

ANSI C – Variabili e Costanti (Parte 1)

Bentornati, nei tutorial precedenti abbiamo imparato i primi concetti di base sulla programmazione in ANSI C, abbiamo visto come nasce il C, i suoi pregi e i suoi difetti, abbiamo visto dove è possibile scrivere il codice sorgente e come questo venga preso e trasformato per essere poi eseguito dal calcolatore. Abbiamo visto poi come si scrive un programma vero e proprio, anche se molto molto semplice, accennando alle direttive al preprocessore e alle funzioni. Abbiamo visto come si commenta il codice sorgente e come si può racchiuderlo all'interno di un blocco attraverso le parentesi graffe. Insomma, abbiamo messo parecchia carne sulla brace della programmazione, ma questo non è che l'inizio, ora andremo a toccare uno degli argomenti fondamentali della programmazione, ovvero le variabili.

Le Variabili

Che cosa sono le variabili? Il concetto di variabile è molto semplice per chi ha una minima conoscenza di programmazione, ma vi assicuro che anche se siete totalmente digiuni in merito, avrete visto e usato più di una volta le variabili. Prendo in prestito alcuni esempi fatti sul libro “Become An Xcoder”, un libro del quale ho partecipato insieme allo staff di xcodeitalia alla traduzione e del quale parleremo sicuramente nei capitoli futuri.

Quando eravamo alle scuole elementari, per insegnarci a fare le operazioni matematiche utilizzavano un metodo semplice, ci chiedevano di scrivere il risultato ottenuto sui puntini:

$$2 + 6 = \dots$$

$\dots = 3 * 4$ (l'asterisco * è il modo standard del computer di rappresentare le moltiplicazioni)

Andiamo avanti negli anni e arriviamo alle scuole medie, i puntini oramai usciti fuori moda sono stati sostituiti dalle lettere e è stata introdotta una nuova parola per definire questa matematica, si è iniziato a parlare di “algebra”.

$$2 + 6 = x$$

$$y = 3 * 4$$

Come potete vedere c'è stato un semplicissimo cambio di notazione, dai puntini si è passati alle lettere, ebbene queste lettere sono state chiamate variabili, appunto perchè il loro valore poteva cambiare, ovvero essere variabile. Che cosa sono le variabili dunque? Semplicemente dei contenitori di valori, caratterizzati da un nome (ad esempio x, y, z, pippo, indice), il nome deve essere **univoco**, in quanto vogliamo sempre sapere in ogni momento a quale variabile ci stiamo riferendo.

Perchè le variabili? Possiamo utilizzare le variabili in qualsiasi parte del programma al posto dei loro valori, sfruttando l'utilissima corrispondenza nome-valore, questo ci aiuta ad esempio a gestire cambiamenti di valore, oppure quando si tratta di effettuare operazioni ripetitive con un minimo sforzo.

La variabili possono contenere un qualsiasi tipo di valore, sia esso un numero, un carattere o una stringa di caratteri. Quando si definisce una variabile è importante specificare oltre al nome, anche il tipo di dati che questa può contenere.

Per convenzione è bene dare alle variabili dei nomi che si commentano da soli, se per

esempio vogliamo creare una variabile per contenere l'area di un quadrato, la possiamo chiamare semplicemente `areaQuadrato`, un nome (chiamato identificatore) può essere composto da una o più caratteri o cifre e deve necessariamente iniziare con una lettera o il simbolo di underscore “_”.

N.B: Quando si definisce un identificatore per una variabile, bisogna tenere conto che l'ANSI C è “case-sensitive”, ovvero c'è differenza tra lettere maiuscole e minuscole, ad esempio la variabile `areaQuadrato` è diversa dalla variabile `areaQUADRATO`, quindi fare sempre attenzione ai nomi delle variabili. Per convenzione i nomi delle variabili sono formati da parole composte (ad esempio “area” + “quadrato”) e la prima parola inizia con la lettera minuscola, mentre tutte le altre con la lettera maiuscola (ad esempio “sommaTreNumeri”).

Abbiamo detto prima che oltre all'identificatore, quando si dichiara una variabile, è necessario indicarne anche il tipo, questo è molto importante perchè andiamo a definire qual'è il contenuto della variabile stessa, se ad esempio definiamo una variabile di tipo intero, questa conterrà al suo interno un numero intero senza cifre decimali (ad esempio `int numero = 5`), mentre se la definiamo di tipo carattere, questa conterrà al suo interno una rappresentazione in codice ASCII del carattere inserito (ad esempio il carattere 5 sarà diverso dall'intero 5, in quanto l'intero 5 ha una rappresentazione binaria seguente “00000101”, mentre il carattere 5 avrà il codice ASCII 53, quindi la sua rappresentazione binaria è la seguente “00110101” entrambi occupano uno spazio in byte in memoria, ma come possiamo vedere sono completamente differenti. Potete trovare una tabella di codici ASCII relativi ai caratteri nell'appendice A messa a complemento dei tutorial).

Come abbiamo appena detto nella parentesi precedente, ogni variabile occupa uno spazio in memoria, ovviamente a differente tipo, corrisponde un differente impiego di spazio, nei prossimi tutorial spiegheremo come fare a gestire la memoria, per ora vi basti sapere che in base al tipo, per rappresentare un dato, ho bisogno di uno spazio in memoria.

Tutte le variabili che decido di utilizzare nel mio programma devono essere prima di tutto dichiarate e successivamente inizializzate, che cosa significa dichiarare una variabile? Dichiarare un variabile significa, appunto, dire al compilatore che quell'identificatore si riferisce a una variabile di un certo tipo:

```
int primoValore;  
char primoCarattere;  
double numeroRealeADoppiaPrecisione;
```

Inizializzare una variabile significa assegnare alla variabile appena creata una valore iniziale, si può anche dichiarare e inizializzare una variabile nella stessa istruzione:

```
primoValore = 0;  
int secondoValore = 1;
```

A questo punto mi aspetto una domanda, perchè dobbiamo inizializzare una variabile? Innanzitutto non è obbligatorio inizializzare una variabile, ma caldamente consigliato, in quanto quando andiamo a definire una variabile, non facciamo che assegnare una certa memoria a questa variabile, ovviamente la memoria che gli viene assegnata e memoria libera, ma al suo interno potrebbe contenere dei dati sporchi che sono stati scritti da altri programmi prima di rilasciarla, quindi inizializzare una variabile serve per pulire la zona di memoria che stiamo andando a utilizzare, in questo modo non corriamo il rischio di avere dei valori distorti da quelli che ci aspettiamo.

Nel prossimo tutorial andremo a vedere nel dettaglio tutti i tipi di dati primitivi e andremo a mostrare degli esempi di funzionamento, vedrete che dichiarare, inizializzare e utilizzare

le variabili è davvero molto semplice. Inoltre andremo anche a vedere come si definiscono variabili costanti, ovvero che non variano nel nostro programma.

Manuel (aka Jekrom)

per segnalazioni : manuel.montini@gmail.com

<http://iphonecodes.wordpress.com/>

LICENZA:

Questo tutorial è rilasciato sotto licenza Creative Commons : “Attribuzione-Non commerciale-Non opere derivate 2.5 Italia”,

<http://creativecommons.org/licenses/by-nc-nd/2.5/it/>