

1. Lỗi lặp vô tận - Lỗi từ vòng lặp

1.1 Vòng lặp biết trước số lần lặp For ... Next

Trước khi đi vào chi tiết, tôi muốn nhắc bạn đến một tổ hợp phím là Ctrl + Break. Khi bạn bấm Play và cảm thấy có một điều gì đó không ổn, cụ thể là vào khoảng 10 giây sau mà code bạn vẫn chưa hoàn tất thì bạn nên nhấn tổ hợp phím trên để dừng công việc lại. Bởi vì, ngoại trừ bảng tính Excel của bạn cần phải thao tác với rất nhiều công thức Excel phức tạp cùng với dữ liệu khổng lồ, thông thường, một thủ tục đơn giản sẽ không khiến bạn phải chờ lâu đến thế.

Khi bạn nhấn Ctrl + Break, bạn sẽ bắt gặp một lỗi Run-time.



Lỗi Run-time sau khi nhấn Ctrl + Break

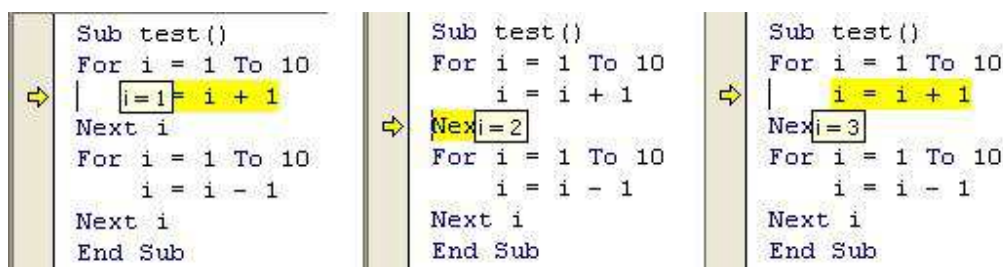
Nếu bạn nhấn Debug, nó sẽ giúp bạn xác định cụ thể đâu là vòng lặp bị lặp vô tận. Bạn có thể thấy rõ trong hình, không phải là vòng lặp thứ nhất mà chính là vòng lặp thứ hai bị lỗi.

```
Sub test()  
  For i = 1 To 10  
    i = i + 1  
  Next i  
  For i = 1 To 10  
    i = i - 1  
    Next i  
End Sub
```

Vòng lặp thứ hai bị lỗi lặp vô tận

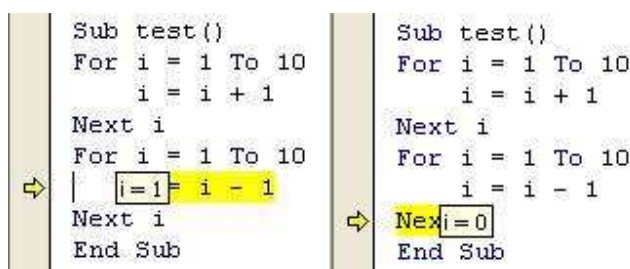
Tôi sẽ bắt đầu với vòng lặp For ... Next. Bạn có thể nhìn thấy trong hình ở phía trên có hai vòng lặp, trong đó, với vòng lặp 1, như bạn đã biết, vòng lặp For sẽ chạy từ i = 1 đến i = 10 thì dừng. Và dòng lệnh i = i + 1 bên trong sẽ chỉ giúp cho vòng lặp chạy nhanh hơn, vì sau một lần chạy, i sẽ tăng lên từ i = 1 thành i = 2 và khi đến lần thực hiện

thứ hai, vòng lặp sẽ có $i = 3$ và cứ thế tiến đến $i = 10$ (bạn có thể dễ dàng nhìn thấy cách chương trình thực hiện vòng lặp bằng cách sử dụng công cụ Debug Step Into).



Quá trình thực hiện vòng lặp thứ nhất

Tuy nhiên, tới vòng lặp thứ hai thì mọi chuyện lại không được tốt đẹp như vậy.



Biến i bị giảm giá trị

Nếu bạn sử dụng Step Into từ từ đến vòng lặp thứ hai, bạn sẽ thấy khi thực hiện vòng lặp lần đầu, biến i sẽ bằng 1, nhưng với dòng lệnh tiếp theo $i = i - 1$ đã khiến cho i giảm xuống bằng 0. Do đó, khi vòng lặp thực hiện lần hai, i sẽ lại tăng giá trị lên bằng 1 rồi lại bị giảm xuống bằng 0. Quá trình cứ tiếp diễn mãi mà i không thể nào thỏa mãn giá trị bằng 10 khiến cho vòng lặp bị lặp vô tận không có điểm dừng.

