

9° incontro - Finalità Corso - Sul linguaggio-macchina



Cari ragazzi, devo dire che dopo due settimane vi trovo cresciuti. Moltiplicando 14, i giorni trascorsi, per il numero di persone in quest' aula, si ha un' idea di quanta saggezza in più si può respirare. Approfittiamone per iniziare il nostro Corso di che? di programmazione? Non direi proprio e vi spiego perché.

Che ne direste voi se la vostra Scuola vi insegnasse **solamente** le regole d'uso della Lingua italiana? Alla fine, avremmo tante persone che parlano correttamente l' Italiano ; sì, ma **di che cosa parlerebbero?** Comunicerebbero tutte cose riguardanti la loro vita quotidiana, vuote di ogni traccia di **cultura**, delle esperienze di chi ha vissuto prima di loro, naturale linea di partenza per tagliare nuovi traguardi. In altre parole, se tutti fossero in questa condizione non esisterebbe progresso.

L'insufficienza di quella condizione sarebbe analoga a quella di una società nella quale tutti fossero esperti muratori, ma nella quale mancassero ingegneri e architetti. Si costruirebbero tutte case a un solo piano, magari anche bruttine, non si farebbero ponti, monumenti, acquedotti e così via. È chiaro che, in una società bene organizzata, sono necessari sia i muratori che gli ingegneri e gli architetti.

Abbiamo parlato del problema del settore delle costruzioni, ma è evidente che un discorso analogo potrebbe essere fatto per ogni altro tipo di attività.



Sì, Carlo, anche se ti prego di abituarti a chiamarla **Elaborazione automatica**. Vedi, nell'attributo **automatica** è il concetto di "moto proprio", cioè, in qualche modo, esso riconosce all'Elaborazione un certo grado di autonomia, il che corrisponde alla realtà ed ha un importante significato. Ma torniamo a noi.

Ora, un semplice Corso di Programmazione, che mira alla conoscenza di un linguaggio di programmazione, ti insegna a parlare, ad esempio, in Visual Basic, ma **per dire cosa?** Ecco che cadremmo in un caso analogo a quello della Scuola che insegnasse solamente la Lingua italiana. Ebbene, nel settore dell'Elaborazione Automatica, questo è quanto si fa (nei casi più fortunati) nella Scuola italiana.



Siamo di fronte a un vero dramma. **In una Nazione che possiede le potenzialità umane per avere i migliori Progettisti di software del mondo** (ingegneri e architetti del software), abbiamo sparuti gruppi di muratori del software, in un mondo ove vale e varrà sempre di più l'equazione:

software = potere

Riusciremo ad uscire da questo buio tunnel? Io spero di sì, non appena la maggioranza degli Italiani capirà che è da stupidi votare sperando nella possibilità di evadere il fisco e che **il bene dello Stato è il bene di tutti noi**.

Comunque, in attesa di tempi migliori, noi impegniamoci a fare la nostra parte e a tentare di porre riparo, per quanto possiamo, ai danni procurati dalle incapacità e dalle avidità dei nostri tristi governanti.



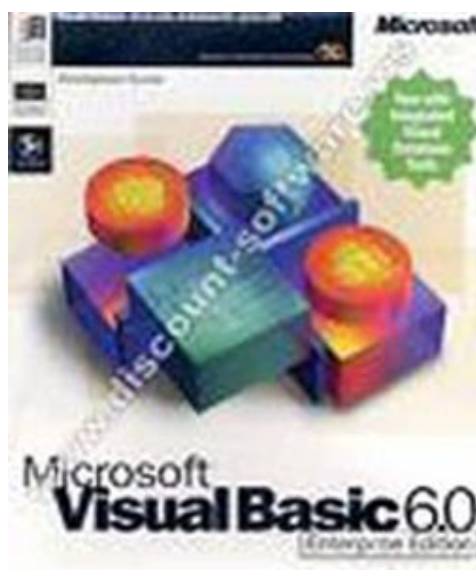
Forse la denominazione più adatta potrebbe essere:

Introduzione alla progettazione del software

Prima di iniziare, però, è in caso che chiarisca un altro aspetto.

All'inizio della mia carriera, quindi molti anni fa, fui obbligato a seguire due corsi di Programmazione, che trovai abbastanza noiosi ; uno di Assembler per l'elaboratore IBM 370 (che occupava una enorme sala ed aveva potenzialità inferiori a questo mio portatile) e l'altro di PL/1 (Programming Language).

Tutti gli altri linguaggi (almeno una decina) che ho usato li ho imparati da solo, con il seguente criterio: imparavo le diverse istruzioni man mano che mi servivano ; con questo criterio sono sempre riuscito ad ottenere dalle macchine tutto quello che mi serviva, senza annoiarmi, anzi divertendomi.



