Ruggero Cerino

Strumenti informatici per indici dei nomi. Come creare un indice analitico con MS-WORD

citare come: R. Cerino, *Strumenti informatici per indici dei nomi. Come creare un indice analitico con MS-WORD*, in «Laboratorio dell'ISPF», VIII, 2011, 1-2, pp. 118-131.
http://www.ispf-lab.cnr.it/2011_1-2_ 303.pdf

Laboratorio dell'ISPF
ISSN 1824-9817

Il problema

Un tipico problema di chi si occupa di curare l'edizione di un libro o anche semplicemente di scrivere una monografia è la realizzazione dell'indice dei nomi. Questo altro non è che un Indice Analitico, cioè una lista di parole chiave posta solitamente alla fine di un documento, che permette al lettore di trovare facilmente gli argomenti che lo interessano. Nel nostro caso le parole chiave sono i nomi degli autori citati nel testo. In genere si procede manualmente con grandi difficoltà. Le soluzioni che si trovano in rete prevedono tutte che la lista di parole chiave da ricercare nel testo sia nota a priori. Per chi si occupa della cura di un testo, ma anche per chi materialmente lo ha scritto, è però difficile ricordare a memoria i nomi degli autori citati.

Il programma WORD (Microsoft Word) offre una soluzione interna, che prevede la marcatura delle parole destinate a comporre la lista durante la digitazione del testo; in un secondo momento si può lanciare la procedura che completerà il tutto con l'aggiunta dei numeri di pagina.

Il problema è quindi trovare o meglio elaborare una soluzione che operi in post-produzione, creando in modo automatico la lista dei presunti nomi[1].

La soluzione proposta

Per ipotizzare una possibile soluzione conviene evitare di preoccuparsi in un primo momento dei dettagli implementativi e cercare di affrontare la questione col tipico metodo di dividere un problema complesso in problemi di più facile soluzione (la tecnica del *divide et impera*[2]).

Immaginiamo di avere un esecutore che sia in grado di eseguire poche ma fondamentali operazioni:

- leggere un file WORD
- scrivere in un file WORD
- riconoscere schemi di parole da una sequenza di caratteri
- gestire indici (indice inteso come una tabella di elementi)
    - creare un indice
    - aggiungere un elemento dall'indice
    - ricavare un elemento dall'indice
    - cancellare un elemento dall'indice
    - ordinare un indice in base ai valori o alle chiavi

[1] Un tentativo è già stato proposto anni fa su questa rivista da L. Pica Ciamarra, *Tecniche di indicizzazione di un testo elettronico*, in «Laboratorio dell'ISPF», III, 2006, 2, pp. 69-101; la soluzione avanzata di seguito supera alcuni limiti di quell'approccio e offre nuove funzioni.

[2] In informatica approccio molto efficace per la risoluzione di vari problemi: si divide ricorsivamente un problema in due o più sotto-problemi sino a che questi ultimi diventino di semplice risoluzione, quindi si combinano le soluzioni al fine di ottenere la soluzione del problema dato: cfr. http://it.wikipedia.org/wiki/Divide_et_impera_%28informatica%29.

Il nostro esecutore, inoltre, è in grado di eseguire tali azioni secondo l'ordine e con le modalità richieste, vale a dire che è in grado si eseguire un *algoritmo*[3]. In pratica, possiamo pensare al nostro esecutore come a un mediocre cuoco in grado di seguire alla lettera una ricetta.

Iniziamo con il considerare in che modo procederebbe un operatore umano. Costui svolgerebbe nell'ordine una serie di operazioni:

1. leggere il libro, ponendo attenzione alle sequenze di carattere che possono essere assimilabili ad un nome proprio (magari tutte quelle che iniziano con una maiuscola);
2. trovato il nome potenziale, confrontarlo con la liste delle parole da escludere;
3. se il nome è composto dal solo cognome, confrontarlo con una lista di verosimili integrazioni (supponiamo di avere una lista nata da passate esperienze, che per un dato cognome propone diverse possibili soluzioni per il nome);
4. prendere nota della pagina dove il nome compare ed eventualmente se esso è presente nel testo e/o nella nota;
5. ordinare la lista cosi ottenuta e stampare il risultato.

Ispirandoci a questa descrizione del comportamento di un operatore umano, possiamo pensare ora ad una soluzione in cui lo sostituiamo con il nostro esecutore automatico, adattando ovviamente alcune azioni alle capacità di cui quest'ultimo è dotato.

