前所未見的 Excel 寶典

{點、陣列、VBA 運算}

egg

October 23, 2017

目錄

Part III

多維

第 11 章

VBA

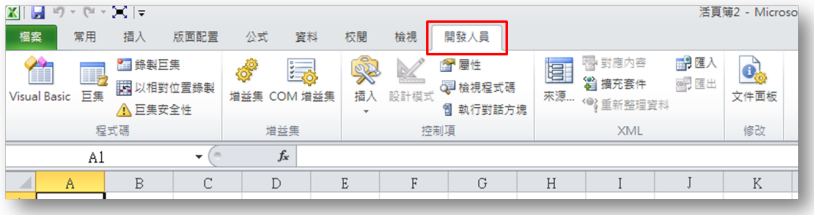
會了陣列的運算, 幾乎可以解決大部分運算的問題, 除非是資料運算量太大或重複性動作太多, 否則陣列運算已經綽綽有餘。既然有資料量與重複動作的問題, 我們就用 Excel VBA 來補救這二個洞, 讓 Excel 更臻完美。

Excel VBA 雖是寫程式讓 Excel 執行, 但它有一個友善的部分是「錄製」, 假設你對語法不熟, 你可以實際操作一遍, 藉由錄製功能來擷取、學習你要的語法, 所以使用者不一定要非常精通程式語言, 可以邊錄邊寫、邊寫邊錄, 以完成使用者所要的目的。

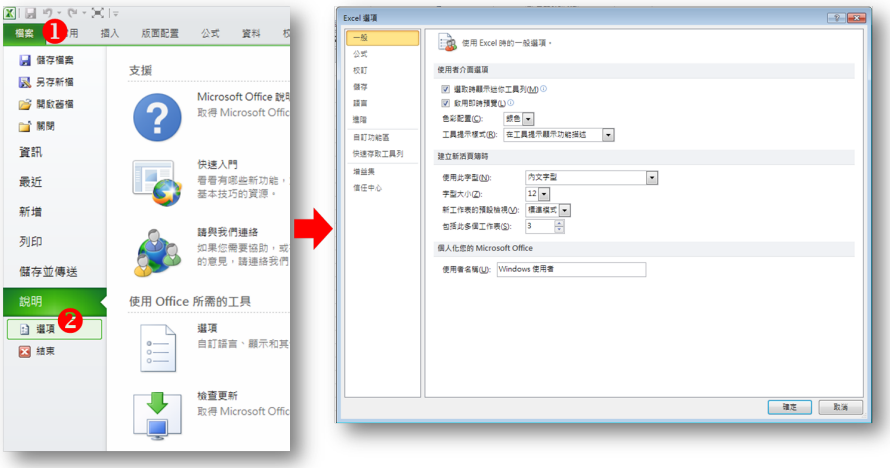
VBA 的功能不是原預設在軟體上, 它被隱藏起來, 要進入這領域前, 我們要先把它的功能叫出來, 所以先介紹設定, 接著先不問青紅皂白照著做, 然後再介紹很基本結構流程, 這裡只是很基礎地介紹, 如果玩出興趣, 可以參照 VBA 專書進入更有趣的領域。

11.1 設定

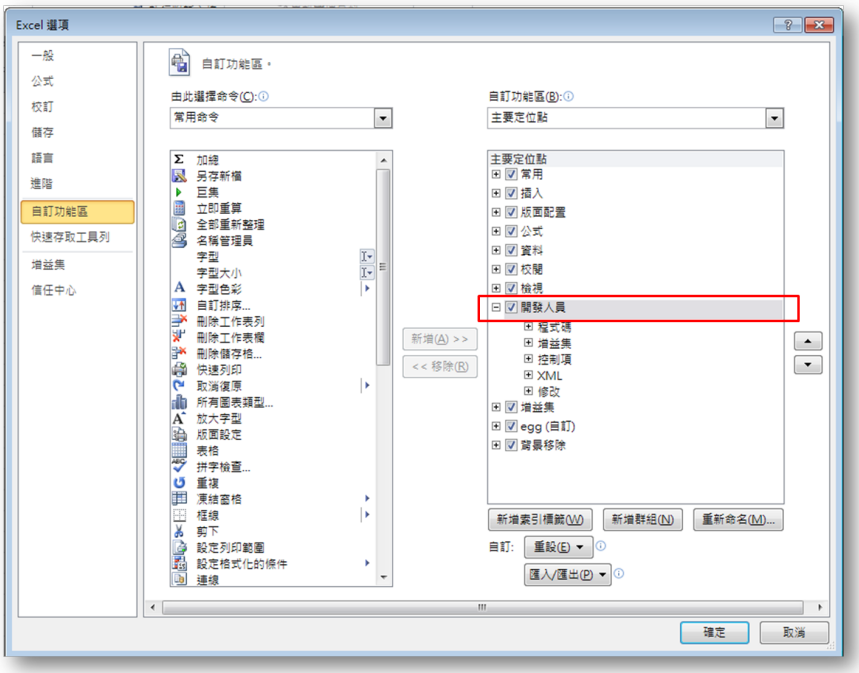
一般預設的 Excel 並沒有 VBA 功能, 想要有 VBA 的功能, 要把工具列的「開發人員」選單叫出來, 以方便之後的操作。



