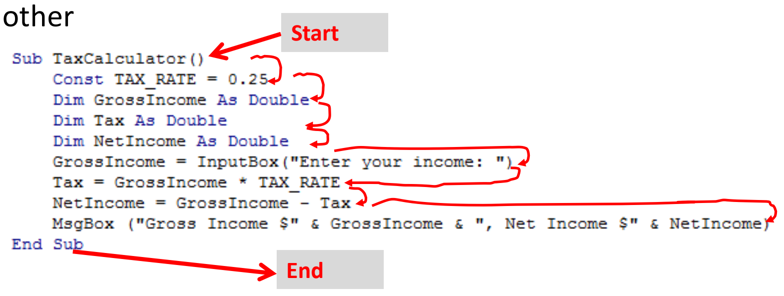
VBA Programming: Part III

- VBA structures: IF branching, Do-While repetition/looping
- Return to VBA collections
- The DIR function
- Basics of the VBA debugger

Recap: Programs You've Seen So Far Produces Sequential Execution

- Each instruction executes from beginning to end, one after the other

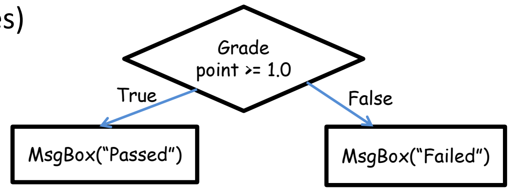


```
Sub TaxCalculator()
    Const TAX_RATE = 0.25
    Dim GrossIncome As Double
    Dim Tax As Double
    Dim NetIncome As Double
    GrossIncome = InputBox("Enter your income: ")
    Tax = GrossIncome * TAX_RATE
    NetIncome = GrossIncome - Tax
    MsgBox ("Gross Income $" & GrossIncome & ", Net Income $" & NetIncome)
End Sub
```

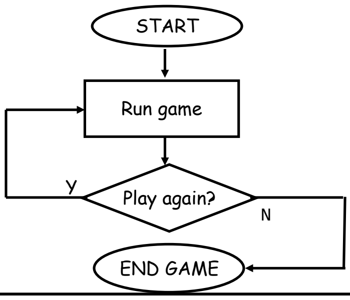
- When the last instruction is reached then the program ends

New Program Writing Concepts (Non-Sequential)

- Branching (alternatives)



- Looping (repetition)



New Terminology

- **What you know; Boolean expression**: An expression that must work out (evaluate to) to either a true or false value.
  - e.g., it is over 45 Celsius today
  - e.g., the user correctly entered the password
- **New term, body**: A block of program instructions that will execute under a specified condition (for branches the body executes when a Boolean is true)

```
Sub Document_Open()
    MsgBox ("Fake virus!")
End Sub
```

This/these instruction/instructions run when you tell VBA to run the macro, the 'body' of the macro program

  - Style requirement
    - The 'body' is indented (1 tab)
    - A "sub-body" (IF-branch) is indented by an additional 1 tab (2 or more tabs)

Branching: Alternative Courses Of Execution

- Similar to the Excel (IF-Function): Check if some condition has been met (e.g., password for the document correctly entered): Boolean expression
- But the IF-structure employed with programming languages is not just a function that returns a value for the true or false cases.
- For the programming IF: **a statement or a collection of statements can be executed** (again this is referred to as "the body" of the if or else case.
  - The programming IF is far more flexible (powerful) that the function equivalent.

Branching: Alternative Courses Of Execution (2)

- Example where alternatives are possible: Checking if the keyboard has caps lock enabled when the user is typing in some text.
  - A popup with the text "KEYBOARD CAPS LOCK ON" when the **caps lock is on**.
  - A popup with the text "Caps lock off" when the **caps lock is off**.

```
If (Application.CapsLock = True) Then
    MsgBox ("KEYBOARD CAPS LOCK ON")
Else
    MsgBox ("Caps lock off")
End If
```

  - Explanations regarding the IF-ELSE structure will be coming shortly.

Branching Mechanisms

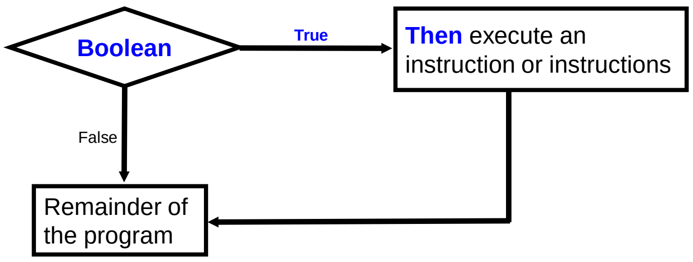
- `If-Then` **Similar to Excel IF function (no false case)**
- `If-Then, Else` **Similar to Excel IF function (true and false case specified)**
- `If-Then, ElseIf, Else` **The Excel equivalent are nested IF functions (may not have been covered).**

Allowable **Operators** For Boolean Expressions (Same Symbols As Excel)

`if (value operator value) then`   `e.g. if (age >= 0) then`

VBA operator	Mathematical equivalent	Meaning	Example
`<`	<	Less than	5 < 3
`>`	>	Greater than	5 > 3
`=`	=	Equal to	5 = 3
`<=`	≤	Less than or equal to	5 <= 5
`>=`	≥	Greater than or equal to	5 >= 4
`<>`	≠	Not equal to	x <> 5

Branching With ‘If-Then’



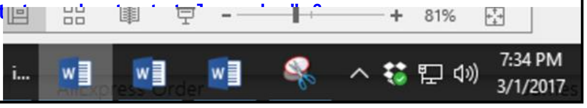
If-Then

- **Format**:
```
If (Boolean expression) Then
    If-Body
End if
```
- **Learning Objective:** Program reacts for the **true case**, counting words in a document.
- **Example usage**:
```
If (totalWords < MIN_SIZE) Then
    MsgBox ("Document too short, total words " & _
    totalWords)
End If
```