1. Carico la lista alias (nomi suggeriti):
   prelevo suggerimenti contenuti in un file ascii (testo) formattato in modo che ogni riga contiene il cognome trovato separato tramite il segno *pipe* "|" dal nominativo sciolto (esempio: Cerino | Cerino Ruggero).
2. Carico il filtro (la lista delle parole da escludere):
   apro un file testo con il quale creo un stringa secondo le regole delle espressioni regolari formata dalle parole che vogliamo escludere (esempio: [Una][Uno]…).
3. Inizio la ricerca nomi nel testo:
   facendo uso delle espressioni regolari cerco nel testo i possibili schemi di caratteri che posso assimilare a potenziali nomi filtrando quelli che appartengono alla tabella delle parole escluse (Filtro).

[3] Un algoritmo si può definire come un procedimento che consente di ottenere un risultato atteso eseguendo, in un determinato ordine, un insieme di passi semplici corrispondenti ad azioni scelte solitamente da un insieme finito di passi: cfr. http://it.wikipedia.org/wiki/Algoritmo.

4. Ricavo il numero pagina:
   partendo dalle parole contenute nella tabella dei nomi costruita nel punto precedente passo a ricercare per ognuna di esse una tabella contenente le pagine dove queste appaiono nel testo e/o nelle note.
5. Stampo l'indice:
   al termine creo un nuovo file word contente l'indice creato proponendo per i nomi per i quali ho il riferimento nella lista alias i potenziali nomi sciolti seguiti dai numeri di pagina dove compaiono.

Il nostro esecutore ha la capacità di riconoscere gli schemi di parole nelle sequenze di caratteri secondo i *pattern* definiti per mezzo delle espressioni regolari[4]. Quindi l'indice delle concordanze nel punto 3 si crea semplicemente ricercando negli schemi di simboli (stringhe di caratteri) quelle sequenze che si possono presumibilmente assimilare ad un nome.

Ad esempio con il seguente pattern:

"\b[A-Z][a-z]{1,}\b"

indico al mio esecutore di ricercare stringhe nel testo il cui primo carattere sia maiuscolo. Questo però non risolve problemi di nomi composti come "Della Torre", che non verrebbe trovato; o meglio, troveremmo "Della" e "Torre", dove la prima parola però verrebbe persa nel filtro delle parole da escludere. Ovviamente è possibile raffinare ulteriormente lo schema da utilizzare o meglio definire uno schema predefinito e dare insieme la possibilità di definirne di propri.

Implementazione

Per implementare la soluzione appena esposta ho pensato di far riferimento allo stesso MS Word, in virtù del fatto che tale software è dotato della possibilità di automatizzare operazioni semplicemente facendo uso di macro. Dal menu Strumenti->Macro->Visual Basic Editor accedo quindi all'editor di codice e da qui posso impartire le istruzioni che il mio esecutore (WORD) dovrà eseguire.

Il listato che segue alle prossime pagine può essere copiato e incollato nell'editor delle macro. Prima di poter eseguire lo script però sono necessari alcuni accorgimenti.

[4] Una espressione regolare definisce una funzione che prende in ingresso una stringa, e restituisce in uscita un valore del tipo sì/no, a seconda che la stringa segua o meno un certo *pattern*. Le espressioni regolari sono utilizzate principalmente da editor di testo per la ricerca e la sostituzione di porzioni del testo: cfr. http://it.wikipedia.org/wiki/Espressione_regolare.

In primo luogo vanno creati – nella stessa cartella che contiene il documento che si intende elaborare – i file *alias.txt*, contenente i nomi di battesimo suggeriti per i cognomi trovati, e *filtro.txt*, contenente la lista delle parole da escludere. I file, di cui di seguito riporto un esempio, vanno creati usando un editor tipo notepad (dal menu Start di Windows: Tutti i programmi->Accessori->Blocco note).

