Creare e richiamare procedure Sub e Funzioni

Se volete sviluppare una applicazione VBE, una delle prime cose che dovete sapere è la differenza tra *Funzioni* e *Subroutine*, quest'ultime note anche come *Procedure Sub*. In entrambi i casi si tratta di una sorta di "raggruppamento" di istruzioni che svolgono un'operazione comune, e per questo sono molto simili, praticamente tutto il codice di un programma è contenuto all'interno di funzioni e routine. Conoscere la differenza tra i due tipi di procedure aiuterà a prendere la decisione giusta su quale usare, in quanto consentono di suddividere il codice in blocchi, ognuno dotato di una propria specifica funzionalità, evitando ripetizioni e garantendo una miglior leggibilità del codice stesso. La differenza fondamentale tra le due è che le procedure eseguono operazioni di tipo generale, come una serie di istruzioni collegate ad un evento o richiamate nel codice principale, mentre le funzioni rappresentano un procedimento di calcolo o una elaborazione che deve restituire un valore come risultato al programma chiamante.

Se si scrive lo stesso codice più volte, l'applicazione che state costruendo potrebbe beneficiare di una procedura, piuttosto che duplicare il codice in più posizioni, che oltre a rendere più difficoltosa l'operazione di *Debug* rende il programma stesso più grande di quanto dovrebbe essere. Una procedura è indicata anche con il termine di *macro*, che è rappresentata come un insieme di codici che rendono Excel in grado di eseguire un'azione. Una procedura è una sequenza di istruzioni che inizia con la parole chiave *Sub*, seguita dal nome della routine, e poi da due parentesi, al cui interno è possibile inserire dei parametri (opzionali) richiesti della procedura, ai quali non c'è limite, come numero, e termina con la parola chiave *End Sub*

La procedura Sub

Sub è la forma breve di sotto-routine, ed è utilizzata per gestire una certa attività all'interno di un programma. Le subroutine vengono usate per dividere i task in singole procedure, cosa molto utile in quanto la divisione di un programma in procedure e sotto procedure lo rende più leggibile e riduce le probabilità di errore. Le subroutine possono accettare alcuni argomenti come parametri ma non restituiscono nessun valore alla procedura o alla funzione chiamante. Una procedura Sub inizia con la parola chiave *Sub*, seguita dal nome e poi da una serie di parentesi e termina con l'istruzione *End Sub* nel mezzo tra le due parole chiave va inserito il codice. La sintassi è:

```
Sub nome_routine ()
'istruzioni della subroutine
End Sub
```

Oppure, nel caso si voglia passare dei parametri alla Sub si usa questa sintassi:

```
Sub nome_routine (param1 As tipo, param2 As tipo)
 'istruzioni della subroutine
End Sub
```

Dove *param1* e *param2* rappresentano i parametri che vengono passati alla procedura. Si noti come la dichiarazione di una subroutine è molto simile a come si dovrebbe dichiarare una variabile, specificando il nome del parametro e il tipo di dati, dove ciascun parametro passato alla procedura è trattato come una variabile locale nella subroutine, il che significa che la "durata" del parametro è la stessa di quella della procedura. Ci sono due modi per passare una variabile in ingresso ad una procedura: per *valore* (o ByVal) oppure per *riferimento* (o ByRef). La differenza fra le due modalità sta nel rendere disponibile alla procedura o una copia del valore della variabile, o il contenuto vero e proprio della stessa. Sintatticamente si indica la modalità per valore o per riferimento premettendo *ByVal* o *ByRef* rispettivamente prima della dichiarazione dei parametri di ingresso

Sub nome_routine (ByVal As Type)
Sub nome_routine (ByRef As Type)

Si consideri che in assenza del tipo di dati nella dichiarazione della routine, per impostazione predefinita, una subroutine tratta gli argomenti passati per valore, il che significa che la procedura non può modificare il contenuto della componente variabile nel codice chiamante. Per richiamare una subroutine che non prevede il passaggio di parametri da altra routine, si può alternativamente utilizzare l'istruzione ***Call*** seguita dal nome della routine oppure indicare il solo nome della routine da eseguire.

```
Call aggiungi1
aggiungi1
```

Nel caso si debba richiamare una routine che necessita il passaggio di parametri allora è necessario ricorrere all'istruzione *Call* e inserire, dopo il suo nome, i valori separati da una virgola e racchiusi tra parentesi.