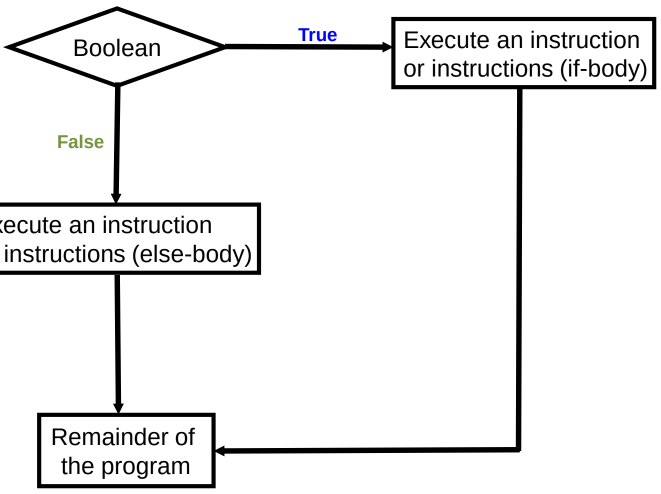
If-Then: Complete Example

- **Learning objective:**
  1. Show how to use an IF structure (program reacts for the true case)
  2. Counting the number of words in Word document
- **Word document containing the macro**: 1wordCountTooFewWords.docm

```
' Try deleting all the words in the Word doc and run the
' macro again
Sub wordCount()
    Dim totalWords As Long
    Const MIN_SIZE As Long = 4
    totalWords =
       ActiveDocument.Range.ComputeStatistics(wdStatisticWords)
    If (totalWords < MIN_SIZE) Then
        MsgBox ("Document
          totalWords)
    End If
End Sub
```



Branching With An 'If, Else'


Boolean
True
Execute an instruction or instructions (if-body)
False
Execute an instruction or instructions (else-body)
Remainder of the program

If-Then (True), Else (False)

- **Format**:

```
If (Boolean expression) Then
    If-Body
Else
    Else-Body
End if
```

- **Example**:

```
If (totalWords < MIN_SIZE) Then
    MsgBox ("Document too short, total words " & totalWords)
Else
    MsgBox ("Document meets min. length requirements")
End If
```

If-Then (True), Else (False): Complete Example

- **Learning objective:**
  - Show how to use an IF-Then structure (program does something for true and false case)
- **Word document containing the macro**: 2wordCountV2TooFewOKCount.docm

```
Sub wordCountV2()
    Dim totalWords As Long
    Const MIN_SIZE As Long = 4
    totalWords =
      ActiveDocument.Range.ComputeStatistics(wdStatisticWords)
    If (totalWords < MIN_SIZE) Then
        MsgBox ("Document too short, total words " &
          totalWords)
    Else
        MsgBox ("Document meets min. length requirements")
    End If
End Sub
```

Applications Of Branching

- **Checking state**

```
IF(program is in some state) then
    Program reacts
End
```

- **Example 1:**

```
If (Application.CapsLock = True) Then
    MsgBox ("Caution: Caps Lock is On!“)
End If
```

- **Example 2:**

```
age = InputBox("Age: ")
If (age < 0) Then
    MsgBox ("Age cannot be negative")
End If
```

Applications Of Branching (2)

- **Example 3:** Learning objective is how to check for **empty user input (“empty string”)**
- (**Name of the Word document that contains the VBA example**): `3checkingForEmptyString.docm`

```
firstName = InputBox("Enter your first name: ")
If (firstName = "") Then
    MsgBox ("You typed in an empty name")
Else
    MsgBox (firstName & " sup?")
End If
```

The `Selection` Object again

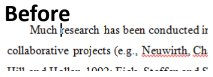
- With previous approaches if no text was selected then the program would produce no visible effect.

```
Sub SelectedFontChange()
    Selection.Font.Bold = wdToggle
End
```

- A modified version automatically selects text.

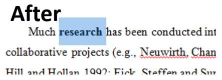
```
Sub AutoSelectedFontChange()
    Selection.Expand
    Selection.Font.Bold = wdToggle
End Sub
```

Before



Much research has been conducted ir
collaborative projects (e.g., Neuwirth, Ch

After



Much **research** has been conducted int
collaborative projects (e.g., Neuwirth, Chan
Hill and Hollan 1992; Fick, Steffen and Sur

The **`Selection` Object** again

- A further modified version (augmented using the `IF` structure):
  - If **no text** has been selected then **display an error message**
  - If **text has been selected** then the **formatting will be changed**

Constants For The Selection Object

Name of constant	Meaning of constant
wdSelectionIP	No text selected
wdSelectionNormal	Text (e.g., word, sentence) has been selected
wdSelectionShape	A graphical shape (e.g., circle, text box) has been selected

Application of these constants coming up on the next slide

The Selection Object And A Practical Application Of Branching

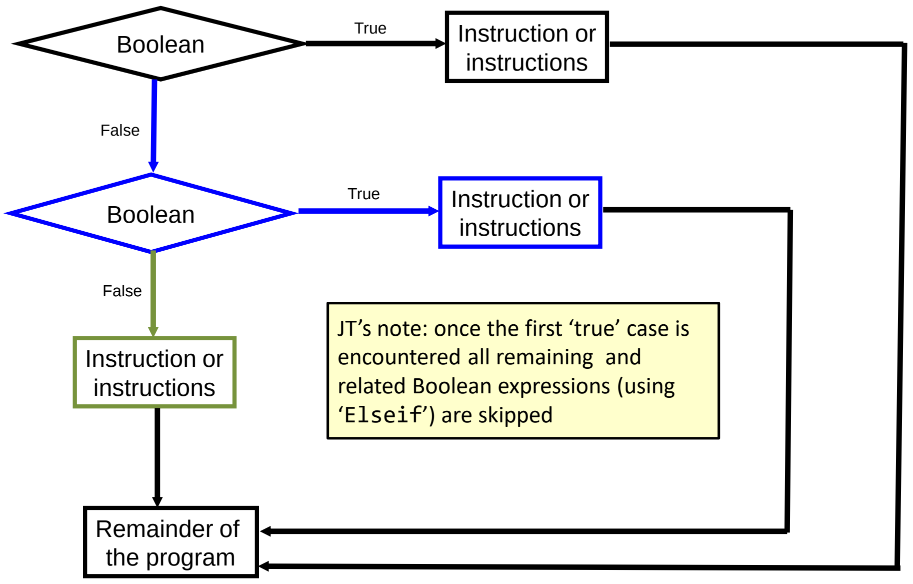
- An example application of branching: check if a selection has been made and only apply the selection if that is the case.
  – Checking if a condition is true
- **Learning objective:** Application of branching (notify user if an action is not valid given the state of Word, empty selection)
- **Word document containing the macro:** "5ifSelectionExample.docm"

```
Sub checkSelection()
    If (Selection.Type = wdSelectionIP) Then
        MsgBox ("No text selected, nothing to change")
    Else
        Selection.Font.Bold = wdToggle  'wdToggle, constant
    End If
End Sub
```

