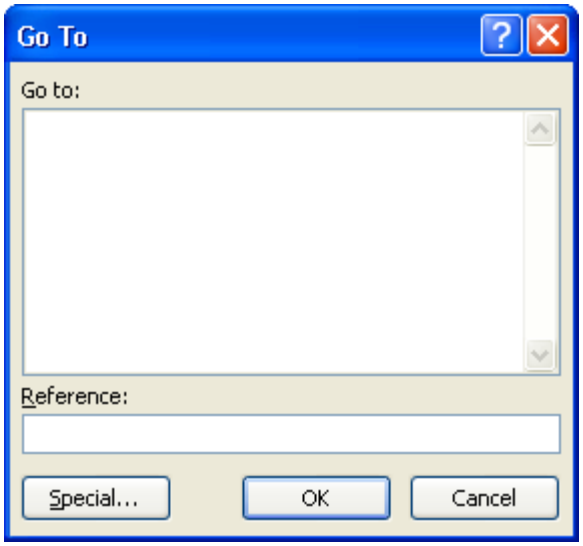
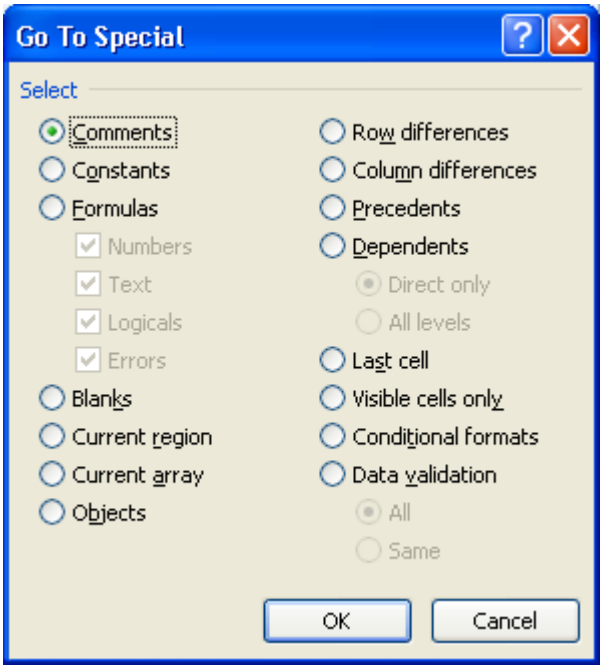
Phương thức SpecialCells trong Excel VBA

Một trong những phương thức có ích nhất (theo kinh nghiệm bản thân) trong Excel là phương thức SpecialCells.

Trong Excel bạn hãy gõ Ctrl + G (hoặc F5) bạn sẽ thấy hộp thoại **Go To** sau:



Sau đó bạn tiếp tục nhấn nút  bạn sẽ được hộp thoại sau:



Hộp thoại này giúp bạn một số thao tác rất hữu ích như: tìm ô cuối cùng trong worksheet, hoặc tất cả các ô là số mà thôi,...hầu hết các thao tác này bạn đều có thể thực hiện bằng VBA, đó chính là việc dùng phương thức SpecialCells.

Khi được gọi, phương thức SpecialCells sẽ trả về một Đối tượng Range đặc trưng cho kiểu của những ô ta chỉ định. Ví dụ: ta có thể dùng phương thức SpecialCells để trả về một đối tượng Range chỉ chứa công thức. Và nếu muốn ta có thể giới hạn để đối tượng Range (chỉ chứa công thức) trả về công thức có lỗi (Tương ứng với hộp thoại Go To Special > Formulas > Errors).

Cú pháp của phương thức SpecialCells là:

expression.SpecialCells(Type, Value)

Trong đó *expression* là một biến đại diện cho một đối tượng Range.

***Type**: Kiểu dữ liệu là XlCellType.*

XlCellType có thể là một trong những hằng số sau:

Kiểu XlCellType	Giá trị	Giải thích
xlCellTypeAllFormatConditions.	-4172	Ô có định dang bất kì
xlCellTypeAllValidation. (having validation criteria)	-4174	Ô có áp đặt điều kiện
xlCellTypeBlanks.	4	Ô trống (rỗng)
xlCellTypeComments.	-4144	Ô có ghi chú
xlCellTypeConstants.	2	Ô chứa hằng số
xlCellTypeFormulas.	-4123	Ô chứa công thức
xlCellTypeLastCell.	11	Ô cuối cùng trong range. Lưu ý rằng kiểu xlCellType này bao gồm cả các ô trống đã thay đổi so với định dạng mặc định.
xlCellTypeSameFormatConditions.	-4173	Các ô có cùng định dạng
xlCellTypeSameValidation.	-4175	Các ô cùng được áp đặt điều kiện
xlCellTypeVisible.	12	Tất cả các ô được hiển thị

Chú ý: Không thể dùng kết hợp nhiều hơn một kiểu XlCellType cho tham số này.

***Value**: đối số này không bắt buộc, có kiểu là variant. Nếu Type có giá trị là xlCellTypeConstants hoặc xlCellTypeFormulas thì đối số này được dùng để xác định kiểu nào của các ô được trả về.*
Các hằng của XlSpecialCellsValue, được thể hiện trong bảng sau:

Hằng số XlSpecialCellsValue	Giá trị
xlErrors	16
xlLogical	4
xlNumbers	1
xlTextValues	2

Ví dụ sau sẽ chọn ô cuối cùng trong vùng đã được dùng trong Sheet1:

```
Worksheets("Sheet1").Activate
ActiveSheet.Cells.SpecialCells(xlCellTypeLastCell).Activate
```

Có thể kết hợp nhiều từ khóa XlSpecialCellsValue với nhau cho tham số này.

