

Gioco del 15, in doppia salsa excelliana

(Il modello **Gioco del 15.xls** è disponibile per il download all'indirizzo <http://www.giannigiaccaglini.it/download/Gioco%20del%2015.xls>)

Sono sicuro che a molta gente, perlomeno a coloro che sono dotati di sufficiente fantasia e pertanto sanno che Excel NON serve solo per aridi bilanci e scontate previsioni di vendite trimestrali, è già venuta in mente la possibilità di implementare questo giochino in Excel + VBA.

A beneficio di tutti i giocherelloni e in omaggio alle sospirate vacanze prossimo venturo, ecco due soluzioni possibili, entrambe fanno ovviamente riferimento a uno schema come il seguente:

GIOCO DEL 15

5	10	7	15
6	2	3	4
11	12		8
14	1	9	13

Prima soluzione

L'intervallo di cui alla figura precedente ha nome **"Gioco15"**, complice l'abuso di fantasia. Venendo subito al sodo, ecco il relativo codice macro, interamente ospitato nel modulo **Foglio1**:

```
Dim VettoriAssegnati As Boolean 'Switch definito a livello modulo
'Vettori di nomi sempre definiti a livello modulo
Dim Vett, Vett1, Vett2, Vett3, Vett4, Vett5, Vett6, Vett7, Vett8, _
    Vett9, Vett10, Vett11, Vett12, Vett13, Vett14, Vett15, Vett16

Private Sub NominaCelle()
    Nomi = Array("Uno", "Due", "Tre", "Quattro", "Cinque", "Sei", _
        "Sette", "Otto", "Nove", "Dieci", "Undici", "Dodici", _
        "Tredici", "Quattordici", "Quindici", "Sedici")
    For Each c In Range("Gioco15")
        c.Name = Nomi(i)
        i = i + 1
    Next
End Sub

Private Sub AssegnaVettori()
    If VettoriAssegnati Then Exit Sub
    Vett1 = Array("Due", "Cinque")
    Vett2 = Array("Uno", "Tre", "Sei")
    Vett3 = Array("Due", "Quattro", "Sette")
    Vett4 = Array("Tre", "Otto")
    Vett5 = Array("Uno", "Sei", "Nove")
    Vett6 = Array("Due", "Cinque", "Sette", "Dieci")
    Vett7 = Array("Sei", "Tre", "Otto", "Undici")
    Vett8 = Array("Quattro", "Sette", "Dodici")
    Vett9 = Array("Cinque", "Dieci", "Tredici")
    Vett10 = Array("Sei", "Nove", "Undici", "Quattordici")
    Vett11 = Array("Sette", "Dieci", "Dodici", "Quindici")
    Vett12 = Array("Otto", "Undici", "Sedici")
    Vett13 = Array("Nove", "Quattordici")
    Vett14 = Array("Dieci", "Tredici", "Quindici")
    Vett15 = Array("Undici", "Quattordici", "Sedici")
    Vett16 = Array("Dodici", "Quindici")
    VettoriAssegnati = True
End Sub

Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, _
    Cancel As Boolean)
    Cancel = True
    If Target.Cells.Count > 1 Then Exit Sub 'Evita selezioni multiple
    If Intersect(Target, Range("Gioco15")) Is Nothing Then Exit Sub
    If IsEmpty(Target) Then Exit Sub
    AssegnaVettori
    Select Case Target.Name.Name
        Case "Uno": Vett = Vett1
        Case "Due": Vett = Vett2
        Case "Tre": Vett = Vett3
        Case "Quattro": Vett = Vett4
        Case "Cinque": Vett = Vett5
        Case "Sei": Vett = Vett6
        Case "Sette": Vett = Vett7
        Case "Otto": Vett = Vett8
        Case "Nove": Vett = Vett9
        Case "Dieci": Vett = Vett10
        Case "Undici": Vett = Vett11
        Case "Dodici": Vett = Vett12
        Case "Tredici": Vett = Vett13
        Case "Quattordici": Vett = Vett14
```

```

    Case "Quindici": Vett = Vett15
    Case "Sedici": Vett = Vett16
End Select
ColorTarget = Target.Interior.ColorIndex
DatoTarget = Target.Value
For i = 0 To UBound(Vett)
    With Range(Vett(i))
        If .Value = "" Then
            .Copy Target
            .Interior.ColorIndex = ColorTarget
            .Value = DatoTarget
        End If
    End With
Next
End Sub

```

Macchinosa anzichenò? È vero, ma le ridondanza è dovuta soltanto all'adozione di una selva di nomi opportuni, dopo di che il funzionamento è relativamente semplice:

1. dare un doppio clic su una cella dello schema;
2. vedere il risultante spostamento della cella cliccata nel buco, se e solo se questo si trova accanto ad essa.

Commenti essenziali

La Sub **NominaCelle** prepara il vettore *Nomi* contenente le stringhe "Uno", "Due", ... "Sedici" quindi le assegna come nomi (dinamici!) alle celle dell'intervallo "Gioco15").