What To Do When Multiple Conditions Must Be Checked

- **Case 1 (mutually exclusive)**:
  - At most one condition is true.
  - The result of one condition affects other conditions (when one condition is true then the other conditions cannot be true)
  - Which of the following is your place of birth? (Answering true to one option makes the options false)
    a) Calgary
    b) Edmonton
    c) Lethbridge
    d) Red Deer
    e) None of the above
  - `If-then`, `elseif`, `else` should be used

Branching With `If-Then`, `Elseif`, `Else`


Boolean
True
Instruction or instructions
False
Boolean
True
Instruction or instructions
False
Instruction or instructions
Remainder of the program

JT's note: once the first 'true' case is encountered all remaining and related Boolean expressions (using '`Elseif`') are skipped

Multiple If-Elseif-Else: Use With Mutually Exclusive Conditions

- **Format:**

```
if (Boolean expression 1) then:
     body 1
elseif (Boolean expression 2) then
     body 2
      ...
else
     body n
' Only one 'end-if' at very end
end if
statements after the conditions
```

Mutually exclusive

- One condition evaluating to true excludes other conditions from being true
- Example: having your current location as 'Calgary' excludes the possibility of the current location as 'Edmonton', 'Toronto', 'Medicine Hat'

If-Elseif-Else: Mutually Exclusive Conditions (Example)

- **Learning objective:** determining which case applies (0 or 1 only applicable)
- **Word document containing the macro (empty document, see macro editor for the important details)**: "6gradesEfficient.docm"

```
If (letter = "A") Then
    grade = 4
ElseIf (letter = "B") Then
    grade = 3
ElseIf (letter = "C") Then
    grade = 2
ElseIf (letter = "D") Then
    grade = 1
ElseIf (letter = "F") Then
    grade = 0
Else
   grade = -1 'A signal that letter was invalid
End If
```

This approach is more efficient when at most only one condition can be true.

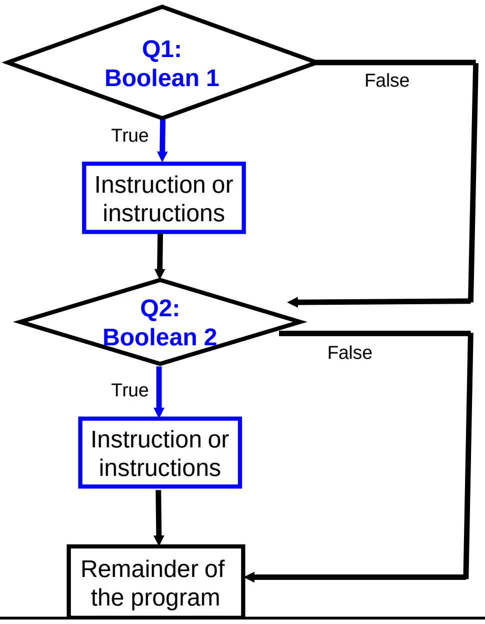
Extra benefit:

The body of the else execu only when all the Boolean expressions are false. (Use for error checking/handling

What To Do When Multiple Conditions Must Be Checked

- **Case 2**: If each condition is independent of other questions
  - Multiple `if-then` expressions can be used
  - Example:
  - Q1: Are you an adult?
  - Q2: Are you a Canadian citizen?
  - Q3: Are you currently employed?

Branching With **Multiple `If-Then`**



Each question is independent (previous answers have no effect on later questions because all questions will be asked).
- Q1: Are you an adult?
- Q2: Are you a Canadian citizen?
- Q3: Are you currently employed?

Multiple Multiple If-Then

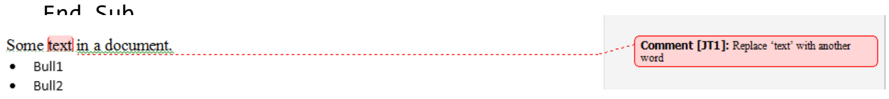
- Any, all or none of the conditions may be true
- Employ when a series of independent questions will be asked
- **Format:**

```
if (Boolean expression 1) then
     body 1
end if
if (Boolean expression 2) then
     body 2
end if
  ...
statements after the conditions
```

Multiple If-Then (2)

- **Learning objective:** a program that handles multiple independent conditions
- **Word document containing the macro:** `7multipleIfs.docm`

```
Sub multipleIf()
 ' Check if there were any 'comments' added to the document.
   If (ActiveDocument.Comments.Count > 0) Then
       MsgBox ("Annotations were made in this document")
   End If
 ' A numbered item includes numbered and bulleted lists.
   If (ActiveDocument.CountNumberedItems() > 0) Then
       MsgBox ("Bullet points or numbered lists used")
   End If
End Sub
```



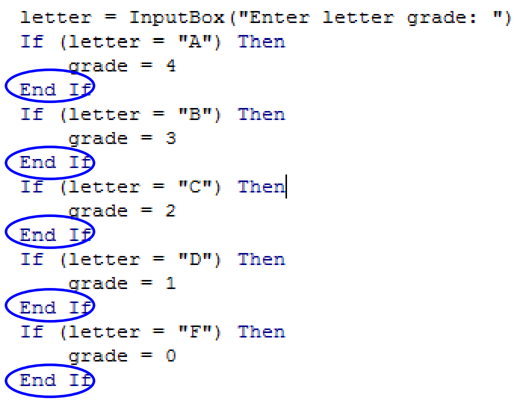
Some text in a document.

