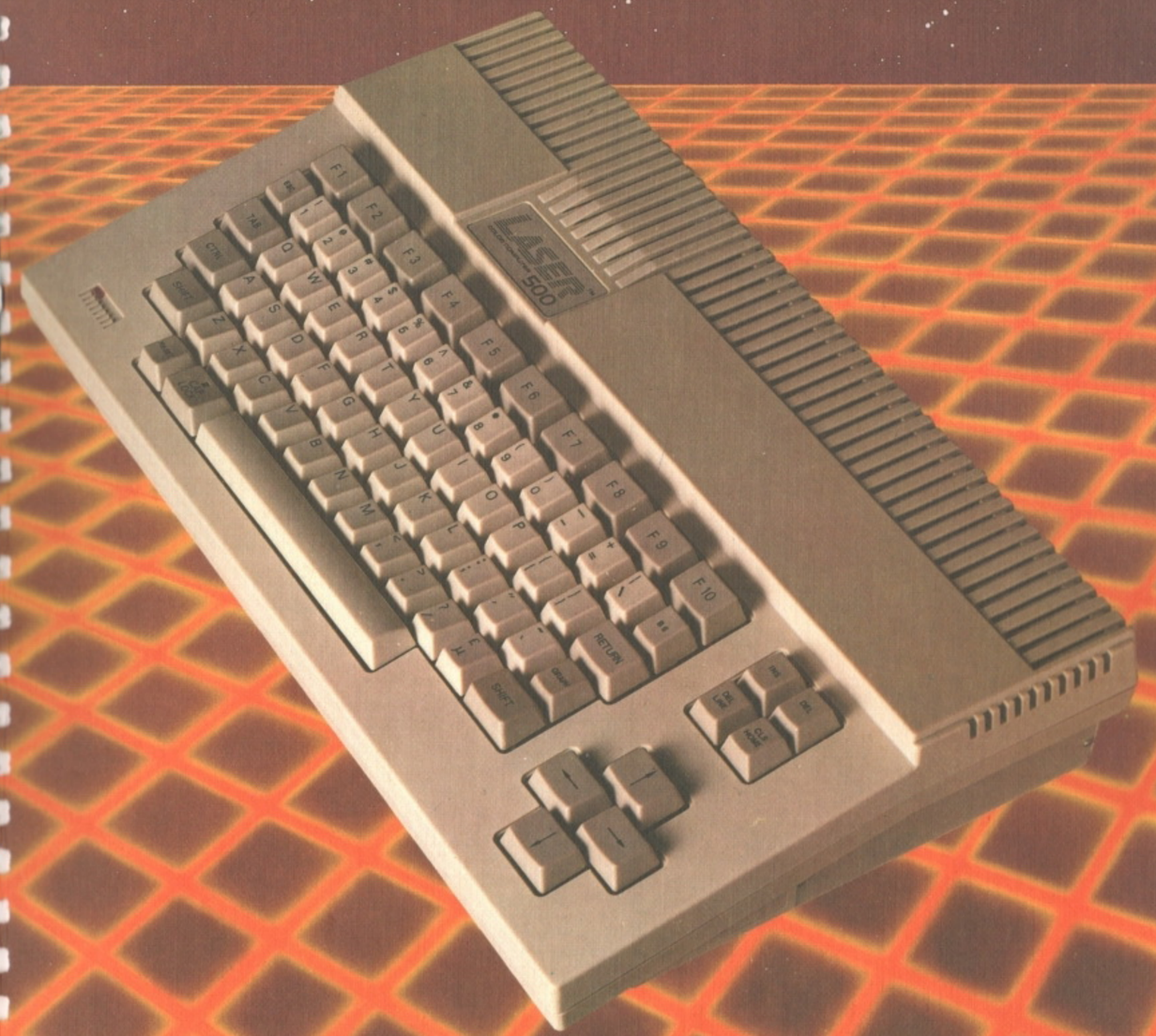
CORSO DI INFORMATICA

LIBRO ESERCIZI - 1



SCUOLA Scheidegger

Longum iter est per praecepta,
breve et efficax per exempla.
(Seneca il giovane)

La Scuola Internazionale Scheidegger, che opera con successo nel campo della formazione professionale ha prodotto questa Guida Didattica per consentire a tutte le persone di accedere con semplicità all'informatica, ad all'uso di un computer.

La nostra intenzione è quella di fornire una chiara visione dei numerosi aspetti dell'informatica, operando con un metodo teorico/pratico, ricco di contenuti ed esemplificazioni.

Ampio spazio è riservato alle esercitazioni pratiche sul computer, ed alla programmazione con il linguaggio BASIC.

Anche gli argomenti più importanti, come l'analisi dei problemi, e le tecniche di programmazione, sono proposti in modo semplice e di facile comprensione.

Tutti potranno vivere da protagonista il più importante sviluppo tecnologico dell'Era attuale, traendone soddisfazioni personali e professionali.

Scuola Internazionale Scheidegger
LA DIREZIONE
Cav. A. Cagienard

Hanno collaborato:
Testo: Enzo Noseda, Graziano Venturini.
Didattica: Alan Garner, Francesco Rossi.

Importanti segnalazione e consigli sono giunte dalle filiali di tutta l'Italia, e dalle sedi europee della Scuola Internazionale Scheidegger, da parte di docenti, ispettori didattici e consulenti.

Luigi Siclari, ha curato la veste tipografica, e la realizzazione delle immagini; Potito Brunato ha eseguito il controllo del materiale didattico.
Stampato dalla Tecnografica di Lomazzo (Como)

LEZIONE 1

OBIETTIVI

1. Presentare il corso in generale e le sue finalità.
2. Far conoscere gli elementi di un sistema per l'elaborazione dati.
3. Chiarire le differenze tra HARDWARE, SOFTWARE e FIRMWARE.
4. Insegnare la funzione e l'uso della tastiera di un computer.
5. Dare chiarimenti sui linguaggi di programmazione e sul BASIC.
6. Insegnare l'uso di home-computer con i comandi diretti.
7. Chiarire l'uso dell'istruzione PRINT da programma (solo con numeri).
8. Definire il concetto di Sistema Operativo in generale, rapportandolo al BASIC con spiegazione dei comandi RUN, SAVE, VERIFY, LOAD, LIST, NEW.
9. Spiegare l'editing nella stesura di programmi BASIC.
10. Spiegare l'uso di costanti numeriche, alfanumeriche e operatori.
11. Chiarire l'uso degli operatori matematici: priorità operazioni.

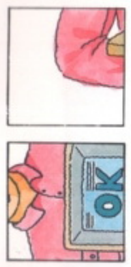
1 PRESENTARE IL CORSO IN GENERALE E LE SUE FINALITÀ

Il corso si articola in nove lezioni di due ore cad.

La frequenza è quindicinale.

Nel corso delle lezioni verranno date informazioni di carattere generale sui sistemi per l'elaborazione dei dati.

Tali informazioni saranno poi rapportate agli home-computer in dotazione, ed al linguaggio di programmazione BASIC, per consentire agli studenti di raggiungere i seguenti obiettivi:



Possedere le conoscenze di base sui sistemi per l'elaborazione dati.

Conoscere le tecniche di programmazione e la loro applicazione per la risoluzione di problemi con l'uso dell'elaboratore.

Conoscere i linguaggi di programmazione in generale, la loro funzione ed il loro utilizzo nei diversi campi dell'informatica.

Conoscere il linguaggio di programmazione BASIC, in forma generale.

Conoscere l'uso degli home-computer e la programmazione degli stessi con il linguaggio BASIC.

Essere introdotti alle problematiche relative alla gestione dei dati ed all'uso delle periferiche.

Possedere una cultura generale, nel campo dell'informatica, che dia l'opportunità di affrontare argomenti più complessi a chi intende dedicarsi all'informatica in modo professionale.

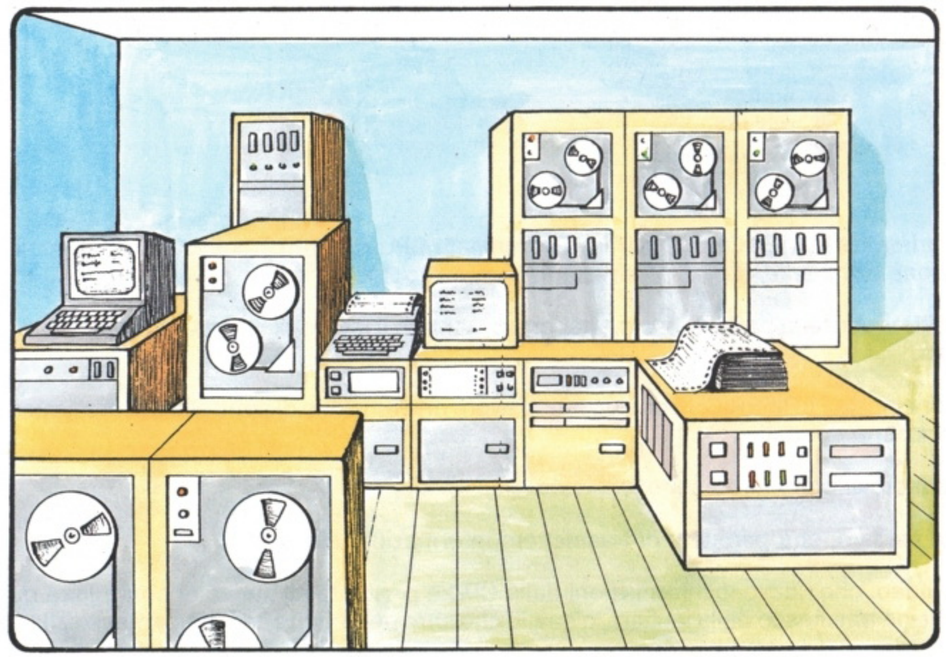
Gli obiettivi principali saranno trattati esaurientemente durante le ore di lezione, mentre alcuni argomenti collaterali saranno solo accennati e necessiteranno di approfondimento, da realizzare sui libri di testo e di verifiche a casa, sul computer in dotazione.

Per ottenere maggior beneficio dalle lezioni in aula è opportuno che gli studenti prendano appunti personali durante le lezioni stesse.

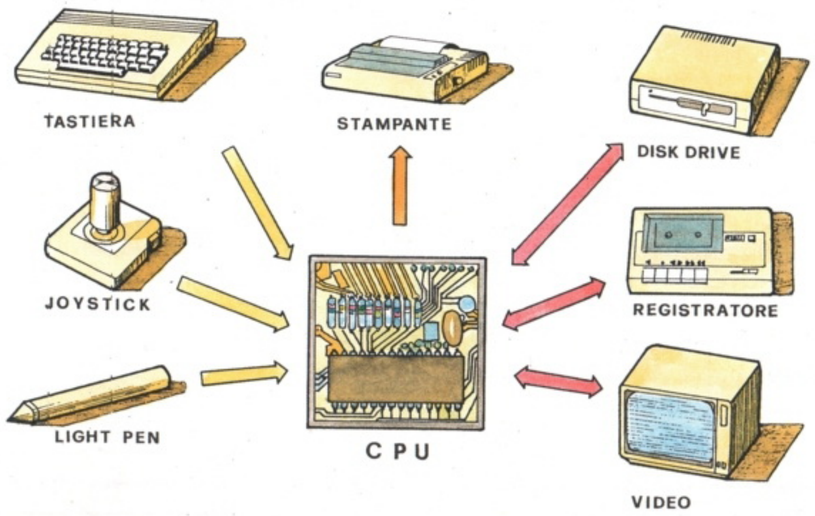
Il livello di apprendimento sarà verificato costantemente attraverso TEST, in modo che si possano indicare agli studenti gli argomenti da ripassare, o per permettere all'insegnante di ripetere quei concetti che non trovano facilità di comprensione da parte degli allievi.

2 FAR CONOSCERE GLI ELEMENTI DI UN SISTEMA PER L'ELABORAZIONE DEI DATI

Un sistema per l'elaborazione dei dati può essere realizzato in diversi modi; a noi interessa quello che utilizza apparecchiature elettroniche che vengono chiamate con il termine generico di COMPUTER.



In realtà tale sistema utilizza diverse componenti che hanno una specifica funzione.



Per prima c'è la MEMORIA CENTRALE, chiamata CPU (Central Process Unit) che svolge la funzione principale, cioè l'elaborazione vera e propria.

La CPU inoltre è la parte del computer che organizza il lavoro di tutti gli altri elementi del sistema.

Ci sono poi le PERIFERICHE che possono avere una funzione di controllo di trasmissione, o di ricezione dei dati.

Le principali periferiche sono:

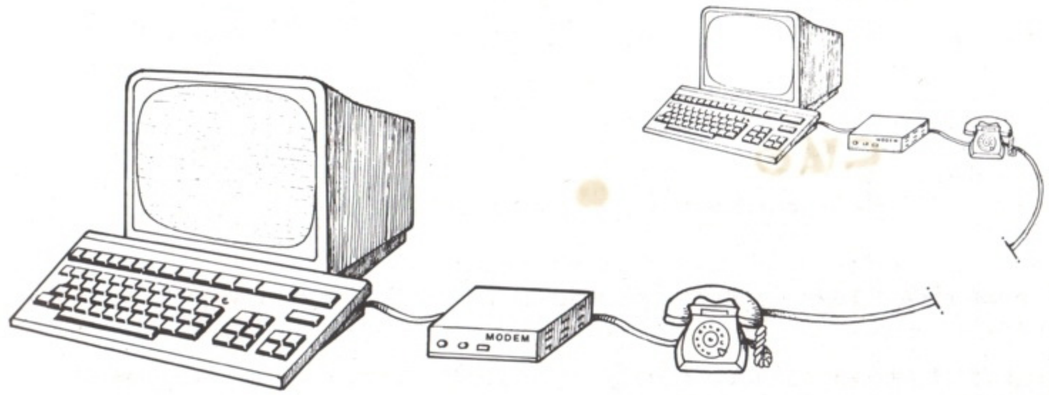
- La tastiera, che permette di trasmettere informazioni alla CPU.
- Il video, che riceve le informazioni dalla CPU e permette all'utente di controllare quello che ha trasmesso dalla tastiera, o quello che altre periferiche hanno trasmesso alla CPU.
- Il registratore o l'unità a dischi, che ricevono informazioni dalla CPU oppure le trasmettono alla stessa e possono memorizzare in modo permanente tali informazioni su nastri o dischi magnetici.
- La stampante, che riceve SOLAMENTE informazioni dalla CPU e può trasferirle su carta.

Tutte queste apparecchiature possono avere, caratteristiche e prestazioni diverse da un costruttore all'altro, o per i diversi modelli di uno stesso costruttore, ma le loro FUNZIONI sono sempre le stesse qualunque sia il modello o il costruttore.

La CPU inoltre può trasmettere informazioni ad altre apparecchiature per usi particolari, attraverso collegamenti che sono chiamati INTERFACCE e può anche ricevere informazioni sempre attraverso tali collegamenti.

Si può quindi collegare il computer al telefono, o ad altri apparecchi domestici, controllandone il funzionamento; oppure collegarlo ad altri computer (es. di una BANCA), per trasmettere e ricevere informazioni da quel computer.

Tali applicazioni esulano dagli obiettivi del corso e pertanto non saranno più riprese.

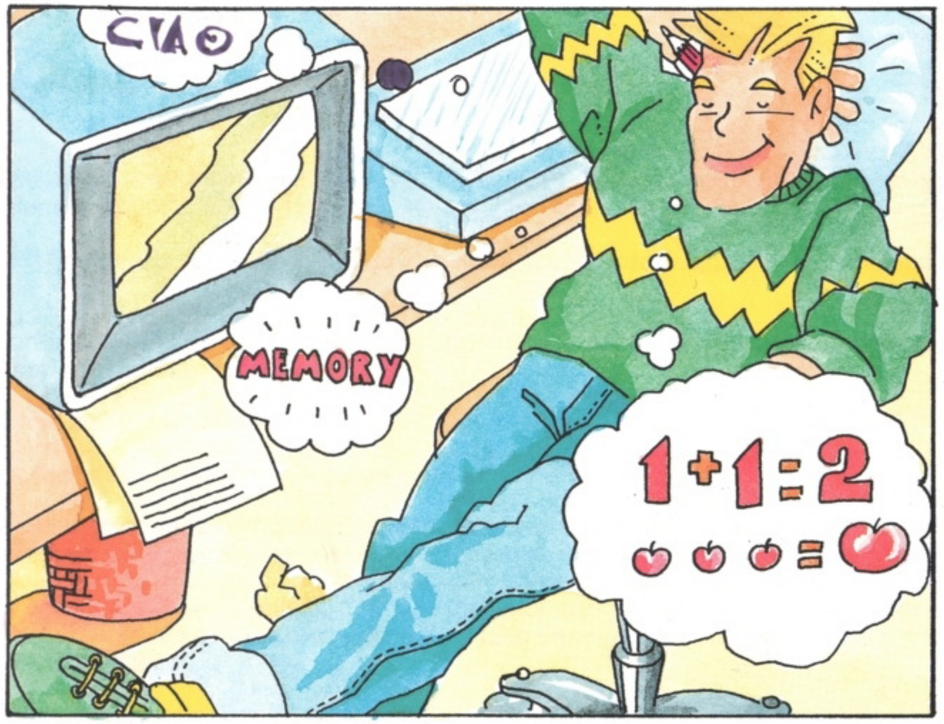


3 CHIARIRE LE DIFFERENZE TRA: HARDWARE SOFTWARE E FIRMWARE

I diversi componenti di un sistema per l'elaborazione elettronica dei dati costituiscono l'HARDWARE (traduzione: HARD = duro WARE = merce), cioè rappresentano la parte visibile e tangibile del sistema.

Però per permettere alle apparecchiature di trasformare i segnali elettrici in qualcosa di comprensibile a chiunque, i costruttori hanno realizzato dei programmi per la trasformazione di tali segnali in numeri ed altri programmi per permettere alla CPU di interpretare questi numeri nel modo voluto dal costruttore.

Questi programmi prendono il nome di SOFTWARE (traduzione: SOFT = soffice WARE = merce).



Il Software che si trova disponibile all'accensione del computer si chiama SOFTWARE RESIDENTE o anche FIRMWARE (FIRM = fermo).

Il software è composto da diverse parti: una è il SISTEMA OPERATIVO, l'altra sono i linguaggi di programmazione.

Un computer può utilizzare più di un sistema operativo e più di un linguaggio di programmazione.

Alcuni SISTEMI OPERATIVI
• CP/M 80
• CP/M 86
• APPLE DOS
• UC SD PASCAL
• MS DOS
• UNIX
• OASIS

Alcuni LINGUAGGI DI PROGRAMMAZIONE
• COBOL
• FORTRAN
• BASIC
• PASCAL
• FORTH
• RPG II

I computer detti HOME-COMPUTER (HOME = casa) non possiedono un vero e proprio sistema operativo, ma sono stati predisposti per l'utilizzo del linguaggio di programmazione BASIC, nei limiti concessi da tale linguaggio.

Tale scelta è dovuta al fatto che il linguaggio BASIC (Beginner's All - purpose Symbolic Instruction Code cioè codice simbolico di istruzioni per tutti gli usi destinato a principianti) è un linguaggio che permette una facile programmazione ed un rapido controllo della stessa e può essere utilizzato anche per la risoluzione di problemi complessi, oppure per la gestione di dati a livello professionale.

HOME COMPUTER

4 INSEGNARE FUNZIONE ED USO DELLA TASTIERA DI UN COMPUTER

La tastiera di un computer ha la funzione di trasmettere informazioni alla MEMORIA CENTRALE. Permette quindi di comunicare con la CPU.

Le informazioni che sono scritte per mezzo della tastiera possono essere controllate sul video.

Alcuni tasti della tastiera svolgono la semplice funzione di scrittura; altri invece svolgono compiti particolari.

I tasti cosidetti "normali" sono simili a quelli di una macchina per scrivere, anche se ci sono delle differenze per ciò che riguarda la loro disposizione e l'uso.

In particolare la tastiera di un computer ha solitamente queste caratteristiche:

In basso c'è la barra spaziatrice (SPACE BAR) che serve appunto per separare con degli spazi vuoti le parole o le lettere.

Sopra tale barra ci sono almeno 4 file di tasti che riportano lettere dell'alfabeto inglese (26 lettere da A a Z), tutti i numeri da 1 a 9 più il numero 0 (zero) che è DIVERSO dalla lettera "O" perchè ha una linea diagonale che lo attraversa

TASTI DI EDITING
TASTI DI FUNZIONE
TASTI ALFANUMERICI
TASTI DI CONTROLLO
TASTO DI IMMISSIONE

Ci sono poi i principali simboli quali % , ; : ? / ! ed altri che non tutte le macchine per scrivere hanno come > < = * + − $ # ↑ non ci sono le lettere accentate, ma esiste l'apostrofo (') che può essere usato come accento. Inoltre ci sono le parentesi () [] @ (AT o A commerciale) & (AMPERSAND o E commerciale) # (NUMBER = numero).