Hay như là với một ví dụ khác, chúng ta hãy cùng nhau lập bảng cửu chương 2. Bạn cũng biết bảng cửu chương 2 là thế nào rồi đấy, nó bao gồm:

$$\begin{array}{lllll} 2 \times 1 = 2 & 2 \times 2 = 4 & 2 \times 3 = 6 & 2 \times 4 = 8 & 2 \times 5 = 10 \\ 2 \times 6 = 12 & 2 \times 7 = 14 & 2 \times 8 = 16 & 2 \times 9 = 18 & 2 \times 10 = 20 \end{array}$$

Khi nhìn vào bài toán, bạn sẽ thấy có một sự lặp lại số 2 là 10 lần và sẽ có một biến số khác có giá trị 1 và tăng dần đến giá trị 10, chúng ta liền nghĩ đến việc sử dụng vòng lặp. Một mở ngoặc cho ví dụ này là tôi muốn bạn nạp vào một mảng một chiều, mà mảng thì bắt đầu từ phần tử thứ 0. Như vậy, chúng ta sẽ có một đoạn code như thế này:

```
Sub BangCuuChuong2()  
Dim i As Integer  
Dim arr(9)  
For i = 1 To 10  
    i = i - 1
```

```

arr(i) = 2 * i
Next i
End Sub

```

Cho phép tôi diễn giải một chút, chúng ta sẽ có một vòng lặp chạy từ 1 đến 10, và do mảng bắt đầu từ phần tử thứ 0 nên chúng ta cần phải giảm giá trị của i xuống để arr(i) trở thành arr(0) mà nạp vào.

Tiến trình diễn ra như trên nghe thì có vẻ rất hợp lý nhưng khi bạn Play, bạn sẽ gặp phải lỗi vòng lặp vô tận, vậy nguyên nhân là do đâu? Thực ra thì cách lý giải cũng giống như ví dụ trước đó vậy, tức là một mặt vòng lặp đưa giá trị i lên một đơn vị, còn một mặt dòng code trong vòng lặp lại giảm giá trị i trở về một đơn vị, cứ như thế khiến cho vòng lặp không thể nào đạt được điều kiện để thoát ra được. Cách giải quyết cho vấn đề này chính là:

```

Sub BangCuuChuong2()
Dim i As Integer
Dim arr(9)
For i = 1 To 10
    arr(i - 1) = 2 * i
Next i
End Sub

```

Bạn có thấy không? Chỉ cần một sai sót nho nhỏ về tính logic thôi là có thể khiến cho chương trình bị vướng vào một lỗi khá nghiêm trọng.

Một biến thể khác của vòng lặp For ... Next mà nhiều người dễ mắc phải lỗi vòng lặp vô tận hơn chính là vòng lặp lặp từ dưới lên For ... Next Step -x (x là một số nguyên nào đó, chẳng hạn như 1, 2, 3). Cũng với bài toán trên, giả sử rằng tôi đã thành công trong vấn đề nạp bảng cửu chương vào mảng, bây giờ tôi muốn xuất mảng ra bảng tính Excel. Có nhiều cách để thực hiện điều này, nhưng do đề tài đang là vòng lặp nên tôi sẽ xuất mảng ra bằng vòng lặp với đoạn code sau:

```

Sub BangCuuChuong2()
Dim i As Integer, a As Integer
Dim arr(9)
For i = 1 To 10
    arr(i - 1) = 2 * i
Next i
For i = 9 To 0 Step -1
    a = arr(i)
    i = i + 1
    Sheet1.Range("A" & i).Value = a
Next i

```

End Sub

Lần này bạn có cảm thấy điều gì vô lý không? Nếu bạn đang nghi ngờ ở dòng code $i = i + 1$ thì bạn đúng rồi đấy. Nếu giải thích rằng do trong bảng tính không có địa chỉ A0 nên phải tăng biến i lên để có thể xuất ra bắt đầu từ A10 nhằm khớp với lúc kết thúc là A1 thì cũng ổn nhưng mà tăng biến i là phải tăng làm sao đó chứ nếu tăng như trên thì chắc chắn sẽ lại vướng vòng lặp vô tận một lần nữa. Bởi vì, xét theo logic khi vòng lặp giảm một đơn vị còn dòng code lại tăng một đơn vị thì quá trình này sẽ bị lặp đi lặp lại mãi mãi. Bạn có thể sử dụng Step Into để chạy thử một lần, bạn sẽ phát hiện ra ngay.

