# Comandi di programmazione

**Etichetta ( Label)**: con questo termine si intende un numero intero seguito dal doppio punto, da inserire all'inizio di una riga, ad esempio:

```
1:
```

Scrivendo a monte di questa riga il comando:

```
goto 1
```

il programma salterà tutte le righe successive per fermarsi all'etichetta che abbiamo denominato `1`, da questo punto in poi inizierà a leggere lo script.

Oppure possiamo scrivere, sempre a monte dell'etichetta `1:`

```
gosub 1
```

In questo caso, oltre all'etichetta, occorrerà scrivere in fondo al testo che si vuole richiamare, la scritta `return:`

```
1:
...
return
```

In tal modo il programma, dopo aver letto il testo compreso tra `1:` e `return`, tornerà alla riga immediatamente sotto il comando iniziale `gosub 1`.

Utilizzando `gosub`, prima della riga con l'etichetta `1:`, dovrà poi essere inserito un ulteriore comando che eviti di attivare ancora il comando `return` relativo all'etichetta medesima, ad esempio inserendo un `end`, che chiude la lettura dei dati.

Esistono poi comandi di condizione, del tutto simili a quelli in uso nella logica:

```
If x=1 then y=2
```

Se il parametro x assume il valore 1 allora il parametro y assumerà il valore 2.

```
If v="ortogonale" then y=1 else y=2
```

Il parametro `v`, in questo caso, deve essere definito nei parametri come testo ed essere associato alla stringa "ortogonale", specificata nel *Testo Master* in seguito al comando `values "v"`. Se `v` corrisponde alla stringa specificata, allora y assumerà il valore 1, altrimenti assumerà il valore 2.

Il comando `not`, seguito da parentesi, esprime negazione logica del contenuto entro parentesi, mentre i comandi `or`, `and` corrispondono rispettivamente alle o, e logiche.

Esempio:

```
if  not( x=1 ) then y=2
```

quando il parametro `x` non assume il valore 1, allora `y` assumerà il valore 2

La scritta:

If x=1 and y=2 then z=3 else z=1

farà assumere alla variabile z il valore 3 soltanto se risultano valide entrambe le condizioni x=1, y=2, altrimenti z assumerà il valore 1.

La scritta:

If x=1.5 or y=2.3 then z=3

farà assumere alla variabile z il valore 3 quando risulta valida almeno una delle condizioni x=1.5, y=2.3

In realtà utilizzare in modo diretto, per i parametri, le forme qui presentate negli esempi, può far apparire una scritta di errore, cliccando su *controlla script*, anche se l'oggetto funzionerà comunque:

*Usare un tipo reale può causare problemi di precisione*

Per aggirare questo problema , si potrebbe scrivere:

If abs(sgn(x-1.5))=0 or abs(sgn(y-2.3))=0 then z=3

Infatti abs(sgn(x)), per qualsiasi numero reale x, può fornire soltanto 1 (se x diverso da 0) o 0 (se x nullo).

Possono essere elencate diverse proposizioni collegate da end o da or, ma con soltanto un if iniziale e un then finale. Se il comando if con la condizione specificata è seguito da goto o gosub, si può omettere then prima di questi: If x=1.5 then goto 1 equivale a: If x=1.5 goto 1 ovvero a: If x=1.5 then 1

Un altro comando fondamentale è quello della concatenazione logica o **loop**, che permette di iterare un procedimento o una forma.

```
FOR i = 1 TO n
…
NEXT i
```

La parte di script compresa tra FOR e NEXT viene eseguita n volte, mentre i viene trattato come una variabile che assume i valori rispettivamente 1,…n ad ogni ciclo. Dato che i è essa stessa una variabile, essa può essere utilizzata per definire funzioni all'interno del loop medesimo. Al di fuori del loop la variabile interna i assume il valore n+ 1, che può essere assunto entro la funzione specificata da una nuova variabile.

Le concatenazioni possono essere nidificate, ovvero possono contenerne altre al loro interno come scatole cinesi:

```
FOR k = 1 TO n3
…
FOR j = 1 TO n2
…
FOR i = 1 TO n1
…
NEXT i
…
NEXT j
…
NEXT k
```