Nota – Palesemente è una finezza, questi battesimi si potevano compiere a mano, con noti comandi **Inserisci > Nome...**

Segue **AssegnaVettori** che inserisce nei vari *Vett1*, *Vett2*, ..., *Vett16* i nomi delle celle circostanti – sopra, sotto, a sinistra e a destra – di ciascuna di quelle dello schema. Tali operazioni si compiono solo se *VettoriAssegnati = True* (con =True sottinteso).

A questo punto interviene la routine dell'evento doppio-click. Tralasciandone l'esordio, incluso il richiamo di *AssegnaVettori*, la struttura **Select Case** discrimina i possibili nomi della cella colpita (*Target*) in ciascuno dei *Case* previsti ponendo il relativo *Vett1*, *Vett2*, ecc. in un solo array **Vett**.. A quel punto si registrano il colore e il valore della cella *Target* in *ColorTarget* poi si avvia un ciclo che spazzola *Vett* individuando, con **Range(Vett(i))** le celle circostanti il *Target* per vedere se è vuota. Nel qual caso si opera lo scambio desiderato, su cui mi affido al comprendonio di chi legge.

Seconda soluzione

Fa ovviamente riferimento a uno schema del tutto simile al precedente, ma con due varianti:

- l'intervallo, supposto collocato in un altro foglio di lavoro, si chiama "GiocoDel15";
- l'evento sfruttato è il clic semplice, ergo la routine è la **Selection_Change** di questo foglio.

Il codice, presente nel modulo, diciamo, **Foglio2** dell'Editor VBA stavolta è più compatto:

```

Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)
    Cancel = True
    MsgBox "In questo foglio devi usare il clic NORMALE...", vbInformation, "Attenzione"
End Sub

Private Sub Worksheet_SelectionChange(ByVal Target As Range)

```

```

If Target.Cells.Count > 1 Then Exit Sub 'Evita selezioni multiple
If Intersect(Target, Range("GiocoDell5")) Is Nothing Then Exit Sub
If IsEmpty(Target) Then Exit Sub
Dim OffRiga, OffColonna
Dim TargOffset As Range
Dim depColore As Integer, depDato As Integer
OffRiga = Array(-1, 0, 1, 0)
OffColonna = Array(0, 1, 0, -1)
depColore = Target.Interior.Color
depDato = Target.Value
For i = 0 To UBound(OffRiga)
    Set TargOffset = Target.Offset(OffRiga(i), OffColonna(i))
    With TargOffset
        If IsEmpty(.Cells(1)) Then
            .Copy Target
            .Interior.Color = depColore
            .Value = depDato
        End If
    End With
Next
End Sub

```

Commenti essenziali

Nessuno! Mi affido all'esegesi autogestito di chi legge, dico solo che:

- a) stavolta, come ripeto, l'evento sfruttato è il cambio selezione, che corrisponde al clic semplice sulla cella Target;
- b) **OffRiga** e **OffColonna** sono vettori contenenti gli scarti (*Offset*) di riga e di colonna del Target, in base ai quali vengono anche qui esplorate le celle circostanti scambiando il Target col buco, se possibile.

Rimescolamento caselle

Sono quasi certo che tutti gl'informatici per ottenere questo scopo penseranno di ricorrere ad operazioni casuali di spostamento del buco, magari con sofisticati algoritmi ricorsivi...

Da praticone invece ho pensato a un altro algoritmo: *eseguire il sort sia per righe che per colonne* (che Excel consente) dello schemino. A tal fine ho inserito nella riga sopra e nella colonna a sinistra funzioni =CASUALE(), occultate con il formato numerico ";;;" assegnato a tali celle. Dopodiché ecco il codice rimescolante, associato a un pulsante **CommandButton1**, ottenuto quasi solo tramite Registratore:

```

Private Sub CommandButton1_Click()
    Disordina
End Sub

Sub Disordina()
    'Sort per righe
    Range("B5:F8").Sort Key1:=Range("B5"), Order1:=xlAscending, _
    Header:=xlGuess, OrderCustom:=1, MatchCase:=False, _
    Orientation:=xlTopToBottom, DataOption1:=xlSortNormal
    'Sort per colonne
    Range("C4:F8").Sort Key1:=Range("C4"), Order1:=xlAscending, _
    Header:=xlNo, OrderCustom:=1, MatchCase:=False, _
    Orientation:=xlLeftToRight, DataOption1:=xlSortNormal
End Sub

```

Va da sé che il riordino secondo numeri casuali, di fatto crea disordine. Aggiungo solo che “sento” che questi scambi di colonne e righe non dovrebbero creare schemi irrisolvibili (*), ma non ne sono sicuro. Ancora una volta ci vorrebbe un matematico che dimostri, se vero, tale *teorema*. Se costui c'è e mi legge, batta un colpo...

Nota (*) – Ricordo che uno schema che ammettesse, ad esempio, una soluzione coi valori 15-14-13 nell'ultima riga non sarebbe compatibile con quello regolare.