Call aggiungi1(x)

Si consideri il seguente programma come esempio:

```
Sub prova1()
Dim x As Integer
x = 5
MsgBox x
Call aggiungi1(x)
MsgBox x
End Sub

Sub aggiungi1(ByVal i As Integer)
i = i + 1
End Sub
```

In questo esempio viene assegnato il valore 5 alla variabile **x**, e poi tramite la funzione *MsgBox* ne viene stampato il valore a video, successivamente viene chiamata la procedura *aggiungi1* che prende in ingresso una copia del valore di **x** e lo incrementa. Questa operazione però non ha effetto sul contenuto della variabile x infatti quando successivamente viene stampato il valore di x rimane sempre 5. Se invece consideriamo il seguente listato

```
Sub prova2()
Dim x As Integer
x = 5
MsgBox x
Call aggiungi2(x)
MsgBox x
End Sub

Sub aggiungi2(ByRef i As Integer)
i = i + 1
End Sub
```

Questa volta la variabile **x** è stata passata per riferimento (ByRef), questo significa che la procedura va a modificare il contenuto della variabile, Infatti alla fine dell'elaborazione il valore di x è stato incrementato e vale 6

La procedura Function

Abbiamo già detto che una procedura sub è un insieme di codici VBA o istruzioni che svolgono un'azione, ma non *restituiscono un valore*, mentre invece una funzione è anche usata per eseguire un'azione o un calcolo, cosa che può essere eseguita anche da una sub routine, ma la Function *restituisce un valore*. Il nome della funzione può essere utilizzato nell'ambito della procedura per indicare il valore restituito dalla funzione stessa e non può essere creata utilizzando il registratore di Macro. Una routine *Function* può essere richiamata utilizzando il nome in un'espressione che non è possibile per una procedura sub, che invece necessita di essere richiamata solo da una dichiarazione autonoma. Poiché una funzione restituisce un valore, può essere utilizzata direttamente in un foglio di lavoro inserendo il nome della funzione dopo aver digitato un segno di uguale

Una Function inizia con la parola chiave *Function*, seguito da un nome e poi da una serie di parentesi, inoltre è necessario specificare il tipo di dati che corrisponde al tipo di valore che la funzione restituirà, digitando la parola chiave **As** dopo le parentesi seguita dal tipo di dati.

```
Function nome_funzione(param1 As tipo_param1, param2 As tipo_param2)  As tipo_valore
... istruzioni funzione
End Function
```

Esempio di funzione:

```
Function calcola_area(Raggio As Double) As Double
 calcola_area = Raggio * Raggio * 3.14
End Function
```

Le funzioni possono essere create senza argomenti e con qualsiasi numero di argomenti che sono elencati tra le parentesi e in caso di argomenti multipli, si deve separarli con una virgola e termina con un'istruzione ***End Function*** che può essere digitato, andando alla riga successiva dopo aver digitato la dichiarazione viene inserita automaticamente da VBE. Le istruzioni espresse in codice VBA vanno collocate tra la dichiarazione *Function* e la dichiarazione finale *End Function*. La funzione può essere richiamata da un'altra procedura, vale a dire una sub o una function e può essere utilizzata direttamente in una formula del foglio di lavoro inserendo il nome della funzione dopo aver digitato un segno di uguale. Vediamo alcuni esempi di Funtion e come si chiamano:

Esempio: Utilizzare una function per inserire il valore restituito in una cella del foglio attivo

```
Function part_nome() As String
Dim nome As String
Dim cognome As String
nome = InputBox("Inserire nome")
cognome = InputBox("Inserire il cognome")
part_nome = nome & "" & cognome
End Function

Sub nome_int()
ActiveSheet.Range("A1") = part_nome
End Sub
```