Ad esempio, del **Visual Basic** (1991), che ormai uso da almeno una dozzina d'anni, non conosco l'intero set di istruzioni. Ora, se io fossi costretto a tenere un corso di Programmazione, avrei il dovere di conoscere tutto del linguaggio e sarei condannato ad annoiare a morte me e voi. Ma noi non abbiamo la vocazione dei muratori del software, noi ambiamo a divenire architetti del software e, come tali, dobbiamo conoscere il lavoro del muratore, ma non nei dettagli.



Sì, ma per un paio di semplici motivi: primo, perché ci metterei più tempo a spiegare a un programmatore ciò che voglio che a scrivere un po' di istruzioni in Visual Basic ; secondo, forse più importante, perché il denaro che dovrei dare a lui, lo spendo volentieri per andare qualche volta in più a un concerto, a un teatro o, se permettete, in pizzeria. L'importante è, però, avere chiara la distinzione dei ruoli. Ma, a dirla tutta, devo confessarvi una cosa, che neppure Marco sa. Vanni non vive da solo ; a casa sua c'è una folla di persone, Vanni Progettista, Vanni Analista di Sistemi, Vanni Analista di Procedure, Vanni Programmatore, Vanni Operatore. Una gran confusione, ma c'è un grande vantaggio: se accade qualcosa di storto, ognuno può dare la colpa a qualcun'altro.



Adesso guardate un po' lo schermo. Cosa vuol fare quella bella signora? Secondo me è evidente: vuol comunicarci che lei sta pensando al **numero 5**. Sapendo di non poter essere ascoltata e non avendo a portata di mano carta e penna, usa il mezzo più comodo che ha a disposizione, **le mani**. E se avesse pensato al numero **15**, cosa avrebbe fatto? La signora è sveglia; avrebbe indicato prima il numero 10 con le mani aperte, quindi il segno + con i due indici incrociati e, infine, il numero 5 come aveva fatto prima. E se avesse pensato al numero **115** ? Probabilmente, tenendo le mani aperte, avrebbe indicato 100 spostandole per dieci volte in avanti, poi il segno + e quindi avrebbe indicato il 15 come prima.

La serie di segnali inviati dalla signora, corrisponde a un messaggio che, per iscritto sarebbe stato: $10 \cdot 10 + 10 + 5$, che equivale a: $100 + 10 + 5$ o anche a:

$$1 \cdot 10^2 + 1 \cdot 10^1 + 5 \cdot 10^0$$

che è la corretta **scrittura polinomiale di un numero naturale**, che si legge: “*uno per dieci alla seconda più uno per dieci alla prima più cinque per dieci alla zero*” (qualunque numero elevato all'esponente 0 dà come risultato il numero 1). In altri termini, abbiamo scritto in **forma estesa** un numero secondo il **sistema di numerazione decimale, cioè a base 10**, il sistema di numerazione usato ai nostri giorni. Sappiamo bene che lo stesso numero si scrive in **forma abbreviata: 115**, ove i simboli numerici assumono un valore posizionale: l'ultimo indica il numero di unità, il penultimo il numero di decine e il terzultimo il numero di centinaia.

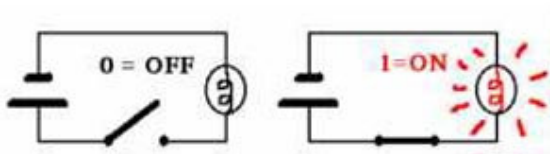
In tale sistema di numerazione, si usano 10 simboli numerici (cifre):

0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 .

Il sistema di numerazione decimale adottato risulta di uso particolarmente semplice, se paragonato a quelli adottati nel corso della storia dell'uomo (Babilonese, Egiziano, Romano), ma la sua **base (10)** è stata scelta fra altre possibili (ad es.: 8 > sistema ottale oppure 16 > sistema esadecimale). Sono sicuro che i vostri insegnanti vi hanno detto per quale motivo è riuscito naturale scegliere, fra tutte le possibili, la base 10.



Esattamente, per lo stesso motivo per il quale lo ha adottato la nostra signora sullo schermo. Ora pensiamo al caso dei semplicissimi "esseri viventi" che stiamo imparando a creare noi. Ho azzardato a chiamarli "esseri viventi", in quanto anche i nostri Elaboratori automatici hanno un loro, seppure ancora estremamente limitato, grado di autonomia, capacità di decisione, piccolo, piccolo, piccolo proprio "**libero arbitrio**".



