checked property has value True when check box is checked and False when not
- CheckedChanged event is raised when the user clicks on the check box
Example 4: Form

chkDrug
chkDental
chkVision
chkMedical

Benefits Menu
- Prescription Drug Plan ($39.15)
- Dental Plan ($10.81)
- Vision Plan ($2.25)
- Medical Plan ($55.52)

Determine Total Monthly Cost

Total monthly cost:
Example 4: Code for Button

```vbnet
Dim sum As Double = 0
If chkDrugs.Checked Then
    sum += 39.15
End If
If chkDental.Checked Then
    sum += 10.81
End If
If chkVision.Checked Then
    sum += 2.25
End If
If chkMedical.Checked Then
    sum += 55.52
End If
txtTotal.Text = FormatCurrency(sum)
```
Example 4: Output

![Benefits Menu]

- Prescription Drug Plan ($39.15)
- Vision Plan ($2.25)
- Medical Plan ($55.52)

Determine Total Monthly Cost

Total monthly cost: $96.92
Events Raised by a Selection

• `SelectedIndexChanged` – raised when a new item of a list box is selected
• `CheckedChanged` - raised when the user clicks on an unchecked radio button or a check box; that is, when the value of the Checked property is changed.
Example 5: Code

Private Sub checkBox_Changed(...) Handles _
    chkDrugs.CheckedChanged,
    chkDental.CheckedChanged,
    chkVision.CheckedChanged,
    chkMedical.CheckChanged

    Dim sum As Double = 0
    If chkDrugs.Checked Then
        sum += 39.15
    End If

(continued on next slide)
Example 5: Code (continued)

```vbnet
If chkDental.Checked Then
    sum += 10.81
End If
If chkVision.Checked Then
    sum += 2.25
End If
If chkMedical.Checked Then
    sum += 55.52
End If
txtTotal.Text = FormatCurrency(sum)
End Sub
```
Example 5: Output

```
Benefits Menu

- Prescription Drug Plan ($12.51)
- Dental Plan ($9.68)
- Vision Plan ($1.50)
- Medical Plan ($25.25)

Total monthly payment: $39.26
```