<pre>Sub BangCuuChuong2() Dim i As Integer, a As Integer Dim arr(9) For i = 1 To 10 arr(i - 1) = 2 * i Next i For i = 9 To 0 Step -1 i = arr(i) i = i + 1 Sheet1.Range("A" & i).Value = a Next i End Sub</pre>	<pre>Sub BangCuuChuong2() Dim i As Integer, a As Integer Dim arr(9) For i = 1 To 10 arr(i - 1) = 2 * i Next i For i = 9 To 0 Step -1 a = arr(i) i = i + 1 Sheet1.Range("A" & i).Value = a Next i End Sub</pre>
<pre>Sub BangCuuChuong2() Dim i As Integer, a As Integer Dim arr(9) For i = 1 To 10 arr(i - 1) = 2 * i Next i For i = 9 To 0 Step -1 a = arr(i) i = i + 1 Sheet1.Range("A" & i).Value = a Next i End Sub</pre>	

Biến i tăng giảm giá trị nhưng mãi không thể thoát khỏi giá trị 9

Do đó, đoạn code trên cần phải sửa như thế này thì mới chính xác:

```
Sub BangCuuChuong2()  
Dim i As Integer, a As Integer  
Dim arr(9)  
For i = 1 To 10  
    arr(i - 1) = 2 * i  
Next i  
For i = 9 To 0 Step -1  
    a = arr(i)  
    Sheet1.Range("A" & i + 1).Value = a  
Next i  
End Sub
```

Khi sử dụng cách lặp từ dưới lên thì biến *i* sẽ phải giảm giá trị xuống thì mới đạt được điều kiện chấm dứt vòng lặp. Tuy nhiên, do thói quen viết code, nhiều người lại quên hẳn mà lại tưởng rằng vòng lặp đi từ trên xuống dẫn đến gặp phải lỗi không đáng có này.

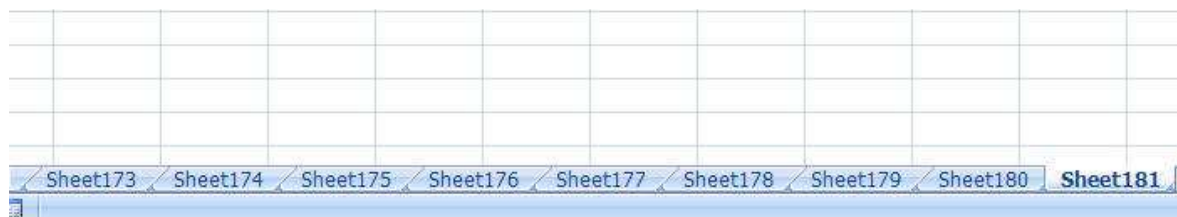
Tổng kết lại cho 2 loại For ... Next, khi lập trình, bạn cần phải lưu ý biến chạy vòng lặp mà trong các ví dụ trên là biến *i* và cần cẩn thận với biến đó, bởi, mọi sự can thiệp của bạn (tăng, giảm hay thậm chí là cố định giá trị biến chạy) mà khiến cho vòng lặp không thể đạt đến giá trị thoát ra thì sẽ dẫn đến lỗi lặp vô tận.

1.2 Vòng lặp For Each ... Next

Lỗi vòng lặp vô tận cũng có thể xảy ra cả với vòng lặp For Each ... Next. Đây là vòng lặp dùng để duyệt qua các phần tử trong một nhóm đối tượng. Tuy nhiên, vấn đề ở đây là giả sử, các phần tử trong nhóm đối tượng này cứ không ngừng tăng lên thì vòng lặp của bạn sẽ gặp phải rắc rối.

```
Sub test()  
Dim sh As Worksheet  
For Each sh In ThisWorkbook.Sheets  
    Sheets.Add After:=Sheets(Sheets.Count)  
Next  
End Sub
```

Với thủ tục trên, vòng lặp sẽ duyệt qua các sheet có trong workbook mà bạn đang sử dụng. Nhưng, cứ mỗi lần vòng lặp của bạn duyệt qua một sheet là y như rằng nó sẽ tăng thêm một sheet và cứ như thế. Hậu quả là số sheet của bạn sẽ tăng lên không dứt và dung lượng file của bạn cũng “khổng lồ” không kém.



