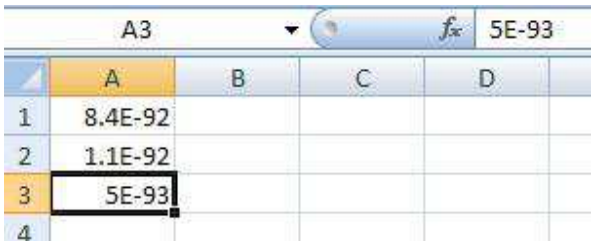
1.1.1 Lỗi vòng lặp lặp vô tận

1.1.1.1 Lỗi từ sự kiện Worksheet_Change

Giả sử bạn là giáo viên và bạn nhập điểm cho học sinh. Thang điểm của học sinh là từ 1 đến 10 và mỗi khi nhập điểm lẻ, bạn sẽ phải gõ "8.4". Để tiết kiệm thời gian, bạn mong muốn khi gõ "84" thì máy sẽ tự hiểu và chuyển đổi thành "8.4" cho bạn. Như phát kiến ra một ý tưởng, bạn bắt đầu tìm hiểu và biết rằng, sáng kiến này của bạn có thể trở thành hiện thực bởi "thầy phù thủy" VBA. Bạn vui mừng và bắt đầu học thì biết được rằng, cần phải viết một đoạn code vào trong sự kiện Worksheet_Change. Sau một thời gian mày mò, bạn đã làm được một đoạn code như sau:

```
Private Sub Worksheet_Change(ByVal Target As Range)
If Intersect(Target, Range("A1:A10")) Is Nothing Then Exit Sub
If IsNumeric(Target) Then Target = Target / 10
End Sub
```

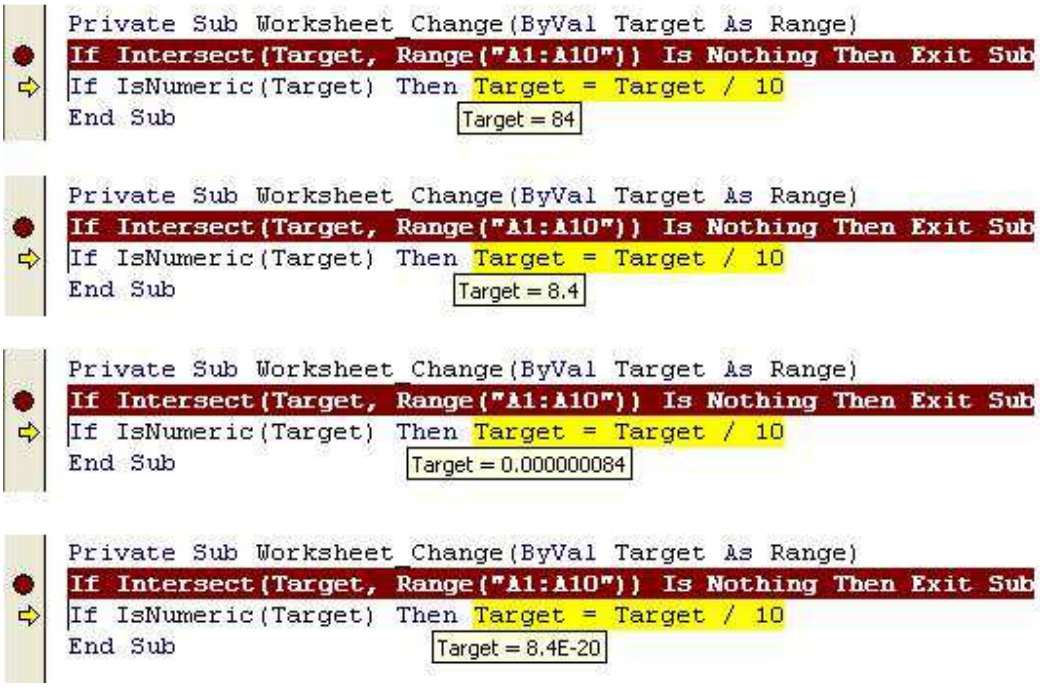
Chúng ta sẽ tìm hiểu về từng dòng code nhé. Dòng code dùng hàm Intersect có nghĩa là nếu dữ liệu nhập vào trong phạm vi A1 đến A10 thì sự kiện mới có tác dụng, còn mọi dữ liệu được nhập ngoài phạm vi này thì sẽ chẳng có chuyện gì xảy ra cả. Tất nhiên, chúng ta có thể tùy biến giá trị này cho phù hợp với yêu cầu của vấn đề. Dòng code thứ hai dùng để kiểm tra dữ liệu nhập vào có phải là kiểu số không, nếu là dữ liệu kiểu số thì nó sẽ được chia cho 10 nhằm để biến "84" thành "8.4", "11" thành "1.1", "05" thành "0.5",… Tuy nhiên, kết quả bạn nhận được là thế này:



