Lavorare con stringhe di testo in Excel VBA

Un tipo di variabile che abbiamo toccato marginalmente in questo corso è il tipo **String** che come suggerisce il suo nome, viene utilizzato per contenere le stringhe di testo. Questo tipo di variabile viene utilizzata moltissimo, quindi vale la pena di approfondire questo argomento. Per impostare una variabile per contenere il testo è sufficiente usare la parola chiave **Dim** seguita dal nome della variabile e dalla notazione **As String** in questo modo: *Dim MyString As String*, mentre per memorizzare il testo all'interno della variabile è necessario circondarlo con virgolette doppie così: *MyString = "Testo"*

Si tenga presente che anche se si inseriscono dei numeri racchiusi dalle virgolette vengono trattati come testo e non come numeri: *MyString = "25"*, in questo modo viene assegnato il valore 25 come testo, e non memorizza nella variabile il numero 25. È inoltre possibile inserire il testo in una cella del foglio di calcolo in questo modo:

```
Dim MyString As String
MyString = "Testo"
ActiveCell.Value = MyString
```

Oppure si può ottenere il testo dalla cella del foglio di calcolo:

```
Dim MyString As String
MyString = ActiveCell.Value
```

Molto spesso, però, è necessario fare delle elaborazioni con il testo che si ottiene da una cella di un foglio di calcolo. Ad esempio, potrebbe essere necessario prendere un nome completo da una cella e posizionare il primo nome in un'altra cella e il cognome in un'altra. Per fare le cose in questo modo, è necessario sapere come utilizzare le funzioni stringa incorporate di Excel VBA.

Alcune funzioni elencate in questo articolo sono già state trattate in questo corso, vengono riprese in questo contesto per mostrarne l'uso con altre funzioni VBA

Le funzioni LCase e UCase

Le funzioni **Ucase** e **Lcase** vengono utilizzate per modificare le lettere in caratteri minuscoli o maiuscoli, dove *Ucase* converte tutti i caratteri in lettere maiuscole, mentre *Lcase* converte in lettere minuscole. Vediamo come funzionano con un esempio pratico, creiamo una nuova cartella di lavoro vuota e inseriamo alcune voci nelle celle A1, B1 e C1 come si vede in figura, e inseriamo un nome nella cella A2

	A	B	C
1	Testo	Lcase	Ucase
2	Nelson Mandela		
3			

Fig. 1

A questo punto entriamo nell'editor VBA e inseriamo una routine in un Modulo e scriviamo del codice per leggere il valore presente nella cella A2 come il seguente

```
Sub Prova1 ()
Dim Nome As String
Nome = Range ("A2"). Value
End Sub
```

Con questo codice abbiamo creato una variabile denominata *Nome*, che abbiamo dichiarato come String e per inserire un valore in questa variabile, abbiamo usato l'enunciato *Range ("A2"). Value*. Per convertire il testo in minuscolo, è necessario solo il comando: *LCase (Testo_da_convertire)*

Qualunque cosa si sta cercando di convertire va posto tra le parentesi tonde della funzione *LCase* e il testo che si sta tentando di convertire può essere immesso direttamente, circondato da virgolette, o in una variabile che contiene una stringa di testo. Se vogliamo inserire il testo presente nella cella A2 convertito nella cella B2, tutto quello che dobbiamo fare è scrivere un codice come il seguente:
Range ("A2"). Offset (, 1) .Value = LCase (Nome)

Con *Range ("A2"). Offset (, 1)* ci si sposta di una colonna a destra della cella A2, e poi accediamo alla proprietà *Value* della cella stessa e a destra del segno di uguale inseriamo la funzione LCase col nome della variabile che conterrà il testo da convertire e il codice sarà simile a questo

```
Sub Prova1 ()
Dim Nome As String
Nome = Range ("A2"). Value
Range("A2").Offset(, 1).Value = LCase(Nome)
End Sub
```

Se eseguiamo questa routine possiamo vedere nel foglio la conversione del testo che appare in questo modo

	A	B	C
1	**Testo**	**Lcase**	**Ucase**
2	Nelson Mandela	nelson mandela	
3			

Fig. 2

Al tempo stesso se vogliamo convertire il nome in maiuscolo il codice è molto simile, basta solo cambiare il nome della funzione in questo modo:*Range("A2").Offset(, 2).Value = UCase(Nome)*