- Bull1
- Bull2

Comment [JT1]: Replace 'text' with another word

Location Of The "`End If`": Multiple `If`

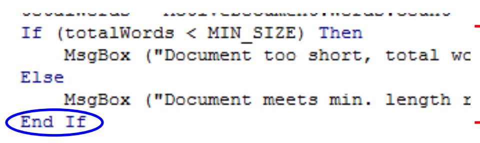
- Independent `If-then`'s:
  - Since each '`if`' is independent each body must be followed by it's own separate '`end if`'



```
letter = InputBox("Enter letter grade: ")
If (letter = "A") Then
    grade = 4
End If
If (letter = "B") Then
    grade = 3
End If
If (letter = "C") Then
    grade = 2
End If
If (letter = "D") Then
    grade = 1
End If
If (letter = "F") Then
    grade = 0
End If
```

Location Of The "`End If`": `If-then`, `Else`

- `If-then`, `Else`:
  - Since the '`if-then`' and the '`else`' are dependent (either one body or the other must execute) the '`end if`' must follow the body of the 'else-body' (last dependent "if-branch")

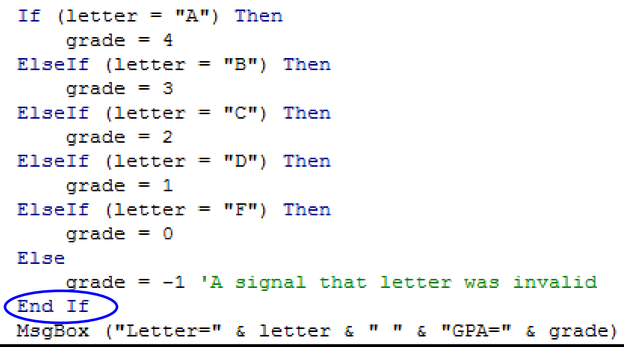


```
If (totalWords < MIN_SIZE) Then
    MsgBox ("Document too short, total wc
Else
    MsgBox ("Document meets min. length r
End If
```

Document either does or does not have enough words (one option IF or the other option ELSE must be applied)

Location Of The "**End If**": If-Then, ElseIf

- Dependent `If-then`, `Else-If`:
  - Since the results of earlier Boolean expressions determine whether later ones can be true (reminder: because at most only one can be true) all of the `if-then` and `Elseif` expressions are dependent (one related block).
  - The "`end if`" belongs at the very end of the block



```
If (letter = "A") Then
    grade = 4
ElseIf (letter = "B") Then
    grade = 3
ElseIf (letter = "C") Then
    grade = 2
ElseIf (letter = "D") Then
    grade = 1
ElseIf (letter = "F") Then
    grade = 0
Else
    grade = -1 'A signal that letter was invalid
End If
MsgBox ("Letter=" & letter & " " & "GPA=" & grade)
```

Logical **AND**: Review From Google Searches

- AND:
  - Requires that a website includes all the words before that site shows up as a search result (all conditions must be true before the entire AND-expression is true)
  - Conversely if a site does not include any of the search words then the site should not appear as a search result (if any condition is false then the entire AND-expression is false)
  - **Format:**
    - *<First word>* (**implicit AND**) *<Second word>*
  - **Example:**
    - `Calgary Canada`

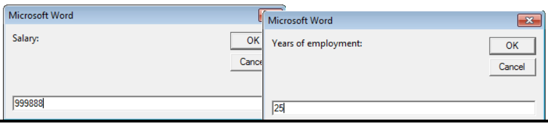
Logic: The VBA "AND" Operator

- **Learning objective:** a program that reacts only if all conditions met
- **Format:**

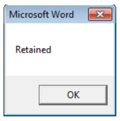
```
If ((Boolean expression) And (Boolean expression)) then
    body
End if
```

- **Word document containing the macro (empty document, see macro editor for the important details)**: `8if_and_firing.docm`

```
salary = InputBox("Salary: ")
years = InputBox("Years of employment: ")
If ((salary >= 100000) And (years < 2)) Then
    result = "Fired!"
Else
    result = "Retained"
```



Salary:
999888
Years of employment:
25



Retained

Firing Example: Example Inputs & Results

```
If ((salary >= 100000) And (years < 2)) Then
```

Salary	Years on job	Result
1	100	Retained
50000	1	Retained
123456	20	Retained
1000000	*0*	*Fired!*

Logical **OR**: Review From Google Searches

- OR:
  - If a website includes any of the search words then the site shows up as a search result (a single true result will make the entire OR-expression is true)
  - Conversely only if a website does not include any of the search words will a site not appear as a result (only if all results are false will the entire OR-expression evaluate to false)
  - **Format:**
    - *<First word>* **OR** *<Second word>*
  - **Example:**
    - Calgary **OR** Edmonton

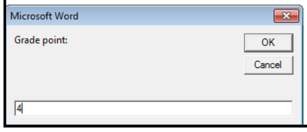
Logic: The VBA "**OR**" Operator

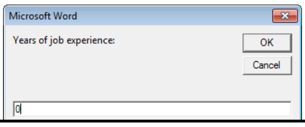
- **Format:**

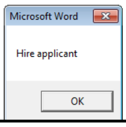
```
If ((Boolean expression) OR (Boolean expression)) then
    body
End if
```

- **Learning objective**: a program that reacts if at least one condition met.
- **Word document containing the macro (empty document, see macro editor for the important details)**: 9if_or_hiring.docm

```
gpa = InputBox("Grade point: ")
experience = InputBox("Years of job experience: ")
If ((gpa > 3.7) Or (experience > 5)) Then
    result = "Hire applicant"
Else
    result = "Insufficient qualifications"
```


Microsoft Word
Grade point:
4


Microsoft Word
Years of job experience:
0


Microsoft Word
Hire applicant

Hiring Example: Example Inputs & Results

```
If ((gpa > 3.7) Or (experience > 5)) then
```