Ci sono poi il tasto SHIFT e SHIFT-LOCK che permettono di selezionare i tasti (come il tasto MAIUSCOLE/minuscole delle macchine per scrivere) in quanto molti tasti svolgono più di una funzione.

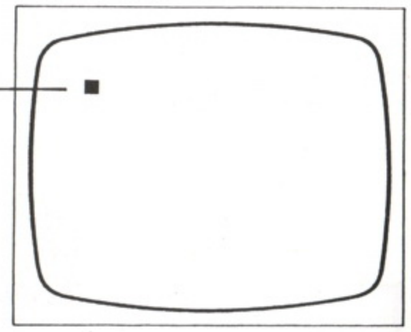
Altri tasti di uso particolare possono avere denominazioni diverse, da una tastiera all'altra, pertanto elenchiamo le principali operazioni svolte da questi tasti, nominando solo alcune delle definizioni più usuali:

Tasti per l'EDITING (edizione): servono per muovere il CURSORE sul VIDEO in senso orizzontale o verticale.

Il CURSORE è quel quadratino lampeggiante che indica, sul video, dove sarà posta la prossima lettera o cifra o simbolo.

TASTI EDITING
Movimenti cursore

↑ sopra
↓ sotto
→ destra
← sinistra



Tasto di cancellazione e tasto per inserimento: il primo cancella i caratteri scritti, il secondo permette di inserire tra due cifre o lettere vicine.

Tasto BREAK di interruzione: solitamente interrompe un programma.

Tasto di IMMISSIONE (solitamente indicato con RETURN o ENTER): è il tasto che permette al computer di "capire" che è stato inserito un comando. È uno dei tasti più importanti della tastiera.

RETURN

Ci sono poi tasti ad uso speciale chiamati tasti FUNZIONE che solitamente sono contrassegnati con la sigla f1 - f2 - f3 etc.

Ogni simbolo, lettera o cifra, che compare sulla tastiera prende il nome di CARATTERE che abbreviato si indica crt.

La pressione dei tasti con le dita si chiama DIGITAZIONE.

Premendo i tasti sulla tastiera i caratteri relativi ad ogni tasto appariranno sul video; (si dice che i caratteri sono VISUALIZZATI), però digitando sulla tastiera NON TRASMETTIAMO NESSUN COMANDO alla memoria centrale.

La trasmissione dei comandi avviene SOLTANTO premendo il tasto di IMMISSIONE (tasto RETURN o ENTER).

RETURN

– Se digitiamo una qualsiasi parola e premiamo il tasto RETURN la CPU interpreta tale parola come un comando.

– Ogni computer è in grado di riconoscere ed interpretare SOLTANTO i comandi per cui è stato programmato dal suo costruttore, cioè quei comandi che fanno parte del SOFTWARE di BASE.

– Se il comando trasmesso non è uno di quelli previsti la CPU segnala di non aver compreso il comando con un'indicazione di errore che viene VISUALIZZATA (cioè scritta sul video).

Per gli HOME-COMPUTER che utilizzano il linguaggio BASIC tale messaggio è solitamente:

```
? SYNTAX ERROR
```

che significa errore di sintassi, cioè NON POSSO ESEGUIRE UN COMANDO CHE NON CONOSCO. Dopo questa segnalazione la CPU si predispone a ricevere un altro comando e sul video compare la scritta READY (trad. PRONTO) cioè: sono PRONTO per ricevere un altro comando.

```
READY
```

È molto importante conoscere e saper utilizzare nel migliore modo la tastiera del computer per poter lavorare celermente.

Altri due tasti molto utili sono: il tasto che permette di portare il CURSORE nell'angolo in alto a sinistra (tale posizione del video viene chiamata HOME trad. = CASA) ed il tasto che permette di cancellare TUTTO quello che compare sul video e CONTEMPORANEAMENTE posizionare il cursore a casa.

Questo tasto può essere indicato con scritte quali CLS oppure CLR abbreviazioni dei termini inglesi CLear Screen (trad. = pulisci schermo) e CLeaR (trad. = Pulisci).

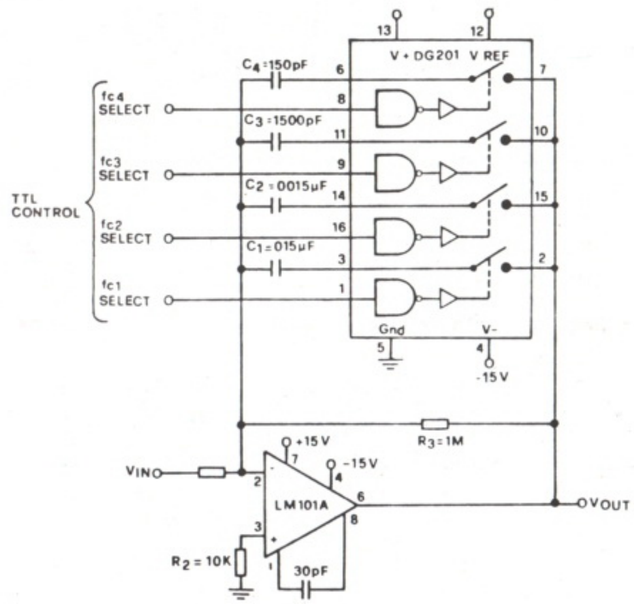
5 CHIARIMENTI SUI LINGUAGGI DI PROGRAMMAZIONE E SUL BASIC

Abbiamo visto che ogni computer è provvisto di un software di base, cioè è stato programmato dal suo costruttore per interpretare dei segnali elettrici, trasformandoli in numeri. Ogni computer è quindi in grado di comprendere solo un linguaggio basato su numeri: ad ogni numero corrisponde una certa operazione che il computer esegue.

Questo linguaggio viene chiamato linguaggio macchina.

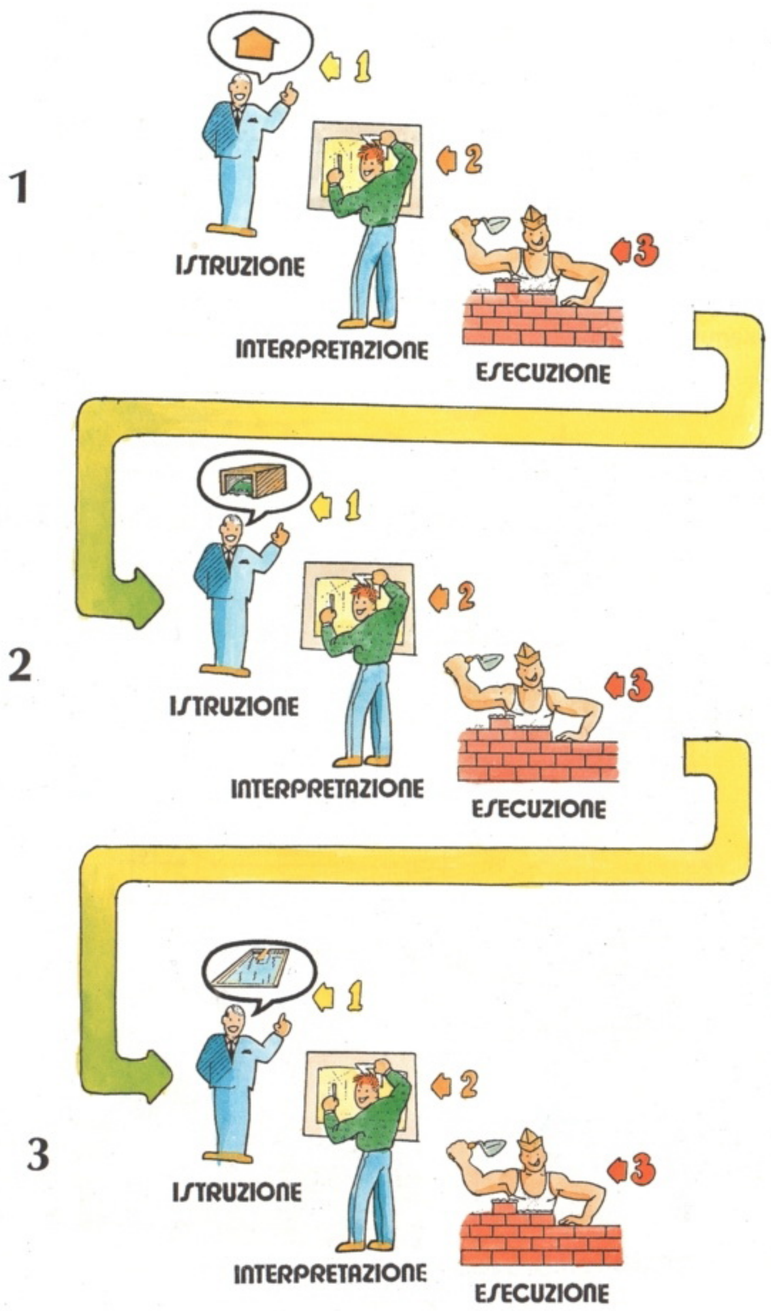
In linguaggio (o codice) macchina però la programmazione è molto complessa, in quanto è possibile eseguire solo operazioni molto semplici, una alla volta

ed occorre anche conoscere molto bene come è stata programmata dal costruttore la memoria del computer.



Per questo ogni computer è predisposto per comprendere linguaggi basati su parole. Il computer stesso provvede a trasformare le parole in numeri, per mezzo dell'INTERPRETE cioè di un programma che converte ogni parola (COMANDO) in una serie di numeri.

LINGUAGGIO INTERPRETATO



Per l'uomo la programmazione diventa più facile, poichè deve solo imparare dei nuovi vocaboli, ognuno dei quali sottointende numerose istruzioni in codice macchina.

I linguaggi cosidetti "evoluti", cioè basati su parole, sono molti e possono essere di tipo generico (GENERAL PURPOSE), o per un uso specifico (SPECIAL PURPOSE). Esempi di linguaggi per uso speciale sono il COBOL ed il FORTRAN mentre il BASIC è un linguaggio di tipo generico.

Ogni linguaggio dispone di una serie di parole e simboli che servono per istruire il computer sulle operazioni da eseguire.

Tale serie di istruzioni è diversa nei diversi linguaggi ed alcune volte può avere delle differenze anche per lo stesso linguaggio usato da computer diversi.

È più importante quindi CAPIRE quello che un'istruzione FA ESEGUIRE al computer che non imparare le istruzioni a memoria.

Il BASIC è un linguaggio molto comodo da usare, perchè ogni comando può essere fatto eseguire immediatamente, altri linguaggi invece permettono di eseguire i comandi solo DOPO una COMPILAZIONE in codice macchina di TUTTE le istruzioni.

In BASIC però l'esecuzione delle istruzioni è più lenta perchè ogni istruzione viene prima interpretata e poi eseguita.

LINGUAGGIO COMPILATO

6 INSEGNARE L'USO DI HOME COMPUTER CON I COMANDI DIRETTI

Un HOME COMPUTER è un tipo di computer che ha come caratteristiche:

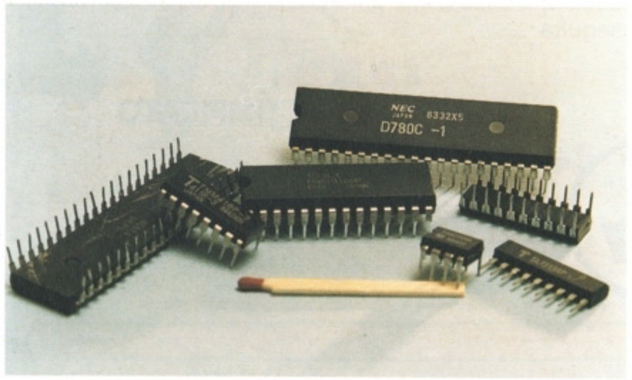
un HARDWARE molto compatto, cioè la MEMORIA CENTRALE è costituita da una o più schede di circuiti stampati su cui trovano posto i MICROPROCESSORI. Tali schede sono inserite sotto la tastiera con cui fanno corpo unico.

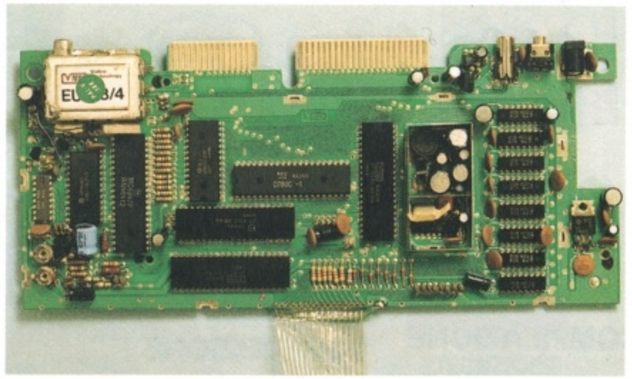
Una capacità di memoria variabile tra 3 e 128 KBytes (1 Byte = 1 crt.).

Un SOFTWARE solitamente limitato all'utilizzo del linguaggio BASIC.

Eventuali estensioni del BASIC per scopi GRAFICI e/o SONORI, cioè con possibilità di realizzare anche programmi con grafica e suoni.

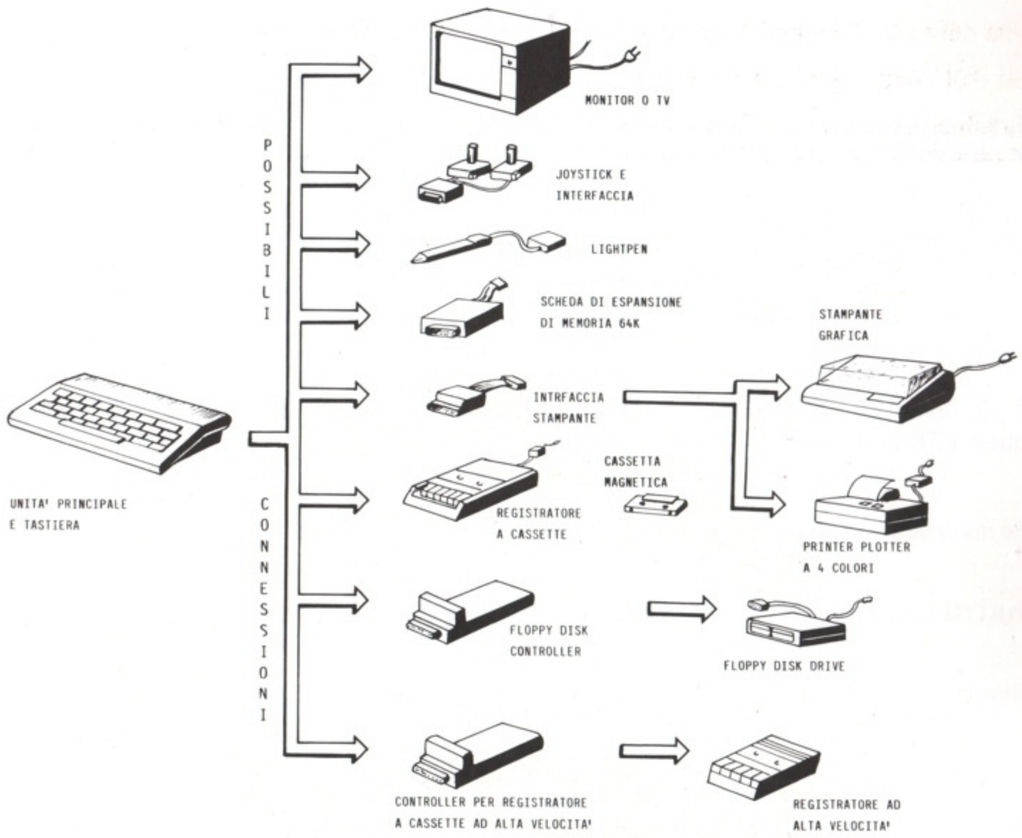
MICROPROCESSORI





Le periferiche disponibili sono solitamente:

- Il registratore a cassette, per archiviare in modo permanente i programmi e/o i dati.
- Il video, che può essere sostituito da un televisore b/n o colori.
- Stampanti di tipo amatoriale ed anche di tipo professionale.
- Unità a dischi flessibili chiamati FLOPPY DISK.
- PADDLE e JOYSTICK che sono accessori utilizzati soprattutto per programmi di divertimento o didattici.

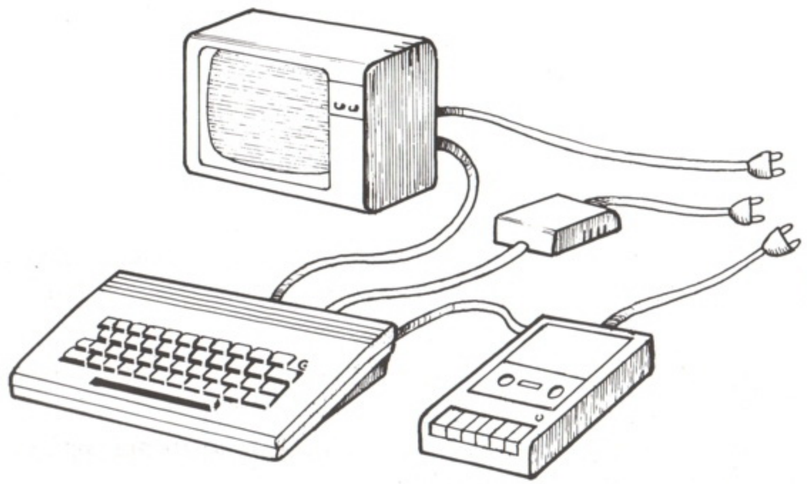


– La messa in funzione di un HOME COMPUTER avviene effettuando gli opportuni collegamenti tra i diversi componenti.

1) La TASTIERA-CPU riceve la tensione di un ALIMENTATORE che deve essere collegato alla RETE elettrica (220 VOLT 50-60 Hertz solitamente).

2) Il Video è collegato alla tastiera tramite un cavetto che viene inserito nella presa di antenna (se il video è un televisore).

3) Il registratore viene collegato alla tastiera con un apposito cavo.



220 Volt
50-60 Hertz

È possibile che per effettuare i collegamenti sia necessario usare piccole apparecchiature quali INTERFACCE e/o MODULATORI.

interfaccia

Il televisore (o il monitor) deve essere collegato alla rete elettrica e così pure il registratore, (in alcuni computer però il registratore riceve l'alimentazione dal computer).

DOPO aver fatto i collegamenti si possono accendere gli apparecchi; di solito È MEGLIO ACCENDERE LA TASTIERA-CPU PER ULTIMA.

Per ricevere il segnale dalla tastiera il VIDEO (se è un televisore) deve essere sintonizzato sulla giusta BANDA di frequenza.

Quando tutto è a posto, sul video compare un riquadro ed alcune scritte, relative al computer usato.

Compare inoltre la scritta READY, e sotto tale scritta lampeggia un quadratino che sta ad indicare la posizione in cui sarà scritto il primo carattere che sarà digitato. Il quadratino si chiama CURSORE.

La parola READY come abbiamo già visto è un messaggio del computer che significa SONO PRONTO PER RICEVERE DEI COMANDI.

È possibile scrivere qualsiasi parola, ma SOLTANTO premendo il tasto di IMMISSIONE (ENTER o RETURN o altro nome), si trasmette il comando.

Inoltre tutte le parole che il computer non riconosce, provocano la segnalazione di errore di sintassi (SYNTAX ERROR) se immesse.

Uno dei comandi che il computer è in grado di riconoscere è PRINT.

Se scriviamo PRINT e premiamo il tasto di IMMISSIONE il computer NON segnala alcun errore, ma riscrive la parola READY.

Ciò significa che il comando PRINT è stato eseguito e che il computer è pronto (READY) a ricevere un altro comando.

L'effetto del comando eseguito però non è visibile in quanto non sappiamo ancora il significato del comando PRINT.

```
PRINT = SCRIVI
```

PRINT (trad. = SCRIVI) è un comando che il computer interpreta così SCRIVI sul video quello che c'è dopo la parola PRINT.

Poichè abbiamo scritto solo PRINT (cioè scrivi) e non abbiamo indicato COSA SCRIVERE il computer NON HA SCRITTO NIENTE.

Se però scriviamo: PRINT 1984 e premiamo il tasto di IMMISSIONE il computer esegue il comando, cioè scrive 1984 e poi attende un altro comando, indicando con READY di essere PRONTO.

È possibile far scrivere sul video dei numeri oppure il risultato di una operazione tra due o più numeri.

ESEMPIO: se digitiamo PRINT 10 + 20 – 8 (più il tasto ENTER o RETURN) il computer non scrive i numeri ed i segni di operazione che abbiamo indicato, ma per prima cosa CALCOLA il RISULTATO delle operazioni e poi SCRIVE sul video tale risultato.