Số lượng rất lớn sheet đang tồn tại

1.3 Vòng lặp không biết trước số lần lặp Do ... Loop, While ... Wend,...

Loại vòng lặp này có rất nhiều biến thể như Do Until ... Loop, Do ... Loop Until, Do While ... Loop, Do ... Loop While, While ... Wend hay thậm chí là Do ... Loop, nhưng tất cả đều có chung một bản chất là vòng lặp không biết trước số lần lặp, và vòng lặp chỉ thực sự được hoàn tất nếu:

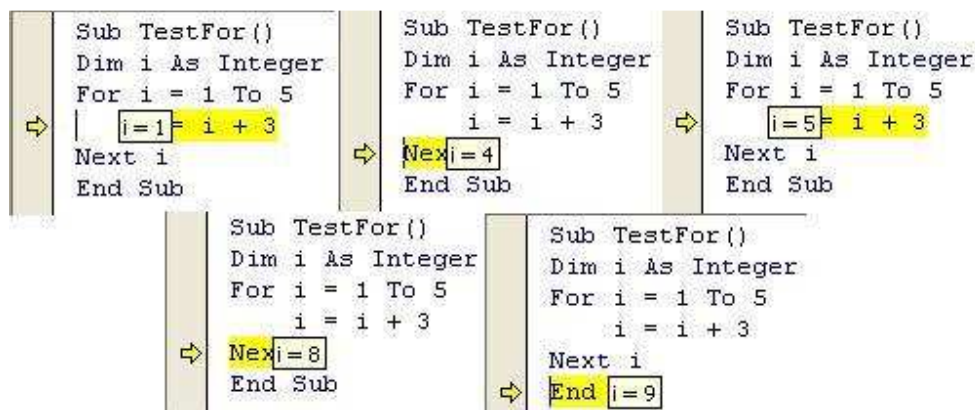
- Với Do While <điều kiện> ... Loop, Do ... Loop While <điều kiện> và While <điều kiện> ... Wend: thực hiện đến khi <điều kiện> không còn đúng nữa thì ngừng.
- Với Do Until <điều kiện> ... Loop và Do ... Loop Until <điều kiện>: thực hiện đến khi <điều kiện> đúng thì ngừng.
- Riêng với Do ... Loop: thực hiện đến khi có lệnh thoát, tức Exit Do thì mới ngừng.

Vậy nguy cơ tiềm tàng lỗi vòng lặp vô tận xuất phát từ đâu? Chúng ta hãy cùng xem một ví dụ. Tôi có hai đoạn code rất đơn giản ở dưới đây.

```
Sub TestFor()  
Dim i As Integer  
For i = 1 To 5  
    i = i + 3  
Next i  
End Sub
```

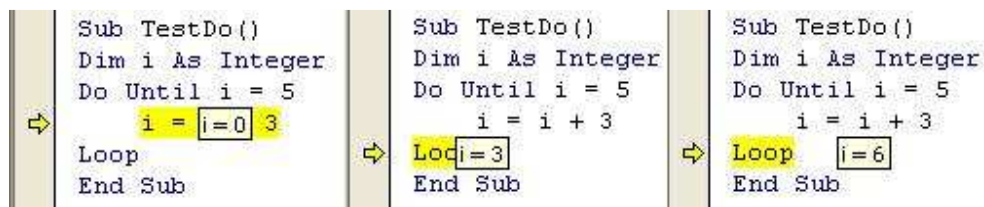
```
Sub TestDo()  
Dim i As Integer  
Do Until i = 5  
    i = i + 3  
Loop  
End Sub
```

Theo như những gì mà tôi đã trình bày ở phần vòng lặp For trên thì đoạn code TestFor dường như sẽ không gặp vấn đề gì, bởi vì ban đầu, biến i sẽ có giá trị là 1, và khi nó thực hiện `i = i + 3`, biến i sẽ có giá trị mới là 4, sau đó do giá trị 4 chưa thỏa mãn vòng lặp nên vòng lặp sẽ được thực hiện lượt thứ hai và biến i sẽ được tăng giá trị lên là 5 và cũng thực hiện dòng code bên trong vòng lặp để tiến đến giá trị là 8. Lượt thứ ba, biến i sẽ được tăng lên bằng 9 nhờ dòng code `Next i`, rồi sau đó vòng lặp sẽ kiểm tra biến i, nó nhận thấy i đã ra khỏi vòng lặp liền kết thúc vòng lặp.