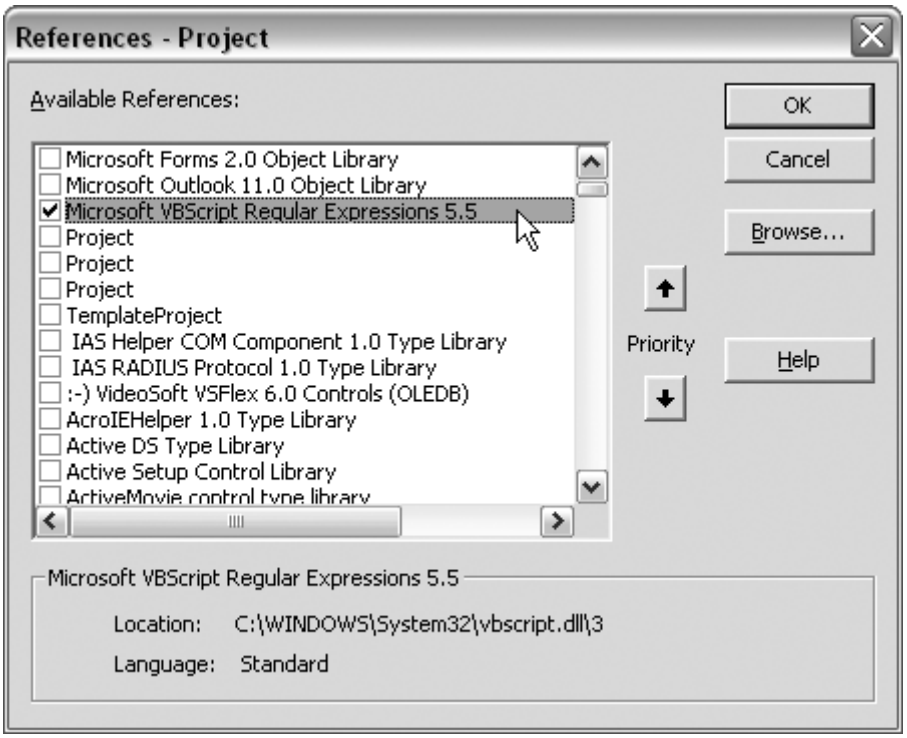
Esempio di file *alias.txt*:

```
Sanna|Sanna, Manuela
Mazzola|Mazzola, Roberto
Cerino|Cerino, Ruggero
```

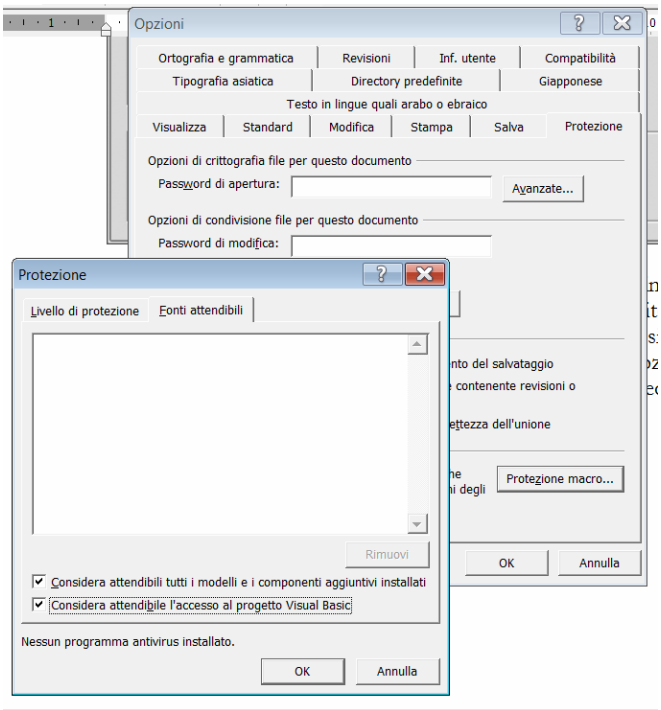
Esempio di file *filtro.txt*:

```
Anche
Ancora
Annali
```

In secondo luogo, dal momento che lo script realizzato fa uso delle "espressioni regolari", queste vanno attivate nel seguente modo: si apre il file di cui occorre fare l'indice, si entra nel Visual Basic Editor selezionando dal menu Strumenti->Macro->Visual Basic Editor; si seleziona Strumenti->Riferimenti; quindi si attiva "Microsoft VBScript Regular Expressions 5.5" come mostrato in figura.



Ancora, è bene accertarsi che le impostazioni sulla sicurezza di Word non blocchino le macro. Se così fosse, si può ovviare andando in Strumenti->Opzioni->Protezione pulsante "Protezione Macro" e quindi impostare a "Medio" il livello di protezione e spuntare la casella "considerare attendibile l'accesso al progetto Visual Basic" come mostrato in figura.