Il nostro piccolo eroe, il nostro bebè, contrariamente alla fortunata signora, non ha dieci dita, ma ha **un solo dito**, un circuito elettrico, e con questo deve

comunicare. Il dito abbassato (circuito aperto, OFF, nessun passaggio di corrente) indica il numero **0**; il dito alzato (circuito chiuso, ON, passaggio di corrente) indica il numero **1**. In questa situazione, al nostro bebè riesce naturale il **sistema di numerazione binario, cioè a base 2**, nel quale sono usati due simboli numerici (cifre binarie): **0** e **1**.

Vediamo ora come un numero (ad esempio, **13**, espresso in decimale abbreviato) viene espresso, nelle forme abbreviata ed estesa, nei due sistemi di numerazione:

Sistema decimale - $13 = 1 \cdot 10^1 + 3 \cdot 10^0$

Sistema binario - $1101 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$

Naturalmente, nell'ambito di ognuno dei due sistemi, sono definite le regole per eseguire le comuni **operazioni aritmetiche** tra due numeri e per risolvere **espressioni algebriche** che coinvolgano un insieme di numeri, regole delle quali non occorre parlare qui; i vostri insegnanti di matematica lo faranno nel migliore dei modi. Qui ci limiteremo a qualche definizione di **termini comunemente usati** nella Elaborazione automatica.

Tutte le informazioni sono rappresentate, in un elaboratore, da un insieme enorme di circuiti, ognuno dei quali, con il suo stato OFF o ON, dà il suo contributo. Ebbene, la quantità di informazione minima, elementare, cioè non divisibile, è quella che può dare un circuito singolo, esprimibile con una cifra (0 o 1), detta **bit** (da **Binary digiT**, cifra binaria). Il **bit** è assunto come **unità elementare di informazione**.

In un elaboratore ogni tipo di informazione, di natura non solamente numerica, troverà una sua opportuna codifica, dando luogo ad un **linguaggio-macchina**, che ha un **alfabeto** costituito da due (binario) simboli (caratteri): **0** e **1**.



Sì, ci stavo proprio arrivando. Vedete, in genere, in un hardware il complesso dei circuiti è assemblato in modo che non si possa accedere al singolo circuito, che cioè non si possa leggere o scrivere un singolo bit di informazione. In altri termini, **il singolo bit non è indirizzabile**.

È indirizzabile una sequenza di **8 bit**, cioè devo leggere o scrivere 8 bit alla volta. Vi chiederete come mai proprio 8, una scelta certamente non necessaria, tanto che in alcuni hardware particolari si sceglie il 16. La scelta dell'**8** risulta la più conveniente per i seguenti motivi:

1. la lettura/scrittura di 8 bit alla volta rende l'operazione 8 volte più rapida rispetto all'operazione sequenziale di 1 bit alla volta ;
2. per il fatto che i dati sono espressi in una codifica binaria, tale numero deve essere una potenza di 2 ;
3. tenuto conto che si devono rappresentare un certo numero di simboli: alfabetici (in maiuscolo e in minuscolo), numerici, segni di interpunzione, operatori numerici e di altro tipo, le 256 (2 elevato all'ottava potenza) combinazioni dei valori possibili (0 e 1) di 8 cifre binarie ci consentono di risolvere il problema.

Allora, come unità di misura delle occupazioni di memoria (elettronica, magnetica, ottica o di altro tipo) è stato assunto il **byte**, cioè l'insieme di 8 bit. La denominazione deriva da **Binary Term**, ad indicare il termine binario, nel senso del più piccolo gruppo di bit richiamabile. Forse non è un caso che **byte** richiami il verbo inglese 'to bite', che significa mordere; il byte come risultato di un morso a una memoria.

Delle 256 combinazioni ottenibili dagli 8 bit, ne basta la metà, 128 per rappresentare **tutti i simboli comuni** usati nel nostro mondo occidentale, per i quali basterebbero, quindi, 7 bit; allora, è stata definita la cosiddetta Codifica **ASCII**, (acronimo da 'American Standard Code for Information Interchange') (pronuncia in inglese: **aschi**). Questa codifica la potete consultare comodamente sullo schermo. Noterete che tutti i codici ASCII hanno la prima cifra a 0, il che sta a indicare che il primo bit non è stato utilizzato.