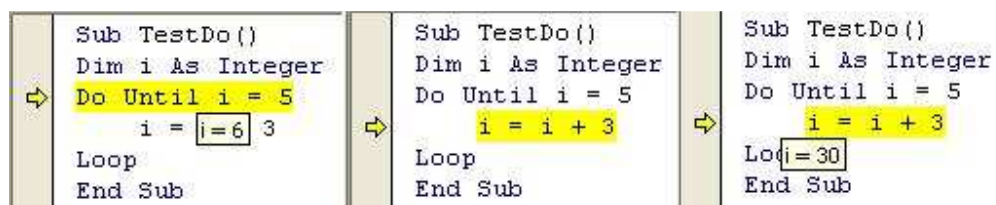
Sự thay đổi giá trị của biến i ở thủ tục TestFor

Nhưng còn với vòng lặp ở thủ tục TestDo thì sao? Chúng ta cùng xem diễn biến của nó nhé.



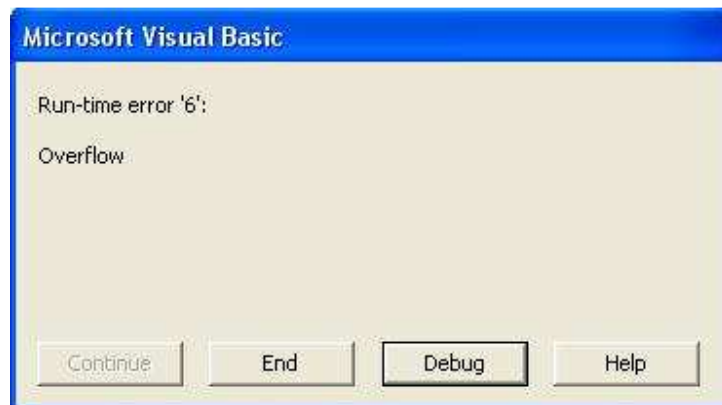
Sự thay đổi giá trị của biến i ở thủ tục TestDo

Khởi đầu, biến i có giá trị bằng 0 và do vòng lặp Do Until ... Loop nên nó sẽ kiểm tra điều kiện trước hết. Nhận thấy i không bằng 5, vòng lặp được tiến hành và dòng code $i = i + 3$ làm cho i tiến lên giá trị 3. Sau đó, nó lại kiểm tra điều kiện và lại thực hiện vòng lặp, i lại một lần nữa tăng lên giá trị 6. Tới đây, một lần nữa vòng lặp sẽ kiểm tra điều kiện, và theo bạn, với giá trị của biến i hiện giờ, vòng lặp có thực hiện tiếp không hay sẽ thoát ra như vòng lặp For ... Next ở trên? Chúng ta cùng xem kết quả với hình dưới đây.



Vòng lặp vẫn tiếp tục thực hiện

Như bạn thấy đấy, do điều kiện không thỏa nên vòng lặp vẫn cứ tiếp diễn mãi, thậm chí biến i đã bằng đến 30 rồi mà vòng lặp vẫn không dứt ra được. Và nếu bạn may mắn, kết quả bạn nhận được sẽ là:



Lỗi Run-time 6

Lỗi Run-time 6 xảy ra do biến *i* của bạn được định nghĩa là Integer nên khi biến *i* tăng vượt quá giới hạn của Integer thì sẽ xảy ra lỗi này. Nhưng trên thực tế, bạn sẽ gặp nhiều bài toán chẳng hạn như biến *i* của bạn là Long (đôi lúc bạn quên khai báo thì nó là Variant) thì chắc chắn bạn sẽ phải cần đến sự trợ giúp của Ctrl + Break.