Se guardiamo questa nuova riga di codice, notiamo che sono cambiate solo due cose, il valore di Offset, in quanto abbiamo un 2 invece di 1, questo perché ci si sposta di due colonne a destra della cella A2 e la funzione che converte in maiuscolo è *UCase*. Se si aggiunge questa la linea al codice e si manda in esecuzione la routine si ottiene

	A	B	C
1	**Testo**	**Lcase**	**Ucase**
2	Nelson Mandela	nelson mandela	NELSON MANDELA
3			

Fig. 3

Così ora abbiamo trasformato il valore della cella A2 a caratteri minuscoli e maiuscoli. Si noti che il nome in A2, Nelson Mandela, è scritto con la prima lettera maiuscola, in gergo si capitalizza la prima lettera di

ogni parola. Excel dispone della funzione di conversione, e usando VBA possiamo usare la funzione chiamata **Proper** che permette la capitalizzazione di una stringa utilizzando il seguente codice:

```
Dim Nome As String
Nome = "nelson mandela"
Range ("A2"). Offset (, 3) .Value = Application.WorksheetFunction.Proper (Nome)
```

	A	B	C	D
1	**Testo**	**Lcase**	**Ucase**	
2	Nelson Mandela	nelson mandela	NELSON MANDELA	Nelson Mandela
3				

Fig. 4

Il codice che converte la prima lettera in maiuscolo è: *Application.WorksheetFunction.Proper (Nome)*
Si consideri che Application è un oggetto di livello superiore, ovvero l'intero Excel,
mentre WorksheetFunction viene utilizzato per accedere alla funzione di Excel, mentre tra le parentesi tonde di Proper, si digita la variabile che si sta tentando di convertire. Così come la digitazione di un nome di variabile, è possibile digitare il testo direttamente circondato da virgolette.

Le funzioni Trim e Len

La funzione **Trim** viene utilizzata per tagliare lo spazio bianco indesiderato nel testo, quindi, se si ha la seguente stringa: " Un testo" usando Trim si eliminerebbero gli spazi e rimane "Un testo", mentre invece la funzione **Len** viene utilizzata per ottenere il numero di caratteri di una stringa. Se si inserisce il codice seguente in una routine:

```
Dim Nome As String
Dim LNome As Integer
Nome = "  Nelson Mandela  "
LNome = Len (Nome)
MsgBox LNome
```

Abbiamo creato due variabili, una chiamata *Nome* e una chiamata *LNome* e quest'ultima è stata dichiarata come Integer. Nella variabile Nome abbiamo archiviato il testo " Nelson Mandela ", ma abbiamo due spazi vuoti alla sinistra del nome e due spazi vuoti a destra, mentre la quarta linea è questa: *LNome = Len (Nome)*

In questa riga di codice stiamo utilizzando la funzione **Len** e tra le parentesi tonde abbiamo la variabile Nome. La funzione Len conterà quanti caratteri sono presenti nel testo che abbiamo memorizzato all'interno della variabile Nome. Quando VBA ha eseguito questo calcolo lo memorizza nella variabile denominata LNome. Poiché la funzione Len conta i caratteri, il valore restituito sarà un numero intero. Una volta eseguito il codice verrà mostrato in una finestra il numero 18, tuttavia, il nome Nelson Mandela è lungo 14 caratteri, la finestra di messaggio visualizza 18 perché ha contato gli spazi extra all'inizio e alla fine della stringa. Per rimuovere lo spazio, si deve utilizzare la funzione Trim in questo modo:*Nome = Trim (" Nelson Mandela ")*

Il testo o la variabile che si sta cercando di tagliare va tra le parentesi tonde alla destra della funzione e VBA quindi rimuovere qualsiasi spazio bianco dalla parte anteriore e dalla fine della stringa. Se si esegue di nuovo il codice la finestra di messaggio visualizza il valore di 14.

