



## 5. Le regole di sintassi

Le regole per scrivere e comunicare le istruzioni con Algobuild seguono le indicazioni riportate nella tabella seguente.

TIPI DI DATI	Ogni espressione e ogni variabile ha associato un <i>tipo</i> .
INT	<p><b>Sono numeri scritti come sequenza di cifre senza il punto oppure espressioni algebriche contenenti SOLO int e variabili di tipo int.</b></p> <p>ESEMPLI: 3, <math>10*2+4</math>, <math>3*k</math> (se k è variabile int)</p> <ul style="list-style-type: none"><li>• numero = 53 (numero viene definita come variabile di tipo intero)</li><li>• numeri interi in basi diverse</li><li>• numero = 12 (valore in base 10) numero = 012</li><li>• valore ottale - base 8 = decimale 8</li><li>• numero = 0x10 (valore esadecimale = decimale 16)</li><li>• numero = 0b10 (valore binario = decimale 2)</li></ul>
STRING	<p><b>Sono sequenze di caratteri racchiuse tra doppi apici.</b></p> <p>ESEMPLI: "ciao", "tanti quanti", "torno\nsubito" (il simbolo \n inserisce un "a capo")</p> <p>nome = "Mondo"</p> <p>messaggio = "Ciao" + nome + "!"</p>
BOOLEAN	<p><b>Possono avere solo due valori true e false.</b></p> <ul style="list-style-type: none"><li>• condizione = true</li><li>• test = <math>x &gt; y</math></li></ul>
NOMI DI VARIABILI	<ul style="list-style-type: none"><li>• Possono essere scritte in minuscolo, MAIUSCOLO o modoMisto.</li><li>• Devono iniziare per carattere alfabetico, possono contenere cifre o trattini di sottolineatura.</li><li>• NON possono contenere spazi o caratteri speciali.</li><li>• <b>CONSIGLIO:</b> usare nomi tutti minuscoli o "a cammello" con iniziale minuscola.</li></ul> <p>ESEMPLI: numero, prezzo, nElementi</p>
ASSEGNAMENTO	<p><b><i>variabile=espressione oppure vettore[indice]=espressione</i></b></p> <p>ESEMPLI: <math>x=7</math> oppure <math>x=x+1</math> oppure <i>giorni[mese]=31</i></p> <p>In un blocco di assegnamento si possono inserire più istruzioni, una per riga. Il tipo della variabile dipende dall'espressione a destra del simbolo di assegnamento uguale.</p> <p><b>NOTA:</b> A una variabile esistente non può essere assegnata un'espressione di tipo diverso.</p> <p>Unica eccezione double che accetta espressioni di tipo int.</p>
INPUT	<p><b>Legge un valore da tastiera. Se non specificato i valori sono di tipo double.</b></p> <p>ESEMPLI:</p> <p>lunghezza (legge un numero double e lo inserisce nella variabile lunghezza)</p> <p>valori[0] (legge un numero double e lo inserisce alla prima posizione del vettore valori).</p> <p>Per leggere un input di un tipo specifico si deve premetterne il nome.</p> <p>int nElementi (legge un numero intero e lo inserisce nella variabile nElementi)</p> <p>string nome (legge una sequenza di caratteri e la inserisce nella variabile nome)</p> <p>In un blocco di input si possono inserire più istruzioni di lettura, una per riga.</p>



OUTPUT	<p><b>Visualizza il valore di una variabile o di un'espressione nel riquadro Output (console).</b></p> <p>ESEMPI: nElementi nElementi*4 "ciao" + nome</p> <p>In un blocco di output si possono inserire più istruzioni di visualizzazione, una per riga.</p>
IF	<p><b>Valuta un'espressione booleana e prosegue sul ramo true (T) se la condizione è vera, sul ramo false (F) altrimenti.</b></p> <p>ESEMPI: <math>y &lt; 5</math> <math>x == z * 2</math> <math>a[i] &lt; a[i+1]</math></p>
WHILE	<p><b>Ciclo precondizionale</b></p> <p>Valuta un'espressione booleana ed esegue un gruppo di istruzioni finché la condizione è vera. Espressione booleana come nell'istruzione if.</p>
DO-WHILE	<p><b>Ciclo postcondizionale</b></p> <p>Esegue un gruppo di istruzioni e alla fine valuta un'espressione booleana. Ripete finché la condizione è vera.</p> <p>Espressione booleana come nell'istruzione if.</p>
FOR	<p><b>È un ciclo usato come contatore.</b></p> <p>Si usa una variabile in tre parti della stessa istruzione: inizializzazione, condizione, aggiornamento.</p> <p>La parte "inizializzazione" è un assegnamento come: <math>i = 0</math>.</p> <p>La parte "condizione" è un'espressione booleana come: <math>i \leq 10</math>.</p> <p>La parte "aggiornamento" è un assegnamento per calcolare il valore successivo come: <math>i = i + 1</math>.</p>
ESPRESSIONI	<p><b>Operatori aritmetici</b></p> <ul style="list-style-type: none"><li>• + addizione definita per i tipi int, double e string</li><li>• - sottrazione definita per i tipi int e double</li><li>• * moltiplicazione definita per i tipi int e double</li><li>• / divisione definita per i tipi int e double</li><li>• % modulo (resto della divisione tra interi) definita solo per i tipi int. Esempio <math>12 \% 7</math> restituisce 5</li><li>• ^ potenza (es. <math>x^2</math>) definita solo per i tipi double</li><li>• &amp; (And aritmetico bit a bit - solo tra interi) definita solo per i tipi int. Esempio <math>7 \&amp; 12</math> restituisce 4</li><li>•   (Or aritmetico bit a bit - solo tra interi) definita solo per i tipi int. Esempio <math>7   12</math> restituisce 15</li><li>• &gt;&gt; (shift bit a destra) definita solo per i tipi int. Esempio: <math>16 &gt;&gt; 2</math> restituisce 4</li><li>• &lt;&lt; (shift bit a sinistra) definita solo per i tipi int. Esempio: <math>16 &lt;&lt; 2</math> restituisce 64</li></ul> <p><b>Operatori logici</b></p> <ul style="list-style-type: none"><li>• == (uguaglianza) definita per i tipi int, double e string restituisce boolean</li><li>• != (disuguaglianza) definita per i tipi int, double e string restituisce boolean</li><li>• &gt; (maggiore di) definita per i tipi int, double e string restituisce boolean</li><li>• &lt; (minore di) definita per i tipi int, double e string restituisce boolean</li><li>• &gt;= (maggiore o uguale a) definita per i tipi int, double e string restituisce boolean</li><li>• &lt;= (minore o uguale a) definita per i tipi int, double e string restituisce boolean</li><li>• &amp;&amp; (And logico) definita per i tipi boolean restituisce boolean</li><li>•    (Or logico) definita per i tipi boolean restituisce boolean</li><li>• ! (negazione logica) definita per i tipi boolean restituisce boolean</li></ul>