Ogni volta che compare READY sul video è possibile dare comandi al computer.

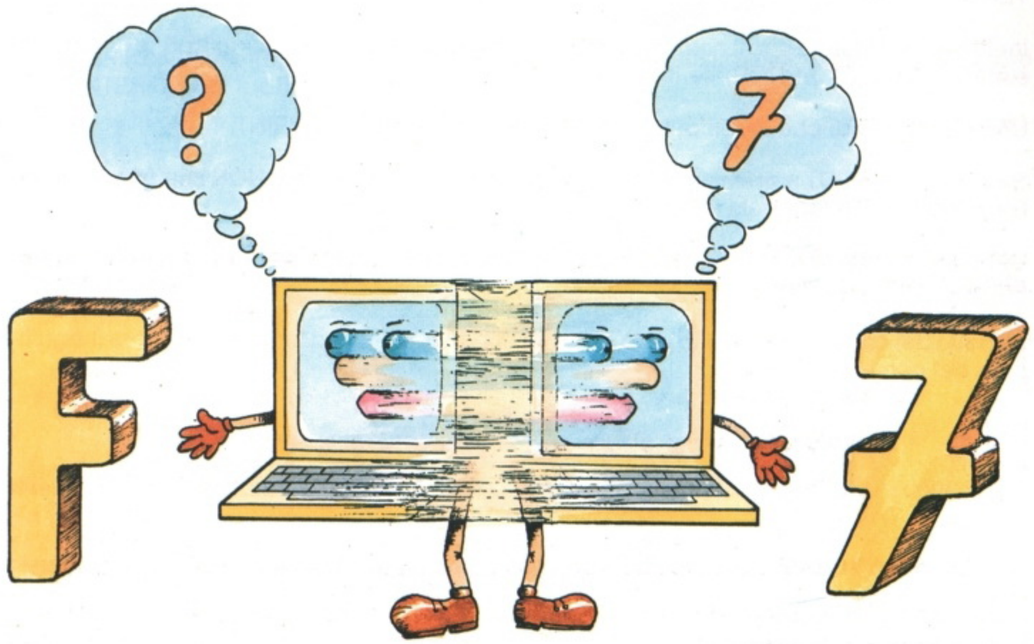
Questo modo di operare con il linguaggio BASIC è chiamato MODO DI COMANDO DIRETTO.

ATTENZIONE

È importante notare che se diciamo di scrivere una parola, digitando PRINT CIAO (più tasto ENTER o RETURN) il computer risponde con un numero cioè scrive la cifra 0 (zero) e poi READY.

Questo è dovuto al fatto che il computer NON è in grado di capire la differenza tra un numero ed una parola.

Questa differenza gli deve essere SEGNALATA, come vedremo più avanti.



7 INSEGNARE L'USO DELL'ISTRUZIONE PRINT DA PROGRAMMA

– Abbiamo visto che è possibile dare dei comandi al computer in MODO DIRETTO se sul video compare la scritta READY. Inoltre il computer esegue i comandi dati SOLO se sono comandi VALIDI.

I comandi vengono eseguiti soltanto quando si preme il tasto di IMMISSIONE (ENTER o RETURN).

È possibile dare comandi anche in un altro modo: se vogliamo far eseguire più di un comando al computer occorre che, prima cosa, indichiamo quale comando deve essere eseguito per primo, poi quale eseguire per secondo, per terzo etc.

Dobbiamo perciò dare una numerazione ai comandi.

Questa numerazione permetterà al computer di eseguire i comandi nell'ordine che noi abbiamo stabilito.

Se vogliamo che il computer scriva il numero 10 e subito dopo scriva il risultato della somma 10 + 9 potremo scrivere:

```
1 PRINT 10      (più tasto ENTER o RETURN)
2 PRINT 10 + 9  (più tasto ENTER o RETURN)
```

Occorre notare che il computer NON risponde più READY.

Questo perchè quando un'istruzione preceduta da un numero NON VIENE ESEGUITA come invece avveniva nel MODO DIRETTO.

Dare una numerazione progressiva alle istruzioni vuol dire: compilare una lista di istruzioni che saranno eseguite, nell'ordine specificato, SOLO IN UN SECONDO TEMPO.

Una lista di ISTRUZIONI NUMERATE é chiamata PROGRAMMA.

La scrittura di questa lista si chiama PROGRAMMAZIONE.

ATTENZIONE

Ricordarsi di premere il tasto di immissione alla fine di ogni riga e non confondere lo zero 0 con la lettera O.

– Dopo aver inserito le istruzioni numerate (cioè il programma) è possibile dare al computer un comando per farle eseguire.

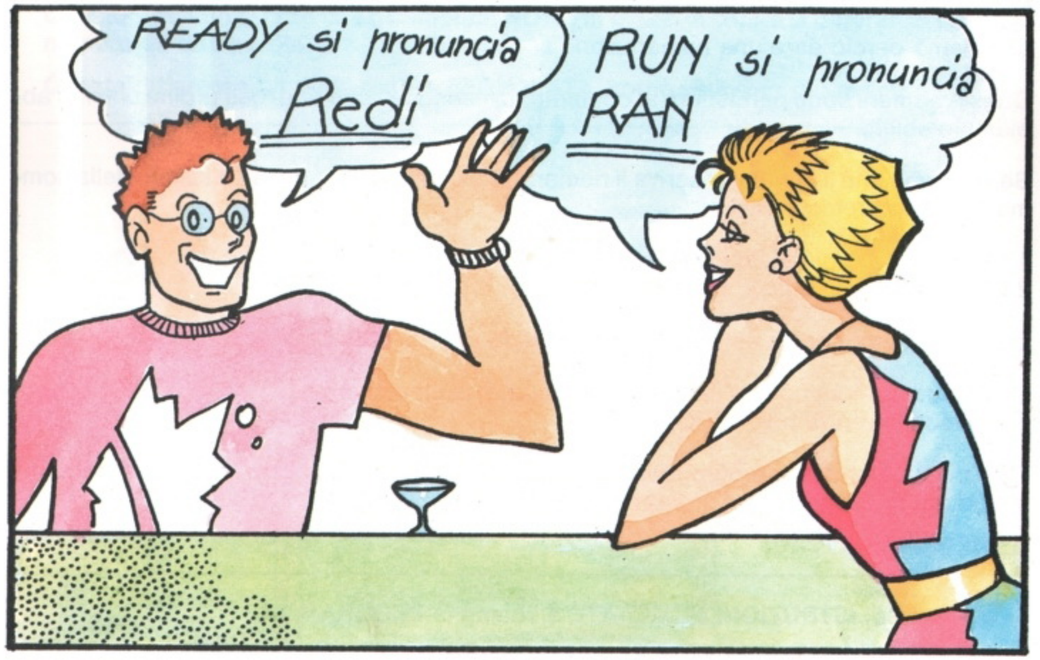
– Il comando che dice al computer di ESEGUIRE una lista di istruzioni è RUN (più tasto RETURN).

RUN = ESEGUI

Con questo comando il computer esegue le istruzioni del programma che noi abbiamo inserito precedentemente.

Nell'esempio di cui sopra eseguirà prima l'istruzione PRINT 10 cioè scriverà 10 sul video e dopo eseguirà PRINT 10 + 9 per cui scriverà 19 sul video.

– Eseguito il programma, sul video tornerà il messaggio READY.



8 DEFINIRE IL CONCETTO DI SISTEMA OPERATIVO

Parlando in generale dei computer abbiamo visto che essi sono dotati dal costruttore di un SOFTWARE di base, cioè di programmi, grazie ai quali possono interpretare comandi, ed eseguire operazioni.

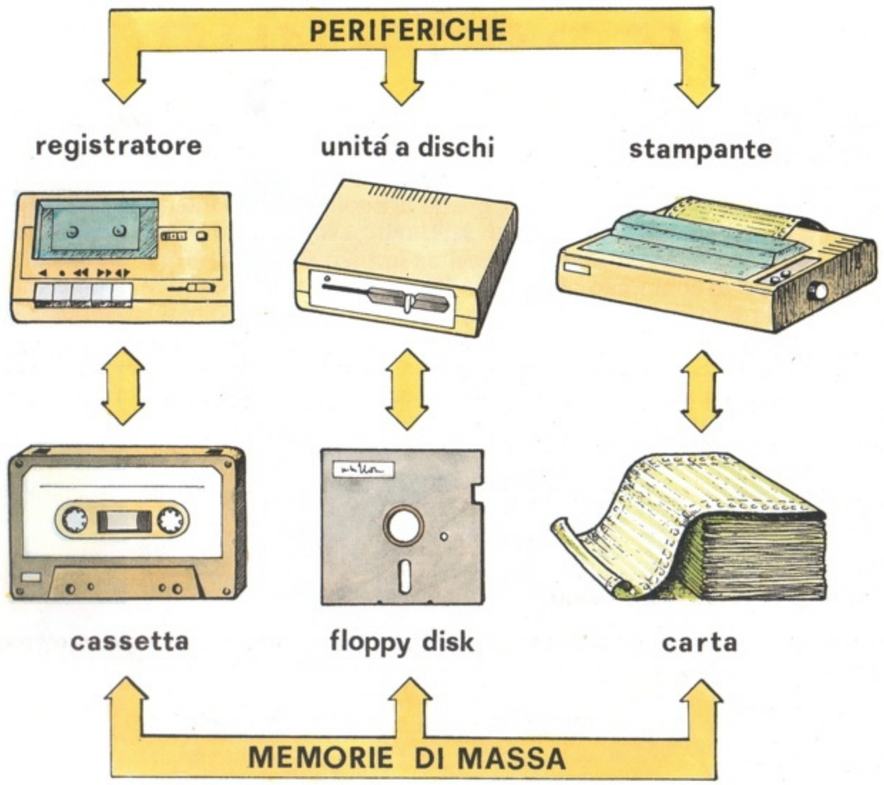
Nel software di base è compreso il SISTEMA OPERATIVO che, come dice il nome, è il MODO in cui il computer è in grado di OPERARE, interpretando determinati comandi.

Il Sistema Operativo (S.O.) consente al computer anche di interpretare un certo tipo di linguaggio di programmazione; permette inoltre la scrittura, la correzione o la compilazione in codice macchina, e l'esecuzione dei programmi, segnalando gli eventuali errori e molte altre cose.

Gli HOME-COMPUTER non dispongono di un S.O. completo, ma sono predisposti soltanto per operare in AMBIENTE BASIC, cioè comprendono solo i comandi relativi al linguaggio BASIC.

Il S.O. di un HOME-COMPUTER permette di scrivere e correggere programmi BASIC ed è in grado di eseguirli, interpretando le istruzioni e segnalando eventuali errori.

Il sistema operativo permette inoltre di TRASFERIRE i programmi dalla MEMORIA CENTRALE, alle MEMORIE DI MASSA (nastri, dischi, carta), o di riportare nella MEMORIA centrale il contenuto, precedentemente archiviato su nastri o dischi.



LA MEMORIA DEL COMPUTER E DELLE PERIFERICHE

UNITÀ CENTRALE O PERIFERICA	TIPO DI MEMORIA	ELEMENTO CHE PERMETTE LA MEMORIZZAZIONE DEI DATI
COMPUTER	R.O.M.• e R.A.M.•	MICROPROCESSORI
REGISTRATORE	MEMORIA DI MASSA	NASTRO MAGNETICO
UNITÀ A DISCHI	MEMORIA DI MASSA	DISCHI
STAMPANTE	MEMORIA DI MASSA	CARTA

• R.O.M. = READ ONLY MEMORY (MEMORIA DI SOLA LETTURA).

• R.A.M. = RANDOM ACCESS MEMORY (MEMORIA AD ACCESSO CASUALE).

Per mantenere la distinzione tra S.O. e linguaggio di programmazione chiameremo **COMANDI di SISTEMA,** tutti quei comandi che fanno eseguire le operazioni di cui sopra, mentre chiameremo **ISTRUZIONI** i comandi che compiono operazioni di tipo diverso, e che servono, soprattutto, a manipolare dati in un programma.

IN GENERALE

SISTEMA OPERATIVO	LINGUAGGIO DI PROGRAMMAZIONE
COMANDI DI SISTEMA	ISTRUZIONI
. . . .	. . . .

Riprendendo i due comandi visti prima possiamo quindi dire che: PRINT è una istruzione BASIC che permette di scrivere sul video. RUN è un COMANDO di SISTEMA, che dice al computer di eseguire un programma BASIC che si trova in memoria.

Vediamo ora altri COMANDI di SISTEMA ed il loro significato:

SAVE (trad. = SALVA): significa trasferisci il programma che c'è in memoria su una MEMORIA di MASSA (nastro o disco).

VERIFY (trad. = VERIFICA): verifica se il programma che c'e in memoria è uguale al programma che c'è sul nastro o disco.

LOAD (trad. = CARICA): leggi, dal nastro o disco, il programma registrato e trasferiscilo in MEMORIA.

LIST (trad. = LISTA): scrivi sul video il programma che è in MEMORIA.

NEW (trad. = NUOVO): cancella dalla MEMORIA il programma esistente.

IN BASIC

SISTEMA OPERATIVO		LINGUAGGIO DI PROGRAMMAZIONE
COMANDI DI SISTEMA	(Pronuncia)	ISTRUZIONI
RUN	Ran	PRINT
SAVE	Seiv	. . . . .
VERIFY	Verifai	. . . . .
LOAD	Lod	. . . . .
LIST	List	. . . . .
NEW	Niù	. . . . .

9 SPIEGARE L'EDITING NELLA STESURA PROGRAMMI

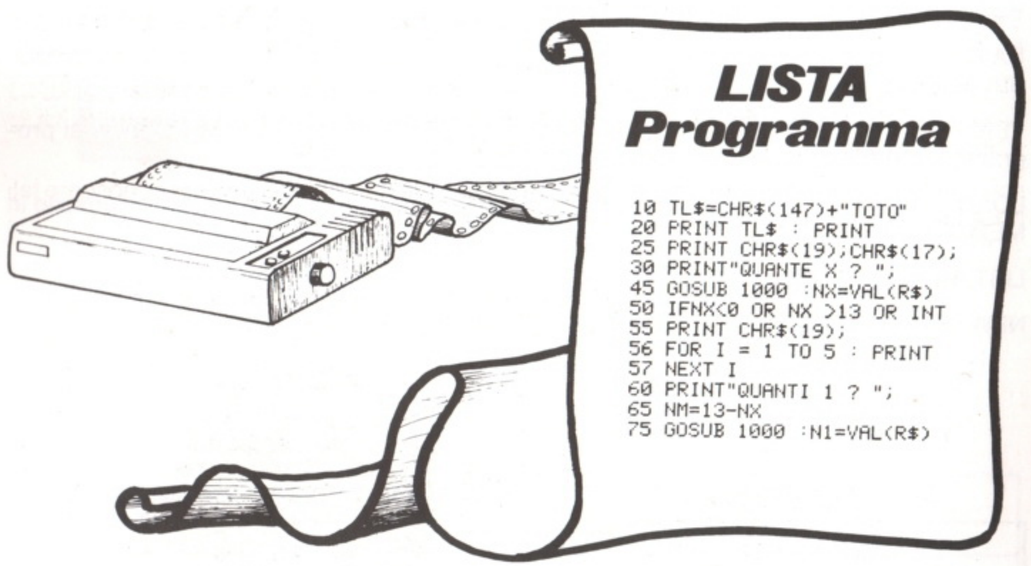
Il Sistema Operativo permette al computer di capire la differenza tra un comando e l'altro; inoltre consente all'utente di scrivere un programma e di correggerlo, o di modificarlo, aggiungendo oppure togliendo istruzioni.

Queste operazioni prendono il nome di EDITING cioè EDIZIONE di un programma.

L'EDITING ha delle regole molto precise che possono essere diverse in relazione al computer usato.

In linea generale, riferendoci agli HOME-COMPUTER ed al BASIC si hanno le seguenti regole:

1 Un programma BASIC è una lista di istruzioni numerate.



2 La numerazione delle righe dice al computer quale è l'ordine di esecuzione delle istruzioni (quale è la prima, la seconda etc.).

3 Dopo aver scritto il numero della istruzione (che si chiama numero di riga) bisogna scrivere una istruzione BASIC e poi premere il tasto ENTER o RETURN per inserire la riga in memoria.

4 Le righe possono essere inserite con una numerazione qualsiasi, ma il numero di riga DEVE ESSERE UN NUMERO INTERO POSITIVO.

5 Il numero di riga minimo è 0 (zero), il numero di riga massimo dipende dal computer usato, ma di solito è 65536.

6 Se si scrive una riga con numero uguale ad una riga già inserita la NUOVA riga sarà inserita al posto della VECCHIA e quest'ultima sarà cancellata dalla memoria.

7 Per cancellare una riga si deve scrivere il numero della riga e premere il tasto ENTER o RETURN. Occorre cioè inserire una riga che non contiene istruzioni.

8 Per sostituire una riga con un altra è sufficiente inserire la riga nuova con il numero della riga vecchia.

9 Per correggere una riga si può operare in diversi modi che dipendono dal computer usato, oppure si può semplicemente RISCRIVERE la riga in modo corretto (sostituendo la riga errata).

10 Alcuni computer segnalano eventuali errori di sintassi ad ogni inserimento, cioè NON accettano righe che contengono errori; altri invece accettano qualsiasi riga di programma ed effettuano i controlli SOLO durante l'esecuzione, cioè dopo il comando RUN.

È molto importante conoscere a fondo le possibilità di EDITING offerte dal proprio computer.

Alcuni possiedono dei comandi di sistema per l'editing, esempi di tali comandi sono: AUTO (abbreviazione di AUTOmatic) che permette la numerazione automatica delle righe, RENUM (abbrev. di RENUMBER) che permette di RINUMERARE le righe.

È sempre necessario fare riferimento ai manuali del computer che si usa per conoscere tali possibilità.

10 SPIEGARE L'USO DI COSTANTI (numeriche, alfanumeriche) e OPERATORI

Stabilita la differenza tra comandi di sistema e linguaggio di programmazione possiamo analizzare più a fondo il linguaggio stesso.

Un linguaggio di programmazione permette di comunicare con il computer per mezzo di parole, facendogli compiere determinate operazioni.

È opportuno fare alcune distinzioni tra queste parole, per capire meglio come il computer le interpreta.

Abbiamo visto che con l'ISTRUZIONE PRINT si comanda al computer di SCRIVERE sul video.

Però se non indichiamo COSA scrivere il computer non scrive niente.

Scrivendo un numero dopo l'istruzione PRINT (es. PRINT 10) il computer scrive il numero che NOI gli abbiamo detto di scrivere.

Se ora scriviamo un programma:

```
10 PRINT 25      (più tasto ENTER o RETURN)
20 PRINT 12      (più tasto ENTER o RETURN)
30 PRINT 2100    (più tasto ENTER o RETURN)
```

e poi chiediamo l'esecuzione del programma con il comando RUN il computer scriverà tutti i numeri che gli abbiamo detto di scrivere.

I numeri inseriti dopo l'istruzione PRINT NON sono una ISTRUZIONE ma il computer li interpreta, cioè capisce che sono numeri.

Questi numeri, inseriti in un programma, prendono il nome di COSTANTI NUMERICHE.