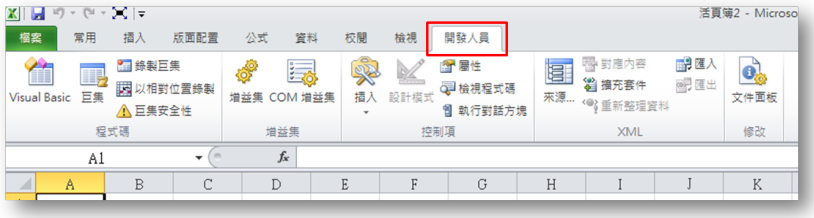
要把「開發人員」選單請出來, 請找到: 檔案 | 選項,按下選項後, 會跳出一「Excel 選項」視窗 如下圖右所示,



在選單區塊點選「自訂功能區」, 右邊自訂功能區(B) 中確認爲「主要定位點」, 選取「開發人員」核取方塊。

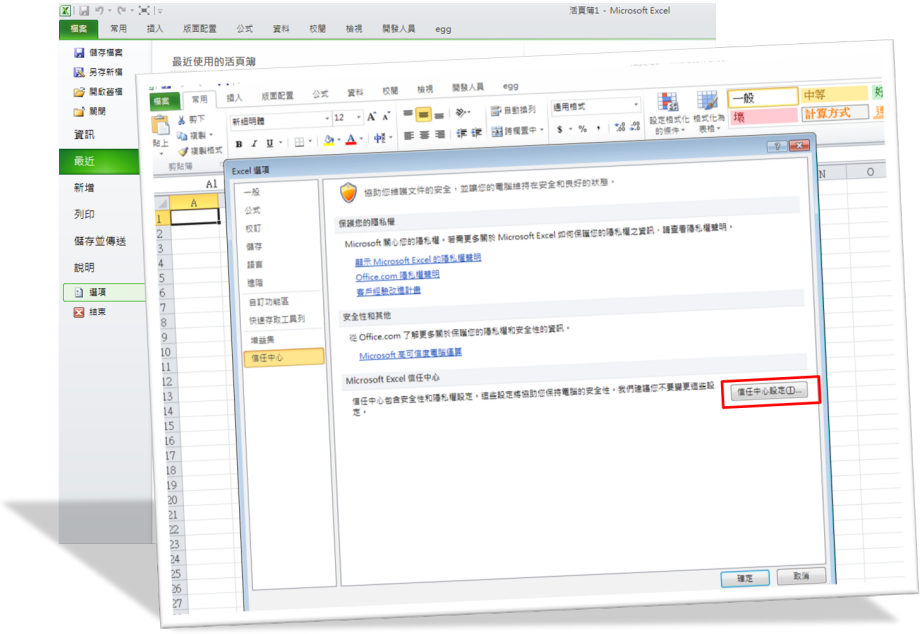


最後選擇「確定」按鈕以關閉「選項」對話方塊, 回到試算表頁, 功能列表列上多了「開發人員」, 如下圖。



開發人員的選單有程式碼、增益集、控制項、XML、修改選單, 其中「程式碼」區塊是撰寫 VBA 程式較常用的功能, 尤其是 Visual Basic、巨集、錄製巨集等三項功能更常用。本章所介紹的功能也主要爲這三個, 若是有要製作按鈕來控制 VBA 程式, 可以點選「控制項」的插入, 我們利用 11.7 來介紹。

另外爲了能順利使用巨集程式, 不會常跳出訊息來問我們安全性問題, 我們把巨集的信任度做以下調整: 檔案 | 選項 | 信任中心 | 信任中心設定 | 巨集設定,

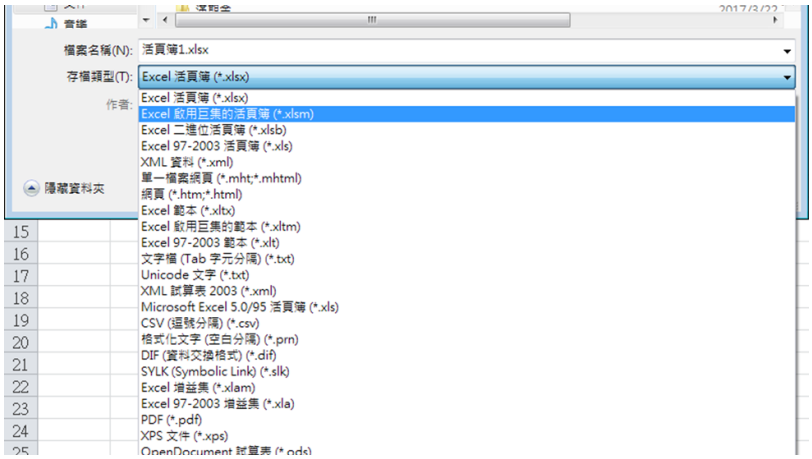


- 啟用所有巨集(不建議使用; 會執行有潛在危險的程式碼)(E)
- 信任存取 VBA 專案物件模型(V)

