



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>Sono numeri scritti come sequenza di cifre senza il punto oppure espressioni algebriche contenenti SOLO int e variabili di tipo int.</p> <p>ESEMPLI: 3, $10*2+4$, $3*k$ (se k è variabile int)</p> <ul style="list-style-type: none">• numero = 53 (numero viene definita come variabile di tipo intero)• numeri interi in basi diverse• numero = 12 (valore in base 10) numero = 012• valore ottale - base 8 = decimale 8• numero = 0x10 (valore esadecimale = decimale 16)• numero = 0b10 (valore binario = decimale 2)
STRING	<p>Sono sequenze di caratteri racchiuse tra doppi apici.</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>Possono avere solo due valori true e false.</p> <ul style="list-style-type: none">• condizione = true• test = $x > y$
NOMI DI VARIABILI	<ul style="list-style-type: none">• Possono essere scritte in minuscolo, MAIUSCOLO o modoMisto.• Devono iniziare per carattere alfabetico, possono contenere cifre o trattini di sottolineatura.• NON possono contenere spazi o caratteri speciali.• CONSIGLIO: usare nomi tutti minuscoli o "a cammello" con iniziale minuscola. <p>ESEMPLI: numero, prezzo, nElementi</p>
ASSEGNAMENTO	<p><i>variabile=espressione oppure vettore[indice]=espressione</i></p> <p>ESEMPLI: $x=7$ oppure $x=x+1$ 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>NOTA: 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>Legge un valore da tastiera. Se non specificato i valori sono di tipo double.</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>Visualizza il valore di una variabile o di un'espressione nel riquadro Output (console).</p> <p>ESEMPLI: 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>Valuta un'espressione booleana e prosegue sul ramo true (T) se la condizione è vera, sul ramo false (F) altrimenti.</p> <p>ESEMPLI: $y < 5$ $x == z * 2$ $a[i] < a[i+1]$</p>
WHILE	<p>Ciclo precondizionale</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>Ciclo postcondizionale</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>È un ciclo usato come contatore.</p> <p>Si usa una variabile in tre parti della stessa istruzione: inizializzazione, condizione, aggiornamento.</p> <p>La parte "inizializzazione" è un assegnamento come: $i = 0$.</p> <p>La parte "condizione" è un'espressione booleana come: $i \leq 10$.</p> <p>La parte "aggiornamento" è un assegnamento per calcolare il valore successivo come: $i = i + 1$.</p>
ESPRESSIONI	<p>Operatori aritmetici</p> <ul style="list-style-type: none">• + addizione definita per i tipi int, double e string• - sottrazione definita per i tipi int e double• * moltiplicazione definita per i tipi int e double• / divisione definita per i tipi int e double• % modulo (resto della divisione tra interi) definita solo per i tipi int. Esempio $12 \% 7$ restituisce 5• ^ potenza (es. x^2) definita solo per i tipi double• & (And aritmetico bit a bit - solo tra interi) definita solo per i tipi int. Esempio $7 \& 12$ restituisce 4• (Or aritmetico bit a bit - solo tra interi) definita solo per i tipi int. Esempio $7 12$ restituisce 15• >> (shift bit a destra) definita solo per i tipi int. Esempio: $16 >> 2$ restituisce 4• << (shift bit a sinistra) definita solo per i tipi int. Esempio: $16 << 2$ restituisce 64 <p>Operatori logici</p> <ul style="list-style-type: none">• == (uguaglianza) definita per i tipi int, double e string restituisce boolean• != (disuguaglianza) definita per i tipi int, double e string restituisce boolean• > (maggiore di) definita per i tipi int, double e string restituisce boolean• < (minore di) definita per i tipi int, double e string restituisce boolean• >= (maggiore o uguale a) definita per i tipi int, double e string restituisce boolean• <= (minore o uguale a) definita per i tipi int, double e string restituisce boolean• && (And logico) definita per i tipi boolean restituisce boolean• (Or logico) definita per i tipi boolean restituisce boolean• ! (negazione logica) definita per i tipi boolean restituisce boolean