Esempio: Usare una Function per fare calcoli e ricevere il risultato come valore ritorno

```
Function area1() As Double
Dim i As Integer
i = InputBox("Immettere il raggio del cerchio")
area1 = i * i * 3.14
area1 = Format(area1, "#. # #")
End Function

Sub calcola_1()
MsgBox "L'area del cerchio è:" & " " & area1
End Sub
```

Esempio: Passare una variabile alla Function per eseguire dei calcoli

```
Function tre(i As Integer) As Long
tre = i * 3
End Function

Sub chiama_3()
Dim a As Integer
a = InputBox("Immettere un numero intero")
MsgBox tre(a)
End Sub
```

Si può osservare che quando la Function non riceve argomenti l'insieme di parentesi, dopo il nome della procedura, è vuoto, tuttavia, quando gli argomenti sono passati a una procedura secondaria o a una funzione da altre procedure, allora questi sono elencati tra le parentesi.

Esempio: La funzione calcola_1 restituisce un valore che può essere utilizzato in una sub

```
Function calcola_1() As Double
Dim a As Double
Dim b As Double
a = InputBox("Inserire il primo numero")
b = InputBox("Inserire il secondo numero")
calcola_1 = (a + b) * 2
End Function

Sub calcola()
Dim c As Double
c = calcola_1 * 5 / 2
MsgBox c
End Sub
```

Esempio: La funzione calcola_2 non restituisce un valore che può essere utilizzato dalla procedura chiamante.

```
Sub calcola_2()
Dim a As Double
Dim b As Double
Dim c As Double
a = InputBox("Inserire il primo numero")
b = InputBox("Inserire il secondo numero")
c = (a + b) * 2
MsgBox c
End Sub

Sub usa_calcola2()
calcola_2
End Sub
```

Regole e Convenzioni per assegnare il nome alle procedure

Ci sono alcune regole che devono essere seguite quando si assegna il nome alle procedure, una di queste è quella di assegnare un nome che rifletta l'azione che eseguirà la stessa, si può usare una frase differenziando le parole da un carattere di sottolineatura o una lettera maiuscola: esempio *cerca_N* può equivalere a cerca nomi, si sconsiglia di usare nomi molto lunghi, inoltre il nome deve iniziare con una lettera, sono da evitare i numeri o un carattere di sottolineatura. Un nome può essere costituito da lettere, numeri o caratteri di sottolineatura, ma non può avere un periodo o caratteri di punteggiatura o caratteri speciali: esempio (.) @ # $% ^ & * () + - = [] {}; ':! ",. / <> \ |? ` ~ .

Il nome dovrebbe essere costituito da una stringa di caratteri continui, senza spazio intermedio e può avere un massimo di 255 caratteri, inoltre i nomi delle procedure non possono utilizzare parole chiave o riservate come And, Or, Loop, Do, Len, Close, data, ElseIf, Else, Select, etc. che VBA utilizza come parte del suo linguaggio di programmazione.

Procedure Public o Private

Una procedura VBA può essere utilizzata in ambito pubblico o privato e questo metodo di procedura può essere specificato con la parola chiave *Public* o *Private*, se non viene specificato il metodo, VBA tratta le procedure come pubbliche di default. Una procedura dichiarata come Private può essere richiamata solo da tutte le procedure nello stesso modulo e non saranno visibili o accessibili alle procedure di moduli esterni nel progetto e non apparirà nella finestra di dialogo Macro. La Procedura pubblica invece può essere richiamata da tutte le procedure dello stesso modulo, ma anche da tutte le procedure di moduli esterni nel progetto e il suo nome verrà visualizzato nella finestra di dialogo Macro e può essere eseguito da esso.