Phương thức SpecialCells có thể được sử dụng trong rất nhiều tình huống, khi ta chỉ cần thao tác với một kiểu dữ liệu đặc trưng của các ô trong vùng. Ví dụ: câu lệnh dưới

sẽ trả về một vùng đối tượng (Range) chỉ chứa đặc trưng là loại công thức trong trang tính hiện hành.

```
ActiveSheet.UsedRange.SpecialCells(xlCellTypeFormulas)
```

Nếu muốn, ta có thể giới hạn để phương thức trả về đối tượng Range mang đặc trưng là các công thức trả về của chúng chỉ là giá trị số.

```
ActiveSheet.UsedRange.SpecialCells(xlCellTypeFormulas,xlNumbers)
```

Một khi ta đã chỉ định kiểu của đối tượng Range trả về, ta có thể thao tác với chỉ những ô đó. Điều này thường có thể được thực hiện chỉ bằng một dòng lệnh, hoặc ta thực hiện một vòng lặp **Loop** cho Range. Mời xem các ví dụ dưới đây để hiểu rõ hơn về phương thức này:

Ví dụ 1

```
Sub SpeacialCells0()
    ActiveSheet.UsedRange.SpecialCells(xlCellTypeFormulas). _
       Interior.ColorIndex = 35
End Sub
```

Hình 1 Tô màu nền xanh nhạt cho những ô có công thức

Vậy là macro chỉ gồm 1 câu lệnh đã tô màu cho cột chứa số thứ tự của hồ sơ, nơi mà ai đó đã nhập vô công thức tại A2 như sau: **=IF(A1 = "TT"; 1; A1 + 1)** và công thức này đã được chép xuống đến ô 16. Như trong hình dẫn ra công thức tại ô 'A13'.
Cần nói thêm rằng thuộc tính UsedRange mà trong câu lệnh có đề cập đến, chúng ta sẽ hay đã gặp trong Ebook này. Nếu chúng ta chưa gặp, thì sẽ được gặp & hiểu kỹ về nó sau, còn bây giờ ta tạm hiểu, thuộc tính sẽ trả về toàn bộ các ô mà ta đã dùng trên trang tính.

Ví dụ 2

```
Sub SpeacialCells1()
 Dim rRng As Range, rClls As Range
 Set rRng = ActiveSheet.UsedRange.SpecialCells(xlCellTypeFormulas,
xlNumbers)
 For Each rClls In rRng
        rClls = -1 * rClls.Value
```

```
 Next rClls
End Sub
```

Trong macro thứ hai ta đã dùng một vòng lặp để mọi giá trị số trên trang tính được chuyển trả về số đối âm của chúng.

Ta có thể dùng phương thức PasteSpecial để thực hiện điều này mà không cần dùng vòng lặp, khi đó các công thức được giữ nguyên trong các ô của chúng. Macro để thực hiện việc như vậy có nội dung như sau:

```
Sub SpeacialCells_()
  With Range("IV65536")
       .Value = -1:     .Copy
            ActiveSheet.UsedRange.SpecialCells _
                (xlCellTypeFormulas, xlNumbers).PasteSpecial _
                xlPasteValues, xlPasteSpecialOperationMultiply
        .Clear
   End With
End Sub
```

Nếu quen thuộc với Excel và những tính năng của nó, như SpecialCells, ta sẽ biết khi ta chỉ định chỉ một ô (bằng Selection hay Range) Excel sẽ cho rằng ta muốn thao tác với toàn bộ các ô của trang tính.

Ví dụ3 : Hai macro dưới đây sẽ chọn toàn bộ các ô trống trong trang tính.

```
Sub ChonToanORong()
    ActiveSheet.UsedRange.SpecialCells(xlCellTypeBlanks).Select
End Sub
```

```
Sub ChonToanORong2()
    Range("A1").SpecialCells(xlCellTypeBlanks).Select
End Sub
```

Ta có thể thấy khi chỉ định Range ở trường hợp sau có lúc sẽ cho kết quả không như ý.

Ví dụ 4: Phương thức SpecialCells với các ô công thức và ô chứa dữ liệu
Chúng ta không thể kết hợp nhiều hơn một XlCellType (ví dụ xlCellTypeConstants + xlCellTypeFormulas). Ta có thể dùng phương thức SpecialCells để chỉ trả về một loại trong các ô đã dùng (công thức hay hằng số) và bỏ đi những ô chứa dữ liệu dạng văn bản (công thức & hằng số).

```
Sub AllNummericCells()
Dim ConClls As Range, ForClls As Range
Dim AllRng As Range
    Set AllRng = ActiveSheet.UsedRange
    On Error Resume Next   'Trường hợp không có công thức hay hằng số
dạng số
     ' Mọi ô chứa số liệu dạng số được gán vô biến:
    Set ConClls = AllRng.SpecialCells(xlCellTypeConstants, xlNumbers)
    ' Các ô công thức dạng số được gán vô biến đã khai báo:
    Set ForClls = AllRng.SpecialCells(xlCellTypeFormulas, xlNumbers)
    ' Xác định kiểu của dữ liệu dạng số (công thức, hằng số hay khác):
    If ConClls Is Nothing And ForClls Is Nothing Then
       MsgBox "Trang Tinh Cua Ban Khong Chua Du Lieu So"
```

```
      End
    ElseIf ConClls Is Nothing Then
       Set AllRng = ForClls    'Công thức
    ElseIf ForClls Is Nothing Then
       Set AllRng = ConClls    'Số liệu thuần
    Else
       Set AllRng = Application.Union(ForClls, ConClls)      ' Cả hai
    End If
    On Error GoTo 0
    AllRng.Select
End Sub
```

Ta luôn nên ghi dòng lệnh **On Error Resume Next** như trong đoạn mã trên. Điều này rất cần thiết khi điều kiện của phương thức SpecialCells không thỏa và gây lỗi. Để bãy lỗi này ta kiểm tra biến kiểu Range mà ta đã gán có tồn tại hay không. Trong đoạn mã trên câu lệnh If (với 2 ElseIf) kiểm tra điều này.

Ví dụ 5: Tiện lợi rất nhiều về thời gian một khi ta dùng phương thức này thay cho cách dùng vòng lặp như thông thường

Ta xét 2 macro sau:

```
Sub DoLoop()
Dim Blclls As Range
   For Each Blclls In Range("b2:F350")
       If IsEmpty(Blclls) Then Blclls = "Rong"
   Next Bcell
End Sub
```

Macro này sẽ phải lần lượt duyệt qua toàn bộ các ô trong vùng chọn (“b2:F350"). Khi gặp ô nào không chứa dữ liệu (rỗng) thì nó điền thay vô đó từ ‘Rong”

Còn macro dưới đâythì tiếp cận với các ô rỗng theo phương thức chọn toàn bộ các ô rỗng cùng một lúc & chỉ bằng 1 dòng lệnh duy nhất.

```
Sub SpecialCells5()
    If WorksheetFunction.CountA(Range("b2:F350")) = 0 Then
       MsgBox "TOAN BO O LA RONG", vbOKOnly, "GPE.COM"
       Exit Sub
    End If
    On Error Resume Next
    Range("b2:F350").SpecialCells(xlCellTypeBlanks) = "Rong"
End Sub
```

Ví dụ 6: Tô màu các ô chứa giá trị & các ô công thức cùng nhỏ hơn mộtr giá trị nào đó.

