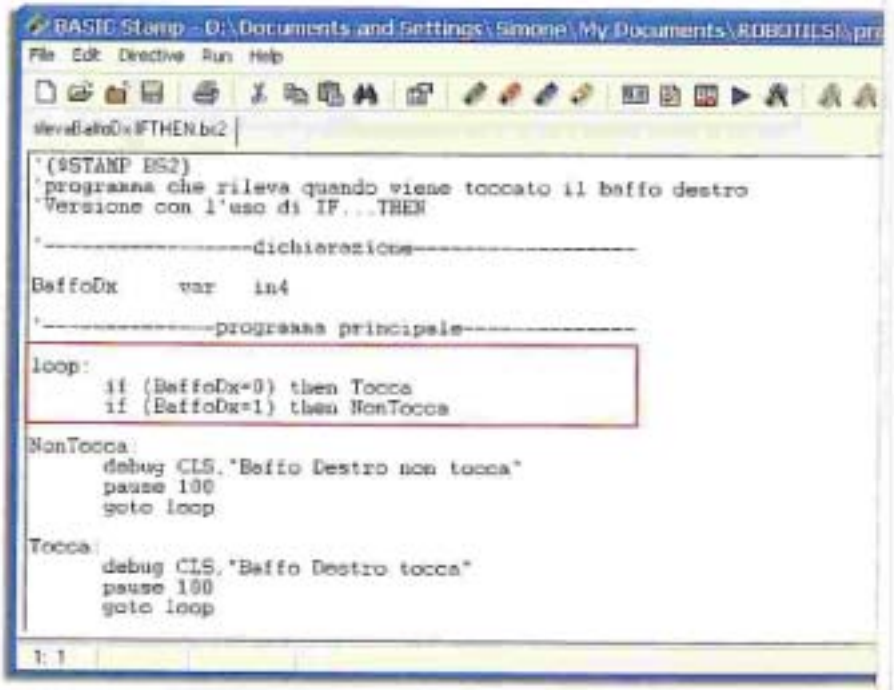
La scatola portacomponenti

La scatola portacomponenti allegata a questo fascicolo ti permetterà di conservare il materiale relativo al montaggio del tuo robot. In questo modo potrai mettere in ordine gli attrezzi (cacciaviti, pinzetta ecc.), i componenti elettrici e quelli elettronici. Ricorda che questi ultimi, ad esempio i driver dei motori, necessitano di particolare attenzione: la scatola, infatti, non è isolata e, dunque, è importante riporli prima sulla spugna conduttiva di protezione, sia perché non accumulino carica elettrica sia per proteggere i contatti dei chip. Con il prossimo fascicolo, inoltre, riceverai la breadboard (la piattaforma mobile di sperimentazione) e, di seguito, diversi componenti elettrici (cavetti, resistori, LED ecc.) riutilizzabili in più esperimenti: la scatola portacomponenti ti permetterà di averli sempre a disposizione, ma anche di proteggerli da urti e polvere. Nella scatola, infine, troverà posto ogni tipo di minutaglia (cilindri plastici, viti, dadi ecc.).

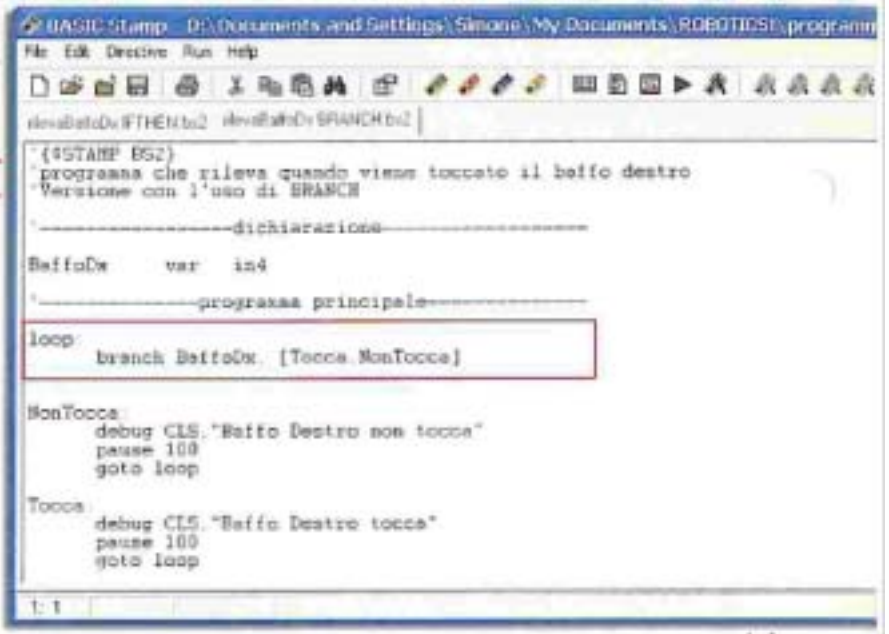
Le fasi di programmazione

Riprendiamo il discorso sull'istruzione di test if... then, per vedere come, in alcuni casi, può essere utile sostituirla con un'altra istruzione di test, detta branch.