Esempi di una procedura pubblica:
Sub calcola1 ()
Public Sub calcola1 ()
Function calcola1 () As Double
Public Function calcola1 () As Double

Esempi di una procedura privata:
Private Sub calcola1 ()
Private Function calcola1 () As Double

Moduli Standard

Questi sono anche indicati come moduli di codice o semplicemente moduli, e ciascun modulo può essere utilizzato per coprire un certo aspetto del progetto. La maggior parte del codice VBA e le funzioni personalizzate (cioè Funzioni definite dall'utente denominate UDF) sono collocati in moduli di codice standard che non vengono utilizzati per le procedure di evento o per gli eventi di applicazioni create in un modulo di classe dedicato. Per quanto riguarda gli eventi non connessi con oggetti, come il metodo *OnTime* (attiva automaticamente un codice VBA ad intervalli periodici o in un giorno o momento specifico) e il metodo *OnKey* (eseguire un codice specifico su pressione di un tasto o una combinazione di tasti), essendo questi metodi non associati a un particolare oggetto il loro codice viene inserito in un modulo standard.

Modulo Workbook

ThisWorkbook è il nome del modulo per la cartella di lavoro e viene utilizzato per inserire eventi nella cartella di lavoro e eventi Application. Gli eventi *Workbook* sono azioni connesse con la cartella di lavoro per innescare un codice VBA o macro, vale a dire, apertura, chiusura, salvataggio, attivazione e disattivazione del foglio di lavoro sono esempi di eventi delle cartelle di lavoro. Con l'evento *Open* della cartella di lavoro, è possibile eseguire una sub automaticamente quando una cartella di lavoro è aperta e il codice dell'evento della cartella di lavoro deve essere inserito nel modulo di codice per l'oggetto ThisWorkbook, mentre se sono posti in moduli di codice standard Excel non sarà in grado di trovarli ed eseguirli. Anche se eventi Application possono essere creati in qualsiasi modulo oggetto, è meglio che siano posizionati in un modulo oggetto ThisWorkbook oppure è possibile creare un modulo di classe per gestirli.

Modulo Foglio

Un modulo foglio ha lo stesso nome del foglio di lavoro con cui è associato (Foglio1, Foglio2 etc.) e in VBE, il codice del nome può essere modificato solo nella finestra *Proprietà* e non a livello di programmazione, inoltre il modulo foglio può essere per un foglio di lavoro o un grafico e viene usato per posizionare gli eventi innescati o da eventi associati al foglio di lavoro, o da macro. Per utilizzare una routine evento del foglio di lavoro, in VBE si deve selezionare il foglio di lavoro dalla casella *Progetti* e quindi selezionare una procedura corrispondente dalla finestra del *Codice* e una volta selezionato l'evento specifico, si deve inserire il codice VBA che si desidera eseguire. Si ricorda che se il codice viene immesso in un modulo standard, Excel non sarà in grado di trovare le istruzioni ed eseguirle. Le principali istanze di evento sono: la selezione di una cella o cambiando la selezione di celle in un foglio di lavoro, cambiare il contenuto di una cella, selezionare o attivare un foglio, calcolare un foglio di lavoro e così via. Ad esempio, con l'evento *Change*(modifica) di *Worskheet*, una procedura viene eseguita automaticamente quando si modifica il contenuto di una cella del foglio di lavoro.

Modulo UserForm

Procedure di eventi per *Userform*, o per i suoi controlli, sono posizionati nel modulo di codice della Form prescelta, e sono moduli pre-determinati, cioè che si verificano per un particolare uso della Form e/o dei relativi controlli, esempi in questo caso includono gli eventi: Initialize, Activate o Click. Per raggiungere un evento è necessario fare doppio clic sul corpo della Form per visualizzare il modulo di codice, quindi selezionare Userform o il suo controllo dalla casella *Oggetti* e quindi selezionare una procedura corrispondente dalla casella *Routine*. Dopo aver selezionato l'evento specifico, inserire il codice VBA che si desidera eseguire, ricordate di impostare la proprietà *Name* dei controlli prima di usare le routine di evento per loro, altrimenti sarà necessario cambiare il nome della procedura che corrisponde al nome del controllo.

