

Codice ASCII

I computer trattano dati di tipo numerico ma anche testuale. Ad esempio un testo scritto con Wordpad presenta parole composte da lettere, magari in alfabeti diversi se si usano lingue diverse, e anche i numeri riportati potrebbero non avere un significato matematico, ma solo essere un codice mnemonico a scopo identificativo.

Per intenderci un numero di telefono non è un dato aritmetico con cui fare delle operazioni aritmetiche, ma viene trattato come un testo, quindi eventualmente ordinato in senso crescente o decrescente, ecc....

il problema è allora il seguente: come codificare, usando il solo codice binario, dati di tipo profondamente diverso?

Si parla infatti di caratteri alfanumerici intendendo:

- le lettere dell'alfabeto, distinguendo le minuscole e le maiuscole
- le dieci cifre da "0" a "9" con le quali costruiamo i numeri decimali
- i simboli speciali, tra i quali quelli di interpunzione (la punteggiatura!) e i segni grafici usati in matematica (le parentesi, simboli che indicano le operazioni matematiche, ..)
- i simboli grafici, linee, archi, gli emoticon,...
- infine esistono caratteri speciali che non sono visibili, ma servono per ordinare il testo, come ad esempio la spaziatura, l'invio, ecc

per rappresentare tutto questo è stato necessario definire una regola generale, che stabilisse una corrispondenza fra ciascun carattere e il codice binario corrispondente.

La prima soluzione storica fu il codice **ASCII**, che è l'acronimo di **American Standard Code for Information Interchange** (ovvero *Codice Standard Americano per lo Scambio di Informazioni*).

La tabella ASCII è un codice convenzionale usato per la rappresentazione dei caratteri di testo attraverso i [byte](#): ad ogni byte viene fatto corrispondere un diverso carattere della tastiera (lettere, numeri, segni).

In realtà lo standard ASCII copre solo i primi 128 byte (da 00000000 a 01111111), i successivi byte fino al 256° costituiscono la *tabella ASCII estesa* che presenta varie versioni a carattere nazionale.

Nella **tabella ASCII standard** si trovano le cifre numeriche, le lettere maiuscole e minuscole (maiuscole e minuscole hanno codici ASCII differenti) la punteggiatura, i simboli aritmetici e altri simboli (\$, &, %, @, #, ecc.). Essendo stata concepita in America, la tabella ASCII standard non comprende le lettere accentate (sconosciute all'ortografia inglese). I primi 32 byte della tabella standard sono inoltre riservati per segnali di controllo e funzioni varie.

L'alfabeto latino, usato nella scrittura di molte lingue nel mondo, presenta una grande quantità di varianti grafiche: si va dalle semplici vocali accentate (accento grave à, acuto á, circonflesso â, dièresi ä, tilde ã) a lettere modificate (lettere con barrette, cediglie, segni), lettere speciali usate solo in una lingua, segni di punteggiatura particolari (il punto interrogativo ed il punto esclamativo capovolti usati nello spagnolo), simboli di valuta, e così via, senza considerare poi che gran parte di questi segni presentano le due forme maiuscola e minuscola. Le varianti sono talmente numerose che i 128 Byte della tabella estesa non sono purtroppo sufficienti a rappresentarle tutte, per questo motivo esistono diverse estensioni della tabella ASCII: lo standard ISO 8859 prevede 15 diverse estensioni, comprese quelle per gli alfabeti diversi dal latino, ma esistono anche ulteriori estensioni non riconosciute dall'ISO e create per esempio dalla Microsoft per i sistemi Windows o dalla Apple per i Macintosh.

La **tabella ASCII estesa** tipicamente utilizzata in Italia è quella dell'Europa occidentale, creata per le lingue germaniche e neolatine (escluso il rumeno). Altre estensioni usate in Europa sono la Centro Europea per i paesi dell'Europa orientale (lingue slave, ungherese, rumeno), la Turca, la Cirillica e la Greca. Questa coesistenza fra diverse versioni del codice ASCII produce spesso discordanze nella visualizzazione dei file di testo. Sarà capitato a molti di aprire un file di testo o ricevere una E-mail e trovare segni assurdi al posto di tutte le lettere accentate, questo perché chi l'ha scritto stava usando una tabella estesa diversa dalla vostra e quindi il vostro computer interpreta alcuni byte del file in modo diverso. Certi tipi di file, come i file html, possono contenere al loro interno il nome esplicito dell'estensione ASCII usata per la loro creazione, così il computer ricevente saprà come regolarsi.

Per cercare di ovviare al problema è stato creato un nuovo standard internazionale detto *Unicode*, definito dalla Unicode Consortium e dalla International Organization for Standardization (ISO 10646), che rappresenta i caratteri usando 2 byte (16 bit). Con 2 byte il numero di combinazioni possibili diventa $256 \times 256 = 65.536$, perciò Unicode supporta 65.536 diversi segni, al posto dei 256 del set ASCII. Si riescono così a rappresentare non solo tutte le varianti dell'alfabeto latino, ma anche tutti gli altri alfabeti (greco, cirillico, arabo, ebraico, hindi, thai, ...) oltre

all'insieme degli ideogrammi cinesi e giapponesi (che sono in tutto circa 30.000, anche se poi ne vengono effettivamente utilizzati solo poche migliaia).