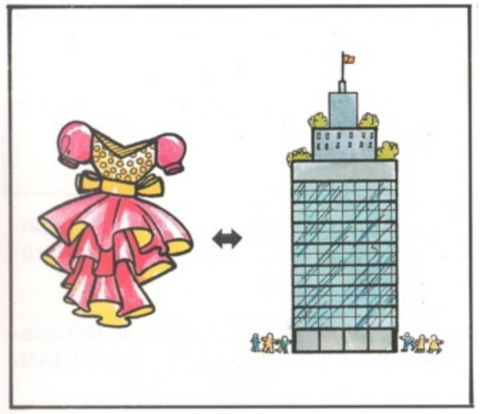
Cerchiamo ora di capire come il computer può interpretare le parole, facendo qualche esempio:

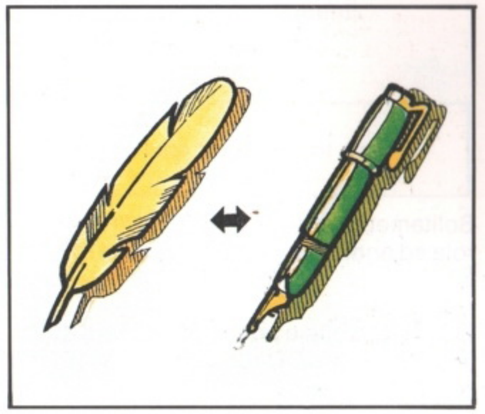
Se noi leggiamo o sentiamo pronunciare parole di uso comune, come PENNA o ABITO o VIA, senza leggere o sentire la frase in cui sono inserite, ci rendiamo conto che ognuna di queste parole può avere più di un significato:

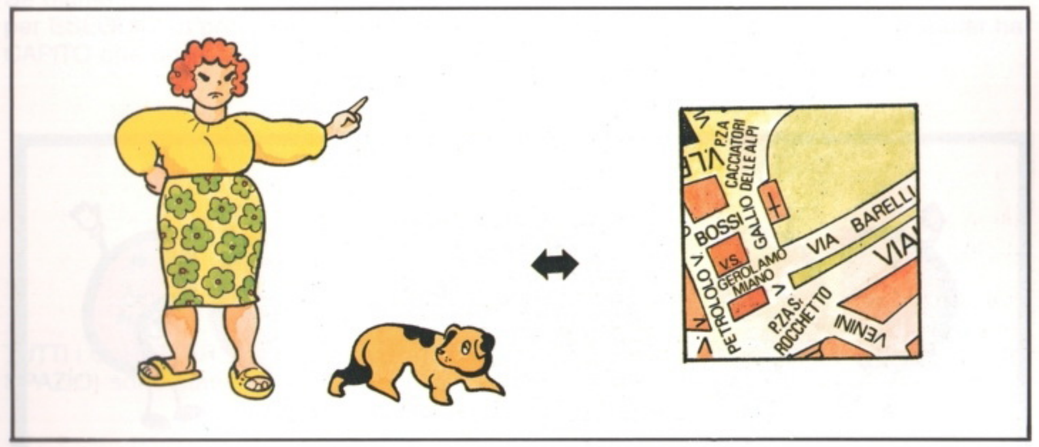
PENNA può essere la penna di un animale o la penna per scrivere.

ABITO può essere l'indumento o la voce del verbo ABITARE (io abito qui).

VIA può significare STRADA o esprimere ALLONTANAMENTO (es: vado via).







È evidente che noi possiamo dare il giusto significato a tali parole SOLTANTO se sappiamo in quale frase o discorso sono usate.

Abbiamo bisogno di qualcosa in più che non la semplice parola.

Allo stesso modo: il computer è predisposto per interpretare sia numeri, sia parole, ma per capire la differenza tra numero e parola ha bisogno di un riferimento, cioè di qualcosa che gli permetta di interpretare il giusto significato.

Il BASIC utilizza il segno delle VIRGOLETTE (") per segnalare al computer la differenza tra parole e numeri.

OGNI VOLTA che il computer trova le virgolette, (chiamate anche impropriamente APICI), interpreta TUTTO QUANTO scritto a destra delle virgolette come parola. Questo fino a quando non trova ancora le virgolette che gli segnalano la FINE della parola.

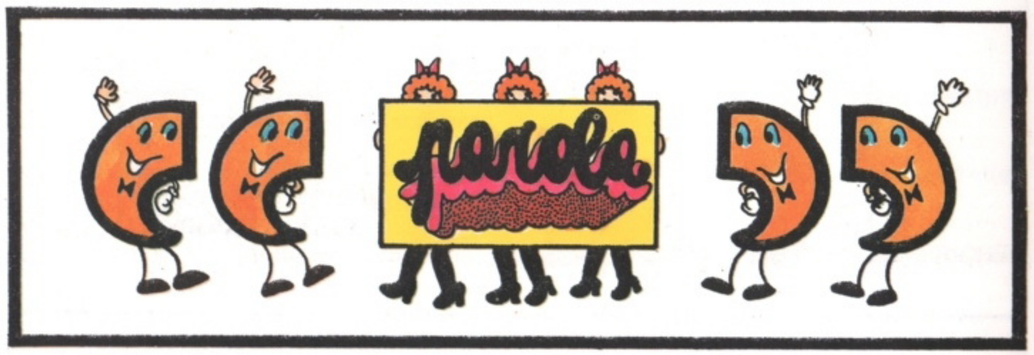
Per il computer una parola è una serie di CARATTERI che sono racchiusi tra le virgolette.

```
"PAROLA"
```

Solitamente si dice APRIRE le VIRGOLETTE quando si mettono le virgolette prima di una parola ed analogamente si dice CHIUDERE le virgolette quando si mettono alla fine della parola.

Tra le virgolette e possibile scrivere qualsiasi lettera, cifra, segno grafico o simbolo ortografico, vale a dire QUALSIASI CARATTERE tra quelli indicati sulla tastiera del computer TRANNE UNO.

L'unico carattere che NON È POSSIBILE mettere tra virgolette è il segno delle VIRGOLETTE che viene usato come segnale per il computer.



Proviamo ora a verificare quanto detto:

Per prima cosa diamo al computer il comando di sistema NEW per cancellare dalla memoria il programma scritto precedentemente.

```
NEW
```


Quando compare la scritta READY., inseriamo le righe di programma seguenti.

RICORDARE di PREMERE il tasto ENTER o RETURN alla fine di ogni riga.

```
10 PRINT "I GIORNI DELLA SETTIMANA SONO 7:"
20 PRINT "LUNEDÌ, MARTEDÌ, MERCOLEDÌ,"
30 PRINT "GIOVEDÌ, VENERDÌ, SABATO,"
40 PRINT "..... E DOMENICA!!!"
```

Se diamo il comando di sistema RUN (più tasto ENTER o RETURN),
per ESEGUIRE il programma esistente in memoria, potremo verificare che il computer ha CAPITO che doveva scrivere delle parole.

```
RUN
```

TUTTI i CARATTERI (lettere, cifre, virgole, punti, due punti, punto esclamativo ed anche lo SPAZIO) sono stati scritti ESATTAMENTE come li abbiamo scritti noi.

Le parole che inseriamo in un programma (scritte tra virgolette) NON SONO ISTRUZIONI, ma prendono il nome di COSTANTI ALFANUMERICHE.

Le COSTANTI ALFANUMERICHE sono chiamate (in BASIC) COSTANTI STRINGA, perchè rappresentano una serie di caratteri allineati in una fila più o meno lunga.

"COSTANTE ALFANUMERICA"

Cerchiamo di comprendere le differenze tra COSTANTI NUMERICHE e COSTANTI ALFANUMERICHE (STRINGHE).

Le costanti numeriche sono SOLO e SOLTANTO NUMERI.

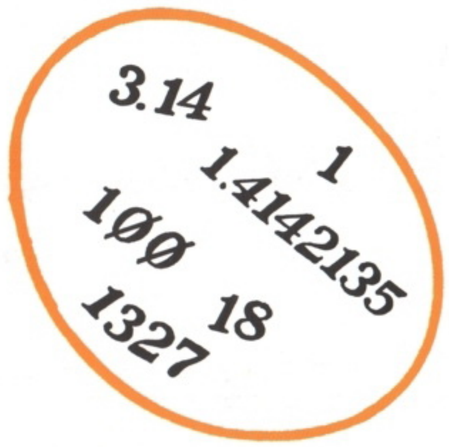
Le costanti ALFANUMERICHE invece possono essere costituite da lettere, cifre o simboli (che da ora in poi chiameremo CARATTERI).

– Ciò significa che l'istruzione PRINT 12 e l'istruzione PRINT "12" SONO DIVERSE anche se il risultato che si ottiene dal computer è apparentemente lo stesso.

– La differenza è questa:

Le COSTANTI NUMERICHE possono essere usate per calcoli MATEMATICI;

Le COSTANTI ALFANUMERICHE NON POSSONO essere usate per calcoli, NEMMENO quando rappresentano CIFRE come nell'esempio di cui sopra.



Chiarita la differenza vediamo quali sono le operazioni matematiche che un computer normalmente può eseguire tra NUMERI ed i simboli che segnalano al computer di ESEGUIRE una operazione matematica.

Le operazioni possibili sono:

1 ATTRIBUIRE il segno ad un numero: cioè precisare se un numero è NEGATIVO o POSITIVO.
Ciò si ottiene semplicemente mettendo il segno – (meno) ai numeri NEGATIVI.
Omettendo il segno – (meno) il computer interpreta il numero come POSITIVO.

2 ELEVARE ad esponente un numero vale a dire fare il quadrato o il cubo di un numero, oppure elevarlo ad altra potenza.
Questa operazione viene indicata al computer per mezzo del segno ↑ .
Per scrivere l'operazione: 10 elevato alla seconda scriveremo: 10 ↑ 2 ed il computer interpreterà nel modo esatto.

3 MOLTIPLICARE un numero per un altro: ad esempio 10 moltiplicato 5 questa operazione si indica con il segno * (asterisco) e NON con il segno X (per) come siamo stati abituati a fare a scuola, per cui si scrive 10*5.

4 DIVIDERE un numero per un altro: es. 10 diviso 5 si indica con il segno / (barra) e NON con : (due punti) o altro, per cui si scrive 10/5.

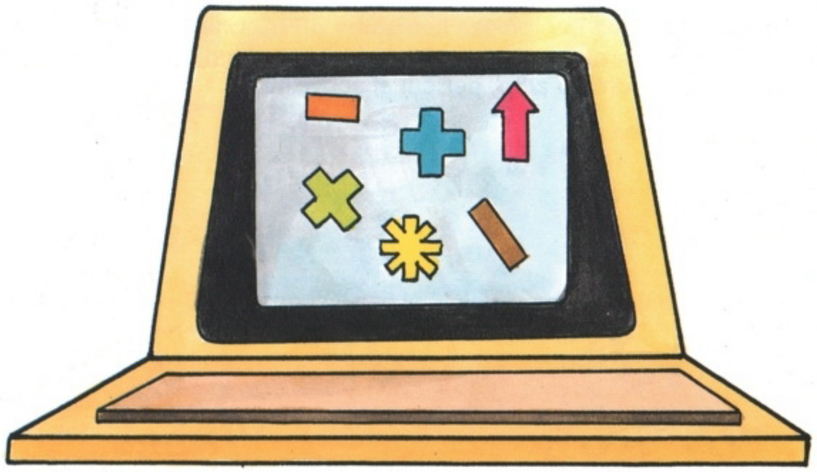
5 SOMMARE due numeri: es. 10 più 5 si usa il segno + (più), per cui si scrive 10 + 5.

6 SOTTRARRE da un numero un altro numero: es. 10 meno 5 i usa il segno – (meno) per cui si scrive 10 – 5.

SIMBOLI RELATIVI AGLI OPERATORI MATEMATICI

SIGNIFICATO	SIMBOLI USATI IN MATEMATICA	SIMBOLI USATI IN BASIC
NUMERO POSITIVO	NESSUN SIMBOLO	NESSUN SIMBOLO
NUMERO NEGATIVO	— (meno)	— MENO
SOMMA	+ (più)	+ PIÙ
SOTTRAZIONE	— (meno)	— MENO
MOLTIPLICAZIONE	X (per)	* ASTERISCO
DIVISIONE	: (due punti) — (opp. linea di fraz.) es. $\frac{1}{2}$	/ BARRA DIAGONALE
ELEVAZIONE A POTENZA	N^2 (esponente) esempio: $20^2 = 400$	↑ (freccia) oppure: E (lettera E) esempi: 20↑2 = 20 elevato a 2 = 400 oppure: 4E2 = 4 volte 10 elev. a 2 = 400

TUTTI i simboli che servono per indicare operazioni matematiche prendono il nome di OPERATORI MATEMATICI.



11 CHIARIRE L'USO DI TUTTI GLI OPERATORI (priorità operazioni)

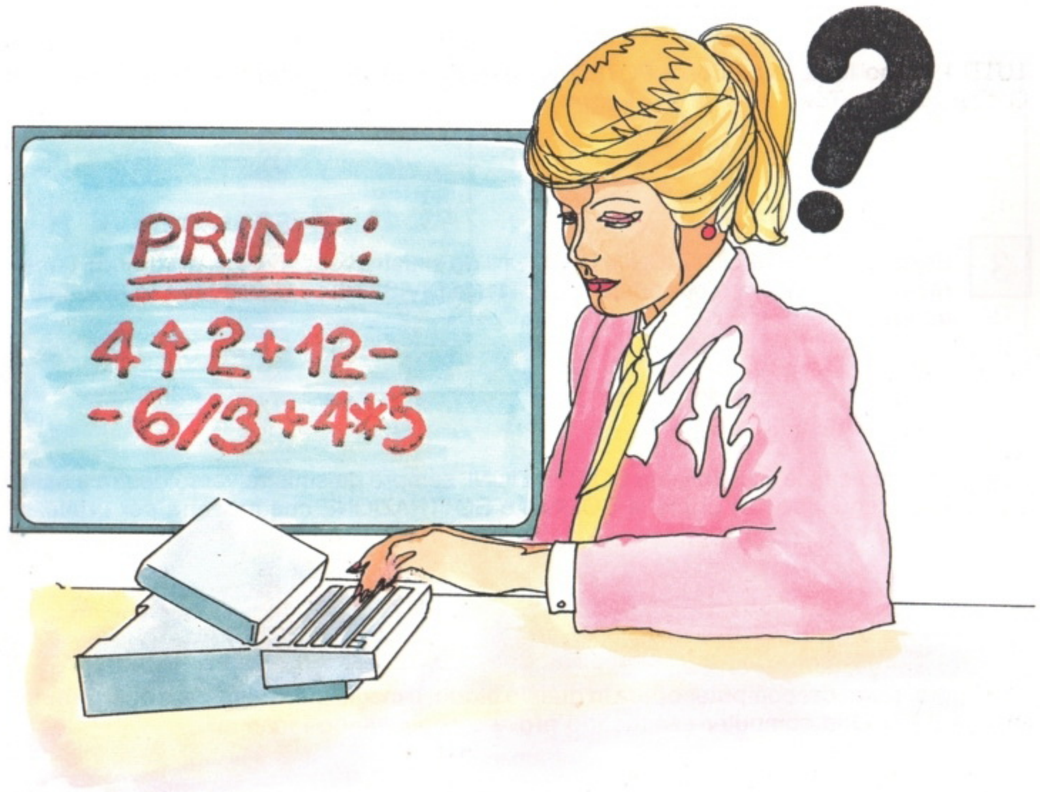
Cerchiamo di approfondire l'uso degli operatori per saperli usare nel migliore dei modi.

Occorre precisare che fino ad ora conosciamo una sola ISTRUZIONE BASIC cioè PRINT che serve per scrivere sul video.

Utilizzeremo quindi questa istruzione per controllare sul video il risultato delle operazioni.

Esempio: PRINT 4↑2+12−6/3+4*5

Se cerchiamo di fare il calcolo a mente dobbiamo scegliere quale operazione eseguire per prima.



Anche il computer fa come noi: per prima cosa stabilisce in quale ordine eseguire le operazioni.

Questa scelta viene chiamata

ATTRIBUZIONE DELLE PRIORITÀ

e significa appunto decidere quale operazione deve essere eseguita per prima, quale per seconda, etc.

La regola usata dal computer è la seguente:

1 ATTRIBUISCE il segno positivo o negativo ai numeri.

2 Esegue le ELEVAZIONI a POTENZA.

3 Esegue le MOLTIPLICAZIONI e DIVISIONI da sinistra verso destra, svolgendo l'operazione che incontra per prima tra TUTTE le DIVISIONI o MOLTIPLICAZIONI che incontra.

4 Per ultimo fa le SOMME e le SOTTRAZIONI, sempre da sinistra verso destra e sempre eseguendo per prima la SOMMA o SOTTRAZIONE che incontra per prima.

La maggior parte dei computer opera in questo modo; è meglio però verificare quali priorità assegna il proprio computer eseguendo prove o consultando i manuali.

Usando la parentesi (SOLO le PARENTESI ROTONDE), il computer esegue per prima i calcoli DENTRO le parentesi rotonde, e successivamente i calcoli FUORI dalle parentesi.

Anche DENTRO le parentesi il computer attribuisce le priorità alle operazioni, secondo le regole già viste.

Se si usano più parentesi vengono eseguite per prime le operazioni nelle parentesi più interne.

Inoltre se si APRE una parentesi la si deve poi CHIUDERE, altrimenti il computer segnala l'errore con un messaggio.

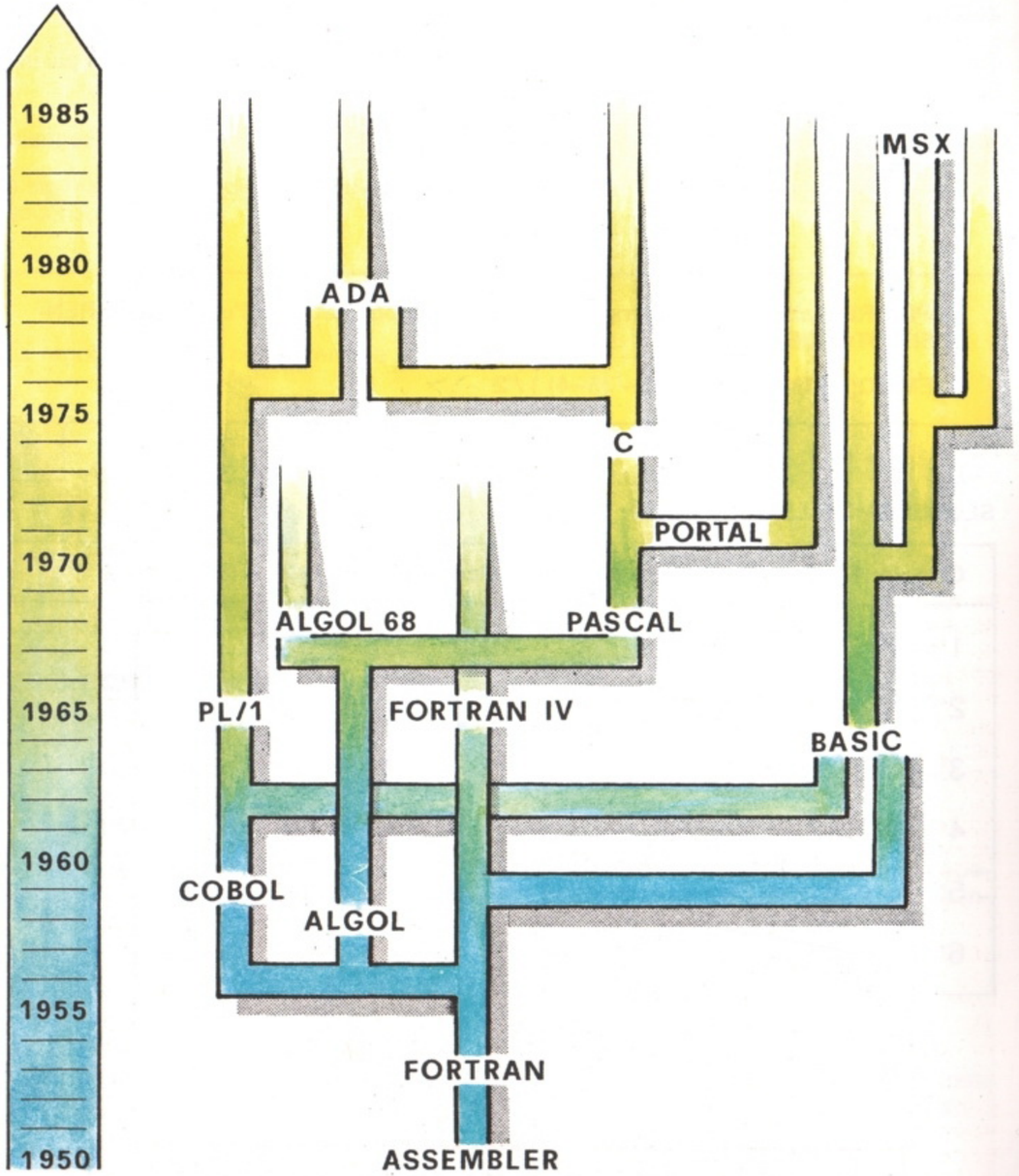
La PRIORITÀ attribuita dal computer può essere modificata con l'uso delle PARENTESI ROTONDE ( ).