Kết quả nhận được sau khi thực hiện đoạn code trên

Tuy rằng đoạn code trên không khiến cho bạn phải nhấn Ctrl + Break, không làm cho máy của bạn bị treo nhưng kết quả rõ ràng là không theo như ý muốn. Chúng ta sẽ khảo sát Step Into để xem lý do tại sao kết quả lại trở nên như vậy.

Do sự kiện Worksheet_Change không thể tự khởi động, bạn cần đặt một nút Breakpoint vào dòng code đầu tiên, sau đó nhập một con số nào đó vào bất kì một ô nào từ A1 đến A10. Ở đây, tôi sẽ nhập "84" vào A1.



Giá trị của ô A1 dần thay đổi

Bạn có thể thấy trên hình, số 84 được nhập vào sẽ được chia cho 10 vào lần đầu tiên. Tuy nhiên, do dòng code `Target = Target / 10` nên giá trị mới - tức là 8.4 - sẽ được nhập thẳng vào A1 để thay thế cho giá trị cũ là 84 dẫn đến một sự thay đổi giá trị trong phạm vi A1 đến A10 và một lần nữa lại kích hoạt sự kiện Worksheet_Change. Cứ như thế mãi, ô A1 sẽ dần bị giảm giá trị đến mức không thể nào giảm được nữa và sẽ ra giá trị như hình kể trên.

Vậy biện pháp nào để khắc phục hiện tượng trên? Thực ra nếu chúng ta đặt thêm một số điều kiện để thoát khỏi sự kiện, chúng ta có thể an toàn qua khỏi lỗi lặp này. Giả sử tôi lý luận rằng, do vấn đề nhập điểm số nên số điểm tối thiểu để nhập chỉ vào khoảng 0.1, tức là chúng ta nhập tối thiểu là số 1 để khi đem chia 10 thì chúng ta sẽ có 0.1. Do đó, tôi sẽ giới hạn đoạn code lại như sau:

```
Private Sub Worksheet_Change(ByVal Target As Range)
If Intersect(Target, Range("A1:A10")) Is Nothing Then Exit Sub
If IsNumeric(Target) And Target > 1 Then Target = Target / 10
End Sub
```

Điều kiện giá trị ô phải lớn hơn 1 đã giúp cho đoạn code chạy suôn sẻ và kết quả ra hệt như mong muốn. Tuy nhiên, trong rất nhiều trường hợp, bạn sẽ không thể nào suy nghĩ ra được những điều kiện hợp lý để có thể bẫy lỗi lặp vô tận kể trên. Do đó, tôi đề nghị với bạn hai dòng code "tuyệt chiêu" mà nó có thể giúp bạn tránh rắc rối lặp đó.

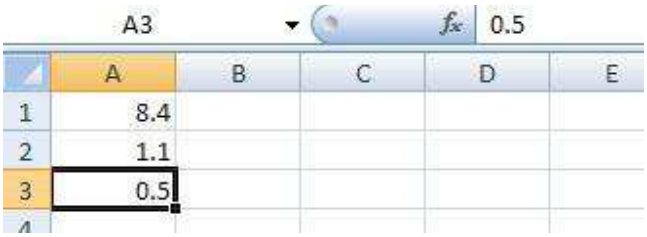
```
Private Sub Worksheet_Change(ByVal Target As Range)
If Intersect(Target, Range("A1:A10")) Is Nothing Then Exit Sub
```

```
Application.EnableEvents = False
If IsNumeric(Target) Then Target = Target / 10
Application.EnableEvents = True
End Sub
```

Bạn có nhìn thấy hai dòng code đó không? Đó chính là `Application.EnableEvents = False` và `Application.EnableEvents = True`. Ý nghĩa của dòng code này tức là, nó sẽ bật/tắt chức năng kích hoạt sự kiện của Excel. Nếu bạn đặt giá trị là False, chức năng kích hoạt sự kiện sẽ tắt và mọi sự kiện như Worksheet_Change, Worksheet_SelectionChange,… đều sẽ không còn hoạt động nữa. Do đó, với sự chi phối của dòng code này, ô A1 dù nằm trong phạm vi của sự kiện nhưng sự kiện vẫn không thể bị kích hoạt, thế nên kết quả là sự kiện này chỉ hoạt động được duy nhất một lần rồi thôi.