A questo punto si può procedere a copiare lo script. Una volta aperto l'editor (Strumenti->Macro->Visual Basic Editor) – sempre dal file che si desidera elaborare – basta selezionare l'intero listato che segue, copiarlo e incollarlo nell'editor stesso, salvando poi le modifiche. Il listato va inserito in "This Document" (facendo doppio click), sotto la voce "Project"[nome_file] che appare nel riquadro a sinistra.

```
'** FUNZIONI AUSILIARIE
'Controlla se un vettore è vuoto
'param a vettore da controllare
'output valore booleano vero (vuoto) o falso (non vuoto)
Public Function IsArrayEmpty(ByRef a As Variant) As Boolean
On Error Resume Next
   IsArrayEmpty = (a(0) = a(0))
   IsArrayEmpty = (Err.Number = 9)
End Function
'Ordina un dizionario in base alle chiavi o ai valori
'param objDict dizionario da ordinare
'param keyorvalur se ordinare in base alla chiave (se vale 1) o al valore
(se vale 2) del dizionario
Function SortDictionary(objDict, keyorvalue)
```

```
Dim strDict()
Dim objKey
Dim strKey, strItem
Dim i, j, h
   h = objDict.Count
   If h > 1 Then
      ReDim strDict(h, 2)
      i = 0
      For Each objKey In objDict
         strDict(i, 1) = CStr(objKey)
         strDict(i, 2) = CStr(objDict(objKey))
         i = i + 1
      Next
      For i = 0 To (h - 2)
         For j = i To (h - 1)
            If StrComp(strDict(i, keyorvalue), strDict(j, keyorvalue),
vbTextCompare) > 0 Then
               strKey = strDict(i, 1)
               strItem = strDict(i, 2)
               strDict(i, 1) = strDict(j, 1)
               strDict(i, 2) = strDict(j, 2)
               strDict(j, 1) = strKey
               strDict(j, 2) = strItem
            End If
         Next
      Next
      objDict.RemoveAll
      For i = 0 To (h - 1)
         objDict.Add strDict(i, 1), strDict(i, 2)
      Next
   End If
End Function

'Ordino in base alla chiave gli elementi di un dizionario
'Param objDict: dizionario dei numeri di pagine per la voce considerata
Function SortNumPag(objDict)
Dim strDict()
Dim objKey
Dim strKey, strItem
Dim i, j, h
   h = objDict.Count
   If h > 1 Then
      ReDim strDict(h, 2)
      i = 0
      For Each objKey In objDict
         strDict(i, 1) = CInt(objKey)
         strDict(i, 2) = CStr(objDict(objKey))
         i = i + 1
      Next
      For i = 0 To (h - 2)
         For j = i To (h - 1)
            If strDict(i, 1) > strDict(j, 1) Then
               strKey = strDict(i, 1)
               strItem = strDict(i, 2)
               strDict(i, 1) = strDict(j, 1)
               strDict(i, 2) = strDict(j, 2)
               strDict(j, 1) = strKey
               strDict(j, 2) = strItem
            End If
         Next
      Next
      objDict.RemoveAll
      For i = 0 To (h - 1)
         objDict.Add strDict(i, 1), strDict(i, 2)
      Next
   End If
End Function
```

```
'Funzione principale che si ispira alla soluzione proposta
Sub indice()
Dim strToSearch As String ' stringa atta a contenere il testo del corpo
Dim strToSearchFootnotes As String ' stringa atta a contenere il testo delle
note
Dim myrange As Range ' testo selezionato nel file word
Dim arrayVociIndice() As String ' tabella dei presunti nomi
Dim arrayVociAlias() As String ' tabella dei nomi suggeriti
Dim arrayVociIndicePagine() As String ' tabella delle pagine rilevate per il
relativo nome
Dim Excludes As String ' parole da escludere
On Error GoTo errhandler ' rinvia al gestore degli errori
   'Ricavo testo in nota
   Set myrange = ActiveDocument.StoryRanges(wdFootnotesStory)
   strToSearchFootnotes = ActiveDocument.StoryRanges(wdFootnotesStory).Text
RetErr5941:
   'Ricavo testo dal corpo
   strToSearch = ActiveDocument.Content.Text
   Application.ScreenUpdating = False
   System.Cursor = wdCursorWait
   Application.StatusBar = "Please Wait..."