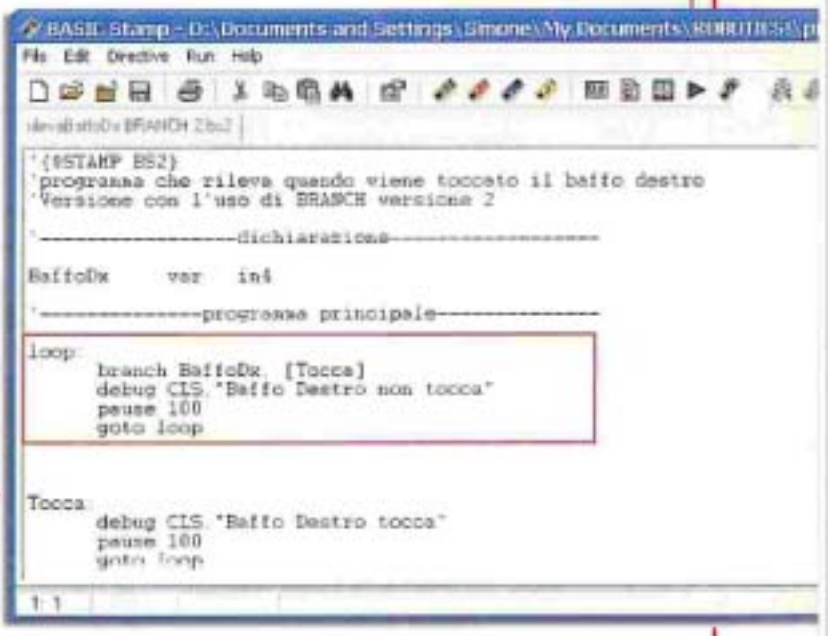
Digita nell'area di editing il listato (a destra), collega il robot al computer, accendilo e premi Run. Tramite il test (if... then) della variabile BaffoDx, di dimensione 1 bit, il programma controlla lo stato (0 oppure 1) del baffo destro del robot. Tramite le istruzioni di debug, stamperà a video Baffo Destro tocca (o non tocca). Il comando CLS, in particolare, ha come effetto quello di cancellare (*CLear*) la schermata (*Screen*) di debug e riposizionare il cursore in alto a sinistra. A ogni visualizzazione, inoltre, segue una pausa di 100 ms (pause 100) che evita l'eccessivo sfarfallio della scritta visualizzata (dovuto al fatto che, a ogni chiamata di un'istruzione debug contenente CLS, tale scritta viene cancellata e riscritta).



Consideriamo ora il listato a destra: digitalo nell'area di editing e mandalo in esecuzione (con il pulsante Run). Noterai che la finestra di debug (sotto), visualizzerà lo stato del baffo destro, esattamente come con il programma precedente. In questo listato, però, la coppia di istruzioni if... then è stata sostituita dall'istruzione branch seguita dal *NomeVariabile* (BaffoDx) e dalla *Lista delle Etichette* (in questo caso due), tra parentesi quadre ([ ]), separate dalla virgola. Letteralmente *branch* significa 'diramazione': infatti, in base al valore assunto dalla variabile su cui esegue il test, questa istruzione permette di scegliere un determinato 'ramo' di codice (NonTocca o Tocca). La lista di etichette che compare tra parentesi quadre è associata, in ordine crescente, ai possibili valori della variabile di branch che, in questo caso, sono solo due (0 e 1). La prima etichetta, dunque, è associata al valore 0: se la variabile BaffoDx è (=) 0, allora il programma salta al 'ramo' con quell'etichetta (Tocca). La seconda etichetta, invece è associata al valore 1: se dunque BaffoDx=1, il programma salta all'etichetta NonTocca.



Consideriamo ora una variazione del precedente programma (sotto). In questo caso la lista di etichette è stata ridotta a una sola (Tocca), associata al valore 0. Poiché ogni caso escluso dalla lista rimanda il programma alla riga che segue il branch, si ottiene lo stesso risultato del listato precedente: il caso BaffoDx=1, infatti, è compreso dall'istruzione debug CLS, "Baffo Destro non tocca".



L'istruzione di test branch, dunque, può sostituire if... then, non sempre, però, e solo in condizioni ben precise.

Per capire l'utilità dell'istruzione branch, ipotizziamo di avere una variabile di tipo nibble, che può cioè assumere 16 valori (compresi tra 0 e 15). Per testare e gestire i corrispondenti 16 casi con l'istruzione if... then, bisognerebbe scrivere una sequenza di 16 test; con branch, invece, basta una sola istruzione con l'intera lista di etichette tra parentesi (da un minimo di una a un massimo di 16). Con una scrittura molto più compatta ed elegante, dunque, una sola istruzione branch sostituisce un'intera sequenza di if... then, a patto però che la condizione di test sia un'uguaglianza (=).

Quando le condizioni di scelta sono invece legate a un set di valori, risulta più comodo usare l'istruzione if... then: se volessimo ad esempio distinguere, di una variabile *x* di tipo nibble, tra il set di valori maggiori e quelli minori di 10, l'istruzione if (x<10) then *NomeEtichetta* è molto più comoda di un branch contenente *NomeEtichetta* per 10 volte di seguito. L'uso di if... then, invece, diventa necessario in caso di condizioni su più variabili legate da operatori logici: if ((z<10) AND (y>2)) then *NomeEtichetta*, per esempio, non può essere realizzata tramite un'istruzione branch.