Thứ tự thực hiện đoạn code

Như bạn đã thấy trên hình, sự kiện chỉ được thực hiện một lần và rồi nó tiến đến kết thúc luôn chứ không lặp nhiều lần như trước nữa.



Kết quả khi thực hiện đoạn code trên

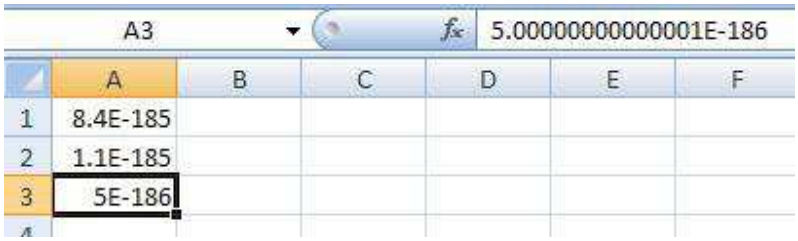
Cũng cần phải nói thêm, giá trị True trong dòng code kế cuối nhằm kích hoạt lại chức năng nhận sự kiện của Excel để khi bạn trực tiếp thay đổi giá trị khác thì sự kiện

Worksheet_Change vẫn còn có thể nhận và thực hiện trở lại. Như trên hình trên, nhờ kích hoạt nên tôi vẫn có thể nhập tiếp "11" để ra "1.1" và "05" để ra "0.5".

Nhưng bạn cần lưu ý rằng, bạn phải bảo đảm rằng sẽ không có dòng code nào ảnh hưởng trực tiếp lên ô có thể kích hoạt sự kiện Worksheet_Change (mà ở trong ví dụ này là từ ô A1 đến A10) sau dòng code `Application.EnableEvents = True` nhé. Vì nếu có, mọi thứ sẽ trở về lại như cũ. Hãy cùng xem đoạn code sau:

```
Private Sub Worksheet_Change(ByVal Target As Range)
If Intersect(Target, Range("A1:A10")) Is Nothing Then Exit Sub
Application.EnableEvents = False
If IsNumeric(Target) Then Target = Target / 10
Application.EnableEvents = True
If IsNumeric(Target) Then Target = Target / 10
End Sub
```

Với đoạn code này, tôi đã tác động tiếp vào ô và thế là mọi tác động của `Application.EnableEvents = False` đã trở nên "công cốc".



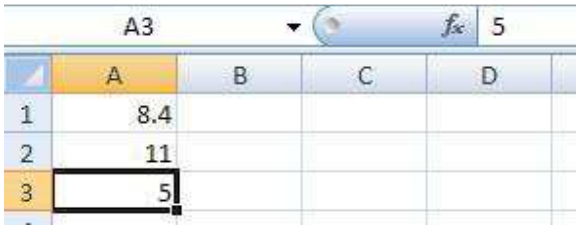
Kết quả vô cùng "thảm thương"

Do đó, tôi khuyên bạn vị trí lý tưởng cho `Application.EnableEvents = True` chính là ở dòng code kế cuối, tức là trước `End Sub`. Ngoài ra, nếu trong đoạn code của bạn có dòng `Exit Sub`, bạn cũng nên đặt trước dòng code này dòng `Application.EnableEvents = True`.

Nếu như bạn không có dòng code kế cuối đó, sự kiện của bạn sẽ vĩnh viễn không thể kích hoạt lại được nữa. Để bạn có thể hiểu hơn về điều này, chúng ta cùng nhau thử nghiệm nhé. Giờ tôi sẽ bỏ đi dòng code kế cuối đó:

```
Private Sub Worksheet_Change(ByVal Target As Range)
If Intersect(Target, Range("A1:A10")) Is Nothing Then Exit Sub
Application.EnableEvents = False
If IsNumeric(Target) Then Target = Target / 10
End Sub
```

Kết quả nó sẽ như thế này:



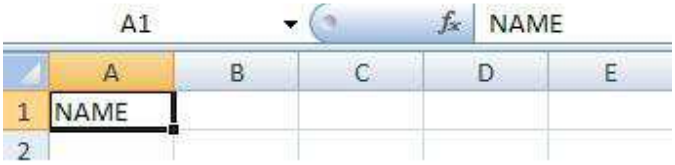
Sự kiện Worksheet_Change chỉ thực hiện một lần