   alias ActiveDocument.Path & "\alias.txt", arrayVociIndice, arrayVociAlias
' aggiunge i nomi suggeriti alla tabella dei nomi e crea una tabella di nomi
sciolti secondo i suggerimenti
   Excludes = Filtro(ActiveDocument.Path & "\filtro.txt") ' crea una stringa
secondo gli schemi delle espressioni regolari delle parole che voglio che
siano escluse
   Concordanze strToSearch, arrayVociIndice, Excludes ' eseguo la ricerca
dei nomi nel corpo del testo
   Concordanze strToSearchFootnotes, arrayVociIndice, Excludes ' eseguo la
ricerca dei nomi nelle note
   If Not (IsArrayEmpty(arrayVociIndice)) Then ' se la tabella dei nomi non
è vuota
      ReDim Preserve arrayVociIndicePagine(UBound(arrayVociIndice)) '
ridimensiono la tabella delle pagine in modo che sia della stessa dimensione
della tabella delle voci trovate
      NumeraPagine2 arrayVociIndice, arrayVociIndicePagine ' ricavo il
numero di pagine
      Stampa arrayVociIndice, arrayVociAlias, "", arrayVociIndicePagine '
creo un nuovo file word dovo riporto i nomi trovati
   Else
      MsgBox "Didn't match anything. Try again."
   End If
   Application.ScreenUpdating = True
       Application.StatusBar = "Task completed"
       System.Cursor = wdCursorNormal
       MsgBox "Task completed"
       Exit Sub
errhandler:
   If Err = 5941 Then ' nel caso non vi sia testo nelle note
      MsgBox "The footnotes story is not available."
      strToSearchFootnotes = ""
      Resume RetErr5941
   ElseIf Err.Number <> 0 Then
      MsgBox Err.Number & " - " & Err.Description & " in sub indice"
   End If
End Sub

'preleva i suggerimenti contenuti in un file ascii (testo) formattato in
modo che ogni
'riga contenga il cognome suggerito separato tramite il segno pipe '|' dal
nominativo sciolto
'esempio: Cerino | Cerino Ruggero
'param strFileAlias: nome file da caricare nella tabella
'param arrayVociIndice: tabella contenente i nomi
'param arrayVociAlias: tabella contenente i nomi sciolti
```

```
Sub alias(strFileAlias As String, ByRef arrayVociIndice() As String, ByRef
arrayVociAlias() As String)
On Error GoTo errhandler
Dim fs, f, ts
Dim s As String
Dim pos As Long
Dim i As Integer
Dim j As Integer
Dim flag As Boolean
Const ForReading = 1, ForWriting = 2, ForAppending = 3
Const TristateUseDefault = -2, TristateTrue = -1, TristateFalse = 0
   Set fs = CreateObject("Scripting.FileSystemObject")
   Set f = fs.GetFile(strFileAlias)
   Set ts = f.OpenAsTextStream(ForReading, TristateUseDefault)
   If IsArrayEmpty(arrayVociIndice) Then
      i = 0
   Else
      i = UBound(arrayVociIndice) + 1
   End If
   Do While Not ts.AtEndOfStream
      s = ts.ReadLine
      pos = InStr(1, s, "|", 1)
      If Not (IsArrayEmpty(arrayVociIndice)) Then
         flag = False
         For j = 0 To UBound(arrayVociIndice)
            If arrayVociIndice(j) = Trim(Left(s, pos - 1)) Then
               flag = True
               Exit For
            End If
         Next j
         If flag = False Then
            ReDim Preserve arrayVociIndice(UBound(arrayVociIndice) + 1)
            arrayVociIndice(i) = Trim(Left(s, pos - 1))
            ReDim Preserve arrayVociAlias(UBound(arrayVociIndice) + 1)
            arrayVociAlias(i) = Trim(Right(s, Len(s) - pos))
            i = i + 1
         Else
            arrayVociAlias(j) = arrayVociAlias(j) & "/" & Trim(Right(s,
Len(s) - pos))
         End If
      Else
         ReDim Preserve arrayVociIndice(0)
         arrayVociIndice(i) = Trim(Left(s, pos - 1))
         ReDim Preserve arrayVociAlias(0)
         arrayVociAlias(i) = Trim(Right(s, Len(s) - pos))
         i = i + 1
      End If
   Loop
   ts.Close
   Set fd = Nothing
   Exit Sub
errhandler:
   If Err.Number = 5 Then
      MsgBox Err.Number & " - " & Err.Description & " Controllare file alias