Macro mà chúng ta dùng để tô màu nền các ô chứa các trị bế hơn 10 & kể cả các ô chứa công thức, mà kết quả công thức trả về cũng là những giá trị bé hơn 10. Nó gồm các dòng lệnh sau:

```
Sub BackColor()
Dim ConClls As Range, ForClls As Range, AllClls As Range, Clls As Range
Set AllClls = ActiveSheet.UsedRange:  AllClls.Select
On Error Resume Next
Set ConClls = AllClls.SpecialCells(xlCellTypeConstants, xlNumbers)
Set ForClls = AllClls.SpecialCells(xlCellTypeFormulas, xlNumbers)
    If ConClls Is Nothing And ForClls Is Nothing Then
```

```
       MsgBox "Trang Tinh Không Chứa Số":     End
    ElseIf ConClls Is Nothing Then
       Set AllClls = ForClls     'Công thức
    ElseIf ForClls Is Nothing Then
       Set AllClls = ConClls     ' Số Trị
    Else
       Set AllClls = Application.Union(ForClls, ConClls)     'Cả Hai
    End If
    On Error GoTo 0
    For Each Clls In AllClls
      With Clls
          If .Value < 10 Then   .Interior.ColorIndex = 36
       End With
 Next Clls
End Sub
```

Những gì mà macro vừa thực hiện, được hiển thị trên hình 2:
Tại cột 'A', là những giá trị do các công thức tại A(j+1) = IF(A(j) = "TT"; 1; A(j)1 + 1).
Tại cột 'E', những giá trị bé hơn 10 đều được tô màu vàng nhạt

	A	B	C	D	E
	TT	Ma	Ho	Ten	SoLuong
1	1	A01	Nguyễn	An	14
2	2	A03	Trần	Anh	9
3	3	A05	Lê	By	5
4	4	A07	Lý	Chi	11
5	5	A09	Vương	My	2
6	6	A02	Lại	Phi	50
7	7	A06	Đỗ	Hoa	36
8	8	A10	Huỳnh	Phai	22
9	9	A14	Hoàng	Mai	8
10	10	A18	Dương	Ty	-6
11	11	A89	Đậu	Yên	15
12	12				
13	13				
14	14				
15	15				

Hình 2 Tô màu nền những ô chứa dữ liệu bé hơn 10

Ví dụ 7: Về ô cuối cùng chứa dữ liệu
Nếu ta viết macro có nội dung sau:

```
Sub LastCells()
       Range("A1").SpecialCells(xlCellTypeLastCell).Select
       MsgBox Selection.Address
End Sub
```

Thì hộp thoại sẽ đưa ra địa chỉ ô, mà ô này chưa chắc phải là ô cuối cùng chứa dữ liệu trong vùng sử dụng. Để trả lời vấn đề này, ta đọc lại phần đầu của đề mục viết về tham số xlCellTypeLastCell.
Vậy khi ta muốn tìm tới ô cuối cùng chứa dữ liệu của trang tính thì làm sao đây? Lúc đó ta phải dùng macro như sau để đạt mục đích:

```
Sub FindLastCell()
Dim LastColumn As Integer:
Dim LastRow As Long
Dim LastCell As Range
   If WorksheetFunction.CountA(Cells) > 0 Then
      LastRow = Cells.Find(What:="*", After:=[A1], _
               SearchOrder:=xlByRows, SearchDirection:=xlPrevious).Row
            LastColumn = Cells.Find(What:="*", After:=[A1], _
               SearchOrder:=xlByColumns,
SearchDirection:=xlPrevious).Column
            MsgBox Cells(LastRow, LastColumn).Address
   End If
End Sub
```

Với cơ sở dữ liệu như trong h.3, ta sẽ thấy hộp thoại của macro đưa ra địa chỉ như trong hình đã dẫn.

	A	B	C	D	E	F	G	H	K	P	T
1	DANH SÁCH HỌC SINH THAM DỰ KỲ THI TUYỂN SINH VÀO LỚP 6										
2						*Năm học 2008 - 2009*					
3	**PT**	**SBD**	**Hdem**	**Ten**	**NgSinh**	**Nsinh**	**Nu**	**TThg**	**AS**	**HTRL**	**TgDm**
4	1	1	Vương Hải	Anh	1/1/1977	Hà Nội	x	x		25	21.0
5	1	6	Nguyễn Hoàng Diệu	Linh	1/6/1977	Hà Nội	x			25	19.0
6	1	7	Lê Trí		1/7/1977	Hồ Chí Minh			x	25	23.0
7	1	8	Lê Thanh		1/8/1977	Quảng Ninh	x	x		25	19.0
8	1	12	Lê Vũ Ph		1/12/1977	Quảng Ninh		x		24.5	21.0
9	2	25	Lê Hà Ma		1/25/1977	Ba Lan				25	21.0
10	2	26	Phạm Ph		1/26/1977	Hà Nội		x		25	16.0
11	2	27	Trần Thu		1/27/1977	Nam Định			x	25	20.0
12	2	28	Nguyễn Quỳnh	Như	1/28/1977	Hà Nội		x		25	19.0
13	2	29	Phạm Minh	Thanh	1/29/1977	Hà Tây				25	21.0
14	2	38	Nguyễn Ngọc	Ánh	2/7/1977	Hà Nội	x	x		25	19.0
15	2	39	Nguyễn Anh	Duy	2/8/1977	Kiến An			x	25	22.0
16	2	40	Trần Bùi Phương	Dung	2/9/1977	Thanh Hóa		x		25	19.0
17	2	41	Hoàng Gia	Hưng	2/10/1977	Hà Nội	x			25	22.0
18	2	42	Trần Minh	Hoàng	2/11/1977	Hà Nội	x	x		25	16.0
19	2	43	Nguyễn Quốc	Dũng	2/12/1977	Hà Nội	x		x		18.0
20	2	44	Vũ Hà	Phương	2/13/1977	Hà Nội		x		25	21.0
21	2	45	Lương Nguyễn Ngọc	Lê	2/14/1977	Hà Nội				24	22.0
22	2	46	Nguyễn Hồng Minh	Châu	2/15/1977	Hà Nội	x	x		23,5	14.0
23	2	47	Phạm Minh	Đức	2/16/1977	Hà Nội	x		x	25	22.0
24	2	48	Nguyễn Danh Phương	Thảo	2/17/1977	Hà Nội	x	x		25	21.0
25											