Bạn thấy đấy, sự kiện Worksheet_Change chỉ bắt được một lần mà thôi.[1]

Chúng ta sẽ cùng đến với một ví dụ đơn giản khác. Giả sử bạn muốn mỗi lần bạn gõ vào ô A1 một từ gì đó thì ngay lập tức nó sẽ được viết hoa toàn bộ. Ví dụ bạn gõ "name" thì trên màn hình, ô A1 sẽ có giá trị là "NAME". Thuật toán của bài này cũng rất đơn giản, bạn chỉ cần sử dụng hàm UCase là đủ rồi. Do đó, chúng ta sẽ có đoạn code như sau:

```
Private Sub Worksheet_Change(ByVal Target As Range)
With Target
    If .Address = "$A$1" And .Value <> "" Then
        .Value = UCase(.Value)
    End If
End With
End Sub
```

Đoạn code này được diễn tả như sau, nếu địa chỉ của ô bị thay đổi là A1 và giá trị của nó không phải là giá trị rỗng thì dữ liệu sẽ được viết hoa lên qua hàm UCase. Chúng ta hãy chạy thử xem thế nào nhé, bây giờ bạn hãy gõ từ "name" vào ô A1 đi. Kết quả mà bạn nhận được sẽ là như thế này:

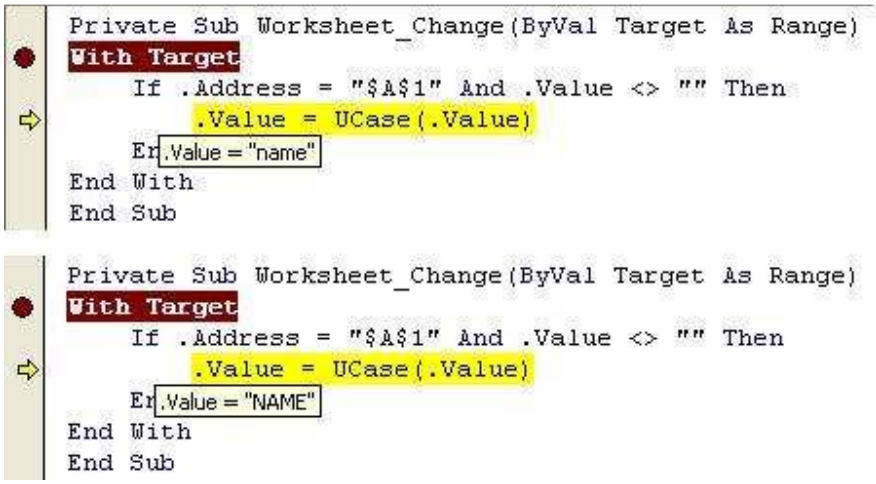


Kết quả của đoạn code biến "name" thành "NAME"

Bạn thấy bình thường phải không? Đó là vì bạn chưa nhận thấy "nguy cơ tiềm tàng" của đoạn code trên. Bây giờ bạn hãy thử làm lại và lần này hãy dùng Breakpoint để chúng ta từ từ khảo sát bằng Step Into nhé.

---

[1] Nếu bạn đã thử đoạn code mà tôi đưa ra ở trên, chắc hẳn rằng kể từ giây phút này, mọi sự kiện trong Excel của bạn sẽ bị vô hiệu hóa. Để thoát khỏi tình trạng này và giúp cho nó trở lại như cũ, bạn có thể sử dụng Immediate Window (Ctrl + G) và nhập vào đó `Application.EnableEvents = True` sau đó, hãy chỉnh sửa lại đoạn code của bạn để không bị rơi vào tình trạng trên nữa.



Khảo sát bằng Step Into

Bạn có thể thấy, lần đầu giá trị của A1 là “name” và sự kiện Worksheet_Change sẽ được kích hoạt và biến nó thành “NAME”. Tuy nhiên, khi nó được biến thành “NAME” và được gán lại vào A1, sự kiện Worksheet_Change lại tiếp tục được kích hoạt và dù rằng giá trị của A1 là “NAME” thì nó cũng không quan tâm, nó vẫn cứ tiếp tục biến “NAME” thành “NAME” để gán lại vào A1. Sau đó, sự kiện lại tiếp tục được kích hoạt do có sự biến đổi trong ô A1, thế là vòng lặp lặp vô tận cứ thế mà tiếp diễn mãi không ngừng. Tuy rằng kết quả bạn có thể thấy nó bình thường thật nhưng thật ra ẩn trong chương trình, đoạn code ấy vẫn cứ âm thầm chạy mãi, nó sẽ chạy đến lúc mà bạn sẽ gặp lỗi như sau:

Lỗi Run-time 28

Khi lỗi này xuất hiện, Excel của bạn sẽ không thể nào bắt sự kiện được nữa dù rằng Application.EnableEvents của bạn vẫn là giá trị True. Bạn chỉ còn cách thoát ra vào lại Excel để cho nó trở lại như cũ mà thôi.