GPA	Years job experience	Result
2	*0*	*Insufficient qualifications*
1	10	Hire
4	1	Hire
4	7	Hire

Line Continuation Character (Repeated Again For Branching)

- To increase readability of long `IF` statements the line continuation character can split the Boolean expressions (one Boolean per line)

```
If (income  > 99999) And _
   (experience <=  2) And _
   (numRepramands > 0) Then
     MsgBox ("You're fired!")
End If
```

- Reminder:
  - To split the line the line continuation character (underscore) must be preceded by a space.
- Keywords cannot be split between lines e.g.

```
Msg _
Box
```

For more details see: http://support.microsoft.com/kb/141513

Application: IF-Branching (Marking Program)

- Case 1, Failure: document has any spelling mistakes
- Case 2, Pass: document has no spelling mistakes
- **Learning Objective:** Application of branching and other concepts, marking a document based on the number of typographical errors and formatting the marking feedback.
- **Name of the Word document that contains the program**: `10Marking programV1_IF.docm`

Marking Program

```
Sub MarkingProgram()
    Dim totalTypos As Long
    Const MAX_TYPOS = 0
    Dim feedback As String
    totalTypos = ActiveDocument.SpellingErrors.Count
    feedback = "Marking.."
    Selection.HomeKey Unit:=wdStory
    If (totalTypos > MAX_TYPOS) Then
       feedback = feedback & "Has typos: Fail"
    Else
       feedback = feedback & ": Passing grade"
    End If
```

Marking Program (2)

```
    feedback = feedback & vbCr & vbCr
    Selection.Font.ColorIndex = wdRed
    Selection.Font.Size = 16
    Selection.Font.Name = "Arial"
    Selection.TypeText (feedback)
End Sub
```

Conditions Inside Of Conditions

- This is referred to as '**nesting**' (one form of nesting)
- An `IF` can contain within its body a second `IF`

```
IF (Boolean expression 1 for outer IF)
          ..
Outer     IF (Boolean expression 2 for inner IF)     Nested
body          ..                                     inner
          End IF  'Inner If                          body
End IF  'Outer If
```

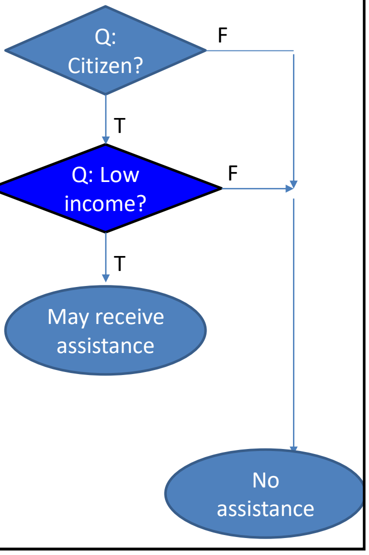
  - In other words: Boolean expression 2 is checked only when Boolean expression is true

Recognizing When **Nesting** Is Needed

- **Scenario 1**: A second question is asked only if a first question answers true:
  - Example: If it's true the applicant is a Canadian citizen, then ask for the person's income (checking if eligible for social assistance).
  - Type of nesting: an IF-branch nested inside of another IF-branch

```
 If (Boolean) then
    If (Boolean) then
        ...
    End If
End if
```

Nested branch/IF



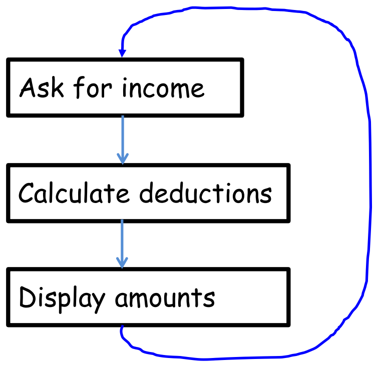
Nested IFs

- **Learning objective:** Conditions checked only if other conditions are true.
- **Word document containing the example**: `11nestingIFinsideIF.docm`

```
Sub nestedCase1()
    Dim country As String
    Dim income As Long
    Const INCOME_CUTOFF = 24000
    country = InputBox("What is your country of citizenship?")
    If (country = "Canada") Then
        income = InputBox("What is your income $")
        If (income <= INCOME_CUTOFF) Then
            MsgBox ("Citizenship: " & country & "; " & _
              "Income $" & income & _
              ": eligible for assistance")
        End If
    End If
End Sub
```

Looping/Repetition

- How to get the program or portions of the program to automatically re-run
  - Without duplicating the instructions
  - Example: you need to calculate tax for multiple people

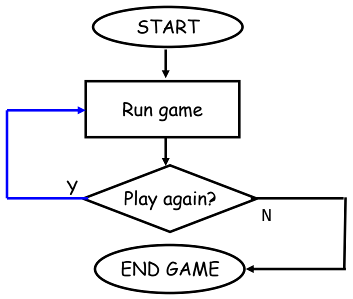


Loop: allows you to repeat the same tasks over and over again

Looping/Repetition (2)

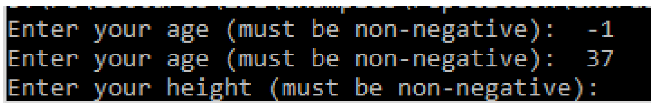
- The entire program repeats

www.colourbox.com



Looping/Repetition (3)

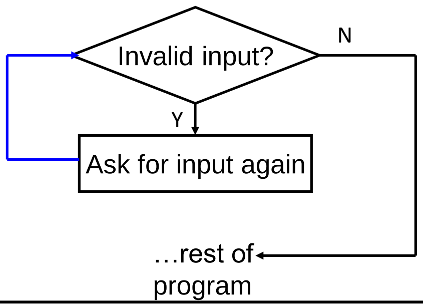
- Only a specific part of the program repeats



```
Enter your age (must be non-negative):  -1
Enter your age (must be non-negative):  37
Enter your height (must be non-negative):
```