如果不放心此步驟的人可以不做, 又或者使用完 VBA 後就馬上調回來。

最後是副檔名, Excel 試算表檔案預設副檔名爲 *.xlsx, 但該類型檔案不允許存有巨集或程式, 若是含有巨集程式的檔存成 *.xlsx 檔, 其中程式將不會被保留, 再此開啟時程式會不見。所以要特別注意, 若含有巨集或程式的檔案格式應爲 *.xlsm (啟用巨集的活頁簿), 否則巨集或程式將不被存取, 辛苦寫得程式會因此不見, 實在不得不小心。

要存成 *.xlsm, 只要在存檔時以「另存新檔」模式, 在另存新檔選單中「存檔類型(T):」選「Excel 啟用巨集的活頁簿 (*.xlsm)。



以上設定好之後, 我們可以準備進入巨集或 VBA 程式了。

11.2 錄製

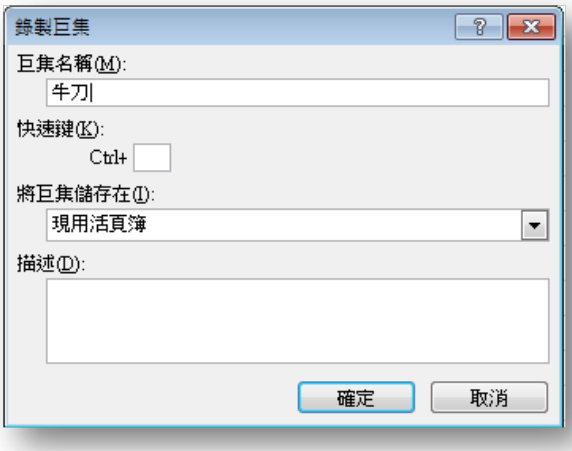
Excel VBA 一個很好的功能是錄製功能, 它可以完全不懂不會程式語法, 按照一般步驟操作, Excel 可以把所有程序步驟錄下來, 供使用者重複自動執行。這節讓我們先看 Excel 的錄製巨集功能, 錄完巨集, 重複執行看看, 體會一下甚麼叫錄製巨集, 接著我們打開所錄製巨集的程式, 試著了解它的內容。

「牛刀」就是我們第一個錄製的巨集, 也是我們要試著解讀的程式。然後應著牛刀寫一個「小試」的最初程式, 試著體驗一下所謂的程式。

11.2.1 牛刀

先錄一個「牛刀」巨集, 巨集內容爲 A1 儲存格等於 1、A2 儲存格等於 2、A3 儲存格等於 3。

在功能列表列選「開發人員」, 在「程式碼」區點選「錄製巨集」。Excel 會跳出一新視窗, 其中有「巨集名稱(M):」預設爲「巨集1」, 請將之改爲「牛刀」,



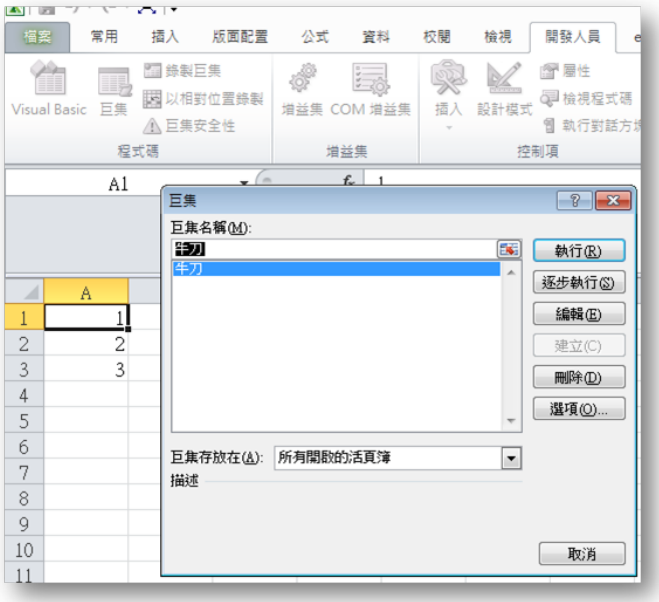
按確定。回到 Excel 試算表中, 接著操作以下程序:

1. 選 A1 儲, 存格在 A1 儲存格輸入 1, 按 Enter;
2. A2 儲存格輸入 2, 按 Enter;
3. A3 儲存格輸入 3, 按 Enter;
4. 再把游標移回 A1 儲存格;
5. 按停止錄製 (開發人員 | 程式碼 | 停止錄製) 。

以上程序在試算表中輸入 1, 2, 3 後, 停止錄製, 整個巨集就錄製完畢。

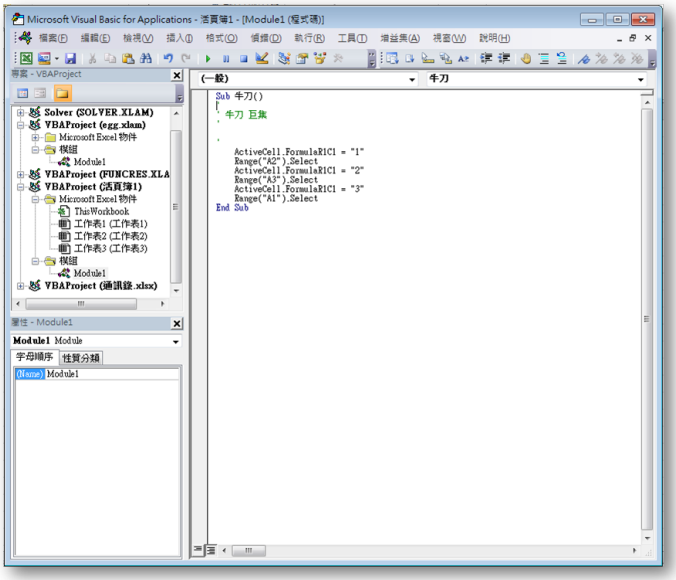
錄好了巨集, 來看看巨集有什麼用。巨集的用處在於可以重複執行, 所以我們就來牛刀一下。牛刀的內容是 A1:A3 爲 {1, 2, 3}, 爲了試它是可以重複執行的, 我們先把 A1:A3 的內容清除, 讓 A1:A3 爲空白。然後把牛刀掏出來, 按一下開發人員 | 程式碼 | 巨集, 跳出一視窗, 選「牛刀」, 再按「執行」, 原先被我們刪除的 A1:A3 儲存格又恢復了 {1, 2, 3} 的內容, 這就是巨集能重複執行的結果。

我們來檢視到底錄了甚麼東西。點選開發人員 | 程式碼 | 巨集, 跳出一視窗,



先選「牛刀」, 再按旁邊「編輯」鈕, 原視窗會轉爲一更大視窗—Microsoft Visual Basic for Applications - 程式碼,

我們得到了程式碼:



我們把其中內容拿出來大概瀏覽一下,

牛刀() VBA

```
Sub 牛刀()
'
'牛刀 巨集
'
'

    ActiveCell.FormulaR1C1 = "1"
    Range("A2").Select
    ActiveCell.FormulaR1C1 = "2"
    Range("A3").Select
    ActiveCell.FormulaR1C1 = "3"
    Range("A1").Select
End Sub
```

開始試圖了解一下我們所錄的程式碼究竟是甚麼, 首先, Sub (子程序) 是以 Sub 這個關鍵字加上一個子程序名稱開始的, 子程序名稱後方會接著一對小括號 (小括號內部可放置參數), 而最後面的 End Sub 就是子程序的結尾, 中間的部分就是子程序的程式內容, 也是我們想執行的內容, 牛刀是我們所取的程式名稱。所以在 Sub 和 End Sub 之間就是所謂的程式碼, 以下我們簡單介紹一下:

程式	說明
'	通常拿來當註解提示, 不會被執行
ActiveCell	使用中儲存格
FormulaR1C1	傳回或設定物件的公式, 並以 R1C1 樣式標記法表示
Range(“A2”)	指 A2 儲存格
Select	游標選到

整體來解釋一下所錄得的程式, 起錄前, 我們的游標停留在 A1 儲存格, 接著按下錄製鍵, 依序說明錄製內容:

程式	說明
Sub 牛刀()	牛刀程式開始
ActiveCell.FormulaR1C1=“1”	游標所在的儲存格 (A1) 設定爲 “1”
Range(“A2”).Select	游標選到 A2 儲存格 (Enter 鍵)
ActiveCell.FormulaR1C1=“2”	游標所在的儲存格 (A2) 設定爲 “2”
Range(“A3”).Select	游標選到 A3 儲存格 (Enter 鍵)
ActiveCell.FormulaR1C1=“3”	游標所在的儲存格 (A3) 設定爲 “3”
Range(“A1”).Select	把游標選到 A1 儲存格
End Sub	程式結束

