Object Oriented Programming with Visual Basic .Net

By:

Dr. Hossein Hakimzadeh
Computer Science and Informatics
IU South Bend

(c) Copyright 2007 - 2015, H. Hakimzadeh
What do we need to learn in order to write computer programs?

- Fundamental programming constructs:
  - Variables,
  - Arithmetic operators,
  - Input and output
  - Conditionals,
  - Loops,
  - Procedures and functions,
  - Arrays,
  - Structures, classes and objects,
  - Files
Loops

- A loop is a portion of a program that repeats the execution of one or more statements.

- For example, assume we know how to calculate and display the grade for a single student. What if we want to repeat this process for all the student in a course?
Do Loop

- The semantics of the DO loop is:

  ‘Do this work while a condition is true’

  Or

  ‘Do this work until a condition becomes true’
**Do Loop**

- **Syntax:**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>do while</strong></td>
<td>condition is check at the beginning statements Loop</td>
</tr>
<tr>
<td><strong>do until</strong></td>
<td>condition is check at the beginning statements Loop</td>
</tr>
<tr>
<td><strong>do</strong></td>
<td>statements Loop <strong>while</strong> condition condition is check at the end</td>
</tr>
<tr>
<td><strong>do</strong></td>
<td>statements Loop <strong>until</strong> condition condition is check at the end</td>
</tr>
</tbody>
</table>
Condition in a Loop..

- The **Condition** is expressed as follows:

  - `< expression >  < operator >  < expression >`
  - `X                       =                  Y`

- The **Operators** are:

  - `=, <>, <, >, <=, >=`

- **And some other ones:** AND, OR, NOT
Examples of WHILE Loop

' DO WHILE

Dim counter As Integer = 0

Console.WriteLine("Let's try counting up")

Do While (counter < 10)
    Console.WriteLine("Counter = {0}", counter)
    counter = counter + 1
Loop
Examples of Until Loop

' DO UNTIL

Dim counter As Integer

Console.WriteLine("Now lets try counting down.")

counter = 10
Do Until counter = 0
    Console.WriteLine("Counter = {0}", counter)
    counter = counter - 1
Loop
Don’t Forget these **Necessary Components of a loop:**

- 1) A loop condition must be **true** at one point! Otherwise we will never enter the loop.

- 2) There must be a way to exit a loop. In other words, the loop condition must become **false** at some point. Otherwise we will never stop the loop!
What is the result of the following loops:

Dim X As Integer

X = 0
Do While (X > 0) 'a loop that never executes
    Console.WriteLine("hello")
Loop
What is the result of the following loops:

Dim X As Integer

X = 0
Do While (X < 10) 'an infinite loop (never stops)
    Console.WriteLine("hello")
Loop
Placing the loop condition at the end:

- **Syntax:**

  ```plaintext
do
  one or more statements
Loop  while (condition-is-true)
do
  one or more statements
Loop  until (condition-becomes-true)
```
Placing the loop condition at the end:

Examples:

- counter = 0
- Do
  - Console.WriteLine("This line will execute at least once. ")
- Loop While counter > 0

Note:

- In a do-while or do-until loop, if the condition is checked at the end, the body of the loop is executed one or more times.
For loops:

- For Loop is used when the programmer know exactly how many times a loop should execute.

- Syntax:

```c
For  Loop_variable = starting_point to ending_point [ step ] 
    statement 
Next [Loop_variable]
```
For loops:

Example 1:

Module Module1

Sub Main()

    Dim LoopCounter As Integer

    For LoopCounter = 0 To 10 Step 1
        Console.WriteLine("hello: {0}", LoopCounter)
    Next LoopCounter

    Console.ReadLine() " pause

End Sub

End Module
For loops:

Example 2:

Module Module1

Sub Main()

Dim LoopCounter As Integer

For LoopCounter = 10 To 0 Step -1
    Console.WriteLine("Goodbye: {0}", LoopCounter)
Next LoopCounter

Console.ReadLine() ' pause

End Sub

End Module
Guess what the following two nested For Loops do:

Module Module1
Sub Main()
    Dim Row, Col As Integer
    For Row = 1 To 10
        For Col = 1 To 10
            Console.Write("{0,4}", Row * Col)
        Next Col
        Console.WriteLine()
    Next Row
End Sub
End Module
Guess what the following nested For loops do:

- For Row = 1 To 10
  - For Col = 1 To 10
    - Console.Write("{0,4}", Row * Col)
    - Next Col
  - Console.WriteLine()
- Next Row

Here is the output:

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>10</td>
<td>12</td>
<td>14</td>
<td>16</td>
<td>18</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>9</td>
<td>12</td>
<td>15</td>
<td>18</td>
<td>21</td>
<td>24</td>
<td>27</td>
<td>30</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>12</td>
<td>16</td>
<td>20</td>
<td>24</td>
<td>28</td>
<td>32</td>
<td>36</td>
<td>40</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>15</td>
<td>20</td>
<td>25</td>
<td>30</td>
<td>35</td>
<td>40</td>
<td>45</td>
<td>50</td>
</tr>
<tr>
<td>6</td>
<td>12</td>
<td>18</td>
<td>24</td>
<td>30</td>
<td>36</td>
<td>42</td>
<td>48</td>
<td>54</td>
<td>60</td>
</tr>
<tr>
<td>7</td>
<td>14</td>
<td>21</td>
<td>28</td>
<td>35</td>
<td>42</td>
<td>49</td>
<td>56</td>
<td>63</td>
<td>70</td>
</tr>
<tr>
<td>8</td>
<td>16</td>
<td>24</td>
<td>32</td>
<td>40</td>
<td>48</td>
<td>56</td>
<td>64</td>
<td>72</td>
<td>80</td>
</tr>
<tr>
<td>9</td>
<td>18</td>
<td>27</td>
<td>36</td>
<td>45</td>
<td>54</td>
<td>63</td>
<td>72</td>
<td>81</td>
<td>90</td>
</tr>
<tr>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td>60</td>
<td>70</td>
<td>80</td>
<td>90</td>
<td>100</td>
</tr>
</tbody>
</table>