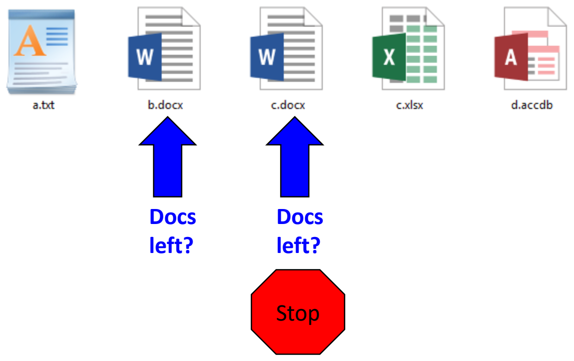
Re-running specific parts of the program

Flowchart



Looping/Repetition (4)

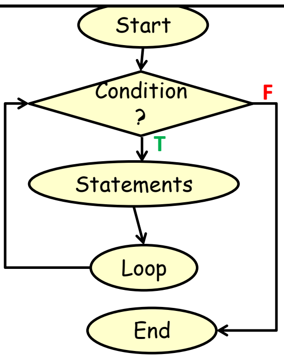
- Process Word documents in a folder as long as there are unprocessed documents remaining in the folder.



Characteristics Of Do-While Loops

- Described as variable repetition loops: runs as long as some condition holds true (number of times that the loop repeats is variable)
  - e.g., while the user doesn't quit the program re-run the program
  - e.g., while the user enters an erroneous value ask the user for input.
  - e.g. while there are unprocessed documents (0? 1? 50?)

Do-While Loop

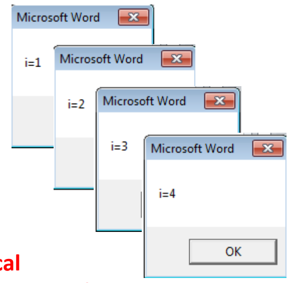


- **Format**:

```
Do While <Condition>
    <Statement(s)>
Loop
```

- **Learning objective:** illustrating loop with a program that counts up by 1
- **Example**: "12whileUpOne.docm"

```
Dim i As Long
i = 1
Do While (i <= 4)
    MsgBox ("i=" & i)
    i = i + 1
Loop
```



i=1

i=2

i=3

i=4

Any valid mathematical expression here e.g. count up by 10, decrease by 1, exponential function etc.

Programming Style: Variable Names

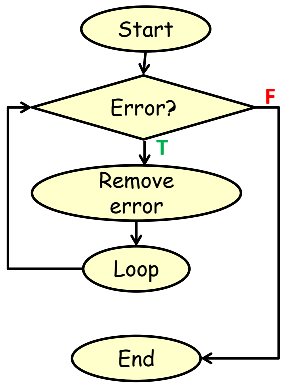
- In general variable names should be self-descriptive e.g., ‘age’, ‘height’ etc.
- Loop control variables are an exception e.g., ‘i’ is an acceptable variable name
  - It’s sometimes difficult to come up with a decent loop control name
  - Loop control variables are given shorter names so the line length of a loop isn’t excessive

```
Dim loopControl As Integer
loopControl = 1
Do While (loopControl <= 4)
    ...
```

Application Of Looping: Error Handling

- General structure:

```
Do While (Error occurring)
    Instructions to deal handle error
Loop
```



Error Handling Example

- **Learning objective:** a program that uses a loop to prevent the user from entering a value outside a valid range.
- **Name of the Word document containing the complete program**: `13errorHandlingLoop.docm`

```
Dim income As Long
Dim tax As Long
Const TAX_RATE = 0.2
income = InputBox("Enter a non-negative income $")
Do While (income < 0)
    MsgBox ("Income cannot be less than zero")
    income = InputBox("Enter a non-negative income $")
Loop
tax = income * TAX_RATE
MsgBox ("Income $" & income & " requires $" & tax & _
  " _taxes paid")
```

Logic And Loops

- Both AND, OR logic can be employed with loops
- AND: when a loop repeats while all conditions are true.
- OR: when a loop repeats when at least one condition is true.

Error Handling Loop: **OR**

- **Learning objective:** a program using a loop to ensure entry of value that's not outside of a valid range (either too low or too high). Program repeats if input is either too low or too high.
- **Name of the Word document that contains the complete program**: `14errorHandlingLoopOR.docm`

```
Dim age As Long
Dim catAge As Long
Const MIN_AGE = 0
Const MAX_AGE = 118
Const CAT_HUMAN_AGE_RATIO = 7
age = -1
Do While ((age < MIN_AGE) Or (age > MAX_AGE))
    age = InputBox("Enter age (0-118): ")
        MsgBox ("Age must be in the range of 0-118")
Loop
catAge = age * CAT_HUMAN_AGE_RATIO
```

Error Handling Loop: **AND**

- **Learning objective:** A program that only allows one out of a set of valid values (loops as long as the value is not the first valid value and not the second value and not the third value).
- **Name of the Word document that contains the complete program**: `15errorHandlingLoopAND.docm`

```
Dim province As String
province = "ON"
Do While ((province <> "BC") _
   And (province <> "AB") _
   And (province <> "SK"))
    province = InputBox("Enter a Western Canadian " & _
      " province: ")
Loop
MsgBox (province & " is a Western province.")
```

Looping And Collections

- Because the number of objects within a collection can vary (e.g. number of documents currently open) and loops can repeat a variable number of times it's common to employ a `do-while` loop when accessing a collection.