這正是我們錄製的 {1, 2, 3} 程序。

回到 Microsoft Visual Basci for Applications - 程式碼視窗畫面, 我們可透過鍵盤上「F8」鍵讓 VBA 逐步執行, 每按一次 F8 鍵程式碼視窗會先標記執行的程式碼, 然後就只執行該行程式, 如此功能是最初學者藉由錄製功能去學習程式碼的好方法。

11.2.2 小試

現在我們試著寫一個簡單的程式「小試」來對應牛刀, 依序照下列程序來操作,

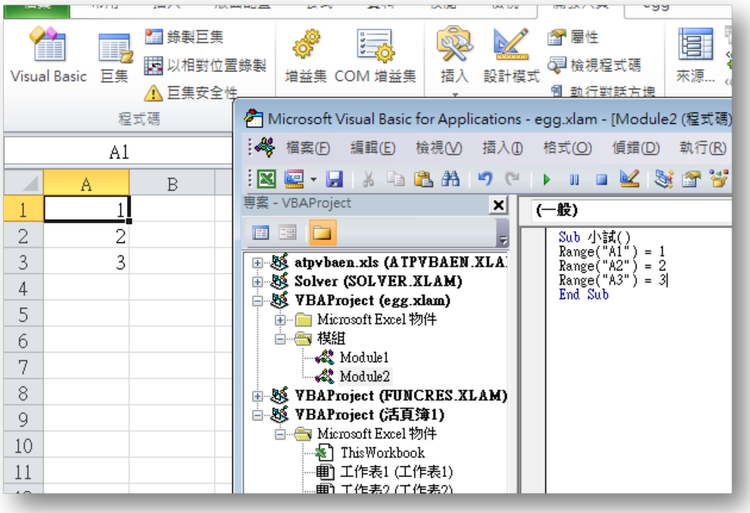
1. 開發人員 | Visual Basic: 開啟一個新編輯視窗。
2. 在新視窗中點選 插入(I) | 模組(M) 指令, 接著視窗的右側會出現空白的程式碼視窗。
3. 在空白的程式碼視窗中運用前小節學到的 Sub 程式和 Range() 儲存格, 輸入以下內容:

小試() **VBA**

```
Sub 小試()
Range("A1")=1
Range("A2")=2
Range("A3")=3
End Sub
```

4. 按 執行(R) | ▶ 執行 Sub 或 UserForm F5 指令

回到試算表視窗, 該分頁的 A1、A2、A3 儲存格已分別被 VBA 鍵入 {1, 2, 3} 了。



這個小試程式與前牛刀程式都可以達到相同的結果, 使用者試著體會一下, 牛刀是以試算表視窗操作概念, 一個動作一個動作執行; 小試是以使用者目的概念, 想要哪個格要多少就多少。對初學者言, 這二個方法沒有優劣勝負, 全看使用者如何發想出發。

同牛刀一樣, 小試程式就是要用來重複執行, 回到試算表中清除 A1:A3 的值, 再執行一次, VBA 會迅速填入 {1, 2, 3}。如果你也想寫一個小程式來清除 A1:A3 儲存格的值, 試著在小試程式下方鍵入以下,

清除() **VBA**

```
Sub 清除()
Range("A1:A3").Clear
End Sub
```

之後一樣按 執行(R) 或 ▶ 來執行, 則 A1:A3 格就會被清除。把游標移「小試」執行, A1:A3 就會出現 {1, 2, 3}; 把游標移到「清除」, A1:A3 的內容就會被清除, 多試幾次, 就很有寫程式重複執行自動化的感覺了。

圖圖

各程式的逐步內容皆可以使用 F8 鍵來看執行狀態。

11.3 宣告變數

變數的宣告是寫程式一個必要程序, 宣告的主要目的是告訴編譯器這個變數屬於哪一種資料型態, 好讓編譯器預先替該變數保留足夠的記憶體空間。宣告的方式很簡單, 就是型態名稱後面接空格, 然後是變數的識別名稱。

Dim 語法

```
Dim 變數名稱 As 資料類型
```

宣告變數是以 Dim 陳述式表示, 定義其資料類型。在語法中, 也可以省略 As 之後的資料類型定義。另外也可以在同一行中以逗號 (,) 分隔數個變數, 一次宣告宣告幾個變數。

資料的型態有:

資料類型	說明
String	字串
Integer, Long, Single, Double	數
Boolean	布林值, True 與 False
其他如 Date, Object, Variant, …	其他

當我們不確定某個變數資料型態時, 可以將變數的資料型態宣告爲不定型變數 Variant, 如此這個變數就可以儲存任何型態的資料了。

以下舉幾個例,

DimTest VBA

```
Sub DimTest()
Dim i As Integer
Dim X as String
Dim mydate, hisdate
End Sub
```

陣列的宣告仍是用 Dim 陳述式, 與一般變數相同, 主要不同點是可以指定大小維度。

Dim 語法

```
Dim 陣列名稱(一維, 二維, 三維, ...)  As 資料類型
```

在這裡要注意的一點是陣列變數的大小是以 0 起算的基底數系, 當我們宣告 AA(100) As Integer, 其代表自 0 至 100, 共具有 101 個元素。