ESEMPIO: PRINT 10* ((4+2) – (5–4)) / 2

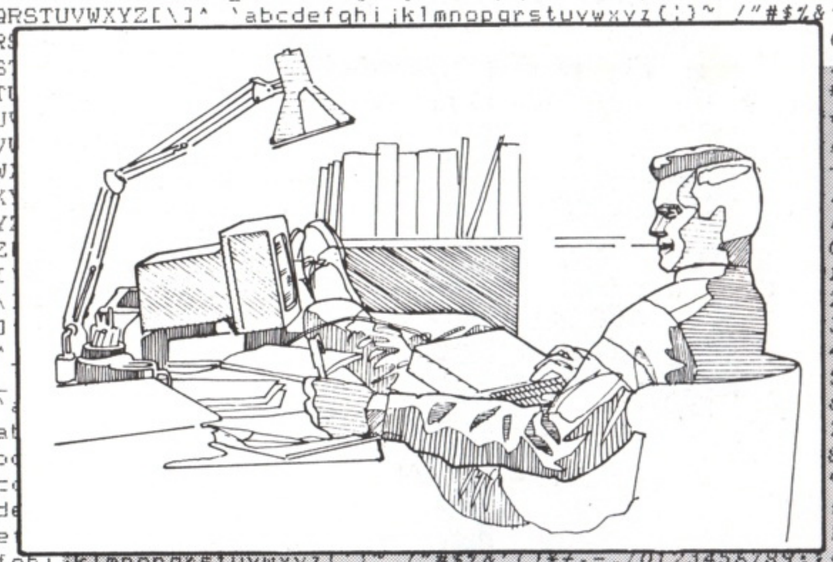
SEQUENZA DELLE OPERAZIONI:

Calcolo: 4 ↑ 2 + 12 – 6 / 3 + 4 *5	Calcolo: 10 * ((4 + 2) – (5 – 4)) / 2
1°) 4 ↑ 2 = 16	1°) (4 + 2) = 6
2°) 6 / 3 = 2	2°) (5 – 4) = 1
3°) 4 * 5 = 20	3°) 6 – 1 = 5
4°) 16 + 12 = 28	4°) 10 * 5 = 50
5°) 28 – 2 = 26	5°) 50 / 2 = 25
6°) 26 + 20 = 46	

L'EVOLUZIONE DEI LINGUAGGI DI PROGRAMMAZIONE



ESERCIZI



ESERCITAZIONI E STUDI DA ESEGUIRE A CASA:

1 Ripassare tutti gli argomenti trattati, rapportandoli al computer in dotazione.

2 Esercitarsi nell'uso della tastiera del proprio computer, con particolare studio dei tasti per l'editing, e per cancellare lo schermo.

3 Studio dei comandi di sistema del proprio computer:
a) per LISTARE sul video un programma (solitamente LIST);
b) per CANCELLARE un PROGRAMMA dalla MEMORIA (solitamente NEW);
c) per TRASFERIRE un PROGRAMMA dalla MEMORIA CENTRALE al NASTRO del registratore (solitamente SAVE oppure CSAVE);
d) per ESEGUIRE un PROGRAMMA ESISTENTE in MEMORIA CENTRALE (solitamente RUN);
e) per TRASFERIRE un programma DAL NASTRO del registratore, ALLA MEMORIA CENTRALE (solitamente LOAD oppure CLOAD);
f) per VERIFICARE un programma esistente in MEMORIA con quello memorizzato su nastro (solitamente VERIFY).

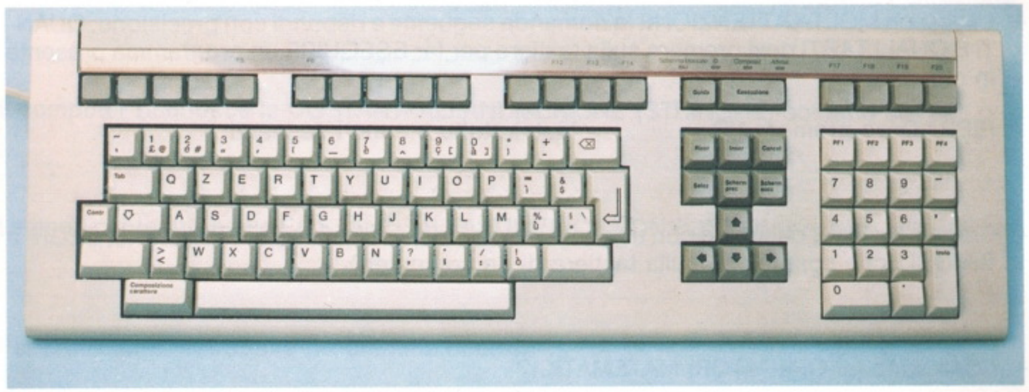
4 DIGITARE IL PROGRAMMA ALLEGATO con MOLTA ATTENZIONE in particolare per i seguenti segni: virgole, punto e virgola, segni speciali come $ (dollaro), parentesi, spaziature, (che devono essere scritte esattamente come nel LISTATO).

RICORDARSI di PREMERE il TASTO di IMMISSIONE (ENTER o RETURN), alla FINE DI OGNI RIGA DEL LISTATO.

5 Dopo l'inserimento di TUTTE le righe provate a:
- Fare ESEGUIRE il programma.
- CORREGGERE gli eventuali ERRORI (segnalati dal messaggio SYNTAX ERROR del computer).
- SALVARE il programma su NASTRO (quando è corretto e funziona).
- VERIFICARE il programma salvato.
- Dopo la VERIFICA spegnere il computer e provare a CARICARE dal nastro il programma, per ESEGUIRLO ancora.

Potrete incontrare delle difficoltà nella realizzazione delle operazioni elencate sopra, dovute SOPRATTUTTO alla vostra inesperienza, ma non scoraggiatevi per così poco.

Consultate attentamente i manuali, per quanto riguarda le modalità da seguire e PROVATE a ripetere da capo tutto quanto, finchè non diventerà più semplice utilizzare la TASTIERA del vostro computer ed i COMANDI di SISTEMA di cui sopra.



Leggete attentamente quanto segue, cercando di dare le risposte che ritenete più opportune. Se una domanda vi sembra esposta in modo poco chiaro vi preghiamo di segnalarlo indicando NON CHIARO come risposta.

– ELENCA almeno tre comandi "di sistema" del linguaggio BASIC spiegando molto brevemente a cosa servono.

PRINT >CRIVI ..... LIST LISTA .....
RUN ESEGUI ....

– ELENCA i tipi di costanti che si possono usare in BASIC facendo almeno un esempio per ogni tipo.

TIPO: ..... esempio: ..... TIPO: ..... esempio: .....

– INDICA la differenza tra COMANDI DIRETTI e COMANDI PROGRAMMATI:

.....

.....

– Leggi con MOLTA ATTENZIONE la domanda seguente e rispondi con precisione: QUANTI E QUALI TASTI devi premere sulla tastiera per far ESEGUIRE un programma presente in memoria?

RISP.: Devo premere NR ..... tasti in tutto e sono i seguenti:

.....

– Come possiamo chiamare con una sola parola l'insieme delle LETTERE, CIFRE, SEGNI e SIMBOLI che compaiono sulla tastiera di un computer?

.....

– Cosa sono gli OPERATORI MATEMATICI?

.....

– Scrivi il risultato che si ottiene inserendo questa istruzione nel computer:

PRINT 20 – 2 ↑ 2 * ((8 – 4)) + 12 / 4 .....

– Quale segnale si deve usare per indicare al computer una COSTANTE ALFANUMERICA?

.....

Leggete attentamente quanto segue, cercando di dare le risposte che ritenete più opportune. Se una domanda vi sembra esposta in modo poco chiaro vi preghiamo di segnalarlo indicando NON CHIARO come risposta.

COMPLETA CON PAROLE TUE LE SEGUENTI FRASI:

Scrivere un programma BASIC vuol dire .....

.....

Cancellando lo schermo ..... il programma in memoria.

Per muovere il CURSORE sul video occorre utilizzare .....

Il computer esegue i COMANDI DIRETTI quando .....

Il computer esegue le righe di un programma quando .....

Per vedere il risultato della divisione 50/10 è necessario scrivere la seguente istruzione

.....

Il computer riconosce le COSTANTI ALFANUMERICHE (STRINGHE) soltanto se .....

I numeri contenuti in una COSTANTE ALFANUMERICA NON POSSONO essere utilizzati per

.....

Per cancellare una riga di programma occorre .....

Per cancellare TUTTE le righe di un programma occorre .....

Quando il computer è PRONTO per ricevere comandi scrive ..... sul video

Il CURSORE lampeggiante sul video indica il posto in cui verrà .....

Per trasferire un programma, DALLA memoria centrale, ad una MEMORIA DI MASSA, è necessario dare il comando di sistema .....

Inserire istruzioni numerate nella memoria significa .....

```
1 REM ************************
2 REM ***  TOTO  RANDOM   ***
3 REM ************************
4 REM ** SCUOLA SCHEIDEGGER *
5 REM ************************
6 REM **PER COMMODORE VIC 20*
7 REM **  E COMMODORE 64   **
8 REM ************************
10 TL$=CHR$(147)+"TOTO RANDOM"
20 PRINT TL$ : PRINT
25 PRINT CHR$(19);CHR$(17);CHR$(17);
30 PRINT"QUANTE X ? ";
45 GOSUB 1000 :NX=VAL(R$)
50 IFNX<0 OR NX >13 OR INT(NX)<>NX THEN25
55 PRINT CHR$(19);
56 FOR I = 1 TO 5 : PRINT CHR$(17);
57 NEXT I
60 PRINT"QUANTI 1 ? ";
65 NM=13-NX
75 GOSUB 1000 :N1=VAL(R$)
80 IFN1<0 OR N1 >NM THEN55
85 N2 = 13 - N1 - NX
90 A=N1 : B=N2 : C=NX :TT=0
95 PRINT TL$ :PRINT
100 FOR J=1 TO 13
110 N$ = STR$(J) : R$= RIGHT$(N$,2)
120 PRINT "PARTITA ";N$;
130 N=INT(RND(1)*3)
140 IF N = 1 AND N1 = 0 THEN 130
141 IF N = 2 AND N2 = 0 THEN 130
142 IF N = 0 AND NX = 0 THEN 130
144 IFN=0THENPRINTTAB(11+TT)" X":NX=NX-1
145 IFN=1THENPRINTTAB(11+TT)" 1":N1=N1-1
146 IFN=2THENPRINTTAB(11+TT)" 2":N2=N2-1
150 NEXT : PRINT
160 W$="":PRINT"ALTRA COLONNA ? (S/N)";
170 GETW$:IFW$="N"THENPRINTCHR$(147):END
175 IF W$<>"S" THEN 170
180 N1=A : N2 = B : NX = C
190 TT=TT+2:IFTT>7THEN TT=0 :GOTO95
200 PRINT CHR$(19);CHR$(17);CHR$(17);
210 GOTO 100
1000 R$=""
1020 POKE204,0:GET A$:IFA$=""THEN1020
1022 IF A$=CHR$(20)ANDLEN(R$)>0THEN1070
1024 POKE204,1
1025 IFA$=CHR$(13)THENPRINTCHR$(32):RETURN
1030 IF A$ < "0" OR A$ > "9" THEN 1020
1035 IF LEN(R$)>1 THEN 1020
1040 PRINT A$;
1050 R$=R$+A$
1060 GOTO 1020
1070 PRINTA$;:R$=RIGHT$(R$,LEN(R$)-1)
1071 GOTO1020

READY.
```



```
1 REM ************************
2 REM **   TOTO  RANDOM   **
3 REM ************************
4 REM ** PER COMPUTERS    **
5 REM ** LASER 110 E 310  **
6 REM ************************
7 REM
8 REM    SCUOLA SCHEIDEGGER
9 REM
10 CLS
20 TL$="TOTO RANDOM"
30 S$="QUANTE X"
40 GOSUB 1000
50 NX=VAL(A$):IF NX>13 THEN 40
60 MAX=13-NX
70 S$="QUANTI 1"
80 GOSUB 1000
90 N1=VAL(A$):IF N1>MAX THEN 80
100 N2=13-N1-NX
110 A=N1:B=N2:C=NX
120 CLS : PRINT TL$
130 FOR I = 1 TO 13
140 I$ = CHR$(32)+STR$(I) : I$ = RIGHT$(I$,2)
150 PRINT"PARTITA ";I$;CHR$(32);
160 N=RND(3) : NA=RND(100)
165 IF N=2 AND NA > 25 THEN 160
170 IF N=1 AND N1=0 THEN 160
180 IF N=2 AND N2=0 THEN 160
190 IF N=3 AND NX=0 THEN 160
200 IF N=1 THEN PRINT " 1" : N1=N1-1
210 IF N=2 THEN PRINT " 2" : N2=N2-1
220 IF N=3 THEN PRINT " X" : NX=NX-1
230 NEXT
240 PRINT@448,"VUOI ALTRA COLONNA"
250 INPUT"(S/N)";R$
260 IF R$ <> "S" AND R$ <> "N" THEN 240
270 IF R$ = "N" THEN CLS : END
280 N1 = A : N2 = B : NX = C
290 GOTO 120
1000 CLS : PRINT TL$
1010 PRINT@64,S$
1020 INPUT"PREVEDI IN SCHEDINA";A$
1030 IF ASC(A$) > 57 OR ASC(A$)<48 THEN 1000
1040 RETURN

READY.
```

```
1 REM ************************
2 REM **   TOTO  RANDOM   ***
3 REM ************************
4 REM **  PER  COMPUTERS  ***
5 REM **    LASER  500    ***
6 REM ************************
7 REM
8 REM   SCUOLA SCHEIDEGGER
9 REM
10 CLS
20 TL$="** TOTO RANDOM **"
30 S$="QUANTE X"
40 GOSUB 1000
50 NX=VAL(A$):IF NX>13 THEN 40
60 MAX=13-NX
70 S$="QUANTI 1"
80 GOSUB 1000
90 N1=VAL(A$):IF N1>MAX THEN 80
100 N2=13-N1-NX
110 A=N1:B=N2:C=NX
120 CLS:PRINT TAB(11);TL$:PRINT :PRINT
130 FOR I=1 TO 13
140 I$=CHR$(32)+STR$(I):I$=RIGHT$(I$,2)
150 PRINT TAB(14);"PARTITA ";I$;CHR$(32);
160 RANDOMIZE:N=INT(RND(1)*3)
170 IF N=0 AND N1=0 THEN 160
180 IF N=1 AND N2=0 THEN 160
190 IF N=2 AND NX=0 THEN 160
200 IF N=0 THEN PRINT "1":N1=N1-1
210 IF N=1 THEN PRINT "2":N2=N2-1
220 IF N=2 THEN PRINT "X":NX=NX-1
230 NEXT
240 PRINT CHR$(27);CHR$(161);CHR$(39);CHR$(52);
250 INPUT "VUOI ALTRA COLONNA (S/N)";R$
260 IF R$<>"S" AND R$<>"N" THEN 240
270 IF R$="N" THEN CLS:END
280 N1=A:N2=B:NX=C
290 GOTO 120
1000 CLS:PRINT TL$
1010 PRINT :PRINT S$:PRINT
1020 INPUT "PREVEDI IN SCHEDINA";A$
1030 IF ASC(A$)>57 OR ASC(A$)<48 THEN 1000
1040 RETURN
```

LEZIONE 2

OBIETTIVI

1. Fornire i concetti relativi all'ingresso e all'uscita dei dati.
2. Chiarire cosa sono le variabili e come sono rappresentate nella memoria del computer.
3. Far conoscere i diversi tipi di variabili ed il loro uso, escludendo le variabili con indice (che saranno spiegate in seguito).
4. Definire l'uso dell'istruzione LET ed il concetto di assegnazione, con l'uso delle variabili numeriche e degli operatori matematici.
5. Chiarire l'uso dell'istruzione REM e dei segni di sintassi (, ; :).
6. Approfondire l'uso delle istruzioni di assegnazione in generale, e dell'istruzione INPUT nelle sue diverse forme.
7. Fornire le tecniche complementari di programmazione (pulizia schermo, controllo dell'INPUT, prompting).
8. Fornire le basi per la risoluzione di problemi elementari, attraverso l'uso dello pseudo-linguaggio.
9. Applicare le nozioni acquisite per la risoluzione di un problema specifico, con il computer ed il linguaggio BASIC.

1 FORNIRE I CONCETTI RELATIVI ALL'INGRESSO E USCITA DATI

Cerchiamo di capire come operano i diversi comandi e istruzioni nei confronti della MEMORIA CENTRALE.

Esistono TRE tipi di operazioni possibili:

1. Operazioni di INPUT, che permettono di INSERIRE informazioni NELLA memoria centrale.
2. Operazioni di OUTPUT, con cui è possibile RICEVERE informazioni DALLA memoria centrale.
3. Operazioni di ELABORAZIONE e di CALCOLO che avvengono DENTRO la memoria centrale.

Facciamo qualche esempio:

Premendo il tasto ENTER o RETURN sulla tastiera si effettua un INSERIMENTO cioè una operazione di INPUT nella CPU.

I messaggi READY. e SINTAX ERROR sono inviati DALLA CPU, perciò sono operazioni di OUTPUT. Anche i comandi che noi introduciamo nella memoria fanno eseguire operazioni di INGRESSO, o di USCITA, o di ELABORAZIONE:

Il comando RUN ordina di ESEGUIRE il programma residente in memoria, perciò genera una operazione di ELABORAZIONE.

Il comando SAVE trasferisce il programma DALLA memoria centrale ad una memoria di massa, perciò genera una operazione di OUTPUT.

L'istruzione PRINT scrive sul video qualcosa, prelevandolo dalla memoria centrale, quindi genera una operazione di OUTPUT.

Il comando LOAD trasferisce un programma DA una memoria di massa, ALLA memoria centrale, quindi esegue una operazione di INPUT.

È MOLTO IMPORTANTE CAPIRE la funzione di ogni operazione, per capire come opera il computer.

2 CHIARIRE COSA SONO LE VARIABILI E COME SONO RAPPRESENTATE NELLA MEMORIA DEL COMPUTER

Cerchiamo di capire come è fatta la memoria di un computer, per comprendere le operazioni di ELABORAZIONE.

Ci sono due tipi di memoria: una per il software di base, ed un'altra, che in parte è utilizzata dal Sistema Operativo, ed in parte è a nostra disposizione per programmi ed altro, come vedremo.

La memoria del software di base NON può essere modificata, in quanto viene programmata dai costruttori.

Per questo motivo si chiama MEMORIA di sola LETTURA, oppure con la sigla ROM (Read Only Memory).

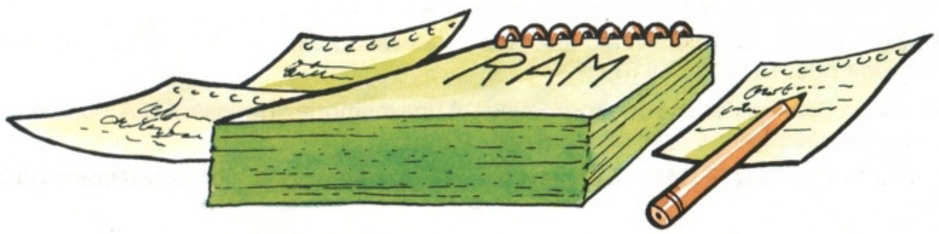
La memoria utilizzata dal Sistema Operativo e la memoria a nostra disposizione, invece POSSONO essere modificate, e prendono il nome di memoria RAM (Random Access Memory), cioè Memoria ad accesso casuale.

Quando inseriamo le righe di un programma il sistema operativo trasforma le nostre istruzioni in numeri e mette questi numeri nella memoria RAM, occupandone una parte.

ROM	RAM s.o.	RAM utente

Noi possiamo usare la memoria RAM anche per conservare dei dati, (numeri o anche parole) se abbiamo bisogno di DEPOSITARE questi DATI momentaneamente.

Possiamo usare la memoria RAM come un foglio su cui scrivere le annotazioni che ci interessano.



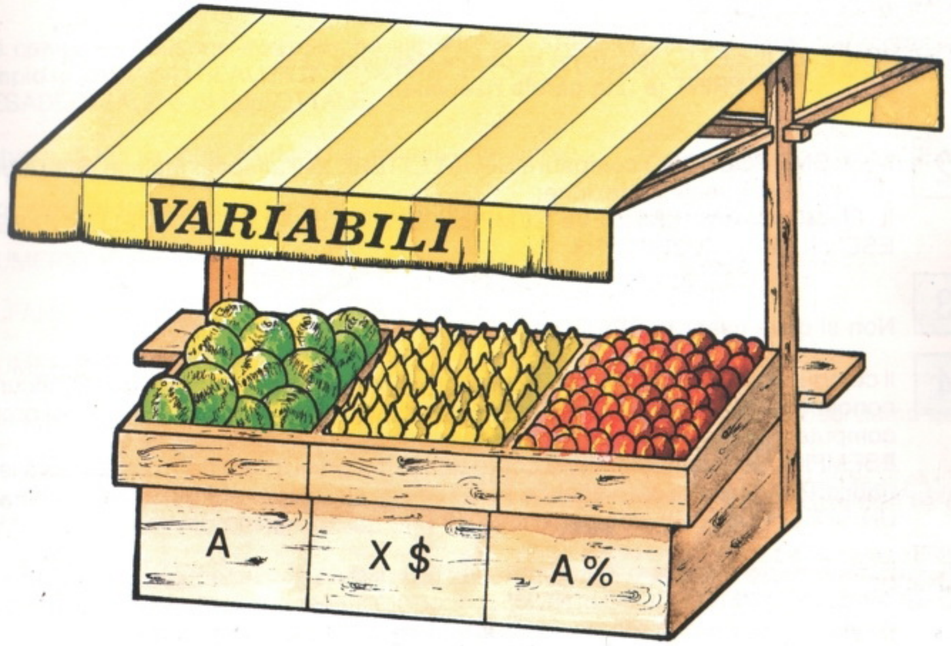
Per usufruire di questa possibilità occorre però seguire delle regole MOLTO SEMPLICI, ma anche MOLTO precise.

