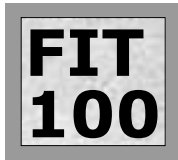


# Iteration -- Once Is Not Enough





# Congratulations!

---

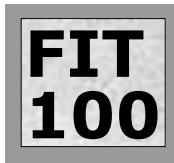
- ❖ The Day Find project is done! -- Reflect
- ❖ This is a significant accomplishment
  - ❑ Understand a fundamental algorithm -- binary search
  - ❑ Know how to search “across a month boundary”
  - ❑ Have encoded the solution in VB6, showing that you know
    - + Declarations and types
    - + Assignment and expressions
    - + Conditional control (If-Then-Else)
    - + Procedure definitions
    - + Procedure calls
  - ❑ Getting it working shows skill in trouble shooting and debugging

## Why Has It Been So Challenging?

- ❖ Algorithm design, programming, application development, etc. are intellectually tough ... why?
  - ❑ There is no “cookbook solution” ... each case has its own logic and requires its own reasoning
  - ❑ The solution must be *exactly right* in every detail
  - ❑ The language used to express the solution (Basic) is new, strange and unforgiving
  - ❑ The context -- Windows operating system, the VB6.0 development environment, the UW computing facilities -- is new and complicated
  - ❑ *The instructors present examples that are “all prepared” so you do not see the actual programming, thinking, debugging and mistakes they make*

Learn by example and analogy

# Iteration -- Once Is Not Enough



Though people don't like to repeat themselves, repetition is a valuable facility that a computer can provide. If program instructions are to be performed more than once, as in Alphabetize CDs, repetition is needed

## Two Additional Control Statements

- ❖ The conditional statement (If-Then-Else) is the only way (so far) to *control* which statements are executed
- ❖ Two more are needed
  - ❑ Elself -- a variation on the If-Then-Else for long sequences of tests
  - ❑ Do While -- a control facility allowing statements to be repeated as long as some condition is true

Programming languages have other control statements, but these are enough to do any programming

# **FIT 100** Elself

---


- ❖ Elself solves the problem of testing a long sequence of alternatives

```
If <T/F condition> Then
    <statement list>                               Stmts for 1st cond
Elself <T/F condition> Then
    <statement list>                               Stmts for 2nd cond
Elself <T/F condition> Then
    <statement list>                               Stmts for 3rd cond
Elself <T/F condition> Then
    <statement list>                               Stmts for 4th cond
Elself <T/F condition> Then
    <statement list>                               Stmts for 5th cond
Else
    <statement list>                               Stmts for otherwise
EndIf
```

## Example

---

- ❖ If txtNum.Text = 1 Then  
    MsgBox("John")                      Executed if Text = 1
- ❖ Elself txtNum.Text = 2 Then  
    MsgBox("Paul")                      Executed if Text ≠ 1 and Text = 2
- ❖ Elself txtNum.Text = 3 Then  
    MsgBox("George")                    Executed if Text ≠ 1 or 2 and Text = 3
- ❖ Elself txtNum.Text = 4 Then  
    MsgBox("Ringo")                    Executed if Text ≠ 1 or 2 or 3 and Text = 4
- ❖ Else  
    MsgBox("Who?")
- ❖ EndIf



The screenshot shows a Windows application window titled "Form1". Inside the window, there is a text box on the left containing the number "1". To the right of this text box is a large button with the text "Pick A Beatle". Further to the right is another text box with the label "Name" above it.

## Contrast With Nested If

---

- ❖ Elself is not a nested test as seen before, though it is similar

```
If txtNum.Text = 1 Then
    MsgBox("John")
Elself txtNum.Text = 2 Then
    MsgBox("Paul")
Elself txtNum.Text = 3 Then
    MsgBox("George")
Elself txtNum.Text = 4 Then
    MsgBox("Ringo")
Else
    MsgBox("Who?")
End If
```

```
If txtNum.Text = 1 Then
    MsgBox("John")
Else
    If txtNum.Text = 2 Then
        MsgBox("Paul")
    Else
        If txtNum.Text = 3 Then
            MsgBox("George")
        Else
            If txtNum.Text = 4 Then
                MsgBox("Ringo")
            Else
                MsgBox("Who?")
            End If
        End If
    End If
End If
End If
```





## Caution With Else If

---

- ❖ An If statement that uses Else If passes through all of the previous cases before reaching a given test ... think about the consequences

```
If someVar < 20 Then
    MsgBox("Less than 20")
Elseif someVar < 10 Then
    MsgBox("Less than 10")
Else
    ...
EndIf
```

Will this MsgBox ever be executed?