Vậy để khắc phục đoạn code trên, bạn không chỉ cần tính đến trường hợp thỏa mãn điều kiện mà còn phải tính đến trường hợp nếu như điều kiện của bạn bị vượt qua thì bạn phải làm thế nào. Cụ thể, bạn có thể thêm vào như sau:

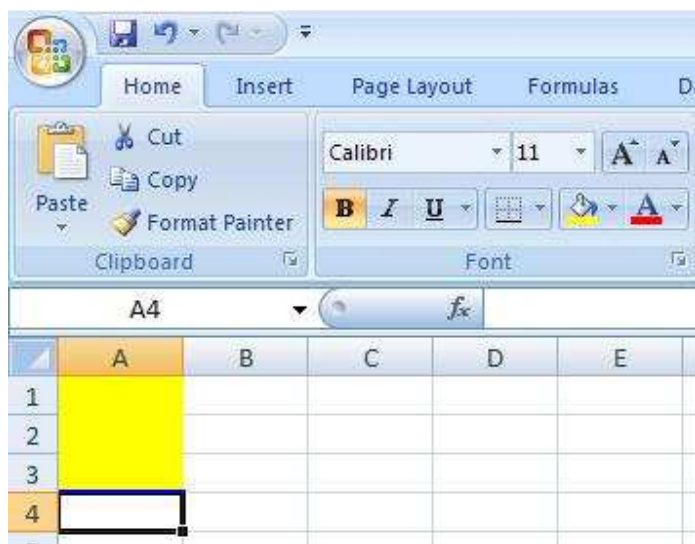
```
Sub TestDo()  
Dim i As Integer  
Do Until i = 5 Or i > 50  
    i = i + 3  
Loop  
End Sub
```

Như thế này thì dù cho biến *i* của bạn không thể đạt được giá trị 5 thì vòng lặp cũng sẽ không bị lặp vô tận khi *i* vượt quá giá trị 50. Bạn có thể hiểu rằng *i* = 5 là điều kiện bạn muốn vòng lặp đạt được còn *i* > 50 là một đường thoát khỏi vòng lặp cho bạn giả sử như điều kiện không thành. Đây chính là một kinh nghiệm cho bạn khi sử dụng loại vòng lặp này.

Chúng ta qua một ví dụ khác với vòng lặp Do While ... Loop nhé. Tôi có một đoạn code sau:

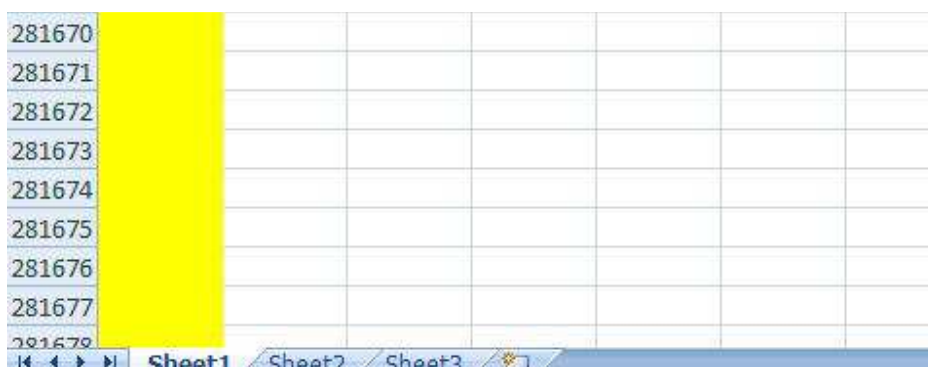
```
Sub TestDo()  
Dim i As Long  
i = 1  
Do While Sheet1.Cells(i, 1).Font.Bold = False  
    Sheet1.Cells(i, 1).Interior.Color = 65535  
    i = i + 1  
Loop
```


Đoạn code này sẽ chạy vòng lặp kiểm tra cột A trong Sheet1 xem có ô nào sử dụng Font chữ in đậm không, nếu có thì dừng, còn nếu ô nào không có sử dụng Font chữ in đậm thì sẽ được tô màu vàng. Với hình dưới đây, tôi đã sử dụng in đậm ở A4 nên bạn có thể thấy được là vòng lặp sẽ ngừng ngay tại A4 và ba ô phía trên đã được tô vàng.



Vòng lặp dừng khi có ô sử dụng Font chữ in đậm

Tuy nhiên, nếu như không có ô nào sử dụng Font chữ in đậm thì sẽ thế nào đây?



Vòng lặp chạy vô tận sẽ tô hết tất cả các ô thành màu vàng

Bạn có thể thấy trên hình số lượng ô được tô vàng là rất lớn, và con số này sẽ còn tiếp diễn cho đến khi nào mà trong cột A không còn ô nào để tô nữa thì chương trình mới báo lỗi. Nhưng, đây cũng có thể là một minh chứng cho bạn thấy rằng lỗi vòng lặp vô tận có thể hiện diện khắp mọi nơi nếu bạn không tính toán kỹ lưỡng. Vậy phương án sửa chữa lỗi ở trên thế nào? Một cách đơn giản là bạn hãy giới hạn biến i lại sao cho khoảng 1000 ô nếu không tìm thấy thì có thể thoát ra được rồi chứ không nên cứ để cho nó chạy hoài như thế.