1. Per riservare un posto nella memoria, allo scopo di conservare dei dati, occorre DARE UN NOME al dato che vogliamo depositare, (vedremo in seguito come fare).

2. Dopo aver dato il nome, occorre ASSEGNARE UN VALORE al dato.

3. In qualsiasi momento potremo usare QUEL DATO per dei calcoli, oppure potremo farlo scrivere sul video con l'istruzione PRINT, o cambiargli valore ma, per fare queste operazioni, dovremo chiamarlo con il SUO NOME.

I DATI che possiamo depositare nella memoria RAM prendono il nome di VARIABILI in quanto, come già detto, possiamo MODIFICARE il loro valore come più ci piace.



3 FAR CONOSCERE I DIVERSI TIPI DI VARIABILI

Vediamo ora le regole da seguire per DARE UN NOME alle variabili.

Bisogna precisare che le regole possono cambiare da un computer all'altro, ma ci sono regole COMUNI a tutti i computer.

1 Il NOME di una VARIABILE può essere una PAROLA QUALSIASI, a nostra scelta, purchè NON si utilizzino le PAROLE RISERVATE del linguaggio di programmazione. Il nome usato NON PUÒ NEMMENO CONTENERE una parola RISERVATA.

ESEMPI:

NOMI VALIDI: A, IV, LIRE

NOMI NON VALIDI: LETTO (contiene la parola riservata LET)
PRINT (è una parola riservata).

2 Nel NOME POSSONO comparire SIA le LETTERE dell'alfabeto inglese, SIA i NUMERI, con la seguente eccezione:
IL PRIMO CARATTERE DEVE SEMPRE ESSERE UNA LETTERA DELL'ALFABETO;
ESEMPI: A, A1, A12BC, NR3.

3 Non si deve usare lo SPAZIO nel NOME.

4 Il computer riconosce SOLTANTO un certo numero di caratteri del NOME, (alcuni riconoscono SOLO 2 CARATTERI), per cui bisogna consultare, il manuale del proprio computer, o fare delle prove, per sapere quanti caratteri riconosce.
ESEMPIO: Se il computer tiene conto SOLO dei primi 2 caratteri, considererà la variabile TEMA uguale alla variabile TELO ed uguale alla variabile TE, per cui riserverà UN SOLO POSTO invece dei TRE posti che noi vorremmo riservare.

5 È necessario segnalare al computer il TIPO di dato che inseriremo, vale a dire che IMMEDIATAMENTE DOPO IL NOME, occorrerà mettere un simbolo, per precisare se vogliamo riservare posto per un numero, per una parola, o altro.

I SIMBOLI che indicano il TIPO della VARIABILE possono cambiare da un computer all'altro, ma SOLITAMENTE sono i seguenti:

1 NESSUN SIMBOLO per indicare VARIABILI di TIPO NUMERICO REALE, cioè numeri POSITIVI o NEGATIVI, con parte intera e decimale.
ESEMPI di NOMI validi per VARIABILI NUMERICHE REALI: A, A1, NUMERO.
ESEMPI di NUMERI REALI: –12.25, 357.00.
NOTA: Il linguaggio BASIC richiede il punto e NON la virgola per separare la parte intera, dalla parte decimale di un numero.

2 Il SIMBOLO $ (dollaro) per indicare VARIABILI di TIPO ALFANUMERICO cioè che CONTENGONO PAROLE o (in generale) CARATTERI ALFANUMERICI;
ESEMPI di NOMI di VARIABILE ALFANUMERICA: A$, NOME$, INDIRIZZO$;
ESEMPI di VARIABILI ALFANUMERICHE: "CIAO", "AZ 732/QB **".

3 Il SIMBOLO % (percento) per indicare VARIABILI di TIPO NUMERICO INTERO cioè NUMERI (positivi o negativi) che hanno SOLO la parte INTERA.
ESEMPI di NOMI di VARIABILI INTERE: A%, COD%, NR%;
ESEMPI di NUMERI INTERI: 123, –12505, 22100.
Solitamente una VARIABILE INTERA può assumere un valore massimo di 32767 ed un valore minimo di –32768.

Molti computer possono avere anche altri TIPI di VARIABILE per usi particolari come ad esempio le VARIABILI REALI IN DOPPIA PRECISIONE, o le VARIABILI di tipo BINARIO o di tipo ESADECIMALE o di tipo OTTALE.

Lo studio di tali variabili esula dagli obiettivi del corso.

Riferendoci ai tre tipi di variabili spiegati, cioè:

– NUMERICHE REALI

– ALFANUMERICHE

– NUMERICHE INTERE

abbiamo visto che quando attribuiamo il NOME, e dichiariamo il TIPO, ci riserviamo uno spazio della MEMORIA RAM in cui depositare i dati che ci interessano.

Questi dati potremo riprenderli, per eseguire qualunque operazione, SEMPLICEMENTE CHIAMANDOLI CON IL LORO NOME.

Esiste anche la possibilità di riservare spazio per TABELLE complete di dati, per mezzo di variabili chiamate VARIABILI CON INDICE.

Tale possibilità sarà spiegata nelle prossime lezioni del corso, in quanto è meglio approfondire lo studio delle VARIABILI SEMPLICI, prima di affrontare le VARIABILI CON INDICE (VETTORI e TABELLE).

4 SPIEGARE L'USO DELL'ISTRUZIONE LET ED IL CONCETTO DI ASSEGNAZIONE CON LE VARIABILI NUMERICHE E OPERATORI MATEMATICI

Vedremo ora come ASSEGNARE un certo valore ad una VARIABILE.

Possiamo procedere sia nel MODO DIRETTO, sia scrivendo un programma.

Occorre utilizzare una ISTRUZIONE BASIC.

TUTTE le ISTRUZIONI che permettono di ASSEGNARE un valore ad una VARIABILE vengono chiamate ISTRUZIONI DI ASSEGNAZIONE.

La prima di queste istruzioni è l'istruzione LET.

LET

ESEMPIO: se vogliamo depositare il numero 135, chiamandolo col nome NUMERO, sarà sufficiente scrivere:

LET NUMERO = 135

premendo il tasto ENTER o RETURN per trasmettere l'istruzione.

Il computer eseguirà l'istruzione e risponderà READY.

Per verificare se il computer ha VERAMENTE depositato il numero, possiamo chiedere di scrivere sul video il valore della VARIABILE che abbiamo chiamato NUMERO, con l'istruzione:

PRINT NUMERO

(ricordandosi di premere il tasto ENTER o RETURN dopo l'istruzione).

Il computer eseguirà il comando, vale a dire scriverà sul video il numero 135 che noi avevamo depositato con l'istruzione LET.

Il significato dell'istruzione LET che abbiamo scritto sopra è il seguente: LASCIA un posto nella MEMORIA, chiama questo posto NUMERO ed ASSEGNAGLI il valore 135.
L'istruzione PRINT, che abbiamo scritto, invece deve essere interpretata così: SCRIVI SUL VIDEO il numero che trovi nella memoria, nel posto che io ho chiamato NUMERO.

Dopo questa verifica sappiamo che il computer si ricorda il numero 135, e noi potremo usarlo per dei calcoli, semplicemente CHIAMANDOLO PER NOME.

ESEMPIO: Per scrivere il prodotto 135*10 potremo scrivere:

PRINT NUMERO*10 (più il tasto ENTER o RETURN) il computer scriverà 1350, cioè eseguirà l'operazione richiesta.

Possiamo anche CAMBIARE valore alla VARIABILE che abbiamo chiamato NUMERO, usando ancora l'istruzione LET;

ESEMPIO: LET NUMERO = 200 (più tasto ENTER o RETURN);

Dopo l'istruzione LET NUMERO = 200 il computer si ricorderà il nuovo valore (200) al posto del vecchio valore (135).

Possiamo cambiare valore alla variabile in altri modi: aggiungendo o togliendo una certa quantità o con altre operazioni, come negli esempi che seguono:

ESEMPIO 1: se la variabile NUMERO ha il valore 200 e vogliamo aggiungere 300 (cioè fare la somma 200 + 300 e memorizzarla nella variabile NUMERO) scriveremo la seguente ASSEGNAZIONE: LET NUMERO = numero + 300 che significa: LASCIA che il posto chiamato NUMERO prenda un nuovo valore: questo valore è dato dalla somma del contenuto ATTUALE del posto chiamato NUMERO (cioè 200) più la COSTANTE NUMERICA 300.

DOPO QUESTA ISTRUZIONE LA VARIABILE NUMERO AVRÀ IL VALORE 500.

ESEMPIO 2: se la variabile NUMERO vale 500 e noi scriviamo: LET NUMERO = NUMERO/5 (più tasto ENTER o RETURN). Il valore della variabile NUMERO sarà cambiato da 500 a 100 (cioè 500/5).

ESEMPIO 3: se la variabile NUMERO vale 100 e scriviamo: LET NUMERO = 7*5+NUMERO (più tasto ENTER o RETURN). Il valore di NUMERO diventa 135 (cioè 7*5+100).

ISTRUZIONI	VALORI DEPOSITATI IN MEMORIA RAM
LET NUMERO = 135	NUMERO = 135
LET NUMERO = 200	NUMERO = 200
LET NUMERO = NUMERO + 300	NUMERO = 500
LET NUMERO = NUMERO/5	NUMERO = 500/5 = 100
LET NUMERO = 7 * 5 + NUMERO	NUMERO = 35 + 100 = 135

Un'altra cosa importante è la seguente:

Se vogliamo depositare il risultato di una operazione, senza cambiare il valore della variabile chiamata NUMERO, possiamo dire al computer di mettere il valore che ci interessa da un'altra parte.

Naturalmente dobbiamo dare anche un altro NOME, cioè dobbiamo riservare posto ad un'altra VARIABILE.

ESEMPIO: se la variabile NUMERO vale 135 e noi vogliamo depositare il risultato dell'operazione 135/5 possiamo scrivere:

LET N2 = NUMERO/5 (più tasto ENTER o RETURN).

Questa istruzione può essere tradotta così:

LASCIA un posto nella MEMORIA, chiamalo N2, ed ASSEGNAGLI il valore della variabile NUMERO diviso per la COSTANTE NUMERICA 5.

Dopo l'esempio di cui sopra potremo verificare i risultati chiedendo di scrivere il contenuto della variabile NUMERO, e quello della variabile N2 con le seguenti istruzioni:

PRINT NUMERO (più tasto ENTER o RETURN)

PRINT N2 (più tasto ENTER o RETURN).

Il computer scriverà sul video i due numeri (cioè 135 e 27).

Vediamo ora di usare quanto appreso per la scrittura di un programma:

DIGITARE IL PROGRAMMA CHE SEGUE RICORDANDOSI DI PREMERE IL TASTO ENTER (o RETURN) ALLA FINE DI OGNI RIGA.

Per prima cosa diamo il comando di sistema NEW per cancellare eventuali programmi esistenti.

```
NEW
```

Poi inseriamo le righe di programma con una numerazione crescente, come segue:

```
 10 LET N1 = 100
 20 LET N2 = 50
 30 LET N3 = N1 * N2
 40 PRINT "PRIMO NUMERO"
 50 PRINT N1
 60 PRINT "SECONDO NUMERO"
 70 PRINT N2
 80 LET R$ = "IL PRODOTTO TRA I DUE NUMERI È"
 90 PRINT R$
100 PRINT N3
```

Dopo aver controllato bene il programma, CANCELLIAMO lo schermo (con gli appositi tasti della tastiera, o con l'istruzione CLS, se il computer la possiede).

```
CLS
```

CANCELLANDO LO SCHERMO NON SI CANCELLA IL PROGRAMMA DALLA MEMORIA MA SI CANCELLA SOLO LO SCHERMO.

Per verificare ancora il programma inserito diamo il comando di sistema LIST, (più tasto ENTER o RETURN).

```
LIST
```

Il computer scriverà sul video il programma che ha in memoria.

Dopo aver controllato che il programma sia IDENTICO a quello che VOLEVAMO INSERIRE, possiamo eseguire ANCORA la PULIZIA DELLO SCHERMO e poi dare il comando di sistema che ordina al computer l'ESECUZIONE DEL PROGRAMMA.

Digitiamo perciò RUN (più tasto ENTER o RETURN).

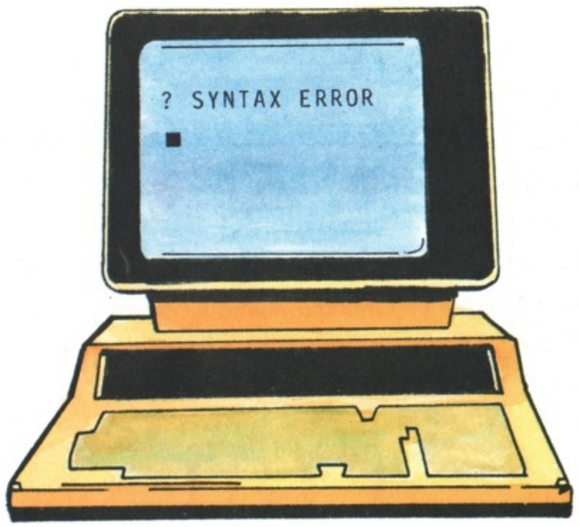
```
RUN
```

Il computer eseguirà TUTTE le istruzioni ESATTE che ha in memoria, partendo dalla prima (cioè la nr. 10) fino all'ultima.

Se (NONOSTANTE i nostri controlli) ci sono istruzioni errate, l'esecuzione si interromperà con un MESSAGGIO:? SINTAX ERROR indicando anche il numero della riga errata.

Se non ci sono errori il computer eseguirà CORRETTAMENTE il programma.

```
? SYNTAX ERROR
```



5 CHIARIRE L'USO DELL'ISTRUZIONE REM E DEI SEGNI DI SINTASSI

Vediamo ora di commentare il programma scritto, in base alle informazioni in nostro possesso, per apprendere altri concetti.

Per prima cosa SAPPIAMO che è un programma BASIC (e non COMANDI DIRETTI) in quanto le istruzioni sono numerate in modo progressivo.

Inoltre possiamo identificare:

- l'istruzione PRINT alle righe 40,50,60,70,90 e 100
- l'istruzione LET alle righe 10,20,30 e 80
- l'operatore matematico * (per) alla riga 30
- i NOMI di VARIABILI NUMERICHE REALI (N1, N2, N3) alle righe 10,20,30, ed alle righe 50, 70 e 100.
- le COSTANTI NUMERICHE 100 e 50 utilizzate alle righe 10 e 20 per assegnare un valore alle variabili N1 ed N2.
- il NOME R$ di una VARIABILE ALFANUMERICA alle righe 80 e 90
- la COSTANTE ALFANUMERICA "IL PRODOTTO TRA I DUE NUMERI È", usata alla riga 80 per ASSEGNARE un valore alla VARIABILE ALFANUMERICA R$
- le COSTANTI ALFANUMERICHE "PRIMO NUMERO" e "SECONDO NUMERO" nelle righe 40 e 60, utilizzate con l'istruzione PRINT che le scrive sul video.

Se ora vogliamo commentare con poche parole le diverse operazioni che il programma prevede, possiamo fare i seguenti commenti:

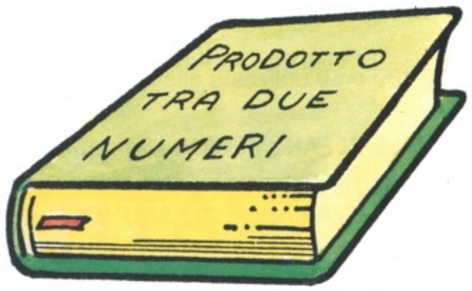
RIGA 10: Assegna il valore 100 alla variabile N1
RIGA 20: Assegna il valore 50 alla variabile N2
RIGA 30: Esegue il prodotto N1 * N2 assegnandolo alla variante N3
RIGA 40: Scrive un messaggio sul video
RIGA 50: Scrive sul video il valore della variabile N1 (cioè 100)
RIGA 60: Scrive sul video un altro messaggio
RIGA 70: Scrive il valore della variabile N2
RIGA 80: Assegna alla variabile R$ un valore ALFANUMERICO
RIGA 90: Scrive il valore contenuto nella variabile R$
RIGA 100: Scrive il contenuto della variabile N3.

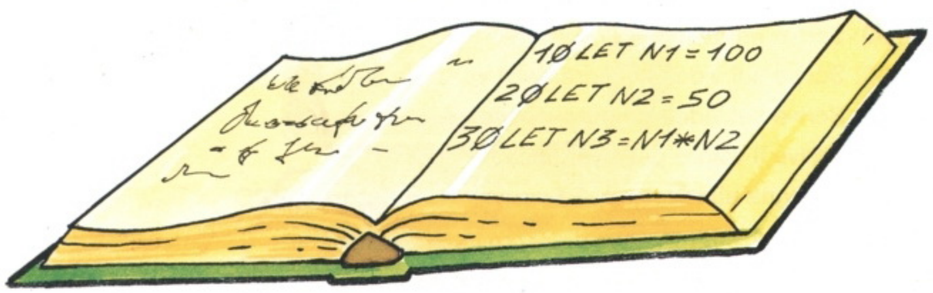
Inoltre possiamo raggruppare tutte le operazioni di cui sopra con un commento unico, che spiega quale è la FUNZIONE PRINCIPALE del programma, cioè possiamo dire:

QUESTO PROGRAMMA ESEGUE IL PRODOTTO TRA DUE NUMERI.

Questo commento serve anche per dare un TITOLO al nostro programma.

Anche se è sempre bene COMMENTARE tutte le RIGHE di un programma, (soprattutto al nostro livello) per abituarsi ad interpretare nel modo CORRETTO le diverse istruzioni, è evidente che possiamo anche RAGGRUPPARE un certo numero di righe, attribuendo un unico COMMENTO per identificare con un TITOLO quello che il programma ESEGUE.





Vediamo ora come è possibile inserire il TITOLO al nostro programma:

Il BASIC ha una ISTRUZIONE che permette di inserire commenti: questa istruzione è REM (abbreviazione di REMark = commento).

REM

L'uso dell'istruzione REM è molto semplice: quando vogliamo inserire un COMMENTO QUALSIASI, in un programma, è sufficiente scrivere una RIGA di programma con l'istruzione REM, scrivendo il nostro commento DOPO l'istruzione REM.

Nel programma precedente potremo quindi INSERIRE una NUOVA RIGA con il titolo come segue:

5 REM PRODOTTO TRA DUE NUMERI

Dobbiamo dare un numero di riga 5 per fare in modo che il computer metta questa RIGA, prima di tutte le altre, cioè prima della PRIMA RIGA, che nel nostro programma era la numero 10.

Se proviamo a LISTARE sul video il programma (con il comando LIST) possiamo vedere che la riga 5 è stata messa al posto giusto.

Provando ad ESEGUIRE il programma (con il comando RUN) possiamo anche constatare che il programma NON HA CAMBIATO ESECUZIONE, (cioè fa le stesse cose che faceva prima).

Ciò significa che l'istruzione REM non MODIFICA un programma ma serve SOLAMENTE a noi, per inserire dei commenti.

Vedremo ora alcuni simboli che il computer usa in modo particolare: questi simboli sono chiamati SEGNI DI SINTASSI perchè aiutano, nella scrittura di programmi, ed alle volte, possono MODIFICARE l'effetto di alcune istruzioni come vedremo più avanti.

I segni di sintassi sono SOLITAMENTE i seguenti:

1 : due punti

2 , virgola

3 ; punto e virgola

Vediamo, per ora, solo qualcuna delle loro applicazioni:

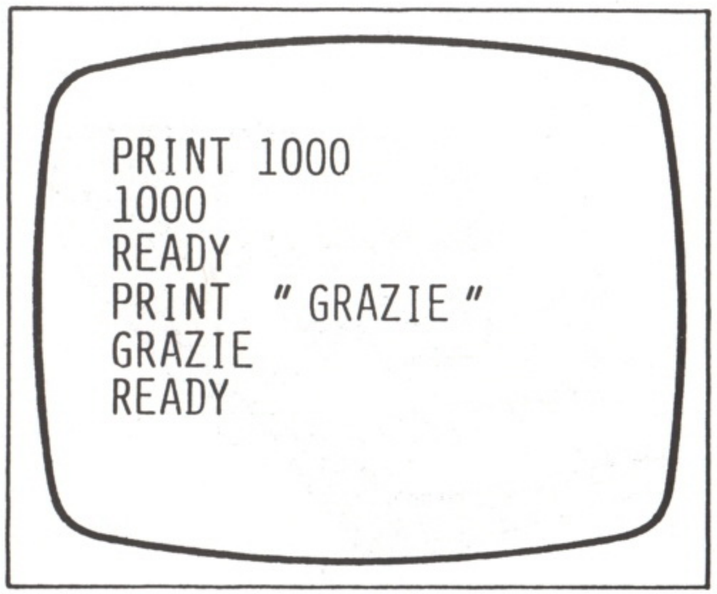
Il segno : (due punti) indica al computer che c'è un'altra istruzione in una RIGA di PROGRAMMA.