Lo svantaggio dell'Unicode, rispetto all'ASCII, è che le dimensioni dei file di testo risultano comunque raddoppiate (vengono usati 2 byte per carattere, invece di 1 solo).

Se si sta usando Windows si può ottenere ogni carattere ASCII tenendo premuto il tasto **Alt** e digitando il codice decimale corrispondente col tastierino numerico (se il tastierino numerico non fosse attivo, premere prima il tasto *Num lock* o *Bloc Num* per attivarlo). per esempio la chiocciola @ si ottiene digitando 64 mentre si tiene premuto il tasto Alt. Nella tastiera inglese sono già presenti tutti i caratteri della tabella standard; nella tastiera italiana invece mancano l'apice (96), le parentesi graffe (123,125) e la tilde (126).

Unicode: codifica per rappresentare caratteri latini, greci, ebraici, arabi, cirillici, etc. Cfr: ASCII (7 bit, 128 caratteri), ISO-Latin-1 (8 bit, 256 caratteri).

Sebbene la sua utilità sia ridotta per gli alfabeti occidentali, esso si rende prezioso per quei linguaggi, come il Giapponese e il Cinese, in cui l'alfabeto è molto ricco di simboli, dove 256 caratteri distinti non bastano per dare una buona rappresentazione. Si prevede che nel corso degli anni questo standard soppianderà ASCII.
[di Marco Lizza]

I codici Unicode vanno da 0 a 1.114.111 ($2^{16} \times 17$, esadecimale 0x10FFFF). Qualcuno ha detto, scherzando, che Unicode ha 20 bit e mezzo perché 1.114.111 (esadecimale 0x10FFFF) è un po' più di 2^{20} e un po' meno di 2^{21} . In realtà, non ha senso dire che Unicode "ha un certo numero di bit". I codici Unicode, come del resto quelli di qualsiasi altra codifica, sono solo dei numeri, non delle unità di memoria e, quindi, non "hanno" bit. Esistono diversi modi di rappresentare i caratteri Unicode in memoria o su file. Questi formati sono detti **UTF** (*Unicode Transformation Formats*) e al momento lo standard ne definisce tre: a 8, 16 e 32 bit.

- **UTF-7:** obsoleto e ufficialmente rimosso dallo standard. Utilizzava byte da 7 bit e alcuni caratteri nell'intervallo 0x00..0x7F erano codificati con un solo byte di valore corrispondente. Tutti gli altri caratteri invece utilizzavano una combinazione di byte che iniziava col byte 0x2B ("+", in ASCII) e terminava con 0x2D ("-", in ASCII). Lo scopo di questo UTF era la compatibilità con i vecchi sistemi a 7 bit, specialmente certi sistemi di e-mail. Questi sistemi sono ormai praticamente scomparsi.
- **UTF-8:** utilizza unità da 8 bit, dette ottetti. I caratteri nel range 0x00..0x7F sono codificati con una sola unità di valore corrispondente. Tutti gli altri caratteri utilizzano una combinazione di 2, 3, o 4 unità, comprese nell'intervallo 0x80..0xFF. È il formato più comodo per i file e per la trasmissione perché è il più compatto e, soprattutto, perché è compatibile con tutte le vecchie applicazioni che si basano su ASCII e sul rapporto tra byte, otetto, carattere; è ancor più compatibile con le applicazioni progettate per le codifiche *multibyte* dell'Estremo Oriente.
- **UTF-16:** utilizza unità da 16 bit. I caratteri nell'intervallo 0x0000..0xFFFF sono codificati con una sola unità di valore corrispondente. Tutti gli altri caratteri utilizzano una combinazione di 2 unità, comprese nel range 0xD800..0xDFFF. Queste unità speciali per codificare i caratteri maggiori o uguali a 0x10000 sono detti *surrogati*. È un compromesso che scontenta tutti; esiste principalmente per compatibilità con le vecchie applicazioni progettate per Unicode 1.x, che era ancora effettivamente limitato a 65536 codici.
- **UTF-32:** utilizza unità da 32 bit. Tutti i caratteri sono codificati con una sola unità di valore corrispondente. Gli 11 bit più alti sono sempre a zero (*N.d.A.* il codice non supera il numero 0x0010FFFF che in notazione binaria equivale a 00000000 00010000 11111111 11111111). È il formato più adatto per l'elaborazione in memoria, in quanto mantiene il rapporto uno a uno fra unità di memoria e caratteri. Lo spreco di memoria, inoltre, è abbastanza trascurabile, considerata la piccola mole di dati che risiedono in memoria in un dato momento.