Loops And Collections: Example #1

- **Learning objective:** Using a loop to automatically print (one at a time) all the documents currently opened in Word.
- **Word document containing the macro example:** `16printMultipleDocuments.docm`

```
Sub PrintDocumentsCollection()
   Dim numDocuments As Integer
   Dim count As Integer
   numDocuments = Documents.count
   count = 1
   Do While (count <= numDocuments)
       Documents.Item(count).PrintOut
       count = count + 1
   Loop
End Sub
```

Loops And Collections: Example #2

- **Learning objective:** Using a loop to automatically **sort all of the tables** in the currently active Word document.
- **Word document containing the macro example:** 17sortingMultipleTables.docm

```
Dim CurrentTable As Integer
Dim NumTables As Integer
NumTables = ActiveDocument.Tables.Count
' Case 1: No tables in document
If NumTables = 0 Then
    MsgBox ("No tables to sort")
```

Loops And Collections: Example #2 (2)

```
'At least one table in the document.
Else
    CurrentTable = 1
    Do While (CurrentTable <= NumTables)
        MsgBox ("Sorting Table # " & CurrentTable)
        ActiveDocument.Tables(CurrentTable).Sort
        CurrentTable = CurrentTable + 1
    Loop
End If
```

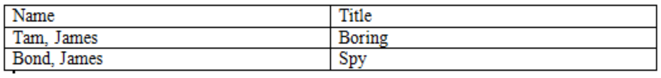
More On Sort

- A **parameter** that can be used to leave out the first (header) row during the sort
- **Format**

  Sort (*<Boolean whether there is one in the table – True or False>*)

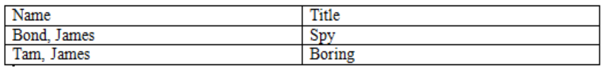
  – Exclude the header (1st row) from sort
- **Example**

  – `ActiveDocument.Tables(CurrentTable).Sort(True)`



Name	Title
Tam, James	Boring
Bond, James	Spy

  – After



Name	Title
Bond, James	Spy
Tam, James	Boring

Sorting A Table With Headers: Variant Example #2

- **Learning objective:** same as previous program but excludes table headers from the sort.
- **Word document containing the macro example:** `18sortingMultipleHeaderedTables.docm`

```
Dim CurrentTable As Integer
Dim NumTables As Integer
NumTables = ActiveDocument.Tables.Count
If NumTables = 0 Then
    MsgBox ("No tables to sort")
Else
    CurrentTable = 1
    Do While (CurrentTable <= NumTables)
        MsgBox ("Sorting Table # " & CurrentTable)
        ActiveDocument.Tables(CurrentTable).Sort (True)
        CurrentTable = CurrentTable + 1
    Loop
End If
```

The DIR Function

- If used in conjunction with a loop:
  - It can be used to go through all the documents in a folder (this will be illustrated gradually in advanced examples but the first one will be rudimentary)
  - It can be used to go through the entire contents of a folder including sub-folders and sub-sub folders (very advanced use: well beyond the scope of the this course)
- Basic use: this function takes a location (e.g., `C:\temp\`) and a filename as an argument and it determines if the file exists at the specified location.
  - If the file is found at this location then the function returns the name of the file.
  - If the file is not found at this location then the function returns an empty string (zero length)

Simple Use Of The DIR Function

- **Learning objective:** a learning example to show how the DIR function works
- **Word document containing the macro example:** 19DIRFunctionSimple.docm

```
Dim location As String
Dim filename As String
Dim result As String
location = "C:\temp\203\dirExample1\" 'Always look here
result = Dir(location) ' Opens first file
MsgBox (result)
result = Dir(location & "*.xls*") 'Any Excel document
MsgBox (result)
filename = "b.docx"
result = Dir(location & filename) 'Always look 4 Doc1.dox
MsgBox (result)
```

Nesting: Loop Within A Branch

- The upcoming example will employ another form of nesting:

```
If (Error: empty folder path)
    Display popup error message
Else
    While (there is another unopened Word document)
        Open document
        Move onto the next document
```

Practical Use Of `Dir:` Access Each File In A Directory

- **Learning objective:** a program that can automatically open and modify in succession all the Word documents in a folder specified by the user.
- **Word document containing the macro example:** 20`loopFolder.docm`
- Features:
  - Prompts the user for the location to the Word documents ('path')
  - Error handling ("`IF`-body")
    - Empty path (i.e. no location entered by the user) or valid path but a path points to an empty folder
  - Non-error case ("`ELSE`-body")
    - Path is okay: using a loop open each Word document in turn

VBA Program: Successively Access Word Documents

```
Dim directoryPath As String
Dim currentFile As String
directoryPath = InputBox("Location for files: ")
currentFile = Dir(directoryPath)

' Dir returns name of a file or empty string if no files
If (currentFile = "") Then
    MsgBox (directoryPath & " does exist/folder is empty")
Else
    ' *.doc* access Word 2003 (doc) or 2007+ (docx)
    currentFile = Dir(directoryPath & "*.doc*")
```

VBA Program: Successively Access Word Documents (2)

```
    ' Path is OK, contains Word documents
    Do While (currentFile <> "")
       ' Display file name in popup
        MsgBox (currentFile)

         ' Use filename to open the Word document from
         ' currentFile = Dir(directoryPath & "*.doc*")
        Documents.Open (directoryPath & currentFile)

         'Modify each document (next slide)

       'Move onto next document in folder
        currentFile = Dir
    Loop

End If
```

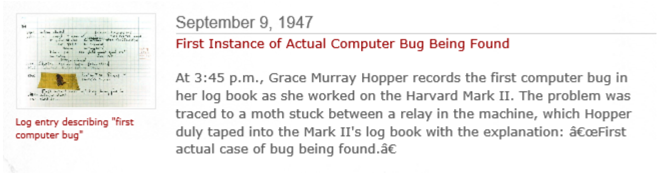
VBA Program: Successively Access Word Documents (How Each Open Document Is Modified)

```
 'How the program modifies each document (these
'instructions should be inserted into the specified
'location on the previous slide
Selection.HomeKey Unit:=wdStory
Selection.Font.ColorIndex = wdBlue
Selection.TypeText ("typos " & _
  ActiveDocument.SpellingErrors.Count)
ActiveDocument.Close (wdSaveChanges)
```

The VBA Debugger

- 'Bug':
  - An error in the logic of your program.
  - The program "doesn't do what it is supposed to do"
  - Example: an erroneous formula for calculating an area of a rectangle

    `area = length + width`
  - Bugs will seldom be this obvious
- Debuggers can be used to help find errors in your program
- Normally more information on using the VBA debugger will be provided in tutorial



Log entry describing "first computer bug"

