Visual Basic for Applications
Programming

Damiano SOMENZI

School of Economics and Management

Advanced Computer Skills

damiano.somenzi@unibz.it

Week 9

Outline

1. Logical Operators
2. Algorithms
   - Searching
   - Exercises

AND Operator

And

And operator is used to perform a *logical* conjunction on two expressions

- The syntax is $result = expression1$ **And** $expression2$
- *result* (required) should be a boolean variable, *expression1* and *expression2* (required) are any expression
- If both expressions evaluate to **True**, *result* is **True**. If either expression evaluates to **False**, *result* is **False**

```
Dim validNum As Boolean
```

```
validNum = ((n >= 1000) And (n <= 2000))
```

`validNum` holds `TRUE` if the number in `n` is larger than or equal to 1000 **and** is smaller than or equal to 2000, `FALSE` otherwise

AND Operator

Example

example

1. `result = ((x >= 10) AND (x <= 20))`

x	$x \geq 10$	AND	$x \leq 20$	Result
15	TRUE	And	TRUE	TRUE
5	FALSE	And	TRUE	FALSE
25	TRUE	And	FALSE	FALSE

2. `result = ((font = "blue") AND ( bgcolor = "yellow"))`

font	bgcolor	font = ”blue”	AND	bgcolor = ”yellow”	Result
blue	blank	TRUE	And	FALSE	FALSE
red	blank	FALSE	And	FALSE	FALSE
blue	yellow	TRUE	And	TRUE	TRUE

OR Operator

Or

Or operator is used to perform a logical disjunction on two expressions

- The syntax is $result = expression1$ **Or** $expression2$
- *result* (required) should be a numeric variable, *expression1* and *expression2* (required) are any expression
- If either or both expressions evaluate to **True**, result is **True**

```
Dim answer As Boolean
answer = ((s = "YES") Or (s = "yes"))
```

`answer` holds `TRUE` if the string (representing the answer of the question) in `answer` is is the string ”YES” **or** ”yes”, `FALSE` otherwise

OR Operator

Example

example

1. `result = ((s = "YES") Or (s = "yes"))`

s	s = "YES"	OR	s = "yes"	Result
"YES"	TRUE	Or	FALSE	TRUE
"yes"	FALSE	Or	TRUE	TRUE
"Yes"	FALSE	Or	FALSE	FALSE

2. `result = ((x <> 0) Or (y <> 1))`

x	y	x <> 0	OR	y <> 1	Result
2	1	TRUE	Or	FALSE	TRUE
0	0	FALSE	Or	TRUE	TRUE
0	1	FALSE	Or	FALSE	FALSE

Outline

1 Logical Operators

2 Algorithms
- Searching
- Exercises

Searching

Searching for a number or a string in a sequence

1. Problem: sequence of n ($n > 0$) numbers $n_1, n_2, n_3, .. , n_n$; a single number q
2. Return the index of the found number or 0
3. Examples:
   - 2, 5, 4, 10, 11; 5 $\rightarrow$ 1
   - 2, 5, 4, 10, 11; 5 $\rightarrow$ 0

1. Problem: sequence of n ($n > 0$) strings $s_1, s_2, s_3,\ .. , s_n$; a single string q
2. Return the index of the found string or 0
3. Examples:
   - Peter, Laura, Markus, Monica; Monica $\rightarrow$ 4
   - Peter, Laura, Markus, Monica; Susanne $\rightarrow$ 0

Outline

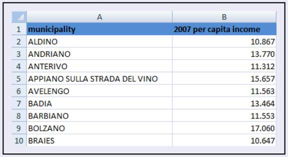
1 Logical Operators

2 Algorithms
- Searching
- Exercises

Exercise

one

Per capita incomes of the year 2007 of the 116 municipalities of the province of Bozen (source: www.ilsole24ore.com) were reported into a worksheet, as shown in the picture below. This data set consists exactly of 116 entries. The VBA range object `Range("A1:B117")` could refer to this data set.



	A	B
1	municipality	2007 per capita income
2	ALDINO	10.867
3	ANDRIANO	13.770
4	ANTERIVO	11.312
5	APPIANO SULLA STRADA DEL VINO	15.657
6	AVELENGO	11.563
7	BADIA	13.464
8	BARBIANO	11.553
9	BOLZANO	17.060
10	BRAIES	10.647

A VBA tool should be implemented in order to extract the list of municipalities that had per capita income belonging to an income range, provided by the user. The tool should perform the following tasks:

1. Ask the user for the income range: lower and upper bound (for example 9000, 10000)
2. Select municipalities that had per capita income belonging to the income range
3. Display in a message box one by one the selected names and incomes

Exercises

1/2

```
Sub perCapita()
    'it displays one by one, municipalities that had a per capita income
    'that belongs to the range provided by the user
    Dim list As Range
    Set list = Worksheets(1).Range("A2:B117")
    Dim lower As Single, upper As Single
    lower = Val(InputBox("Please, the lower"))
    upper = Val(InputBox("Please, the upper"))
    Call selection(list, lower, upper)
End Sub
```