Hình 3: Tìm đến ô cuối chứa dữ liệu

Phần 2: Các ứng dụng có liên quan đến phương thức SpecialCells

1. Xóa các dòng rỗng trong cơ sở dữ liệu

```
Sub DeleteBlankRows1()
'Xóa dòng rỗng trong vùng đã chọn
On Error Resume Next
Selection.EntireRow.SpecialCells(xlBlanks).EntireRow.Delete
On Error GoTo 0
End Sub
```

```
Sub DeleteBlankRows2()
'Cũng xóa dòng rỗng trong vùng đã chọn
Dim BlRng As Range
If WorksheetFunction.CountA(Selection) = 0 Then
   MsgBox "Khong Tim Ra Du Lieu", vbOKOnly, "GPE.com"
   Exit Sub
End If
With Application
   .Calculation = xlCalculationManual
   .ScreenUpdating = False
   Selection.SpecialCells(xlCellTypeBlanks).Select
   For Each BlRng In Selection.Rows
      If WorksheetFunction.CountA(Selection.EntireRow) = 0 Then
         Selection.EntireRow.Delete
      End If
   Next BlRng
   .Calculation = xlCalculationAutomatic
   .ScreenUpdating = True
 End With
End Sub
```

2. Cẩn thận với xlCellTypeVisible khi viết trong hàm người dùng

Nếu chúng ta có ý tưởng viết hàm người dùng để nó đếm cho ta số ô trong vùng nào đó, ví dụ bằng cách thức sau:

```
Public Function KhongHieuNoi(Rng As Range)
    KhongHieuNoi = Rng.SpecialCells(xlCellTypeVisible).Cells.Count
End Function
```

Thì hàm này hoàn toàn đúng về cú pháp, cũng như biên dịch ra ngôn ngữ máy, . . . Thậm chí trong cửa sổ Immediate, ta nhập câu lệnh sau:

```
?khonghieunoi(sheet3.UsedRange)
```

Sau khi {ENTER}, chúng ta sẽ nhận được 1 kết quả rất chi là chính xác nữa là đằng khác.

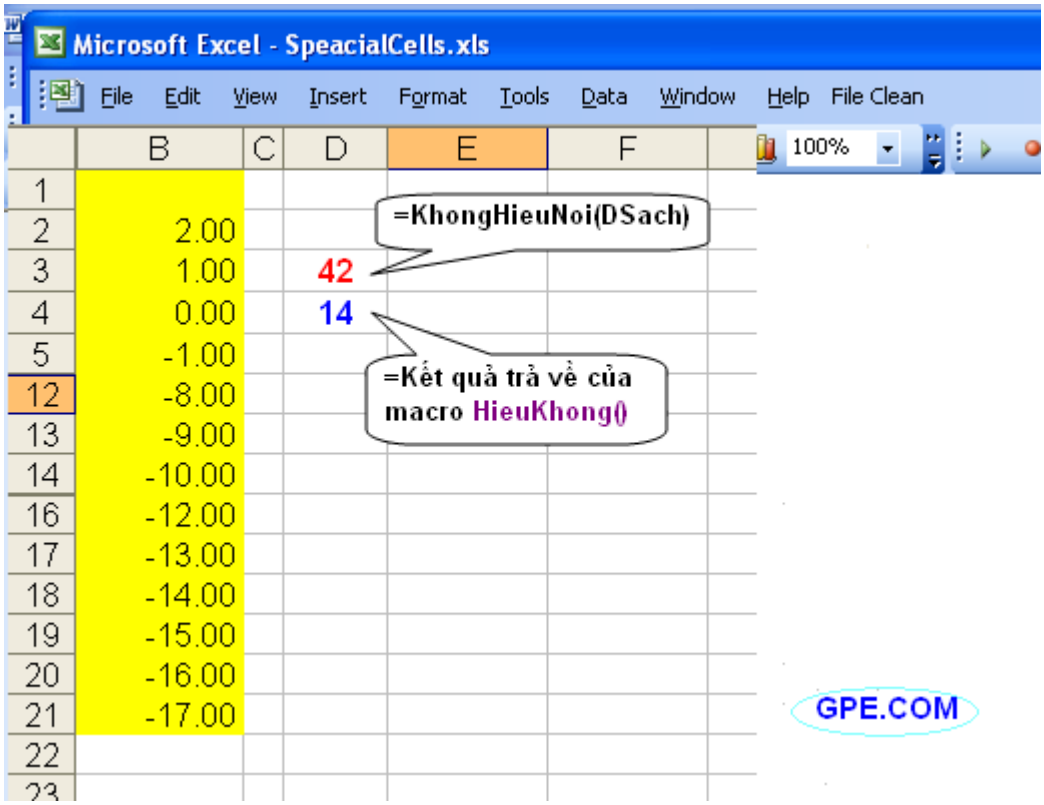
Nhưng vấn đề sẽ khác, một khi ta xét hàm người dùng này & được đối chiếu với 1 macro có những câu lệnh tương tư. Muốn vậy, ta tạo 1 vùng nào đó trên trang tính & gán cho nó cái tên 'DSach' (Trong hình 4 là vùng 'A1:B21'). Tại ô 'D3' ta nhập công thức:

```
=KhongHieuNoi(DSach)
```

Sau khi {ENTER} ta được giá trị 42. Giá trị này vẫn tồn tại cho dù bạn có ẩn vài hàng hay ẩn đi 1 cột, hoặc thậm chí ẩn cả hai cột 'A' & 'B' đi chăng nữa.
Sự thật phủ phàng này (vẫn như cũ, luôn xuất hiện số 42) nên ghi nhớ mãi, một khi làm việc với những ô 'trông thấy' trong các hàm tự tạo trong excel.
Ngược lại với hàm người dùng như trên, với macro HieuKhong() sau đây vấn đề sẽ hoàn toàn khác

```
Sub HieuKhong()
 On Error Resume Next
 [D4]= Sheet3.[DSach].SpecialCells(xlCellTypeVisible).Cells.Count
End Sub
```

Cặp mắt của macro này giống với cặp mắt của chúng ta, và kết quả của hàm hay của macro ghi ra không trùng nhau, như hình dưới đây:



Hình 4 Hàm tự tạo & macro khi đếm ô nhìn thấy

3. Phương thức SpecialCells & vùng được gán tên

Macro dưới đây sẽ tự động gán tên lên các trang tính hiện diện tại vùng bắt đầu từ 'B3'. Còn kết thúc tại điểm giao nhau giữa dòng cuối nhất & cột cuối nhất của trang tính. (Cũng cần nhắc lại rằng, ô giao nhau giữa dòng cuối nhất & cột cuối nhất này không lúc nào trùng hợp với ô cuối nhất chứa dữ liệu)

```
Sub LastColAndRow()
    Dim lRow As Integer, lCol As Integer
    Dim wSh As Worksheet
    Dim AddressName As String
 For Each wSh In Worksheets
   lRow = wSh.Cells.SpecialCells(xlCellTypeLastCell).Row
   lCol = wSh.Cells.SpecialCells(xlCellTypeLastCell).Column
   AddressName = "=" & Range("B3:" & Cells(lRow, lCol).Address).Address
   ThisWorkbook.Names.Add "GPE" & wSh.Name, AddressName
```

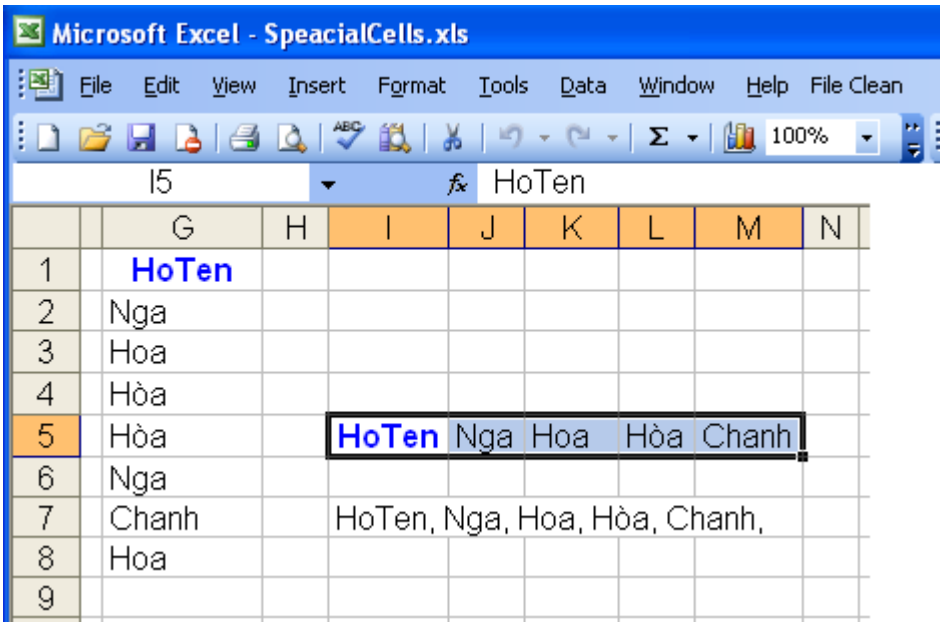
```
 Next wSh
End Sub
```

4. Các phương thức SpecialCells & AdvancedFilter

Ta hãy xét đến macro Filter2() có nội dung như dưới đây:

```
Sub Filter2()
    Dim RngFilter As Range, RngFiltered As Range
    Dim StrC As String
    Set RngFilter = Range("G1:G" & Range("G65432").End(xlUp).Row)
    With RngFilter
        .AdvancedFilter Action:=xlFilterInPlace,
CriteriaRange:=RngFilter, Unique:=True
        Set RngFiltered = RngFilter.SpecialCells(xlCellTypeVisible)
    End With
    RngFiltered.SpecialCells(xlCellTypeVisible).Copy
    Range("I5").PasteSpecial Transpose:=True
    For Each RngFilter In RngFiltered
      StrC = StrC & RngFilter & ", "
    Next RngFilter
    [I7] = StrC
    ActiveSheet.ShowAllData
End Sub
```

Hai dòng lệnh đầu dùng để khai báo 3 biến, trong đó chỉ 1 biến dùng chứa chuỗi.
Tiếp sau, ta gán vùng “G1:G8 (8 : dòng cuối tại cột ‘G’ có chứa dữ liệu) vô biến có tên là RngFilter
Câu lệnh từ dòng 4 đến dòng 7 dùng để xét đến vùng ‘G1:G8’ này
Áp dụng phương thức AdvFilter lên vùng này để lập danh sách duy nhất vế nó.
Câu lệnh 6: Vùng kết quả do phương thức AdvFilter đem lại, ta đem gán vô biến kiểu Range thứ hai đã khai báo tại dòng 1 (RngFitered)
Dòng lệnh 8: áp đặt phương thức Copy lên biến vừa nói tới;
Dòng 9: Chép chuyển vị từ bộ nhớ (do phương thức Copy) vô các ô trên hàng 5, bắt đầu từ cột ‘I’;
Dòng 10 – 12: Tạo vòng lặp để chép nối giá trị trong các ô chứa trong biến RngFiltered vô biến kiểu chuỗi StrC; Các giá trị cách nhau bởi dấu phẩy & khoảng trắng;
Câu lệnh áp chót: Hiển thị vùng dữ liệu. Rất nên thức hiện câu lệnh này, một khi trước đó đã dùng phương thức Filter



Hình 5 Hai phương thức SpecialCells & AdvancedFilter

5. Phương thức SpecialCells & AutoFilter

Để thấy rõ hơn cách thức làm việc của phương thức SpecialCells với vùng lọc bằng AutoFilter, chúng ta khảo sát macro làm việc với CSDL trong hình 6A dưới đây:

```
Sub CopyIsFilter()
    Dim Rng As Range
    With Sheet2
        If Not .FilterMode Then
            MsgBox "AutoFilter?":                Exit Sub
        End If
        Set Rng = .AutoFilter.Range.Offset(1,
0).Resize(.AutoFilter.Range.Rows. _
                  Count - 1).SpecialCells(xlCellTypeVisible)
         'set a range = to visible cells (excluding the header)
        Rng.Copy Destination:=Sheet4.Range("A1")
    End With
End Sub
```

Trong khi chúng ta chưa chuẩn bị điều kiện tối thiểu cho macro nhưng cho 'chạy' nó, ta sẽ nhận được thông báo lỗi (xem hộp thoại báo lỗi trong hình).

Lỗi này, như trong đoạn mã trên đã ghi, là ta chưa áp đặt FilterMod lên CSDL.