Ciò significa che possiamo scrivere più di un'ISTRUZIONE per riga, purchè separiamo le diverse istruzioni con il segno due punti.

USIAMO IL MODO DIRETTO PER NON MODIFICARE IL PROGRAMMA ESISTENTE.

ESEMPIO: (in modo diretto) PRINT 1000 : PRINT "GRAZIE"

il computer eseguirà TUTTE e due le PRINT indicate.



SEGNI DI SINTASSI

Il segno , (virgola) può essere usato con l'istruzione PRINT, per scrivere, CONTEMPORANEAMENTE, più cose sul video.

ESEMPIO: (in modo diretto)
PRINT 1000, "GRAZIE", "A TUTTI"

Il computer scriverà contemporaneamente le tre COSTANTI indicate separandole una dall'altra.

La virgola può essere usata come separatore per MOLTI altri scopi: avremo modo di apprenderli nelle lezioni future.

Il segno ; (punto e virgola) può essere utilizzato, con l'istruzione PRINT, per scrivere sulla stessa riga più cose, SENZA SEPARAZIONE, cioè una in fila all'altra.

ESEMPIO: (modo diretto)
PRINT 1000; "GRAZIE"; "A TUTTI"

il computer scrive le tre COSTANTI sulla stessa riga, senza separazione, come detto.

Ora proviamo a modificare il programma precedente, usando i segni di sintassi spiegati:

UTILIZZARE L'EDITING DI PROGRAMMA PER CANCELLARE, o MODIFICARE, o per CAMBIARE NUMERO ALLE RIGHE.

EDITING DI PROGRAMMA

OPERAZIONE	ESECUZIONE
1 Cancellare una riga	Digitare il numero della riga e premere RETURN
2 Modificare una riga	Usare tasti di movimento cursore ed inserire o cancellare per mezzo dei tasti di inserimento e cancellazione. RETURN Oppure: digitare una riga nuova con il numero della riga vecchia e premere RETURN
3 Cambiare numero ad una riga	Posizionarsi sulla riga con i tasti cursore e cambiare numero alla riga. Ricordarsi di premere RETURN Nota: cancellare la riga vecchia come spiegato in 1

Il nuovo programma dovrà essere come quello che segue:

```
 5 REM PRODOTTO TRA DUE NUMERI
10 LET N1 = 100 : LET N2 = 50 : LET N3 = N1 * N2
20 PRINT "PRIMO NUMERO"; N1
30 PRINT "SECONDO NUMERO"; N2
40 LET R$ = "IL PRODOTTO TRA I DUE NUMERI È"
50 PRINT R$; N3
```

Possiamo notare come l'uso dei segni di sintassi ci ha permesso una SEMPLIFICAZIONE della scrittura del programma.

Inoltre se facciamo eseguire il programma, con il comando RUN, potremo notare anche un MIGLIORAMENTO della esecuzione, perchè i valori, scritti sul video, si possono leggere meglio, in quanto si trovano sulla stessa riga.

6 APPROFONDIRE L'USO DELLE ISTRUZIONI DI ASSEGNAZIONE IN GENERALE E DELL'ISTRUZIONE INPUT NELLE DIVERSE FORME

Il programma scritto precedentemente pur elaborando dei dati non ci permette di INTERVENIRE in prima persona per cambiare il valore alle VARIABILI N1 ed N2 per eseguire una moltiplicazione tra due numeri qualsiasi a nostra scelta.

Ciò significa che se volessimo calcolare un prodotto tra due altri numeri, dovremmo MODIFICARE le istruzioni LET N1 = 100 e LET N2 = 50 della riga 10, cambiando ogni volta, i valori ASSEGNATI alle VARIABILI N1 ed N2.

Questo modo di operare non è molto comodo e nemmeno utile.

Il linguaggio BASIC possiede una istruzione di ASSEGNAZIONE, che permette di MODIFICARE il VALORE delle VARIABILI, senza dover modificare il programma ogni volta.

L'istruzione è INPUT (trad. = METTI DENTRO), e deve SEMPRE ESSERE ASSOCIATA al NOME della VARIABILE che vogliamo modificare.

INPUT

ESEMPIO: per modificare il programma precedente possiamo CANCELLARE le righe 10, 20, 30 e riscriverle nel seguente modo:

```
10 PRINT "PRIMO NUMERO"; : INPUT N1
20 PRINT "SECONDO NUMERO"; : INPUT N2
30 LET N3 = N1 * N2
```

OCCORRE FARE MOLTA ATTENZIONE AI SEGNI DI SINTASSI, (punto e virgola e due punti).

Come potremo verificare, facendo ESEGUIRE il programma, il computer scriverà sul video la richiesta:

PRIMO NUMERO? ed attenderà che noi inseriamo un numero.
Il numero inserito verrà ASSEGNATO alla VARIABILE N1; poi il programma proseguirà con la richiesta:
SECONDO NUMERO? ed analogamente ASSEGNERÀ il numero inserito alla VARIABILE N2; infine il programma proseguirà, come già visto, eseguendo il prodotto N1 * N2, ASSEGNANDO questo valore alla variabile N3 e scrivendo il risultato sul video.

L'istruzione INPUT segnala al computer quanto segue:

1. ARRESTA l'esecuzione del programma nel punto in cui c'è l'istruzione ed attendi l'inserimento di un dato da tastiera.
2. ASSEGNA il dato inserito alla VARIABILE, il cui nome è scritto dopo l'istruzione-INPUT.
3. PROSEGUI nell'elaborazione.

È molto importante notare che l'istruzione INPUT non può essere data in MODO DIRETTO; se si prova a darla in tale modo, il computer non la esegue, ma risponde con un messaggio di errore.

```
ILLEGAL DIRECT ERROR
```

Un'altra segnalazione di errore viene trasmessa nel seguente caso: se in un programma l'istruzione INPUT è associata al NOME di una variabile numerica, (come nel ns. programma INPUT N1), e noi inseriamo DATI NON NUMERICI, (cioè lettere o simboli), quando premiamo il tasto RETURN il computer NON accetta il dato inserito, ma segnala:

```
REDO FROM START
```

che significa: RIFARE DA CAPO.

Alcuni computer segnalano solamente ? REDO, cioè RIFARE. Dopo questo messaggio di errore il computer segnala con un ? (punto interrogativo) di essere ANCORA in ATTESA di un numero. Questa segnalazione di ERRORE NON avviene se l'istruzione INPUT è associata ad una variabile ALFANUMERICA: in tale caso infatti TUTTI i CARATTERI ALFANUMERICI, comprese le cifre, sono accettati.

Resta inteso che, volendo inserire numeri, su cui eseguire calcoli matematici, dobbiamo associare l'istruzione INPUT a variabili di tipo NUMERICO.

L'istruzione INPUT può essere scritta in diversi modi che permettono una stesura più comoda del programma, con l'uso della virgola, punto e virgola e di COSTANTI ALFANUMERICHE come segue:

MODIFICHIAMO LA RIGA 10 del programma precedente, riscrivendola così:

10 INPUT "INSERISCI DUE NUMERI"; N1, N2

CANCELLIAMO LA RIGA 20 CHE NON CI SERVE PIÙ: scrivendo 20 e premendo il tasto ENTER o RETURN.

Facendo eseguire, con RUN, il programma potremo verificare che:

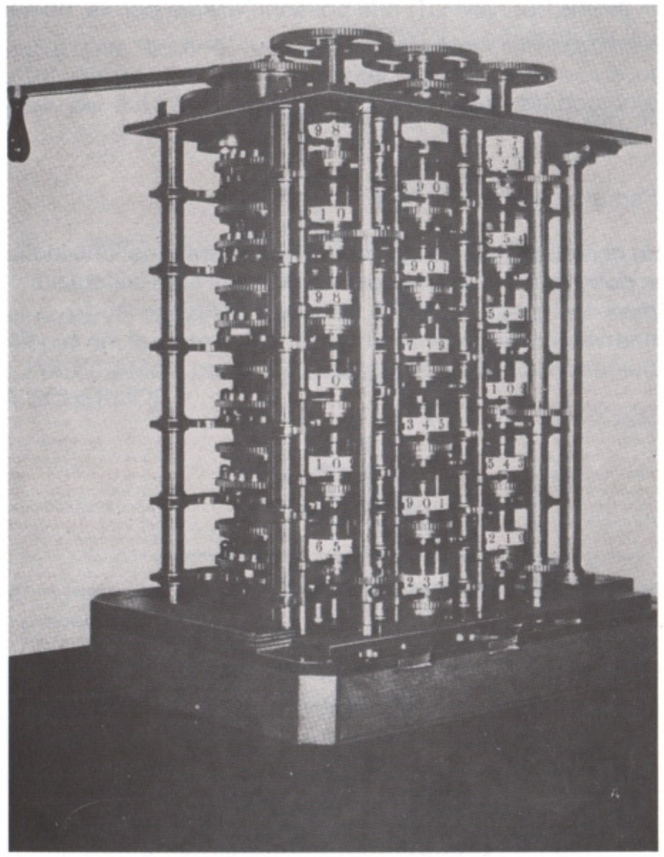
Il programma si interrompe richiedendo l'inserimento CONTEMPORANEO dei due numeri.

Noi dovremo inserire i due numeri, separando un numero dall'altro con la virgola e premendo (ENTER o RETURN) UNA SOLA VOLTA alla fine.

Inoltre se listiamo il programma, possiamo constatare che il numero di righe si è ulteriormente ridotto.

È molto importante conoscere a fondo le possibilità offerte dal proprio computer, circa l'uso delle diverse istruzioni, associate ai segni sintassi, o altro, per poter scrivere, con poca fatica i programmi, ottimizzando le risorse del computer.

E sempre opportuno consultare i manuali del computer ed eseguire delle prove, per constatarne le possibilità.



Macchina delle differenze di Babbage (1822).
Rimase in costruzione tra alti e bassi fino al 1833. In seguito Babbage concepì una Macchina Analitica molto più complessa della precedente, nessuna delle sue macchine però venne interamente realizzata.

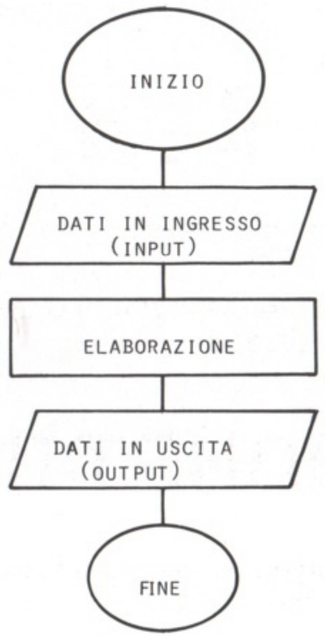
7 FORNIRE LE TECNICHE COMPLEMENTARI DI PROGRAMMAZIONE

Ora siamo in possesso di informazioni sufficienti per scrivere un programma un po' più complesso, che abbia un utilizzo pratico.

Occorre chiarire che, solitamente, un programma per computer, deve avere le seguenti caratteristiche:

1. Uno scopo preciso, cioè deve servire per risolvere un problema specifico (di calcolo, statistica, divertimento etc.).
2. Dei DATI in INGRESSO (INPUT) che possono essere parole, numeri, o segnali (per es. i segnali trasmessi dal JOYSTICK).
3. Dei DATI in USCITA (OUTPUT), di tipo diverso.
4. Un INIZIO ed una FINE.

Noi siamo in grado di realizzare un programma con le caratteristiche indicate sopra, anche se per il momento dobbiamo limitare le nostre ambizioni alla realizzazione di programmi di calcolo numerico.



Occorre considerare anche un altro aspetto della programmazione: tale aspetto non ha riferimenti con le caratteristiche elencate sopra, ma riguarda esclusivamente il MODO in cui il programma opera, cioè un aspetto più TECNICO ed ESTETICO, che possiamo chiamare con il termine TECNICA DI PROGRAMMAZIONE.

TECNICA DI PROGRAMMAZIONE

La TECNICA DI PROGRAMMAZIONE non ha regole ben definite, ma possiamo dire che rappresenta la capacità del programmatore di presentare il programma e di risolvere i problemi di INPUT ed OUTPUT in modo comodo e comprensibile, per chi UTILIZZERÀ il programma, ed altre cose ancora, come vedremo nelle prossime lezioni.

Per ora limitiamoci a tre punti, che noi riteniamo importanti, anche per programmi molto semplici:

1. Un programma deve SEMPRE prevedere tra le prime istruzioni un'ISTRUZIONE per CANCELLARE il VIDEO. Per alcuni computer tale istruzione è CLS, per altri può avere nome diverso, per cui occorre consultare il manuale del computer.

CLS

2. Il programma DEVE SEMPRE avere un TITOLO chiaro, ed eventualmente delle ISTRUZIONI per chi lo deve usare. Il titolo e le spiegazioni dovranno apparire NON SOLO NEL PROGRAMMA come REM, ma anche sopra lo schermo (dopo l'istruzione di PULIZIA SCHERMO).

REM

3. TUTTI i dati in ingresso ed uscita devono essere IDENTIFICATI con scritte MOLTO CHIARE e possibilmente BREVI.

   Questa tecnica prende il nome di PROMPTING e l'abbiamo già usata per identificare i numeri nel programma che abbiamo scritto.

PROMPTING

Per avere una buona TECNICA di PROGRAMMAZIONE occorre conoscere a fondo le possibilità offerte dal proprio computer.

8 FORNIRE LE BASI PER LA RISOLUZIONE DI PROBLEMI ELEMENTARI, ATTRAVERSO L'USO DELLO PSEUDO-LINGUAGGIO

Prima di scrivere un programma cerchiamo di fare una piccola ANALISI del PROBLEMA da risolvere.

PER PRIMA COSA DOBBIAMO CONOSCERE BENE IL PROBLEMA, I DATI CHE NE PERMETTONO LA SOLUZIONE E LA FORMULA RISOLUTIVA, (dobbiamo quindi saper risolvere il problema SENZA il computer).

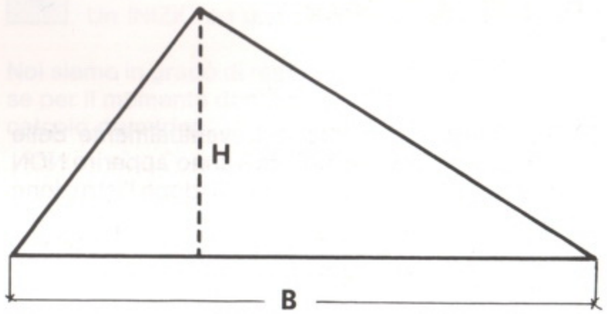
Possiamo poi scrivere alcune frasi per RAPPRESENTARE la soluzione.

Queste frasi dovranno però essere scritte in modo tale da permettere una FACILE TRADUZIONE in BASIC o altro linguaggio di programmazione.

Limitiamoci ad un problema molto semplice e ben conosciuto, come il CALCOLO DELL'AREA DEL TRIANGOLO.

I metodo RISOLUTIVO è noto: UNA MOLTIPLICAZIONE, ed una DIVISIONE per due.

I DATI sono noti: la BASE e l'ALTEZZA del triangolo servono per poter calcolare l'AREA.



Formula A = B*H/2

Dati: A = area
B = base
H = altezza

Per realizzare un programma possiamo scrivere le seguenti frasi:

1 INIZIO : PULIRE LO SCHERMO.

2 SCRIVERE TITOLO PROGRAMMA.

3 RICHIESTA della BASE del triangolo ed ASSEGNAZIONE del valore inserito ad una VARIABILE.

4 RICHIESTA dell'ALTEZZA del triangolo ed ASSEGNAZIONE del valore ad una seconda VARIABILE

5 CALCOLO dell'area (BASE per ALTEZZA diviso 2) ed ASSEGNAZIONE del valore calcolato ad una terza VARIABILE.

6 SCRITTURA sul VIDEO del risultato e sua identificazione.

7 FINE DEL PROGRAMMA.

Queste frasi possono facilmente essere TRADOTTE in BASIC, con le poche ISTRUZIONI a nostra disposizione.

Scrivere frasi di questo tipo permette di CAPIRE le tecniche di programmazione da usare e di definire QUANTE e QUALI VARIABILI usare, facilitando l'ATTRIBUZIONE dei NOMI alle VARIABILI che servono.

Le frasi di sui sopra sono scritte in PSEUDO-LINGUAGGIO, cioè un linguaggio che può essere TRADOTTO FACILMENTE in un linguaggio di programmazione "evoluto".

Considerando il nostro problema vediamo come scegliere i NOMI delle variabili. Per semplicità useremo nomi di una sola lettera, con PRECISI RIFERIMENTI MNEMONICI (cioè che siano FACILI DA RICORDARE)

Ci serve una VARIABILE NUMERICA REALE per depositare il valore della BASE, per cui chiameremo questa variabile B (come Base).

Un'altra per l'ALTEZZA e la chiameremo H.

Un'altra per l'AREA che chiameremo A (Come Area).

È IMPORTANTE NOTARE che se avessimo dato il nome A all'ALTEZZA, non avremmo potuto usarlo per l'AREA, altrimenti il computer avrebbe riservato SOLO un posto in MEMORIA invece di DUE.

9 APPLICARE LE NOZIONI ACQUISITE PER SCRIVERE UN PROGRAMMA

Traduciamo in BASIC le frasi scritte in PSEUDO-LINGUAGGIO, utilizzando TUTTE le informazioni acquisite:

```
10 REM CALCOLO AREA TRIANGOLO
20 CLS
30 PRINT "CALCOLO AREA TRIANGOLO" : PRINT
40 INPUT "BASE"; B : INPUT "ALTEZZA"; H
50 LET A = B * H/2
60 PRINT
70 PRINT "L'AREA È"; A
80 END
```

ATTENZIONE

L'istruzione CLS serve per cancellare lo schermo, controllare quale istruzione BASIC usa il vostro computer per tale scopo.

Occorre notare quanto segue, per apprendere alcune utili tecniche di programmazione:

– In alcuni punti del programma sono state messe delle istruzioni PRINT, senza indicare cosa scrivere; queste istruzioni prendono il nome di PRINT a VUOTO, e servono per SEPARARE con una riga vuota i messaggi sul video.

– Alla fine del programma compare l'istruzione END (trad. = FINE), che indica al computer di non proseguire nell'elaborazione: nel nostro breve programma l'istruzione END non sarebbe necessaria, in quanto il computer interromperebbe comunque l'elaborazione, non trovando più nessuna riga da eseguire, ma in futuro, vedremo altri casi in cui l'istruzione END sarà INDISPENSABILE per comandare al computer di interrompere l'elaborazione.
È meglio pertanto che ci abituiamo fin d'ora ad usare l'istruzione END, anche per rispettare le regole di programmazione viste precedentemente.

END

ESERCIZI

ESERCITAZIONI E STUDI DA ESEGUIRE A CASA

1 Ripassare gli argomenti trattati, rapportandoli al computer in dotazione.

2 Esercitarsi nell'uso dell'istruzione LET per ASSEGNARE valori a VARIABILI NUMERICHE REALI, NUMERICHE INTERE ed ALFANUMERICHE, usando l'istruzione PRINT per VERIFICARNE il CONTENUTO.

3 Esercitarsi nella scrittura di semplici programmi con le istruzioni CLS (per pulire schermo), INPUT, LET, PRINT, REM, END, ricordando che è meglio numerare le righe di DIECI in DIECI. Servirsi del segno : (due punti) per scrivere più istruzioni per riga.

ALLA FINE DI OGNI RIGA è INDISPENSABILE premere il tasto IMMISSIONE (cioè il tasto ENTER o RETURN) altrimenti il computer NON inserisce la riga nel programma.

4 Esercitarsi ad eseguire le seguenti operazioni, affinchè diventino una buona ed utile abitudine del modo di operare al computer:

a) - DOPO aver inserito tutte le righe di un programma PULIRE LO SCHERMO e LISTARE il programma per verificare se è stato inserito nel modo corretto.

b) - Per correggere gli eventuali errori usare le possibilità offerte dall'EDITING di SCHERMO, per quanto riguarda gli spostamenti del CURSORE, la CANCELLAZIONE, la SOSTITUZIONE, o l'INSERIMENTO di CARATTERI.
Utilizzare le possibilità offerte dall'EDITING di PROGRAMMA, per quanto riguarda CANCELLAZIONE o SOSTITUZIONE di RIGHE, ricordandosi di premere il tasto RETURN (o ENTER) dopo ogni correzione o modifica di una riga.

c) - Dopo eventuali modifiche al programma è necessario ripetere quanto già precisato al punto "a".

d) - Rileggere il programma, commentando le righe in PSEUDO-LINGUAGGIO per cercare di CAPIRE cosa verrà eseguito dal COMPUTER ad ogni riga.

e) - Dare il comando RUN per VERIFICARE se il computer ESEGUE ESATTAMENTE quanto PREVISTO.