La funzione Space

A volte si rende necessario inserire uno spazio vuoto prima o dopo una stringa, per fare questo si usa la funzione space inserendo il numero di spazi tra le parentesi tonde. Ecco un codice per illustrare la funzione:

```
Dim Nome As String
Nome = "Nelson Mandela"
MsgBox Len(Nome)
Nome = Space(5) & Nome
MsgBox Len(Nome)
```

Eseguendo il codice viene visualizzata una finestra di messaggio che riporta il valore 14, che è il numero di caratteri che sono nel nome Nelson Mandela, mentre la seconda finestra di messaggio visualizza un valore di 19, che contiene i 14 caratteri originali, più i 5 aggiunti all'inizio del nome. Avremmo potuto aggiungere 5 spazi alla fine del nome in questo modo: *Nome = Nome & Space (5)*. Non deve trarre in inganno l'uso di due volte della variabile Name, infatti se analizziamo la riga di codice: *Nome & Space (5)*

Questa ci dice: "Prendi tutto ciò che è presente nella variabile denominata Nome e aggiungi 5 spazi."(Il simbolo & viene utilizzato per concatenare) e una volta che VBA ha concatenato il testo e lo spazio, si deve memorizzare da qualche parte, e il luogo in cui sarà memorizzato è alla sinistra del segno uguale. A sinistra del segno di uguale, abbiamo di nuovo la variabile Nome, pertanto qualunque sia stato il valore in precedenza della variabile sarà sostituito dal valore dalla destra del segno di uguale.

La Funzione Replace

Abbiamo già visto che la funzione **Replace** permette di sostituire il testo in una stringa con qualcos'altro. Supponiamo, per esempio, di avere una parola errata nella cella A5. È possibile utilizzare replace per modificare le lettere non corrette con quelle corrette.
È possibile utilizzare il foglio di lavoro per provarlo, aggiungendo due voci nelle celle A4 e B4 e digitare il titolo Originale nella cella A4 e la voce Sostituisci nella cella B4. Ora se inseriamo all'interno della cella A5 la parola errata Micrasaft, il foglio di calcolo dovrebbe apparire così:

	A	B	C
1	**Testo**	**Lcase**	**Ucase**
2	Nelson Mandela	nelson mandela	NELSON MANDELA
3			
4	**Originale**	**Sostituito**	
5	Micrasaft		
6			

Fig. 5

Per utilizzare la funzione Replace, sono necessari almeno tre argomenti tra la parentesi tonde:

Replace(string_to_search, string_to_replace, replace_with)

La prima cosa che serve è una stringa di testo da ricercare, poi si specifica il carattere o i caratteri che si vuole sostituire e infino il nuovo personaggio o personaggi. Con la funzione Replace si hanno anche tre optional che è possibile specificare. Queste sono: *start, count, compare.* I parametri opzionali vanno dopo il terzo elemento in sostituzione, con ciascuno dei quali è separato da una virgola:

Replace(string_to_search, string_to_replace, replace_with, start, count, compare)

Il parametro Start rappresenta il carattere della stringa in cui si desidera avviare la ricerca, il valore di default è il carattere 1, che è il primo carattere della stringa. Se si vuole iniziare da una posizione nella stringa diverso dal primo carattere, quindi è necessario digitare il numero di partenza.
Il parametro count è il numero di occorrenze da sostituire, il default è di sostituire ogni occorrenza presente in replace_with . Se si desidera solo per sostituire, le prime due ricorrenze quindi si digita il numero 2.

Il parametro compare ha tre opzioni: vbBinaryCompare, vbTextCompare, vbDatabaseCompare, ma non preoccupatevi di confrontare, in quanto viene utilizzato raramente. A titolo di esempio, aggiungiamo una nuova routine con il seguente codice

```
Dim OTesto As String
Dim CTesto As String
OTesto = Range("A5").Value
CTesto = Replace(OTesto, "a", "o")
Range("A5").Offset(, 1).Value = CTesto
```

Come si può vedere abbiamo due variabili stringa, Otesto e Ctesto, il valore per la variabile Otesto viene ricavato dalla cella A5 sul foglio di calcolo. Abbiamo poi abbiamo questa riga di codice: *CTesto = Replace (OTesto, "a", "o")*, in cui abbiamo la funzione Replace a destra del segno di uguale, per cui la prima voce tra le parentesi tonde del Replace è il nome della variabile Otesto e rappresenta il testo da ricercare. In seguito viene elencato il carattere che non è corretto (la lettera " a "), infine, abbiamo bisogno del nuovo testo che vogliamo nella stringa, che è la lettera " o ". Tutti e tre gli elementi sono separati da virgole. La riga finale inserisce il testo corretto nella cella B5 del foglio di calcolo. Eseguendo il codice il foglio di calcolo dovrebbe sembrare come questo:

	A	B	C
1	**Testo**	**Lcase**	**Ucase**
2	Nelson Mandela	nelson mandela	NELSON MANDELA
3			
4	**Originale**	**Sostituito**	
5	Micrasaft	Microsoft	
6			

Fig. 6

È possibile sostituire più di un carattere, se è necessario. Il seguente codice sostituisce l'errata Microsft con Microsoft: *Ctesto = Replace(Otesto, "sft", "soft")*

È possibile sostituire spazi nel testo digitando due virgolette. La prima serie di virgolette avrà uno spazio tra loro, mentre la seconda serie non ha spazio. Ad esempio: *Ctesto = Replace("M i c r o s o f t ", " ", "")*

Questa volta, la parola Microsoft ha uno spazio dopo ogni lettera, se vogliamo rimuovere gli spazi, il secondo parametro della funzione Replace è di due virgolette con uno spazio tra di loro, mentre il terzo parametro è di due virgolette senza spazio tra loro. Due doppi apici insieme significano "nessun carattere".

Le Funzioni InStr, InStrRev, StrReverse

InStr è l'abbreviazione di inString, e questa funzione stringa viene utilizzata per la ricerca di una stringa all'interno di un'altra. Necessita di almeno due elementi tra le parentesi tonde della funzione InStr: il testo da cercare, e ciò che si desidera trovare. VBA poi vi darà un intero in cambio.

Questo numero sarà 0 se la stringa non viene trovata, mentre se la stringa viene trovata, allora si ottiene la posizione di inizio della stringa di ricerca. Ecco un esempio per provare:

```
Dim Email As String
Dim Posizione As Integer
Email = "mia_email@prova.com"
Posizione = InStr(Email, "@")
MsgBox Posizione
```

Abbiamo dichiarato due variabili, una è una variabile Stringa che contiene un indirizzo email, e l'altra è un intero chiamato posizione. La linea di codice della funzione InStr linea è questa: *Posizione = InStr(Email, "@")*

La prima voce tra le parentesi tonde è la variabile email, mentre il secondo elemento è quello che vogliamo cercare, cioè la chiocciolina dell'indirizzo di posta elettronica. Se il simbolo @ non è nella variabile Email, VBA colloca uno 0 nella variabile Posizione. Quando si esegue il codice verrà visualizzato una finestra di messaggio con il numero 10, in quanto il simbolo @ è il decimo carattere della stringa indirizzo email. Ora se NON inseriamo il simbolo @ nella stringa della E-mail: *Email = "mia_emailprova.com"*

Eseguendo il codice la finestra di messaggio visualizza il valore 0, è possibile utilizzare questo risultato per un test di base sugli indirizzi di posta elettronica con questo codice:

```
Dim Email As String
Dim Posizione As Integer
Email = "mia_emailprova.com"
Posizione = InStr(Email, "@")

If Posizione = 0 Then
MsgBox "Non hai inserito un indirizzo email valido"
Else
MsgBox "Indirizzo email valido"
End If
```

Ci sono due parametri facoltativi per la funzione InStr e sono: *Start* e *Compare* e la sintassi è la seguente: *InStr(start, Text_To_Search, Find, comapre)*

Se si omette il numero di partenza, la funzione InStr cerca dall'inizio della stringa, mentre se si digita un numero per la partenza InStr inizia la ricerca da quel numero di carattere nella stringa.

Il parametro compare ha quattro opzioni:

- vbUseCompareOption
- vbBinaryCompare
- vbTextCompare
- vbDatabaseCompare

Questi parametri vengono utilizzati raramente. Simile a Instr c'è la funzione **InStrRev**, che sarebbe l'acronimo di Reverse, infatti questa funzione è la stessa di InStr con la sola differenza che InStrRev inizia la ricerca dalla fine della stringa invece che all'inizio.

La funzione StrReverse

Questa funzione è abbastanza facile da usare, come suggerisce il suo nome StrReverse inverte le lettere in una stringa di testo. Ecco un po' di codice per provare:

```
Dim Otesto As String
Dim Rtesto As String
Otesto = "Inserisci una frase"
Rtesto = StrReverse(Otesto)
MsgBox (Rtesto)
```

Quando viene eseguito il codice, la finestra di messaggio visualizzerà il testo invertito della variabile Otesto.