Exercises

2/2

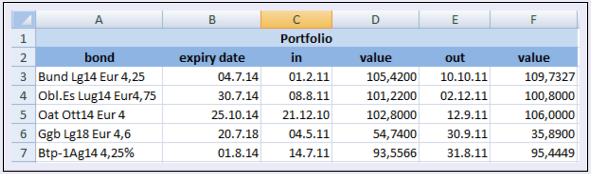
```
Sub selection(source As Range, ByVal low As Double, ByVal up As Double)
    'given the list of the entries
    'the subroutine cretes an array of integer: each slot represents one municipality
    'the array holds:
    '1, if the municipality had an income between "lower" and "upper", 0 otherwise
    'it calls a subroutine to display the result of the selction
    Dim subset(1 To 116) As Long
    Dim r As Long
    For r = 1 To source.Rows.count Step 1
        If (source.Cells(r, 2).Value >= low And source.Cells(r, 2).Value <= up) Then
            subset(r) = 1
        Else
            subset(r) = 0
        End If
    Next r
    Call selected(subset, source)
End Sub

Sub selected(sel() As Long, s As Range)
    'given the array (sel) where the selected municipalities (0/1) are held
    'given the list (s)
    'it displays one by one, name and income of each selected municipality
    Dim i As Integer
    For i = LBound(sel) To UBound(sel)
      If sel(i) = 1 Then
         MsgBox (s.Cells(i, 1).Value & " - " & s.Cells(i, 2).Value)
    End If
    Next i
End Sub
```

Exercise

two

The picture below reports a typical data set about bonds, no more available in your portfolio. For each entry is reported the name, the expiry date, the date of buy (in), the corresponding value, and the date of sell (out) and the corresponding value. This data set has no more than 10 entries. The VBA range object `Range("A3:F7")` could refer to this data set.



	A	B	C	D	E	F
1	Portfolio					
2	bond	expiry date	in	value	out	value
3	Bund Lg14 Eur 4,25	04.7.14	01.2.11	105,4200	10.10.11	109,7327
4	Obl.Es Lug14 Eur4,75	30.7.14	08.8.11	101,2200	02.12.11	100,8000
5	Oat Ott14 Eur 4	25.10.14	21.12.10	102,8000	12.9.11	106,0000
6	Ggb Lg18 Eur 4,6	20.7.18	04.5.11	54,7400	30.9.11	35,8900
7	Btp-1Ag14 4,25%	01.8.14	14.7.11	93,5566	31.8.11	95,4449

A VBA tool should be provided in order to return the following information:

1. The bond that had the maximum performance (i.e. maximum (sell value - buy value))
2. The bond held for the shortest time

Exercises

1/3

```
Sub portfolio()
    'the tool computes and displays the bond yielded the maximum performance
    'then the bond held the shortest time
    Dim source As Range
    Set source = Worksheets(1).Range("A3:F7")
    Call maxPerf(source)
    Call shortest(source)
End Sub
```

Exercises

2/3

```
Function performance(ByVal valueIn As Single, ByVal valueOut As Single) As Single
    'if buy value and sell value are valid
    'then the function computes the performance (valueOut - valueIn)
    'otherwise it returns 0
    If valueIn > 0 And valueOut > 0 Then
        performance = valueOut - valueIn
    Else
        performance = 0
    End If
End Function

Sub maxPerf(table As Range)
    'given the the portfolio
    'it determines the bond that yielded the best performance
    Dim maxPerf As Integer
    Dim t(1 To 10) As Single
    Dim r As Long
    For r = 1 To 10
        t(r) = performance(table.Cells(r, 4).Value, table.Cells(r, 6).Value)
    Next r
    maxPerf = 1
    For r = 2 To 10
        If t(r) > t(maxPerf) Then
            maxPerf = r
        End If
    Next r
    MsgBox ("bonds: " & table.Cells(maxPerf, 1) & " - performance: " & t(maxPerf))
End Sub
```

Exercises

3/3

```
Function time(ByVal timeIn As Date, ByVal timOut As Date) As Long
    'the function returns time in hours between two hours
    time = DateDiff("h", timeOut, timeIn)
End Function

Sub shortest(table As Range)
    'given the the portfolio
    'it determines the bond held the shortest time
    Dim minTime As Integer
    Dim t(1 To 10) As Long 'time in hours
    Dim r As Long
    For r = 1 To table.Rows.count
        t(r) = time(table.Cells(r, 3).Value, table.Cells(r, 5).Value)
    Next r
    minTime = 1
    For r = 2 To 10
        If t(r) > 0 And t(r) < t(minTime) Then
            minTime = r
        End If
    Next r
    MsgBox ("bonds: " & table.Cells(minTime, 1) & " - time: " & t(minTime))
End Sub
```