alla voce" & s
      Exit Sub
   ElseIf Err.Number <> 0 Then
      MsgBox Err.Number & " - " & Err.Description & " in sub alias"
   End If
End Sub

'Creo una stringa secondo le regole delle espressioni regolari
'formata dalle parole che vogliamo escludere
'Param strFile: nome del file le cui righe sono le parole da escludere
Function Filtro(ByVal strFile As String) As String
On Error GoTo errhandler
Dim fs, f, ts
```

```
Dim s As String
Const ForReading = 1, ForWriting = 2, ForAppending = 3
Const TristateUseDefault = -2, TristateTrue = -1, TristateFalse = 0
       Set fs = CreateObject("Scripting.FileSystemObject")
       Set f = fs.GetFile(strFile)
       Set ts = f.OpenAsTextStream(ForReading, TristateUseDefault)
   Do While Not ts.AtEndOfStream
      s = s & "[" & ts.ReadLine & "]"
   Loop
   ts.Close
   Set fd = Nothing
   Filtro = s
   Exit Function
errhandler:
   If Err.Number = 5 Then
      MsgBox Err.Number & " - " & Err.Description & " Controllare file alias
alla voce" & s
      Exit Function
   ElseIf Err.Number <> 0 Then
      MsgBox Err.Number & " - " & Err.Description & " in sub alias"
   End If
End Function

'Facendo uso delle espressione regolari cerco nel testo i possibili schemi
di caratteri che posso
'assimilare a potenziali nomi filtrando quelli che appartengono alla tabella
delle parole escluse
'param strToSearch: testo su cui ricercare
'param arrayVociIndice: tabella dei nomi potenziali aggiunti a quelli già
trovati
'param Excludes: tabella parole da escludere
Sub Concordanze(ByRef strToSearch As String, ByRef arrayVociIndice() As
String, ByRef Excludes As String)
Dim re As RegExp
Dim strPattern As String
Dim oMatches As MatchCollection
Dim oMatch As Match
Dim Umlaut As String
Umlaut = "ÀÁÂÃÄÅÆÇÈÉÊËÌÍÎÏÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞß" ' per gestire caratteri
particolari maiuscoli
Dim Umlautmin As String
Umlautmin = "àáâãäåæçèéêëìíîïðñòóôõö÷øùúûüýþÿ" ' per gestire caratteri
particolari minuscoli
   Set re = New RegExp
   re.Global = True
   re.IgnoreCase = False
   re.Multiline = True
   strPattern = "" & _
   "D[a-z" & Umlautmin & "]{0,}['' ]{0,1}([A-Z" & Umlaut & "][a-z" &
Umlautmin & "]{1,}[ ]{0,1}){1,}|" & _
"[A-Z" & Umlaut & "][a-z" & Umlautmin & "]{1,}" & _
""
   re.Pattern = strPattern
   If strToSearch = "" Then Exit Sub
   Set oMatches = re.Execute(strToSearch)
   If oMatches.Count <> 0 Then
      Dim i As Integer
      Dim f As Integer
      Dim flag As Boolean
      If IsArrayEmpty(arrayVociIndice) Then
         i = 0
      Else
            i = UBound(arrayVociIndice) + 1
      End If
      For Each oMatch In oMatches
         If InStr(Excludes, "[" & Trim(oMatch.Value) & "]") = 0 Then
            If Not (IsArrayEmpty(arrayVociIndice)) Then
```

```
                flag = False
                For j = 0 To UBound(arrayVociIndice)
                   If arrayVociIndice(j) = Trim(oMatch.Value) Then
                      flag = True
                      Exit For
                   End If
                Next j
                If flag = False Then
                   ReDim Preserve arrayVociIndice(UBound(arrayVociIndice) + 1)
                   arrayVociIndice(i) = Trim(oMatch.Value)
                   i = i + 1
                End If
             Else
                ReDim Preserve arrayVociIndice(0)
                arrayVociIndice(i) = Trim(oMatch.Value)
                i = i + 1
             End If
          End If
       Next oMatch
    Else
       MsgBox Chr(34) & strPattern & Chr(34) & _
       " didn't match anything. Try again."
    End If
End Sub

'Partendo dalle parole contenute nella tabella dei nomi passo a cercare
'per ognuna di esse una tabella contenente le pagine dove queste appaiono
'nel testo e/o nelle note