11.4 流程控制

在 VBA 程式中, 通常是由其自動執行接下的程序, 所以免不了加入適當的條件判斷式, 以決定該執行何程序。通常判斷式以 IF 作爲開頭的分歧條件陳述式, 當條件爲真 (True) 或假 (False) 時的程式流程, 好繼續進行接下來的程序。

11.4.1 單一選擇結構

單一選擇結構就是 If…Then 結構, 在條件式成立時, 去執行陳述式 (限單一陳述式), 語法如下, 若條件不成立, 跳過陳述式。

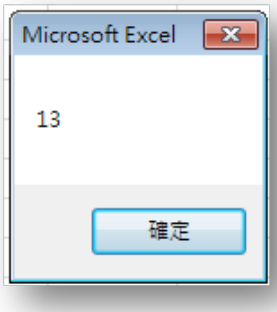
IF…Then 語法

```
If 條件式 Then 陳述式
```

IfTest1 VBA

```
Sub IfTest1()
A=5
B=3
If A>B Then A=B+10
MsgBox A
End Sub
```

程式預設, 假如 A>B, 則去執行 A=B+10, 然後 MsgBox 爲輸出 A。因爲 A=5 的確大於 B, 所以 A 就會被指定爲等於 B+10, 最後爲 13, 是以跳出視窗顯示 13。



若要執行超過一個陳述式, 陳述式不能直接接在 Then 之後同屬一行, 要在 Then 之後斷行, 重新起新一行接續陳述式, 也要在陳述完成後, 多一行 End If 指令。

IF . . . Then 語法

```
If 條件式 Then
  陳述式 1
  陳述式 2
  ...
End If
```

Then 之後的陳述式爲 2 個以上, 自 Then 之後就要換行, 每個陳述式單獨爲一行, 直到陳述式完要在下一行加 “End If”。舉個例子,

IfTest2 VBA

```
Sub IfTest2()
A=5:  B=3
If A>B Then
  A=A+10
  B=0
End If
MsgBox "A=" & A & Chr(10) & "B=" & B
End Sub
```

A=5 和 B=3 要分開二行輸入, 若寫在一行, 中間要用冒號 (:) 隔開。如果 A 大於 B, 執行 A 被指定爲原始 A+10, 等於 5 + 10, 最後答案 A=15; 而 B 就被指定爲 0, 不管原先是多少。之後用 MsgBox 輸出 A 和 B, 其中用 Chr(10)

用以來輸出視窗強制換行。

11.4.2 雙向選擇結構

雙向結構 用 IF ... Then ... Else 來判斷結構, 成立之下續行陳述 1, 不成立則執行陳述 2, 所有的指令都寫在同一行, 即可避免輸入 "'End If"。

IF ... Then ... Else 語法

```
If 條件式 Then 陳述式 1 Else 陳述式 2
```

舉例,

IfTest3 VBA

```
Sub IfTest3()
If A>B Then A=A+10 Else B=B+10
End Sub
```

以上程式內容大概依字面上可以解讀出來, 若 A 大於 B, A 加 10; 若 A 不大於 B, 則 B 加 10。

若要執行多個陳述式, 則拆開來多行陳述, 自 Then 之後斷行, 用 Else 轉折, 最後別忘了加 End If 結束。

IF ... Then ... Else 語法

```
If 條件式 Then
  陳述式區塊 1
Else
  陳述式區塊 2
End If
```

舉例,

IfTest4 VBA

```
Sub IfTest4()
MyAge=InputBox(“輸入年齡”)
If MyAge>20 Then
   MsgBox "可以抽菸"
Else
   MsgBox "抓起來關"
End If
End Sub
```

InputBox 顯示一個視窗接收使用者對話方塊資訊, 輸入之後, 按確定, 電腦把輸入的數字定義爲 MyAge 變數, 輸入的數字若超過 20, 則跳出「可以抽菸」對話框, 若數字小於等於 20, 對話框會出現「抓起來關」。

11.4.3 巢狀結構

以上的條件都只有成立不成立, 若有多層次的條件判定, 則不再像前面的二分。

若有 n 個條件, 各對應著條件符合時的 n 個陳述區塊, 另外若皆不符合以上 n 個條件, 則會有第 $n+1$ 個陳述區塊。

IF ... Then ... ElseIf ... Then ... Else 語法

```
If 條件式 1 Then
  陳述式區塊 1
ElseIf 條件式 2 Then
  陳述式區塊 2
...
ElseIf 條件式 n Then
  陳述式區塊 n
Else
  陳述式區塊 n+1
End If
```