## Repeating Terms

---

- ❖ Iteration is the repeated execution of a series of statements in programming
- ❖ To perform iteration, programming languages include special statements often called *iteration statements*
- ❖ There are two crucial components of all iterations:
  - ❑ The statements that will be repeated -- called the loop body
  - ❑ A test specifying when to repetition stops -- termination test
- ❖ Additionally, loops typically have at least one variable that is explicitly changed “inside” the loop -- this is called the iteration variable

Some value must change between consecutive iterations, or else the loop will never terminate ... it is an infinite loop

## General Form Of VB6 Iteration

---

- ❖ VB6, like most languages, has several iteration statements, but only one form is of interest here

```
Do While <termination condition>  
    <statements>  
Loop
```

- ❖ The semantics are as follows:
  - ❑ The termination condition is tested and if it is false the statements are all skipped; execution continues after Loop
  - ❑ If it is true, the statements are performed once
  - ❑ The termination condition is tested again, and if it is false the loop is over and the statements are skipped; continue after Loop
  - ❑ If it is true, the statements are performed a second time
  - ❑ ...

# An Example

---

- ❖ An easy way to get the idea of iteration is to print out the iteration variable ...

Option Explicit

```
Private Sub Form_Click()  
Dim iterateVar As Integer  
    iterateVar = 0  
    Do While iterateVar < 10  
        iterateVar = iterateVar + 1  
        Print ("iterateVar is" & iterateVar)  
    Loop  
End Sub
```

Declaration of iteration variable

Initialization of iteration variable

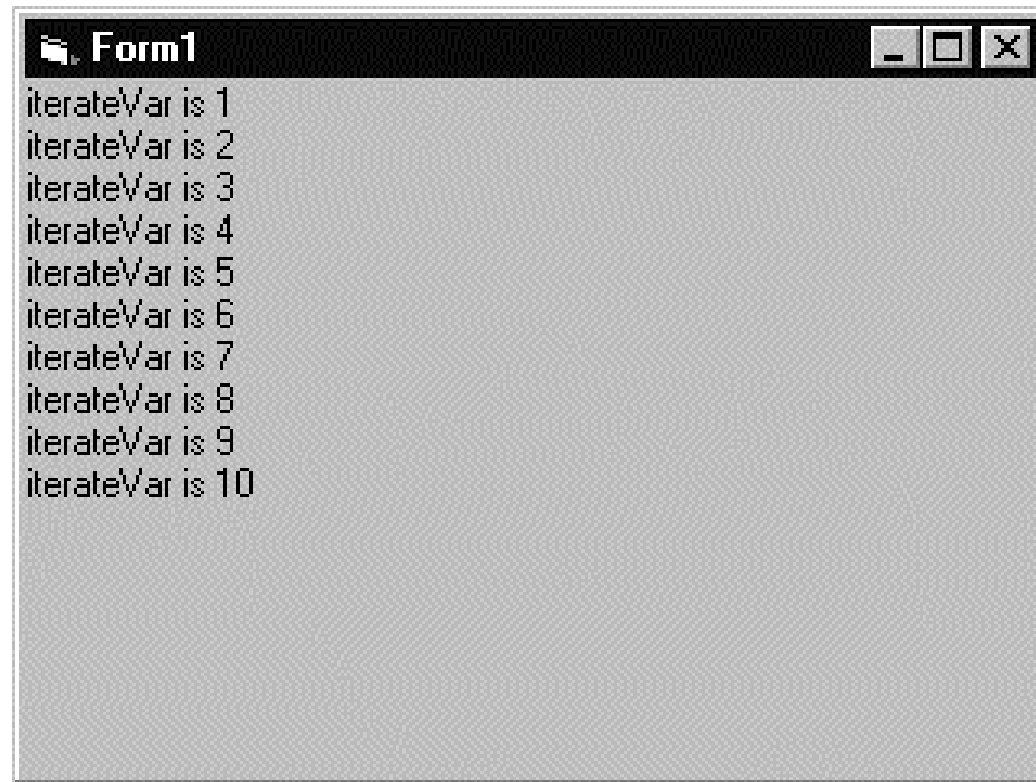
Termination Condition

Increment of the iteration variable

Loop Body

## Execution of Example

---



```
Form1
iterateVar is 1
iterateVar is 2
iterateVar is 3
iterateVar is 4
iterateVar is 5
iterateVar is 6
iterateVar is 7
iterateVar is 8
iterateVar is 9
iterateVar is 10
```

- ❖ Try the same computation with a different termination condition