'Param arrayVociIndice: tabella nomi
'Param arrayVociIndicePagine: tabella contenente le pagine dove compare il
nome relativo
Sub NumeraPagine2(ByRef arrayVociIndice() As String, ByRef
arrayVociIndicePagine() As String)
On Error GoTo errhandler
Dim myrange As Range
Dim numeroPagina As String
Dim Dic, Dickeys, j
Set Dic = CreateObject("Scripting.Dictionary")
Dim Salta As Boolean
   numeroPagina = ""
   Salta = False
   For i = 0 To UBound(arrayVociIndice)
      With ActiveDocument.Content.Find
      .ClearFormatting
      .Forward = True
      .MatchCase = True
      .Format = False
      .MatchWholeWord = True
      Do While .Execute(FindText:=arrayVociIndice(i)) = True
         With .Parent
         .Select
            numeroPagina = Selection.Information(wdActiveEndPageNumber)
            If Dic.Exists(numeroPagina) Then
               If (Selection.Font.Color = wdColorRed) Then
                  If Dic(numeroPagina) = (numeroPagina) Then
                     Dic(numeroPagina) = numeroPagina & " e n"
                  End If
               Else
                  If Dic(numeroPagina) = (numeroPagina & "n") Then
                     Dic(numeroPagina) = numeroPagina & " e n"
                  End If
               End If
            Else
               If (Selection.Font.Color = wdColorRed) Then
                  Dic.Add numeroPagina, numeroPagina & "n"
               Else
                  Dic.Add numeroPagina, numeroPagina
```

```
                End If
             End If
          End With
       Loop
       End With
       If Not (Salta) Then
          With ActiveDocument.StoryRanges(wdFootnotesStory).Find
          .ClearFormatting
          .Forward = True
          .MatchCase = True
          .Format = False
          .MatchWholeWord = True
          Do While .Execute(FindText:=arrayVociIndice(i)) = True
             With .Parent
             .Select
                numeroPagina = Selection.Information(wdActiveEndPageNumber)
                If Dic.Exists(numeroPagina) Then
                   If Dic(numeroPagina) = (numeroPagina) Then
                      Dic(numeroPagina) = numeroPagina & " e n"
                   ElseIf Dic(numeroPagina) = (numeroPagina & "n") Then
                      Dic(numeroPagina) = numeroPagina & " e n"
                   End If
                Else
                   Dic.Add numeroPagina, numeroPagina & "n"
                End If
             End With
          Loop
          End With
       End If
RetErr5941:
       SortNumPag Dic ' Ordino la tabella contente i numeri delle pagine
       Dickeys = Dic.Keys
       For j = 0 To Dic.Count - 1
          arrayVociIndicePagine(i) = arrayVociIndicePagine(i) &
Dic(Dickeys(j)) & ", "
       Next j
       Dic.RemoveAll
    Next i
errhandler:
    If Err = 5941 Then
       Salta = True
       strToSearchFootnotes = ""
       Resume RetErr5941
    End If
End Sub

Sub Stampa(ByRef arrayVociIndice() As String, ByRef arrayVociAlias() As
String, ByRef strPrefisso As String, ByRef arrayVociIndicePagine() As
String)
Dim i As Integer
        'STAMPA INDICE ANALITICO
        Documents.Add Template:="Normal", NewTemplate:=False, DocumentType:=0
        Selection.TypeText Text:="INDICE ANALITICO"
        Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter
        Selection.TypeText (vbCrLf)
        Selection.TypeParagraph
        Selection.InsertBreak Type:=wdSectionBreakContinuous
        If ActiveWindow.View.SplitSpecial <> wdPaneNone Then
       ActiveWindow.Panes(2).Close
    End If
    If ActiveWindow.ActivePane.View.Type <> wdPrintView Then
       ActiveWindow.ActivePane.View.Type = wdPrintView
    End If
    With Selection.PageSetup.TextColumns
    .SetCount NumColumns:=2
    .EvenlySpaced = True
    .LineBetween = False
```