Le Funzioni Left e Right

Le funzioni **Left** e **Right** vengono utilizzate per tagliare i caratteri da una stringa. Si utilizzare Left per tagliare i caratteri dall'inizio della stringa, e Right per tagliare i caratteri a partire dalla fine della stringa. Tra le parentesi tonde di sinistra e destra della funzione si digita il numero di caratteri che si desidera tagliare. Qualche esempio può chiarire le cose.

```
Dim Email As String
Email = "mia_email@prova.com"
MsgBox Left(Email, 9)
MsgBox Right(Email, 9)
```

Nella prima riga si dichiara una variabile String e nella seconda si inserisce un indirizzo e-mail nella variabile Email, mentre nella terza linea si utilizza una finestra di messaggio che utilizza la funzione Left: *MsgBox Left(Email, 9)*

Quando si esegue il codice vedrete che la finestra di messaggio visualizza i primi 9 caratteri dell'indirizzo e-mail, partendo dalla sinistra del simbolo @. La quarta linea è questa: *MsgBox Right(Email, 9)*

La funzione di destra visualizza 9 caratteri a partire dal carattere finale dell'indirizzo e-mail. Questo è un esempio abbastanza semplice, adesso per un uso più complesso di Left e Right, supponiamo di avere un nome completo nella cella A1 in questo formato: Nelson Mandela, tuttavia, si supponga di voler avere il cognome prima del nome in questo modo: Mandela Nelson

È possibile utilizzare le funzioni Left, Right e InStr per raggiungere questo obiettivo. Creiamo una nuova routine e impostiamo quattro variabili, in questo modo:

```
Dim NomeC As String
Dim PNome As String
Dim SNome As String
Dim PosSpc As Integer
```

Posizionare il nome completo nella variabile NomeC: *NomeC = "Nelson Mandela"*. Ora usiamo la funzione InStr per individuare la posizione dello spazio nel nome: *PosSpc = InStr(NomeC, " ")*

Per ottenere solo il primo nome si può cominciare all'inizio del nome completo e andare fino alla posizione dello spazio (PosSpc) e togliere 1: *PNome = Left(NomeC, PosSpc - 1)*

Il motivo per cui è necessario detrarre 1 dalla variabile PosSpc è perché la funzione InStr restituisce la posizione dello spazio, un valore di 7 per il nostro nome, mentre invece il carattere finale del nome, è di 1 in meno di questo valore, infatti Nelson ha solo 6 caratteri.
Per ottenere il cognome, abbiamo bisogno di qualcosa di leggermente diverso. La posizione di partenza è la lunghezza del nome completo meno la lunghezza del nome, con questa operazione si otterrà il numero di caratteri partendo dalla destra del nome. Il codice è questo: *SNome = Right(NomeC, Len(NomeC) - Len(PNome))*. Così come il parametro finale di destra abbiamo questo codice: *Len(NomeC) - Len(PNome)*

Questo utilizza la funzione Len per ottenere la lunghezza delle variabili NomeC e PNome, infine, si visualizzano i risultati in una finestra di messaggio:*MsgBox (SNome & ", " & PNome)*

Ora abbiamo la variabile *SNome* e *PNome* separate dal simbolo di concatenazione (&), ma abbiamo anche bisogno di una virgola, e dobbiamo inserirla tra virgolette in modo che VBA lo vede come testo. Quindi stiamo dicendo, di inserire il cognome, poi una virgola, quindi il Nome. Inoltre si potrebbe anche aggiungere una funzione Trim per i nomi, per sbarazzarsi di qualsiasi spazio bianco. Come questo: *MsgBox (Trim(SNome) & ", " & Trim(PNome))*

Ma non è necessario per questo piccolo esempio, comunque si tenga presente anche questo aspetto nell'utilizzo futuro di queste funzioni. L'intero codice, quindi, dovrebbe essere simile a questo:

```
Sub Prova2()
Dim NomeC As String
Dim PNome As String
Dim SNome As String
Dim PosSpc As Integer

NomeC = "Nelson Mandela"
PosSpc = InStr(NomeC, " ")
```

Eseguendo il codice e si dovrebbe vedere questa finestra di messaggio:

Fig. 7

Il codice precedente funziona solo per i nomi che hanno due parti, se il nome è composto da 3 parti si deve usare un'altra funzione (Split)

La Funzione Mid

Questa funzione viene utilizzata per catturare i caratteri di una stringa di testo. Ha tre parti:

Mid(string_to_search, start_position, number_of_characters_to_grab)

La prima parte è la stringa che si desidera cercare e può essere un testo, una variabile oppure il testo diretto tra virgolette. La seconda parte è dove iniziare la ricerca della stringa e la parte finale è il numero di caratteri che si desidera catturare. Per dimostrare la funzione Mid, esaminare il seguente codice:

```
Dim Email As String
Dim Pcar As String
Email = "mia_emailprova.com"
Pcar = Mid(Email, 15, 4)
MsgBox Pcar
```

Abbiamo creato due variabili stringa, una chiamata Email e una chiamata Pcar, abbiamo poi memorizzato un indirizzo email nella variabile Email e poi viene il codice della funzione Mid: *Pcar = Mid(Email, 15, 4)*

Il testo che stiamo cercando è nella variabile Email variabile e vogliamo iniziare a catturare i caratteri dalla posizione 15 della stringa e Il numero di caratteri che vogliamo prendere è 4. Quando viene eseguito il programma, nella finestra di messaggio verrà visualizzato .com. La funzione Mid è molto utile in un Loop, in quanto permette di esaminare un carattere alla volta da una stringa di testo.

Esercizio con il Metodo String

Per ottenere una certa dimestichezza con i metodi String, vediamo un altro esercizio. Supponete di avere un codice prodotto su un foglio di calcolo che si presentava così: PD-23-23-45, tuttavia, è nel formato sbagliato. Per ottenere il codice giusto si devono rimuovere tutti i trattini, e poi si devono sostituire le lettere PD nel codice del prodotto con PDC, in modo che il codice si presenti simile a questo: PDC232345

La prima parte del problema è la rimozione dei trattini, abbastanza facile, basta usare Replace:

```
Dim Pcode As String
Pcode = "PD-23-2345"
Pcode = Replace(Pcode, "-", "")
```

Tra le parentesi tonde di Replace abbiamo il testo che vogliamo cercare, che è la variabile denominata Pcode, dopo abbiamo una virgola, e in seguito il carattere che vogliamo sostituire, il trattino, che viene sostituito con due doppie virgolette senza spazio tra loro.

La seconda parte del problema, si deve aggiungere la "C" dopo "PD", sarebbe facile, se VBA aveva una funzione di inserimento, ma non è così e questo rende il problema un po' più difficile. Ci sono alcuni modi per inserire la "C" nel posto giusto, noi lo faremo utilizzando le funzioni di stringa Left e Mid. Useremo Left per prendere i primi due caratteri, quindi aggiungeremo la "C", poi usiamo la funzione Mid per ottenere i numeri. Per ottenere i primi due caratteri e aggiungere la "C", il codice è questo:

```
Dim Ncar As String
Ncar = Left(Pcode, 2) & "C"
```

Partendo dalla sinistra della variabile Pcode, prendiamo due caratteri con il codice Left(Pcode, 2) e la lettera "C" viene aggiunta con la concatenazione in questo modo: Left(Pcode, 2) & "C". Il nuovo codice di tre lettere viene memorizzato nella variabile chiamata Ncar e per ottenere i numeri, utilizziamo Mid in questo modo:

```
Dim Num As String
Num = Mid(Pcode, 3)
```

La funzione Mid preleverà 3 caratteri dalla variabile Pcode e poiché non abbiamo specificato un numero finale, Mid prenderà il resto dei caratteri fino alla fine della stringa. L'unica cosa che resta da fare è quello di unire le due parti insieme:

```
Dim NewCar As String
NewCar = Ncar & Num
MsgBox NewCar
```

In questa parte si utilizza solo la concatenazione per unire la variabile Ncar e Num e l'ultima riga utilizza una finestra di messaggio per visualizzare i risultati. Tutto il codice, si presenta così:

```
Sub Prova3()
Dim Pcode As String
Dim Ncar As String
Dim Num As String
Dim NewCar As String
Pcode = "PD-23-2345"
Pcode = Replace(Pcode, "-", "")

Ncar = Left(Pcode, 2) & "C"
Num = Mid(Pcode, 3)
NewCar = Ncar & Num
MsgBox NewCar
End Sub
```

Quando si incontra un problema come quello sopra, la soluzione di solito è di utilizzare una delle funzioni Left, Right, o Mid (o tutti) di spezzare la stringa per poi unirli di nuovo insieme.