Một ví dụ khác với vòng lặp Do ... Loop nhé. Chắc bạn sẽ thấy lạ với vòng lặp này. Thực ra, vòng lặp này rất ít người xài nhưng sở dĩ tôi đề cập đến ở đây là vì, thỉnh thoảng, bạn vô tình viết thiếu While hay Until và sẽ làm cho vòng lặp của bạn trở về Do ... Loop. Nếu bạn xài vòng lặp này, bạn phải bảo đảm rằng trong đoạn code của mình có Exit Do.

```
Sub TestDo()  
Dim i As Long  
Do  
    i = i + 1  
Loop  
End Sub
```

Bạn hãy chạy thử đoạn code này mà xem, tôi dám bảo đảm với bạn 100% rằng vòng lặp của bạn sẽ lặp vô tận. Chỉ với việc nhìn vào nó thôi, bạn cũng không thấy có một cái gì để cho biết vòng lặp sẽ dừng khi nào. Do đó, bạn cần phải thay đổi lại đoạn code này như sau:

```
Sub TestDo()  
Dim i As Long  
Do  
    If i = 10 Then Exit Do  
    i = i + 1  
Loop  
End Sub
```

Nếu bạn thêm vào dòng code tô đậm trên, vòng lặp của bạn sẽ giống như Do Until ... Loop vậy, tức là nó sẽ kiểm tra điều kiện ban đầu và thoát ra nếu thỏa. Nhưng như tôi đã nói ban đầu, khi sử dụng những vòng lặp loại không biết trước số lần lặp, bạn cần phải chừa cho mình một con đường thoát để bảo đảm rằng nếu điều kiện bạn cần không thỏa thì bạn vẫn có thể thoát ra khỏi vòng lặp được. Do đó, cũng với đoạn code trên, nếu tôi thay `i = i + 1` thành `i = i + 2`, bạn vẫn dính lỗi vòng lặp vô tận. Thành ra, bạn nên thay đổi lại điều kiện kiểm tra:

```
Sub TestDo()  
Dim i As Long  
Do  
    If i > 10 Then Exit Do  
    i = i + 2  
Loop  
End Sub
```

Bây giờ, bạn hãy thử nghĩ xem nếu bản thân điều kiện của vòng lặp thay đổi thì có dẫn đến vòng lặp lặp vô tận hay không? Tôi có hai thủ tục như thế này:

```

Sub TestFor()
Dim i As Integer, a As Integer
a = 5
For i = 1 To a
    a = a + 1
Next i
End Sub

```

```

Sub TestDo()
Dim i As Integer, a As Integer
a = 5
Do Until i = a
    a = a + 1
    i = i + 1
Loop
End Sub

```

Với thủ tục TestFor, khi bạn chạy nó, biến a vẫn sẽ tăng giá trị lên qua mỗi lượt vòng lặp. Thế nên, với câu hỏi trên, chắc rằng bạn sẽ nghĩ vòng lặp sẽ lặp vô tận phải không?

<pre> Sub TestFor() Dim i As Integer, a As Integer a = 5 For i = 1 To a a = a + 1 ⇒ Ne a=10 End Sub </pre>	<pre> Sub TestFor() Dim i As Integer, a As Integer a = 5 For i = 1 To a a = a + 1 ⇒ Next i End i=5 </pre>
--	---

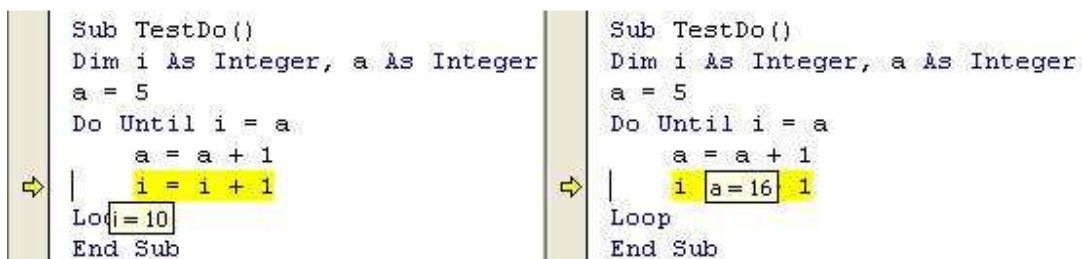
Biến a tăng và biến i đã đến giai đoạn gần kết thúc vòng lặp, vậy vòng lặp sẽ vẫn tiếp tục hay kết thúc?