Muốn có kết quả làm việc của macro, ta vô menu Data trong excel, chọn dòng AotoFilter. Hơn thế nữa, ta còn phải bấm vô hình tam giác nào đó vừa xuất hiện (Ví dụ: [SBD], và chọn dòng có ghi '(Top10. . . .)'. Như vậy tại Shee2 chỉ hiện lên 10 học sinh có số báo danh lớn nhất.

	SBD	Hdem	Ten	NgSinh	Nsinh	Nu	TThg
2	1	Vương Hải	Anh	1/1/1977	Hà Nội	x	x
3	6	Nguyễn Hoàng Diệu	Linh	1/6/1977	Hà Nội	x	
4	7	Lê Trí	Dũng	1/7/1977	Hồ Chí Minh		
5	8	Lê Thanh	Tùng	1/8/1977	Quảng Ninh	x	x
6	12	Lê Vũ Phương	Linh	1/12/1977	Quảng Ninh		x
7	25	Lê Hà Ma		1/25/1977	Ba Lan		
8	26	Phạm Ph		1/26/1977	Hà Nội		x
9	27	Trần Thu		1/27/1977	Nam Định		
10	28	Nguyễn		1/28/1977	Hà Nội		x
11	29	Phạm Mi		1/29/1977	Hà Tây		
12	39	Nguyễn Anh	Duy	2/8/1977	Kiến An		
13	40	Trần Bùi Phương	Dung	2/9/1977	Hà tây		x
14	41	Hoàng Gia	Hưng	2/10/1977	Hà Nội	x	
15	42	Trần Minh	Hoàng	2/11/1977	Hà Nội	x	x
16	43	Nguyễn Quốc	Dũng	2/12/1977	Hà Nội	x	
17	44	Vũ Hà	Phy	2/13/1977	Hà Nội		x
18	45	Lương Ngô	Lê	2/14/1977	Hà Nội		
19	46	Nguyễn Hồng Minh	Châu	2/15/1977	Hà Nội	x	x
20	47	Phạm Minh	Đức	2/16/1977	Hà Nội	x	
21	48	Nguyễn	Thảo	2/17/1977	Hà Nội	x	x

Hình 6A CSDL chưa thực hiện phương thức AutoFilter

Bây giờ thí chúng ta có thể yên tâm & cho macro khởi động. Khi chuyển chọn đến Sheet4, ta sẽ thấy kết quả như hình 6B đưới đây. Đó là liệt kê danh sách các em học sinh có số bao danh lớn nhất trong CSDL

	A	B	C	D	E	F	G	H
1	39	Nguyễn Anh	Duy	2/8/1977	Kiến An			25
2	40	Trần Bùi Phương	Dung	2/9/1977	Hà tây		x	25
3	41	Hoàng Gia	Hưng	2/10/1977	Hà Nội	x		25
4	42	Trần Minh	Hoàng	2/11/1977	Hà Nội	x	x	25
5	43	Nguyễn Quốc	Dũng	2/12/1977	Hà Nội	x		
6	44	Vũ Hà	Phy	2/13/1977	Hà Nội		x	25
7	45	Lương Ngô	Lê	2/14/1977	Hà Nội			24
8	46	Nguyễn Hồng Minh	Châu	2/15/1977	Hà Nội	x	x	23,5
9	47	Phạm Minh	Đức	2/16/1977	Hà Nội	x		25
10	48	Nguyễn	Thảo	2/17/1977	Hà Nội	x	x	25
11								
12								

H. 6B Kết quả tại Sheet4 sau khi chạy macro CopyIsFilter

Sau đây xin giới thiệu với các bạn đọc 1 macro tương tự để các bạn tham khảo:

```
Sub CopyFilterRows()
 On Error Resume Next
    Dim wsData As Worksheet, wsReport As Worksheet
    Dim rFiltered As Range, rNextCl As Range
    On Error GoTo CopyFilterRows_Error
    Set wsData = Worksheets("Sheet2")
    Set wsReport = Worksheets("Sheet4")
    With wsData.AutoFilter.Range
         'Bỏ chọn dòng tiêu đề:
        Set rFiltered = .Offset(1, 0).Resize(.Rows.Count - 1, 1) _
        .SpecialCells(xlCellTypeVisible)
         'Chọn ô rỗng dòng cuối của cột 'A' của Sheet4
        Set rNextCl = wsReport.Cells(Rows.Count, 1).End(xlUp).Offset(1,
0)
         'copy từ Sheet2 đến Sheet4
        rFiltered.Copy rNextCl
        On Error GoTo 0
    End With
    wsData.Select                '.ShowAllData
    Selection.AutoFilter
     'Xóa biến để thu hồi bộ nhớ
    Set wsData = Nothing:              Set wsReport = Nothing
    Set rFiltered = Nothing:           Set rNextCl = Nothing
    On Error GoTo 0
    Exit Sub
CopyFilterRows_Error:
     MsgBox "Error " & Err.Number & " (" & Err.Description & _
      ") in procedure CopyFilterRows of Module Module1"
End Sub
```

6. Xóa trùng từ vùng chọn bất kỳ

Chúng ta khảo sát macro dùng để xóa dữ liệu ở các ô trùng trong một vùng đã chọn từ trước đó. Để tiết kiệm tài nguyên xã hội, chúng ta sẽ viết macro sao cho ứng dụng được hình 6B làm nguồn CSDL (cơ sở dữ liệu) & kết quả hiển thị trên hình 6C như dưới đây