September 9, 1947

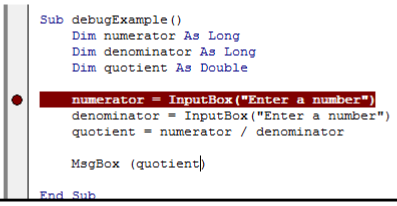
First Instance of Actual Computer Bug Being Found

At 3:45 p.m., Grace Murray Hopper records the first computer bug in her log book as she worked on the Harvard Mark II. The problem was traced to a moth stuck between a relay in the machine, which Hopper duly taped into the Mark II's log book with the explanation: â€œFirst actual case of bug being found.â€

Screenshot: www.computerhistory.org

The VBA Debugger (If There Is Time)

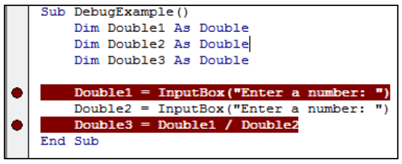
- Debuggers can be used to help find errors in your program
- Setting up breakpoints
  - Points in the program that will ‘pause’ until you proceed to the next step
  - Useful in different situations
    - The program ‘crashes’ but you don’t know where it is occurring
      - Pause before the crash
    - An incorrect result is produced but where is the calculation wrong
- Set up breakpoints
  - Click in the left margin



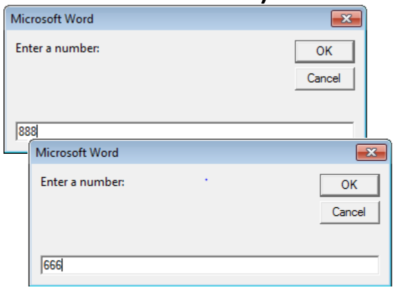
```
Sub debugExample()
    Dim numerator As Long
    Dim denominator As Long
    Dim quotient As Double

    numerator = InputBox("Enter a number")
    denominator = InputBox("Enter a number")
    quotient = numerator / denominator

    MsgBox (quotient)

End Sub
```

The VBA Debugger (2) (If There Is Time)

- Multiple breakpoints



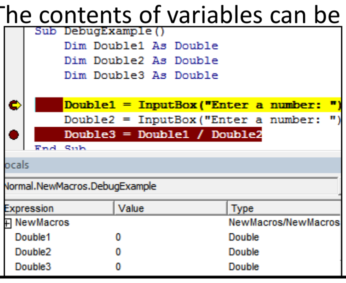
```
Sub DebugExample()
    Dim Double1 As Double
    Dim Double2 As Double
    Dim Double3 As Double

    Double1 = InputBox("Enter a number: ")
    Double2 = InputBox("Enter a number: ")
    Double3 = Double1 / Double2
End Sub
```



Microsoft Word
Enter a number:
OK
Cancel
888

Microsoft Word
Enter a number:
OK
Cancel
666

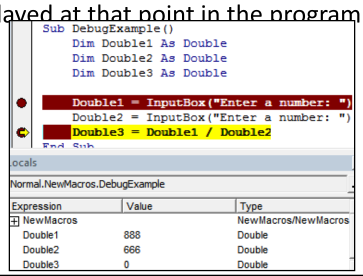
- Program pauses when breakpoints are reached
  - The contents of variables can be displayed at that point in the program



```
Sub DebugExample()
    Dim Double1 As Double
    Dim Double2 As Double
    Dim Double3 As Double

    Double1 = InputBox("Enter a number: ")
    Double2 = InputBox("Enter a number: ")
    Double3 = Double1 / Double2
End Sub
```

ocals
Normal.NewMacros.DebugExample

Expression	Value	Type
⊞ NewMacros		NewMacros/NewMacros
Double1	0	Double
Double2	0	Double
Double3	0	Double



```
Sub DebugExample()
    Dim Double1 As Double
    Dim Double2 As Double
    Dim Double3 As Double

    Double1 = InputBox("Enter a number: ")
    Double2 = InputBox("Enter a number: ")
    Double3 = Double1 / Double2
End Sub
```

ocals
Normal.NewMacros.DebugExample

Expression	Value	Type
⊞ NewMacros		NewMacros/NewMacros
Double1	888	Double
Double2	666	Double
Double3	0	Double

Common Mistake #1

- Mixing up branches (`IF` and variations) vs. loops (`do-while`)
- Related (both employ a Boolean expression) but they are not identical
- Branches
  - General principle: If the Boolean evaluates to true then execute a statement or statements (**once**)
  - Example: display a popup message if the number of typographical errors exceeds a cutoff.
- Loops
  - General principle: As long as (or while) the Boolean evaluates to true then execute a statement or statements (**multiple times**)
  - Example: While there are documents in a folder that the program hasn't printed then continue to open another document and print it.

Common Mistake #1: 2

- Contrast (try running both cases with >1 invalid values)
- **Learning objective:** learning example illustrating the difference between using a branch vs. a loop.
- **Word document containing the complete program**: `21loopVsBranch.docm`

```
age = InputBox("Age (positive only)")
If (age <= 0) then
    age = InputBox("Age (positive only-IF)")
End if
MsgBox(age)
```

Vs.

```
age = InputBox("Age (positive only)")
Do While (age <= 0)
    Age = InputBox("Age (positive only-WHILE)")
Loop
MsgBox(age)
```

After This Section You Should Now Know

- What is a named constant, why use them (benefits)
- What is a predefined named constant and what are some useful, commonly used predefined constants
- Naming conventions for constants
- How to use branches to make decisions in VBA
  - `If`
  - `If-else`
  - `Multiple If's`
  - `If, else-if, else`
  - Using logic (`AND, OR, NOT`) in branches
- How to get a program to repeat one or more instructions using `Do-while` loops

After This Section You Should Now Know (2)

- Nesting:
  - `IF` within an `IF`
  - `Do-While` within an `IF`
  - Writing and tracing/nested structures
  - When to apply nesting
- Applying looping to collections
- How to use the '`Dir`' function to access a folder
  - Using this function to step through all the documents or specific types of documents in a folder

Images

- “Unless otherwise indicated, all images were produced by James Tam

slide 77