Thực ra hoàn toàn không phải vậy, bạn có thể kiểm chứng qua biến i. Với thủ tục trên, nếu vòng lặp hoàn tất thì biến i sẽ có giá trị là 6. Qua đó, chúng ta có thể kết luận được rằng, vòng lặp For ... Next sẽ ghi nhận giá trị a ngay từ lần đầu tiên bắt đầu và nó sẽ vẫn sẽ kết thúc bất chấp biến a có tăng hay không tăng (cả trong trường hợp biến a giảm thì cũng không hề thay đổi).

<pre> Sub TestFor() Dim i As Integer, a As Integer a = 5 For i = 1 To a i=6 a = a + 1 Next i ⇒ End Sub </pre>	<pre> Sub TestFor() Dim i As Integer, a As Integer a = 5 For i = 1 To a a = a + 1 Ne a=10 ⇒ End Sub </pre>
---	--

Bất chấp biến a tăng, vòng lặp vẫn kết thúc

Vậy còn với vòng lặp Do Until ... Loop ở trên thì sao? Liệu vòng lặp sẽ kết thúc khi biến i đạt giá trị là 5 như những gì đã từng xảy ra với vòng lặp For ... Next? Câu trả lời chính là tấm hình dưới đây.



Vòng lặp đã bị lặp vô tận

Bạn có thể thấy đấy, thậm chí cả khi $i = 10$ rồi thì vòng lặp vẫn cứ tiếp tục do không thỏa được điều kiện $i = a$.

Một vấn đề khác mà tôi muốn đề cập chung cho tất cả các loại vòng lặp không biết trước số lần lặp trong mục này là, các vòng lặp đều cần có biến chạy (bạn có thể thấy rất rõ ở các ví dụ trên mà tôi đã đưa ra). Không giống như For ... Next khi mà bản thân dòng code Next đã giúp bạn tăng biến chạy lên rồi, với loại vòng lặp không biết trước số lần lặp này, trong nhiều trường hợp, bạn cần phải thiết lập một biến chạy mà bản thân biến chạy đó phải liên kết với điều kiện bạn thiết lập cho vòng lặp, vì nếu không, bạn cũng sẽ rơi vào vòng lặp lặp vô tận. Để hiểu rõ hơn, mời bạn xem một ví dụ dưới đây:

```
Sub TestDo()
    Dim i As Long, j As Long
    j = 1
    Do Until j > 5
        i = i + 1
    Loop
End Sub
```

Bạn hãy nhìn vào ví dụ trên, bạn có thấy được sự liên kết nào giữa biến i và biến j không? Biến chạy i cứ tăng nhưng biến điều kiện j thì không đổi khiến cho vòng lặp không thể thoát ra được. Do đó, tôi đặc biệt nhấn mạnh sự liên kết giữa biến chạy và biến điều kiện.

Tổng kết lại những gì đã đề cập ở trên:

- Ở vòng lặp For ... Next, hãy chú ý đến biến chạy, đừng cố định nó cũng như đừng làm gì khiến cho biến chạy của bạn bị “giậm chân tại chỗ” (tức là một mặt vòng lặp khiến cho nó tăng lên, một mặt bạn làm cho nó giảm đi).

- Với vòng lặp For Each ... Next và các loại vòng lặp không biết trước số lần lặp khác, bạn hãy cẩn thận trong vấn đề vừa tăng biến chạy vừa tăng biến điều kiện vì nếu không cẩn thận, bạn sẽ rơi vào vòng lặp lặp vô tận.
- Với các loại vòng lặp không biết trước số lần lặp, đừng quên biến chạy vì trong các loại vòng lặp này, sẽ chẳng có một biến chạy nào cả. Bên cạnh đó, hãy nhớ rằng, trong nhiều trường hợp, bạn cần phải chú ý về sự liên kết giữa biến chạy và biến điều kiện.
- Hãy chú ý lường trước về một điều kiện thoát khỏi vòng lặp trong trường hợp vòng lặp khó có thể đạt được điều kiện ban đầu bạn đặt ra.