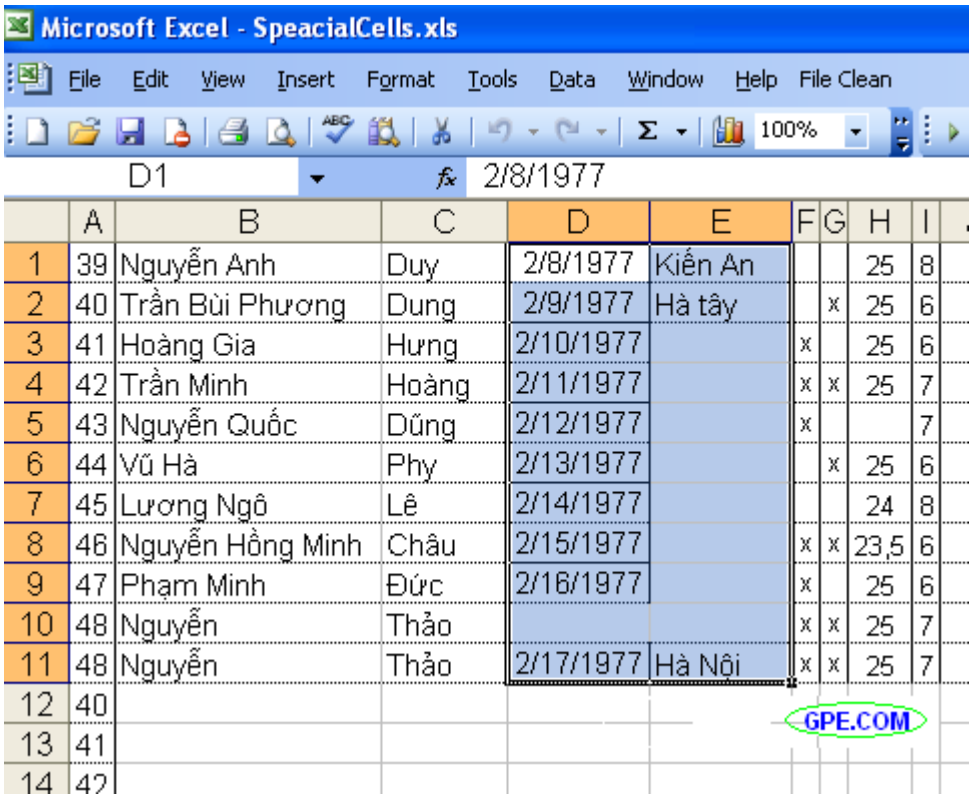
Macro có nội dung sau:

```
Sub KillDupes()
Dim rConstRange As Range, rFormRange As Range
Dim rAllRange As Range, rCell As Range
Dim iCount As Long:           Dim strAdd As String
 On Error Resume Next
Set rAllRange = Selection
If WorksheetFunction.CountA(rAllRange) < 2 Then
MsgBox "You selection is not valid", vbInformation
       On Error GoTo 0:        Exit Sub
End If
Set rConstRange = rAllRange.SpecialCells(xlCellTypeConstants)
```

```
Set rFormRange = rAllRange.SpecialCells(xlCellTypeFormulas)
 If Not rConstRange Is Nothing And Not rFormRange Is Nothing Then
Set rAllRange = Union(rConstRange, rFormRange)
  ElseIf Not rConstRange Is Nothing Then
        Set rAllRange = rConstRange
  ElseIf Not rFormRange Is Nothing Then
        Set rAllRange = rFormRange
  Else
        MsgBox "You selection is not valid", vbInformation
        On Error GoTo 0:       Exit Sub
21    End If
 Application.Calculation = xlCalculationManual
 For Each rCell In rAllRange
strAdd = rCell.Address
        strAdd = rAllRange.Find(What:=rCell, After:=rCell,
LookIn:=xlValues, _
               LookAt:=xlWhole, SearchOrder:=xlByRows,
SearchDirection:=xlNext, _                    MatchCase:=False).Address
        If strAdd <> rCell.Address Then  rCell.Clear
  Next rCell
  Application.Calculation = xlCalculationAutomatic
 On Error GoTo 0
End Sub
```

Các câu lệnh kể từ dòng 21 trở vế trước chúng ta đã gặp, các bạn nếu cần có thể xem lại tại ví dụ 4. Tại đó có giải thích tương đối đầy đủ công việc mà chúng phải thực thi.
Tiếp sau đây, chúng ta xét tiếp các câu lệnh sau dòng 21 nêu trên.
Câu lệnh 22 dùng để tăng tốc chương trình, bỏ qua những phép tự động tính toán lại không cần thiết;
Tạo vòng lặp từ dòng 22 đến dòng 26
(Chú ý nét gạch dưới ở cuối dòng lệnh; chúng báo cho VBA biết là dòng lệnh còn tiếp ngay dòng bên dưới; Khi đó chẳng những máy, mà ta cũng nên bi ết rằng dòng sau dòng có dấu nối như vậy sẽ không được tính là dòng lệnh nữa.
* Cũng chú ý thêm một điều rằng sau dấu gạch nối dòng lệnh này, ta nhất thiết không được để dòng trống. Những ai có thói quen bỏ 1 hay 1 vài dòng trống cho chương trình dễ nhìn & cũng như dễ kiểm soát thìđây ph ải là 1 chú ý đáng có. Tất nhiên ta không thức hiện thì VBA cũng sẽ nhắc ta liền)
Ta trở lại các dòng lệnh trong vòng lặp:
D24: Địa chỉ ô hiện hành (trong vòng lặp) được đem gán vô biến chứa chuỗi
D25: dài & là trái tim của macro này. Nó được giải nghĩa như sau:
Hãy tìm giá trị trong ô hiện thời (thuộc vòng lặp); bắt đầu từ sau ô tìm thấy lần tìm trước;
Tìm theo dạng gì?: dạng dữ liệu (không phải công thức); (. . . mấy tham số sau của phương thức này, các bạn tư tìm hiểu lấy trong phần trợ giúp của VBE, khi có dịp rảnh rỗi);
Nếu tìm thấy thì địa chỉ ô tìm thấy đem gán thay thế vô biến (gán thay vì trước đó biến này đã được gán địa chỉ khác tại dòng 24)
D26: xóa dữ liệu trong ô, nếu thỏa điều kiện
D28: trả về trạng thái tính toán mặc định của VBE (tránh làm phiền người khác hay macro cùng sử dụng excel & VBE tiếp sau đó)
Dòng áp chót: tạo thói quen khử lỗi, tạo thuận tiện cho mình & mọi người dùng tiếp sau.
Kết quả macro được thể hiện trên hình 6C
Tại cột 'D' ta thấy 1 ô trống; Thực ra tác giả muốn các bạn biết thêm rằng, macro xóa được những ô chứa dữ liệu số lẫn dữ liệu kiểu chuỗi, nên mới chọn hai cột cho macro

làm việc. Hơn nữa record cuối trong trang tính là cố tình đưa vô để chúng ta cùng dễ quan sát & rút kết luận.
Khi xem kết quả, các bạn chịu khó đối chiếu với hình trước nó. Có vậy, các bạn sẽ ngẫm ra, tại sao các ô trong cột 'E' bị xóa đi.