舉例,

IfTest5 VBA

```
Sub IfTest5()
分數=InputBox(“輸入分數”)
If 分數>=90 Then
  考績="優"
ElseIf 分數>=80 Then
  考績="甲"
ElseIf 分數>=70 Then
  考績="乙"
Else
  考績="丙"
End If
MsgBox 考績
End Sub
```

本程式是爲將分數判斷爲成績等第, 當 InputBox 方塊出現, 我們填入 95 時, 它會出現「優」的對話方塊來回應。

11.4.4 多向選擇

如果遇到只有單一條件式, 或將運算式與數個不同值比較時, 多向選擇就能派上用場。多向選擇式透過 Select Case 陳述式來處哩, 它的特色是僅有一個表示放在開頭, 隨著狀況不同, 到各個 Case 執行相關程式, 以便得到其結果。

Select Case 語法

```
Select Case 表示式
Case 表示值 1
表示值 1 的陳述式
Case 表示值 2
表示值 2 的陳述式
Case Else
不符合表示式的陳述式
End If
```

舉例,

Case VBA

```
Sub SelectCase()
分數=InputBox(“輸入分數”)
Select Case 分數
Case 90 to 100
  Range("A1")="優"
Case 80 to 89
  Range("A1")="甲"
Case 70 to 79
  Range("A1")="乙"
Case Else
  Range("A1")="丙"
End Select
End Sub
```

當 InputBox 方塊出現, 我們填入 65 時, A1 儲存格會出現「丙」的來回應。

11.4.5 GoTo 強制改變流程

GoTo VBA

```
Sub GoToTest()
GoTo F12
F10:  FontSize=10
Range("A1")="字型 10pt"
GoTo F16
F12:  FontSize=12
Range("A2")="字型 12pt"
GoTo F10
F16:  FontSize=16
Range("A3")="字型 16pt"
End Sub
```

11.5 迴圈

迴圈是在處理重複的動作, 一般重複的動作如果三次、五次, 我們可以耐著性子直接處理掉, 但若是 10 次、50 次、甚至高達 100 次呢? 我想很乏味的動作應該沒多人願意多做。

在 VBA 程式中, 有幾個迴圈控制陳述式, 讓一再重複的動作能夠精簡, 加快執行速度。迴圈控制有 For Next 與 Do Loop 二大類,, 最後說明一下若在迴圈中打轉出不來, 則應該怎麼停止。

- For Next
  - For …Next
  - For Each …Next
- Do Loop
  - Do While …Loop
  - Do Loop While
  - Do Until …Loop
  - Do …Loop Until
- Exit

11.5.1 For …Next

如果很明確知道要執行多少次迴圈次數, 可以用 For …Next 控制。迴圈的計數設計可以遞增、遞減, 直到我們的預設次數, 才會停止。

For …Next 語法

```
For 計數變數 = 開始值  To 終止值 Step 遞增值
  程式
Next
```

舉例,

ForNext1() VBA

```
Sub ForNext1()
Dim i, j As Integer
  j=0
  For i=1 To 10
    j=j+1
  Next
MsgBox j
End Sub
```

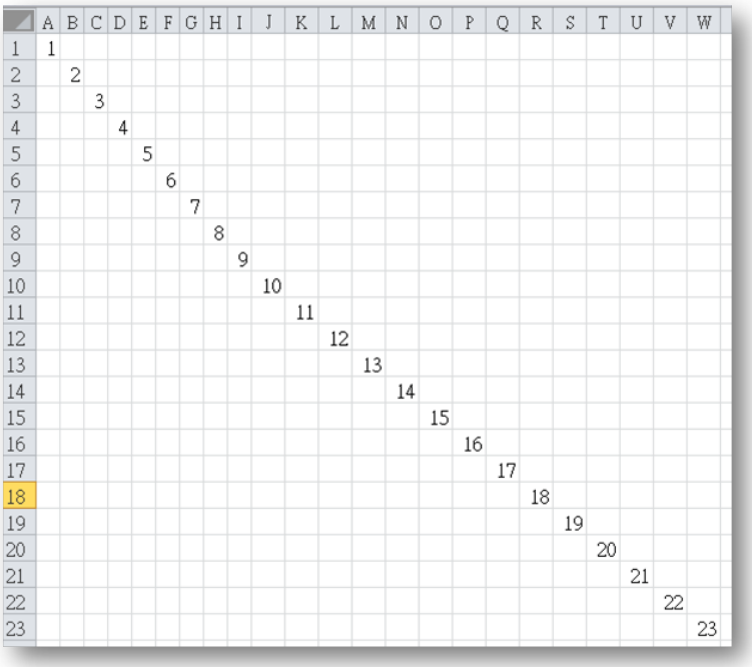
Step 省略就等於遞增值爲 1, i 從 1 計數到 10, 每次遞增 1, 就表示迴圈執行 10 次。這裡迴圈的程式爲 $j = j + 1$, j 從 0 開始, 每次 +1, 加了 10 次, 最後總和爲 10。