Vậy làm cách nào để khắc phục lỗi đó đây? Cũng vậy, bạn sẽ có hai cách để thoát khỏi vấn đề này. Cách đầu tiên, bạn phải tìm được một điều kiện giúp nó thoát ra một cách êm thắm nhưng cũng phải hợp lý. Tôi đề nghị với bạn một đoạn code thế này:

```
Private Sub Worksheet_Change(ByVal Target As Range)
```

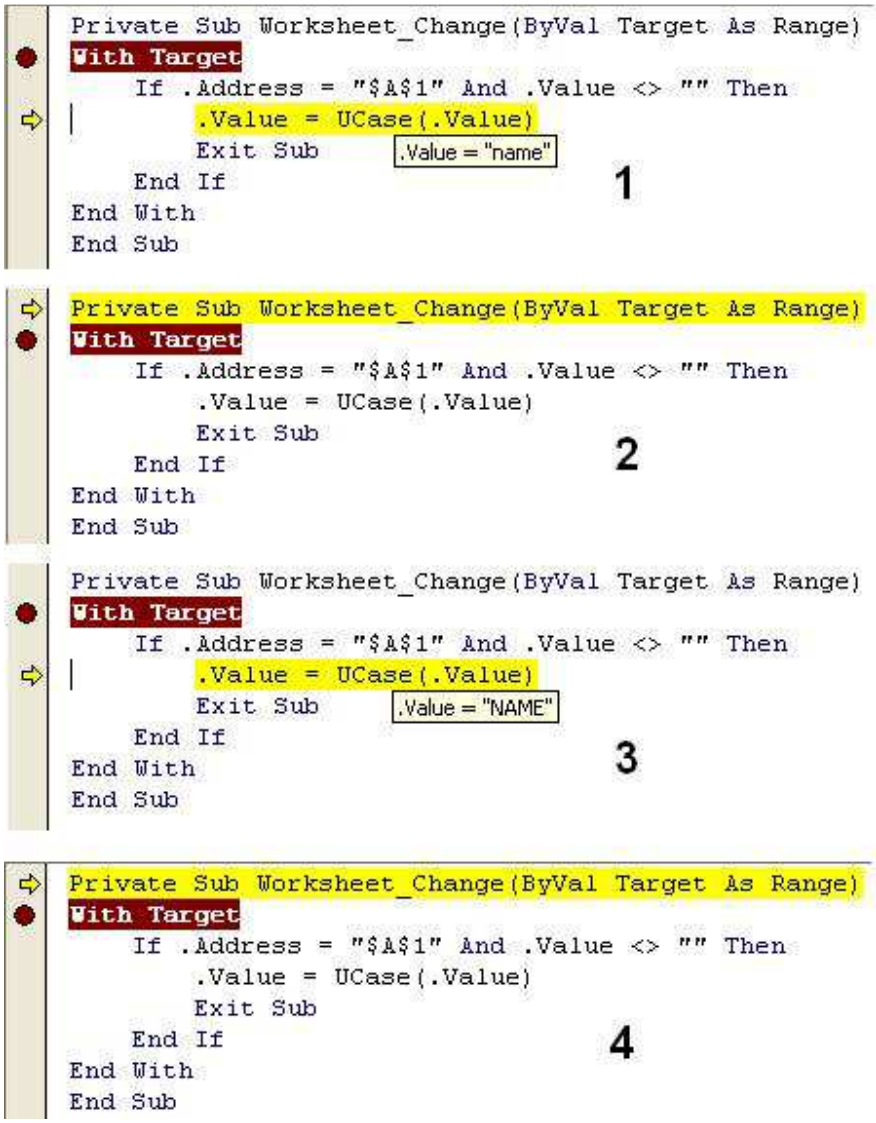
```
With Target
    If .Address = "$A$1" And .Value <> "" Then
        If .Value <> UCase(.Value) Then .Value = UCase(.Value)
    End If
End With
End Sub
```

Điều kiện được thêm vào dùng để so sánh giá trị bạn nhập vào ô A1 có phải là chữ viết hoa toàn bộ không, nếu không thì sự kiện mới tiến hành công việc sử dụng hàm UCase, còn không thì nó sẽ cho qua và kết thúc sự kiện. Với điều kiện này, bạn sẽ tránh khỏi việc sự kiện liên tục dùng hàm UCase và mặc kệ giá trị của A1 ra sao đi nữa.