	A	B	C	D	E	F	G	H	I
1	39	Nguyễn Anh	Duy	2/8/1977	Kiến An			25	8
2	40	Trần Bùi Phương	Dung	2/9/1977	Hà tây		x	25	6
3	41	Hoàng Gia	Hưng	2/10/1977		x		25	6
4	42	Trần Minh	Hoàng	2/11/1977		x	x	25	7
5	43	Nguyễn Quốc	Dũng	2/12/1977		x			7
6	44	Vũ Hà	Phy	2/13/1977			x	25	6
7	45	Lương Ngô	Lê	2/14/1977				24	8
8	46	Nguyễn Hồng Minh	Châu	2/15/1977		x	x	23,5	6
9	47	Phạm Minh	Đức	2/16/1977		x		25	6
10	48	Nguyễn	Thảo			x	x	25	7
11	48	Nguyễn	Thảo	2/17/1977	Hà Nội	x	x	25	7
12	40								
13	41								
14	42								

Hình 6C Xóa trùng từ vùng chọn từ hai cột.

7. Tô màu các ô chứa công thức hàm mảng

Để thấy rõ hơn tác dụng của công việc tô màu, chúng ta sẽ xây dựng một hàm mảng tự tạo. (Tất nhiên chúng ta muốn đưa ra thông điệp rằng, một khi đã tô được màu các ô này thì các ô chứa công thức hàm mảng của excel sẽ là chuyện nhỏ xíu).
Điều trước tiên ta sẽ xây 1 hàm mảng tự tạo giải phương trình bậc hai, như sau:

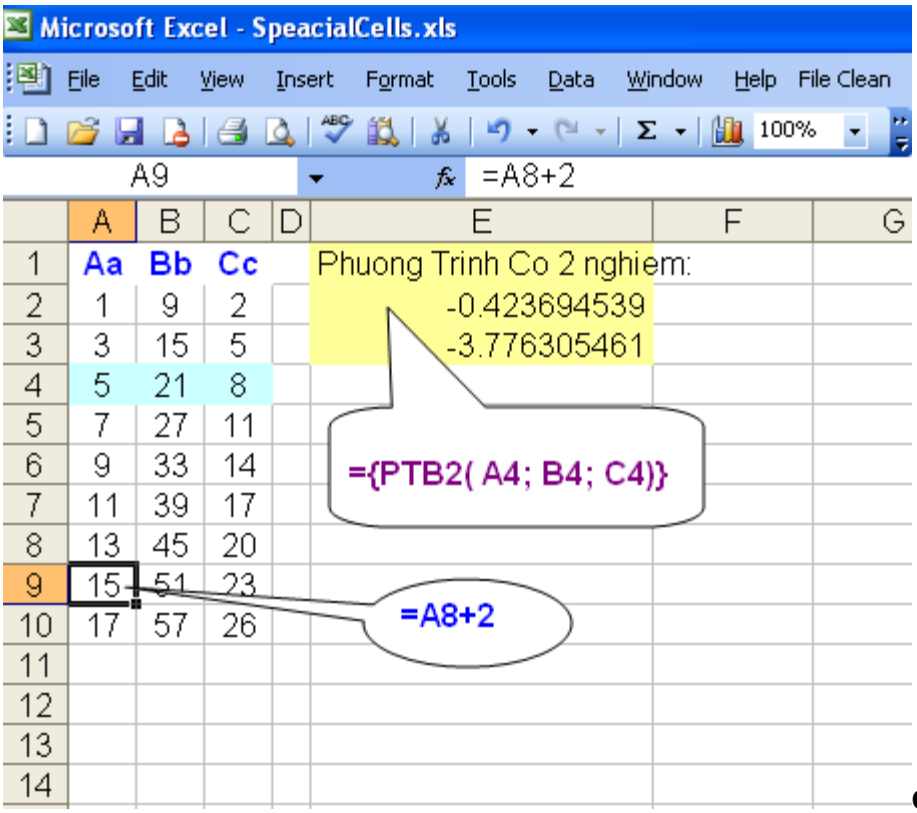
Option Explicit: Option Base 1

```
Function PTB2( aA As Double, Bb As Double, cC As Double)
 ReDim temp( 3, 1):          Dim Delta As Double
 Delta = Bb * Bb - 4 * aA * cC
 Select Case Delta
 Case Is < 0
   temp(1, 1) = "Vo Nghem!"
 Case 0
   temp(1, 1) = "Phuong Trinh Co Nghiem duy nhat:"
   temp(2, 1) = -Bb / (2 * aA)
 Case Is > 0
   temp(1, 1) = "Phuong Trinh Co 2 nghiem:"
   temp(2, 1) = (-Bb + Delta ^ (1 / 2)) / (2 * aA)
   temp(3, 1) = (-Bb - Delta ^ (1 / 2)) / (2 * aA)
 End Select
 PTB2 = temp
End Function
```

Trong hình 7 các ô 'E1:E3' đều có chung công thức như trích đưa ra trong khung chữ nhật. Hơn nữa chúng đang hiển thị kết quả nghiệm của phương trình bậc hai với các hệ số của phương trình ghi tương ứng ở hàng 4 (tô màu nền xanh nhạt). nghiệm thứ nhất ghi tại [e2] & nghiệm thứ hai ghi tại [e3].
Cần lưu ý một điều không thừa là, vì là hàm mảng, nên nó cũng đòi được ứng xử như các hàm mảng trong excel dựng sẵn: kết thúc bằng tổ hợp 3 phím {CTRL}+{ATL}+{ENTER}.
Phần chủ yếu, mà chúng ta quan tâm là macro tô màu cho 3 ô chứa công thức hàm mảng này. Nó có nội dung như sau:

```
Sub Highlight_Arrays()
    Dim Clls As Range
    For Each Clls In Cells.SpecialCells(xlFormulas)
        If Clls.HasArray Then cell.Interior.ColorIndex = 36
    Next Clls
End Sub
```

Thật không có gì phức tạp trong macro này cho lắm. trong vòng lặp chỉ có 1 câu lệnh. Câu lệnh đó là: nếu ô chứa mảng, thì tô màu vàng nhạt cho nó. Nhưng trước đó ta biết rằng, những ô được duyệt chỉ là những ô có công thức trên trang tính mà thôi.
Theo như trong hình, chúng ta có 11 ô chứa công thức. ngoài 3 ô đã liệt kê, chúng ta còn thêm 8 ô tại cột 'A' cũng chứa công thức, nhưng không phải là công thức mảng. các bạn nên tin điều đó là thật, một khi nhìn lên thanh công thức trên hình. Nó đang biểu thị công thức trong ô 'A9' đang được kích hoạt.



GPE.COM

Hình 7 Tô màu vàng nhạt những ô chứa hàm mảng.