再舉個 Step 遞增值爲負數的例子,

ForNext2() VBA

```
Sub ForNext2()
Dim k As Integer
   For k=30 To 1 Step -1
     Cells(k, k)=k
   Next
End Sub
```

k 從 30 遞減到 1, 執行共 30 次, 執行結果會是 A1 至 AD30 儲存格的對角線依序有 1~ 30 的內容。



11.5.2 For Each …Next

在 For Each …Next 的迴圈控制, 它是作用於集合物件中的每個物件, 當迴圈每執行一次時, 則 VBA 會自動設定一個物件變數, 並且針對此變數執行此迴圈中的陳述式。

For Each …Next 語法

```
For Each 物件變數 In 集合物件
  程式
Next 物件變數
```

舉例,

ForEach() VBA

```
Sub ForEach()
For Each Worksheet In Worksheets
  Worksheet.Activate
  Range("A1") = 35
Next
End Sub
```

執行之後每個分頁的 A1 格皆爲 35。試著使用 F8 逐步執行, 可以觀察到先從現行分頁執行爲 Range("A1")=35, 到 Next 準備往下重複執行, Worksheet.Activate 時換到新分頁, 再把 35 鍵進新分頁 A1 儲存格, 依此做到完。

11.5.3 Do …Loop

使用 Do …Loop 執行迴圈次數是不確定的, 它取決於條件敘述的結果。當 While 出現, 條件成立就執行; 當 Until 出現, 就是條件執行到條件成立。換句話說, While 是條件成立起, 執行到條件不成立; 而 Until 是條件不成立起始, 執行到條件成立爲止。

Do While …Loop 在執行之前,「先」測試條件式是否爲 True, 只要條件成立, 它就會開始及繼續執行, 直到條件不成立, 就會跳出迴圈。

Do While …Loop 語法

```
Do While 條件敘述
  程式
Loop
```

舉例,

DoLoop1() VBA

```
Sub DoLoop1()
i=1
 Do While i<=10
   Cells(i, 1)=i
   i=i+1
 Loop
End Sub
```

Do …While Loop 會先執行一次,「再」測試條件式是否爲 True, 只要條件成立, 它就會繼續執行, 直到條件不成立, 就會跳出迴圈。

Do …Loop While 語法

```
Do
  程式
Loop While 條件敘述
```

舉例,

DoLoop2() VBA

```
Sub DoLoop2()
i=1
 Do
   Cells(i, 2)=i
   i=i+1
 Loop While i<=10
End Sub
```

Do Until …Loop 在執行之前,「先」測試條件式是否爲 False, 只要條件沒有成立, 它就會開始及繼續執行, 直到條件成立後, 就會跳出迴圈。

Do Until …Loop 語法

```
Do Until 條件敘述
  程式
Loop
```

舉例,

DoLoop3() VBA

```
Sub DoLoop3()
i=1
 Do Until i=11
   Cells(i, 3)=i
   i=i+1
 Loop
End Sub
```

Do …Loop Until 會先執行一次,「再」測試條件式是否爲 False, 只要條件未達成, 它就會繼續執行, 直到條件成立爲止, 接著跳出迴圈。

Do ...Loop Until 語法

```
Do
  程式
Loop Until 條件敘述
```

舉例,

DoLoop4() VBA

```
Sub DoLoop4()
i=1
 Do
   Cells(i, 4)=i
 i=i+1
   Loop Until i=11
End Sub
```

11.5.4 Exit

Exit Do
Exit For
Exit Sub

11.6 程式撰寫

11.6.1 活頁簿

11.6.2 工作表

11.6.3 儲存格

儲存格是 Excel 工作表最基本的物件, 以下我們介紹幾種方式來參照儲存格。

中括號 []

最最簡單的表達方式就是使用中括號, [A1] 指得就是 A1 儲存格, [A1:C3] 自然就是指工作表中最左上角的 9 格。

Range

參照	說明
Range(“B2”)	[B2] 儲存格
Range(“B2:C5”)	[B2:C5] 儲存格範圍
Range(“B2:C5, D6:E9”)	多重範圍 [B2:C5] 及 [D6:E9], 中間以逗號隔開
Range(“B:B”)	B 整欄
Range(“2:5”)	第 2 列至第 5 列範圍
Range(ActiveCell, “B9”)	從作用儲存格至 [B9] 格的範圍
ActiveCell.Range(“B3”)	以作用儲存格爲左上角參照到第 2 欄第 3 列之格

Cells

參照	說明
Cells(3, 2)	[B3] 儲存格
Cells(*n*)	儲存格範圍中第 *n* 個儲存格 (由列先算)
Cells	[B3] 表示全部儲存格
R(Cells(3, 2), Cells(5, 3))	[B3:C5] 儲存格範圍

Offset

11.6.4 物件、屬性、方法、事件

11.7 按鈕