Vậy ngộ nhỡ bạn chưa thể nghĩ ra được một điều kiện hợp lý nào để thoát ra thì sao? Có một số người nói với tôi rằng chỉ cần thêm Exit Sub là đủ rồi. Họ lý luận rằng khi thực hiện xong công việc thì gặp Exit Sub, sự kiện sẽ tự thoát chứ không có lặp lại. Thực sự thì kết luận của họ là sai lầm, bạn có thể kiểm chứng rõ ràng qua Step Into. Chúng ta hãy xem lại đoạn code trên khi sử dụng Exit Sub.

```
Private Sub Worksheet_Change(ByVal Target As Range)
With Target
    If .Address = "$A$1" And .Value <> "" Then
        .Value = UCase(.Value)
        Exit Sub
    End If
End With
End Sub
```

Bạn cần phải hiểu rõ bản chất của sự kiện Worksheet_Change là hễ có một sự thay đổi nào trong phạm vi thay đổi thì nó sẽ kích hoạt sự kiện bất chất là ở đầu đoạn code, giữa đoạn code, cuối đoạn code hay thậm chí là kết thúc đoạn code. Do đó, khi hàm UCase thực hiện xong công việc của nó thì cũng là lúc một giá trị mới được gán vào phạm vi thay đổi, thế là Worksheet_Change lại được kích hoạt lại và nó sẽ thực hiện sự kiện lại từ đầu.



Thứ tự thực hiện của đoạn code trên

Như bạn đã thấy trong hình, khi đoạn code thực hiện đến dòng `.Value = UCase(.Value)` thì ngay lập tức nó sẽ chuyển ngược lại lên dòng code đầu tiên và tiếp tục thực hiện đến dòng có hàm UCase là lại trở ngược lại ban đầu. VBA hoàn toàn không đả động gì đến `Exit Sub`.

Do đó, bạn cần hết sức chú ý và cẩn thận khi sử dụng loại sự kiện này, tốt nhất bạn nên dùng Step Into để thực hiện từ từ nhằm phát hiện ra những sai sót của mình và để hiểu rõ hơn về cách thực hiện của sự kiện.

Bây giờ, tôi đề nghị bạn hãy đưa hai dòng code mà tôi đã bày cho bạn lúc ban đầu, đó là `Application.EnableEvents = False` và `Application.EnableEvents = True` vào và xem sự thay đổi của nó nhé.

```
Private Sub Worksheet_Change(ByVal Target As Range)
Application.EnableEvents = False
```

```
With Target
    If .Address = "$A$1" And .Value <> "" Then
        .Value = UCase(.Value)
    End If
End With
Application.EnableEvents = True
End Sub
```

Kết quả của đoạn code trên là:

Đoạn code chạy từ đầu đến cuối mà không bị lặp

Nhờ có cặp dòng code trên, đoạn code dù gặp hàm UCase, dù giá trị A1 có thay đổi giữa chừng nhưng nó vẫn được thực hiện từ đầu đến cuối mà không hề bị lặp lại.

Tổng kết lại những gì đã nói ở trên, chúng ta có những kết luận sau:

- Khi sử dụng sự kiện Worksheet_Change và các sự kiện khác tương tự, bạn cần lưu ý đến vấn đề giá trị bị lặp nhiều lần vì đôi khi nó không hiện rõ trên bảng tính Excel, cũng không có thông báo gì, nhưng nó khiến cho chương trình của bạn nặng nề và chậm.

- Nếu có thể, hãy để ý đến cả những điều kiện thoát sự kiện cần thiết.
- Sử dụng Step Into (F8) để theo dõi nhất cử nhất động khi sự kiện được kích hoạt nhằm giúp bạn có được một cái nhìn tổng quát về những gì sẽ diễn ra.
- Hãy sử dụng cặp code Application.EnableEvents = False và Application.EnableEvents = True cũng như đặt nó tại vị trí hợp lý.