CODIFICA ASCII STANDARD

Cod. Binario ASCII	Decimale	Carattere o Controllo
----- azioni di controllo -----		
00000000	000	Nulla (nessun effetto)
00000001	001	Partenza testina
00000010	002	Inizio testo
00000011	003	Fine testo
00000100	004	Fine trasmissione
00000101	005	Interrogazione
00000110	006	Risposta
00000111	007	Campanello (beep)
00001000	008	Posiz. indietro
00001001	009	Tabulazione orizz.
00001010	010	Spaziatura verticale
00001011	011	Tabulazione vert.
00001100	012	Nuova pagina
00001101	013	Ritorno carrello
00001110	014	Scorrimento disattivo
00001111	015	Scorrimento attivo
00010000	016	Cambio trasmissione
00010001	017	Unità controllo 1

00010010	018	Unità controllo 2
00010011	019	Unità controllo 3
00010100	020	Unità controllo 4
00010101	021	Risposta negativa
00010110	022	Riposo sincrono
00010111	023	Fine blocco trasmiss.
00011000	024	Annullamento
00011001	025	Fine supporto
00011010	026	Sostituzione
00011011	027	Cambio
00011100	028	Separatore sequenza
00011101	029	Separatore gruppo
00011110	030	Separatore registraz.
00011111	031	Separatore unità
00100000	032	Spaziatura

caratteri stampabili -----

00100001	033	!
00100010	034	" (doppio apice)
00100011	035	# (segno numerico)
00100100	036	\$
00100101	037	%
00100110	038	& (e commerciale)
00100111	039	' (apice semplice)
00101000	040	(
00101001	041)
00101010	042	* (asterisco)
00101011	043	+
00101100	044	, (virgola)
00101101	045	- (segno meno)
00101110	046	. (punto)
00101111	047	/ (barra)
00110000	048	0 (zero)
00110001	049	1
00110010	050	2
00110011	051	3
00110100	052	4
00110101	053	5
00110110	054	6
00110111	055	7
00111000	056	8
00111001	057	9
00111010	058	:
00111011	059	;
00111100	060	< (minore / ang. c.)
00111101	061	=
00111110	062	> (maggiore /ang.a.)
00111111	063	?
01000000	064	@ (a commerciale)
01000001	065	A
01000010	066	B
01000011	067	C

01000100	068	D
01000101	069	E
01000110	070	F
01000111	071	G
01001000	072	H
01001001	073	I
01001010	074	J
01001011	075	K
01001100	076	L
01001101	077	M
01001110	078	N
01001111	079	O
01010000	080	P
01010001	081	S
01010010	082	R
01010011	083	S
01010100	084	T
01010101	085	U
01010110	086	V
01010111	087	W
01011000	088	X
01011001	089	Y
01011010	090	Z
01011011	091	[
01011100	092	\ (barra inversa)
01011101	093]
01011110	094	^ (esponente)
01011111	095	_ (sottolineato)
01100000	096	` (accento grave)
01100001	097	a
01100010	098	b
01100011	099	c
01100100	100	d
01100101	101	e
01100110	102	f
01100111	103	g
01101000	104	h
01101001	105	i
01101010	106	j
01101011	107	k
01101100	108	l
01101101	109	m
01101110	110	n
01101111	111	o
01110000	112	p
01110001	113	q
01110010	114	r
01110011	115	s
01110100	116	t
01110101	117	u
01110110	118	v

01110111	119	w
01111000	120	x
01111001	121	y
01111010	122	z
01111011	123	{ (graffa aperta)
01111100	124	(barra verticale)
01111101	125	} (graffa chiusa)
01111110	126	~ (tilde)

01111111	127	Cancellazione

La codifica **ASCII standard** è stata ufficialmente introdotta nell'anno **1968**, con la specifica **ISO 646**. Potete vedere anche che i primi 32 codici indicano comandi operativi, le cui denominazioni in chiaro rammentano che la codifica venne introdotta nel mondo delle telescriventi.



Ve lo dicevo io che il nostro Cucciolo diverrà un grande progettista! Lo prenderò presto come mio assistente. Avrei dovuto io farvi notare questa “stranezza”, che alle volte è causa di errori di programmatori principianti.

Utilizzando anche il primo bit a sinistra, l' 8°, avendo cioè la possibilità di rappresentare 256 simboli, si ha un, cosiddetto, **ASCII esteso**, che però NON può definirsi standard, in quanto non è lo stesso per tutti ; a lingue e/o ad ambienti operativi diversi, corrispondono codifiche **ASCII estese** diverse, nelle quali sono considerati caratteri specifici delle diverse lingue e/o segni grafici di uso particolare. Torno a dire, però, che in ogni ASCII esteso i primi 128 codici rimangono identici a quelli dell' ASCII standard.

Quando, ad esempio, da Internet vi arriva uno scritto con un segno strano (sempre uguale) al posto di alcuni caratteri, significa che la macchina di chi ha composto quello scritto adotta un **ASCII esteso** diverso da quello adottato dalla vostra macchina.

Se io fossi costretto a lavorare in linguaggio-macchina, scrivendo una lettera, alla fine, dovrei firmare così :

01010110 01100001 01101110 01101110 01101001

invece di :

V a n n i

Capite perché è tanto, tanto bello lavorare con un Linguaggio di programmazione? Al prossimo sabato, ragazzi.