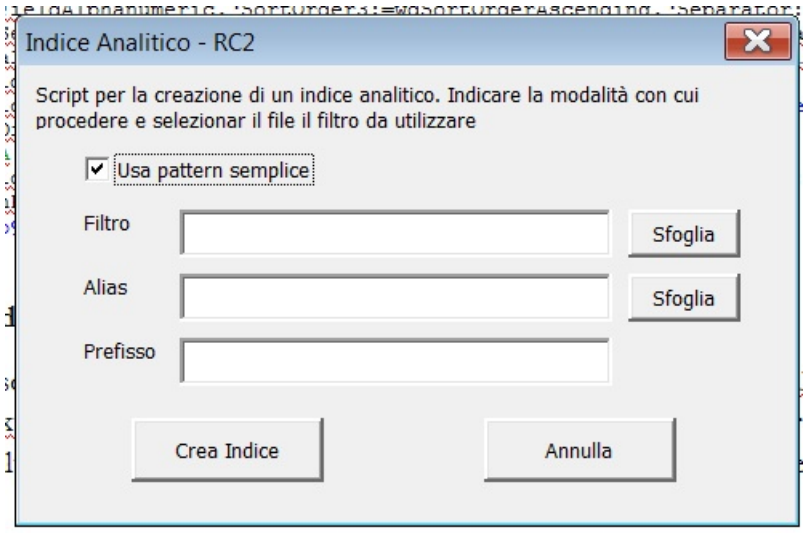
```
   .Width = CentimetersToPoints(7.87)
   .Spacing = CentimetersToPoints(1.25)
   .FlowDirection = wdFlowLtr
   End With
   Selection.ParagraphFormat.Alignment = wdAlignParagraphLeft
   Dim alias As String
   Dim Salta As Boolean
   Dim separa As Boolean
   For i = 0 To UBound(arrayVociIndice)
      alias = ""
      Salta = False
      separa = False
      If i < UBound(arrayVociAlias) Then
         alias = " (" & arrayVociAlias(i) & ") "
         pos = InStr(1, alias, "/", 1)
         If pos > 0 Then
            separa = True
         End If
      End If
      Selection.Font.Color = wdColorAutomatic
      If Len(arrayVociIndicePagine(i)) < 1 Then
         Selection.Font.Color = wdColorRed
         If i < UBound(arrayVociAlias) Then
            Salta = True
         End If
      End If
      If Salta = False Then
         Selection.Font.SmallCaps = wdToggle
         Selection.TypeText (arrayVociIndice(i) & alias)
         Selection.Font.SmallCaps = wdToggle
         Selection.TypeText (", " & strPrefisso & arrayVociIndicePagine(i) &
vbCrLf)
      End If
   Next i
        'Selezione il testo da ordinare supposto di trovarmi alla fine
        Selection.StartOf Unit:=wdSection, Extend:=wdMove
        Selection.EndKey Unit:=wdStory, Extend:=wdExtend
        'ORDINA ELENCO
        Selection.Sort ExcludeHeader:=False, FieldNumber:="Paragrafi", _
        SortFieldType:=wdSortFieldAlphanumeric,
        SortOrder:=wdSortOrderAscending, _
        FieldNumber2:="", SortFieldType2:=wdSortFieldAlphanumeric,
        SortOrder2:= _
        wdSortOrderAscending, FieldNumber3:="", SortFieldType3:= _
        wdSortFieldAlphanumeric, SortOrder3:=wdSortOrderAscending,
        Separator:= _
        wdSortSeparateByTabs, SortColumn:=False, CaseSensitive:=False,
        LanguageID _
        :=wdItalian, SubFieldNumber:="Paragrafi",
        SubFieldNumber2:="Paragrafi", _
        SubFieldNumber3:="Paragrafi"
        Selection.Sort BidiSort:=False, IgnoreThe:=True,
        IgnoreKashida:=False, _
        IgnoreDiacritics:=True, IgnoreHe:=False
        'NUMERA PAGINE
        Selection.Sections(1).Footers(1).PageNumbers.Add
        PageNumberAlignment:= _
        wdAlignPageNumberRight, FirstPage:=True
End Sub
```

Per ottenere l'indice sarà sufficiente aprire il file che si desidera elaborare e avviare la nuova macro, che appare nel menu relativo (Strumenti->Macro ->Macro) col nome appunto di "indice". Se la macro segnalasse errori, è

probabile che i ritorni a capo del listato siano stati falsati nel trasferimento: l'errore sarà visibile in rosso nell'editor e andrà corretto (e la modifica salvata) prima di riavviare la macro.

Considerazioni

La soluzione che ho proposto fa riferimento, per semplicità, ai file *alias.txt* e *filtro.txt* in modo non modificabile. Naturalmente nulla vieta – come ho fatto in un'altra soluzione – di usare un modulo finestra per selezionare tali file facendo uso del metodo show dell'oggetto Form (UserForm1.Show).



Un'altra aggiunta può essere quella di dare la facoltà di selezionare diversi pattern di ricerca magari più complessi, in modo da poter gestire anche il caso dei doppi cognomi.