Esecuzione di Function

La Function può essere chiamata da un'altra procedura vale a dire una procedura Sub o Function, e la funzione può essere utilizzata direttamente nel foglio come una formula inserendo il nome della funzione dopo aver digitato un segno di uguale.

Esecuzione di procedure di evento

Gli eventi sono azioni eseguite, o eventi, che innescano una macro VBA e vengono attivati quando si verifica un evento come apertura, chiusura, attivazione, disattivazione della cartella di lavoro, di una selezione di celle o di una sola cella, in pratica consiste nel fare un cambiamento del contenuto di un foglio di lavoro o di una cella. Le procedure di evento sono collegate agli oggetti, una procedura evento è una procedura con un nome standard che viene eseguito sul verificarsi di un evento corrispondente. Le procedure di evento sono attivate da un evento predefinito e vengono installate all'interno di Excel con un nome standard. Si consideri la procedura di modifica del foglio di lavoro che viene installata con il foglio di lavoro nella procedura *Private Sub Worksheet_Change (ByVal Target As Range)* che viene richiamata automaticamente quando un oggetto riconosce il verificarsi di un evento che possa modificare l'oggetto stesso. Una procedura evento per un oggetto è una combinazione del nome dell'oggetto (come specificato nella proprietà Name), un carattere di sottolineatura e il nome dell'evento.

Esecuzione routine Sub

Nella scheda *Visualizza* della barra multifunzione, cliccare su *Macro* nel gruppo Macro e poi su *Visualizza macro* che aprirà la finestra di dialogo macro. Nella finestra di dialogo Macro, selezionare il nome della macro e fare clic su *Esegui* per eseguire la Sub/macro. È inoltre possibile aprire la finestra di dialogo macro facendo clic su Macro nel gruppo Codice nella scheda Sviluppo sulla barra multifunzione oppure utilizzare la combinazione di tasti *Alt + F8* per aprire la finestra di dialogo macro.

Tasto di scelta rapida: Per utilizzare il tasto di scelta rapida associato alla macro è necessario avere già assegnato un tasto alla macro che può essere fatto sia nel momento in cui si inizia la registrazione di una macro o poi selezionando il nome della macro nell'elenco delle macro presenti nella finestra di dialogo Macro e facendo clic sul pulsante Opzioni .

Per eseguire una macro dall'editor di VBE si deve cliccare all'interno della procedura e premere F5 (o fare clic su Esegui nella barra dei menu) oppure dal menu Strumenti nella barra dei menu e cliccare su macro che apre la finestra di dialogo Macro, e selezionare il nome della macro e fare clic su Esegui per eseguirla. Ovviamente questi metodi non sono molto user friendly per eseguire le macro, un modo migliore per eseguire una macro potrebbe essere quella di fare clic su un pulsante accompagnate da un testo auto-esplicativo che appare sul foglio di lavoro

Si consideri che è possibile assegnare una macro a qualsiasi controllo modulo agendo dalla scheda Sviluppo sulla barra multifunzione, fare clic su Inserisci nel gruppo Controlli, selezionare e fare clic su pulsante nei controlli modulo e quindi fare clic sul foglio di lavoro in cui si desidera posizionare l'oggetto, poi di deve fare clic col destro del mouse sul pulsante e selezionare Assegna macro nella finestra che appare dalla quale è possibile selezionare e assegnare una macro al pulsante

Assegnare più Macro

È possibile assegnare più macro a un oggetto, un'immagine o un controllo, chiamando altre procedure secondarie. Consultare di seguito, ad esempio, nella procedura secondaria (CommandButton1_Click) che viene eseguito facendo clic su un pulsante di comando, altre due procedure secondarie denominate Macro1 e Macro2 sono chiamate ed eseguite. In questo esempio le macro sono chiamati con i loro nomi, ma è possibile visualizzare tutte le macro facendo clic su macro nel gruppo di macro (nella scheda Visualizza della barra multifunzione) e poi selezionando Visualizza macro. Ricordarsi di digitare i nomi di macro su righe separate durante la loro chiamata.

```
Private Sub CommandButton1_Click ()
Macro1
Macro2
End Sub
```