Occorre sapere che il computer segnala soltanto gli ERRORI di SINTASSI, e gli ERRORI dovuti ad un uso errato dei simboli di identificazione, ($, %, etc.), ma NON SEGNALA gli ERRORI LOGICI, cioè I NOSTRI ERRORI.

ESEMPIO: se vogliamo che il computer scriva la parola CIAO ma scriviamo l'istruzione PRINT CIAO (dimenticando di usare le VIRGOLETTE), il computer interpreterà l'istruzione nel modo seguente: SCRIVI il CONTENUTO della VARIABILE NUMERICA che si chiama CIAO, per cui, molto probabilmente, scriverà 0 (zero), al posto della parola.

5 Dare il comando NEW per cancellare il programma che, eventualmente, si trova in memoria ed INSERIRE il programma seguente, facendo MOLTA ATTENZIONE, (come al solito) ad inserire OGNI RIGA così come è scritta.

IN PARTICOLARE: il segno " sta ad indicare le VIRGOLETTE, mentre il segno ' sta ad indicare l'ACCENTO (o l'APOSTROFO).

TUTTI i segni di SINTASSI, gli spazi o altri simboli, DEVONO essere scritti ESATTAMENTE come nel LISTATO.

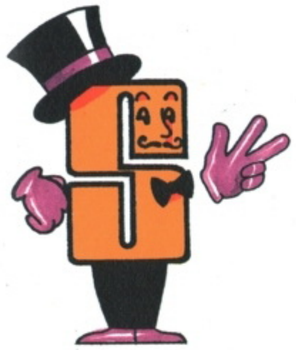
L'istruzione CLS serve per PULIRE lo SCHERMO: se il vostro computer NON ha tale istruzione, occorre sostituire l'istruzione CLS con l'istruzione usata dal vostro computer per cancellare lo schermo.

Ricordarsi di premere il tasto RETURN o ENTER alla FINE di ogni riga.

```
10 CLS
15 INPUT "COME TI CHIAMI";N1$
20 INPUT "QUANTI ANNI HAI";E1
25 INPUT "HAI UN FRATELLO (SI/NO)";R$
30 IF R$="SI" THEN 100
35 IF R$<>"NO" THEN 25
40 INPUT "HAI UNA SORELLA (SI/NO)";R$
45 IF R$="SI" THEN 100
50 IF R$<>"NO" THEN 40
55 INPUT "HAI UN AMICO O UNA AMICA (SI/NO)";R$
60 IF R$="SI" THEN 100
65 IF R$<>"NO" THEN 55
70 CLS
80 PRINT "CIAO ";N1$:PRINT
85 PRINT "IO SONO UN COMPUTER E SARO' TUO AMICO !!"
90 END
100 INPUT "COME SI CHIAMA";N2$
105 INPUT "QUANTI ANNI HA'";E2
110 CLS
115 PRINT N1$
120 IF E1>E2 THEN PRINT "HAI";E1-E2;"ANNI IN PIU' DI"
125 IF E1=E2 THEN PRINT "HAI LA STESSA ETA' DI"
130 IF E1<E2 THEN PRINT "HAI";E2-E1;"ANNI IN MENO DI"
135 PRINT N2$:PRINT
140 INPUT "VUOI RIPROVARE (SI/NO)";R$
145 IF R$="SI" THEN 10
150 IF R$="NO" THEN 70
160 GOTO 140
```

Dopo aver CONTROLLATO, ESEGUITO, e CORRETTO il PROGRAMMA TRASFERIRLO su CASSETTA, con il NOME di LEZIONE 3.

Questo programma servirà nella prossima LEZIONE 3, ricordarsi quindi di portare la cassetta, con il programma registrato, ed il listato di cui sopra alla prossima lezione.



LEZIONE 3

OBIETTIVI

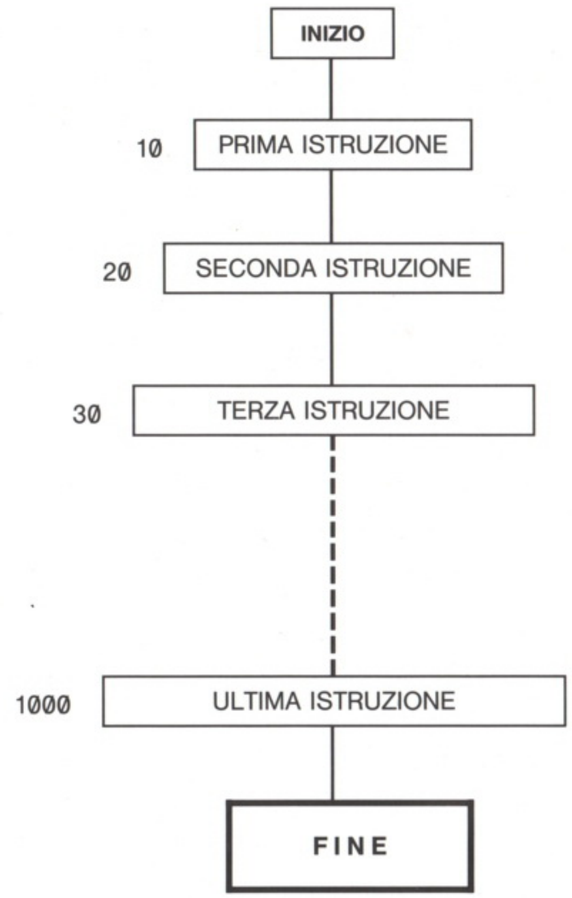
1. Fornire le spiegazioni relative ai salti diretti (GOTO nn).
2. Definire l'uso dell'istruzione GOTO, chiarendone i difetti.
3. Fornire gli elementi per la realizzazione di confronti e scelte con l'uso di operatori relazionali e logici e dei test condizionali.
4. Chiarire l'uso dell'istruzione IF... THEN per eseguire controlli.
5. Definire il concetto di salto condizionato (IF .. THEN .. ELSE).
6. Spiegare cosa sono le funzioni matematiche predefinite, chiarendo l'uso di SQR (x) e INT (x).
7. Spiegare cosa sono le funzioni di editing, chiarendo l'uso di TAB (x) e SPC (x), completando con le eventuali funzioni (o ISTRUZIONI) per controllare il CURSORE.
8. Introdurre il concetto di casualità e di pseudo-casualità.
9. Spiegare l'uso della funzione RND (x) nelle diverse forme.
10. Finalizzare le nozioni acquisite alla stesura di un programma.

1 FORNIRE LE SPIEGAZIONI RELATIVE AI SALTI DIRETTI (GOTO nn)

Un programma viene eseguito dal computer partendo dall'ALTO verso il BASSO, cioè dalla PRIMA RIGA, all'ULTIMA RIGA.

La PRIMA RIGA è quella che ha il numero più basso di tutte le altre.

L'ULTIMA RIGA è quella che ha il numero più alto.



Esiste però la possibilità di MODIFICARE questo modo di operare con alcune istruzioni che sono chiamate ISTRUZIONI DI SALTO.

La prima ISTRUZIONE di SALTO, che prenderemo in considerazione, è l'istruzione GOTO (trad. = VAI A), che è chiamata anche: ISTRUZIONE DI SALTO INCONDIZIONATO.

GOTO

Per prima cosa vediamo come deve essere scritta, cioè quale è la giusta SINTASSI:

L'istruzione GOTO (pronuncia: GOTU).

Deve sempre essere seguita da un numero che indica la riga di programma da cui DEVE PROSEGUIRE l'esecuzione del programma.

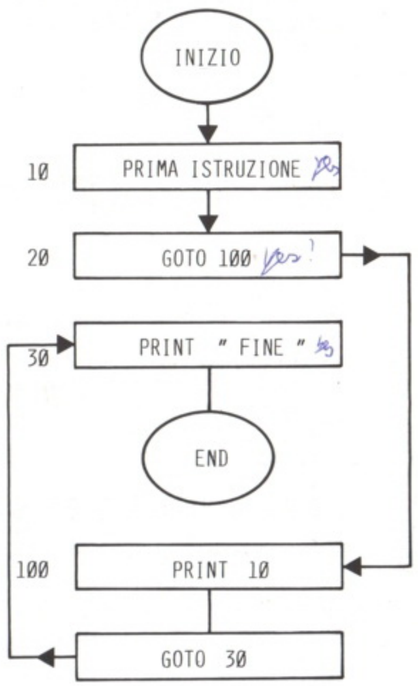
Ciò significa che l'istruzione GOTO 100 viene interpretata così:

L'ESECUZIONE DEL PROGRAMMA PROSEGUE DALLA RIGA NUMERO 100 IN POI.

Questo SALTO viene immediatamente eseguito. Il computer NON ha la possibilità di COMPORTARSI in MODO DIVERSO, ma è obbligato a SALTARE alla RIGA indicata.

Per questo l'istruzione GOTO (+ numero riga) viene chiamata anche: SALTO INCONDIZIONATO DIRETTO.

Esistono anche istruzioni di SALTO INCONDIZIONATO INDIRETTO, come avremo modo di vedere più avanti.



Vediamo ora l'uso del SALTO DIRETTO per uno scopo preciso:

è possibile utilizzare l'istruzione GOTO per fare eseguire più volte un programma, SENZA dovere dare ogni volta il comando RUN, come avviene nel seguente programma:

```
10 INPUT "INSERISCI UN NUMERO"; N1
20 LET N2 = N1 * N1: PRINT "IL QUADRATO DI ";N1;"È";N2
30 GOTO 10
```

In questo programma NON ESISTE la FINE, per cui quando viene eseguito per la PRIMA VOLTA con il comando RUN, il computer ESEGUE le istruzioni della RIGA 10 e della RIGA 20, ed arriva alla RIGA 30, dove l'istruzione GOTO 10 fa eseguire un SALTO all'indietro; l'esecuzione riprende dalla RIGA 10, prosegue con la RIGA 20 e ritorna alla RIGA 30. Il ciclo si ripete continuamente, ed è possibile interromperlo soltanto con il tasto di INTERRUZIONE (BREAK o STOP o altro nome).

BREAK

Il programma, inserito precedentemente, premette di comprendere due cose molto importanti:

1) Che le ISTRUZIONI DI SALTO INCONDIZIONATO possono essere molto utili al programmatore.

2) Che queste istruzioni DEVONO essere utilizzate con MOLTA ATTENZIONE, in quanto possono anche dar luogo, ad effetti NON DESIDERATI, ed ERRORI di LOGICA.

SALTO INCONDIZIONATO

```
1Ø PRINT "RIGA 1Ø"
2Ø GOTO 1ØØ
3Ø PRINT "RIGA 3Ø"
4Ø END

1ØØ PRINT "RIGA 1ØØ"
11Ø GOTO 3Ø
```

Il BASIC mette a disposizione ALTRE ISTRUZIONI di SALTO, molto più utili, come vedremo più avanti.

Per il momento l'istruzione GOTO ci è servita per comprendere il concetto di SALTO all'interno di un programma.

Vedremo poi come utilizzare nel modo migliore l'istruzione GOTO.

La regola da seguire per avere una BUONA TECNICA di PROGRAMMAZIONE però deve essere la seguente:

In un programma le ISTRUZIONI di SALTO INCONDIZIONATO devono comparire in numero MOLTO LIMITATO, e usate SOLTANTO se sono INDISPENSABILI.
Avremo modo di constatare i diversi casi in cui l'istruzione GOTO è molto UTILE, ed altri casi in cui è INDISPENSABILE.

3 FORNIRE GLI ELEMENTI PER LA REALIZZAZIONE DI CONFRONTI E DI SCELTE

Vediamo ora come è possibile far CONFRONTARE due valori qualsiasi al computer, per effettuare una scelta.

Un computer è in grado di eseguire un confronto tra i seguenti valori:

1 – TRA DUE COSTANTI NUMERICHE

2 – TRA DUE COSTANTI ALFANUMERICHE

3 – TRA UNA VARIABILE NUMERICA (INTERA o REALE) ed UNA COSTANTE NUMERICA

4 – TRA UNA VARIABILE ALFANUMERICA E UNA COSTANTE ALFANUMERICA

5 – TRA DUE VARIABILI NUMERICHE qualsiasi (anche di TIPO diverso)

6 – TRA DUE VARIABILI ALFANUMERICHE

La possibilità di confronto è realizzata mediante gli operatori relazionali.

I principali OPERATORI RELAZIONALI sono i seguenti:

> che significa MAGGIORE
< che significa MINORE
= che significa UGUALE
>= che significa MAGGIORE O UGUALE
<= che significa MINORE O UGUALE
<> che significa DIVERSO

La possibilità di scelta è invece data dall'ISTRUZIONE IF (trad= SE).

OPERAZIONI LOGICI = AND - OR - NOT

IF..... THEN .....

L'istruzione IF deve sempre essere seguita dal CONFRONTO che si vuole eseguire, e dall'istruzione THEN (trad= ALLORA), che a sua volta deve essere seguita da un'istruzione BASIC. Si ottiene così una FRASE BASIC che permette di fare un CONFRONTO ed una SCELTA. ESEMPIO di FRASE COMPLETA: IF 10 > 5 THEN PRINT 10 - tale frase può essere TRADOTTA nel seguente modo: SE 10 è MAGGIORE DI 5 ALLORA SCRIVI 10.

Provando a scrivere, in MODO DIRETTO, la FRASE BASIC di cui sopra, e premendo il tasto RETURN o ENTER potremo constatare che il computer esegue le seguenti operazioni:

1) – CONTROLLA se la COSTANTE 10 è MAGGIORE della COSTANTE 5.

2) – Poichè 10 è MAGGIORE di 5, ESEGUE l'istruzione PRINT, che abbiamo messo dopo la parola THEN, per cui scrive 10.

Cambiando l'OPERATORE RELAZIONALE > (maggiore) con l'OPERATORE < (minore) potremo verificare che l'istruzione PRINT 10 non viene più eseguita.

4 CHIARIRE L'USO DELL'ISTRUZIONE IF ... THEN ...

Verifichiamo alcune delle possibilità offerte dalle frasi IF ... THEN ... commentando il listato del programma, dato come esercitazione a casa, alla precedente lezione:

```
10 CLS
15 INPUT "COME TI CHIAMI";N1$
20 INPUT "QUANTI ANNI HAI";E1
25 INPUT "HAI UN FRATELLO (SI/NO)";R$
30 IF R$="SI" THEN 100
35 IF R$<>"NO" THEN 25
40 INPUT "HAI UNA SORELLA (SI/NO)";R$
45 IF R$="SI" THEN 100
50 IF R$<>"NO" THEN 40
55 INPUT "HAI UN AMICO O UNA AMICA (SI/NO)";R$
60 IF R$="SI" THEN 100
65 IF R$<>"NO" THEN 55
70 CLS
80 PRINT "CIAO ";N1$:PRINT
85 PRINT "IO SONO UN COMPUTER E SARO' TUO AMICO !!"
90 END
100 INPUT "COME SI CHIAMA";N2$
105 INPUT "QUANTI ANNI HA'";E2
110 CLS
115 PRINT N1$
120 IF E1>E2 THEN PRINT "HAI";E1-E2;"ANNI IN PIU' DI"
125 IF E1=E2 THEN PRINT "HAI LA STESSA ETA' DI"
130 IF E1<E2 THEN PRINT "HAI";E2-E1;"ANNI IN MENO DI"
135 PRINT N2$:PRINT
140 INPUT "VUOI RIPROVARE (SI/NO)";R$
145 IF R$="SI" THEN 10
150 IF R$="NO" THEN 70
160 GOTO 140
```

Commentiamo le righe del programma precedente:

10 Pulizia schermo
15 richiesta di un nome ed ASSEGNAZIONE alla variabile N1$ (alfanumerica)
20 richiesta dell'età ed ASSEGNAZIONE alla variabile E1 (numerica reale)
25 richiesta di una risposta (SI o NO) ed ASSEGNAZIONE alla variabile R$
30 CONFRONTO: SE il CONTENUTO di R$ è uguale alla COSTANTE "SI" ALLORA vai alla riga 100 (salto condizionato);
SE IL CONFRONTO NON è superato POSITIVAMENTE (cioè se la variabile R$ NON contiene "SI") il programma prosegue alla riga 35.
UN CONFRONTO NON SUPERATO viene detto FALSO (cioè non VERO).
35 CONFRONTO: SE R$ contiene un VALORE DIVERSO da "NO" ALLORA ritorna alla riga 25 (salto condizionato).
Questo confronto serve per ELIMINARE le risposte NON VALIDE in quanto la richiesta alla riga 25 ammetteva SOLAMENTE due possibili risposte (SI/NO); poichè la risposta SI è già stata CONTROLLATA alla riga 30 in questo punto è sufficiente controllare la risposta NO, escludendo TUTTE le risposte DIVERSE da "NO".
Se questo CONFRONTO è VERO (se la risposta è DIVERSA da "NO") il SALTO alla riga 25 riproporrà la domanda; se il confronto è FALSO (cioè se la risposta NON è DIVERSA da "NO") significa che l'utente HA RISPOSTO "NO" per cui il programma prosegue alla successiva riga 40.
40 come riga 25 ma con domanda diversa.
45 SE risposta è "SI" ALLORA SALTO alla riga 100
50 SE risposta NON VALIDA ALLORA ripete la richiesta (salto indietro)
55 come righe 25 e 40 ma con un'altra richiesta ancora.
60 SE risposta è "SI" ALLORA SALTO alla riga 100
65 SE risposta NON VALIDA ALLORA ripete la richiesta (salto indietro)

Le RIGHE da 70 a 90 sono la conclusione del programma:

70 Pulizia schermo
80 SCRIVE messaggio "CIAO" ed il nome contenuto nella variabile N1$ con un ulteriore messaggio di saluto
90 FINE LOGICA del programma (END)

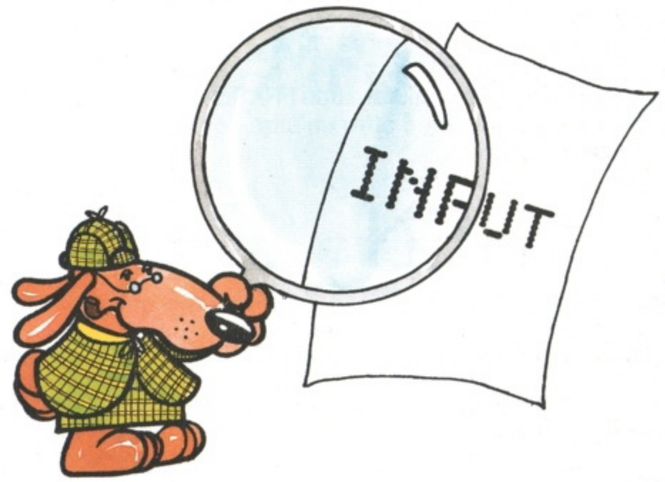
Le righe 100 e 105 formulano ulteriori richieste (come le righe 15 e 20) assegnando i valori digitati alle variabili N2$ ed E2

110 Pulizia schermo
115 scrive il nome contenuto in N1$ (nome dell'utente)
120-125-130 CONFRONTI tra l'età E1 e l'età E2 con stampa di un messaggio, con (o senza) UN NUMERO (DIFFERENZA di età).
135 scrive il nome contenuto in N2$
140 Richiesta ("VUOI RIPROVARE")
145 e 150 CONFRONTO della risposta con le risposte ammesse (SI/NO)
160 SALTO DIRETTO (se non è stata inserita una risposta VALIDA).

5 DEFINIRE IL CONCETTO DI SALTO CONDIZIONATO (IF .. THEN .. ELSE...)

L'istruzione IF ... THEN permette la costruzione di frasi complesse che possono servire per CONTROLLARE le risposte date dall'utente oppure per eseguire operazioni diverse sulla base di un certo CONFRONTO.

È una delle ISTRUZIONI più importanti e potenti di tutto il "vocabolario" BASIC ed è necessario conoscere e sperimentare le diverse possibilità di confronto in quanto TUTTI i programmi di una certa complessità sono basati su due TECNICHE di PROGRAMMAZIONE molto importanti: IL CONTROLLO dell'INPUT e la POSSIBILITÀ di SCELTA.



Il primo è indispensabile per controllare che i dati inseriti dalla tastiera siano CORRETTI e permettano la giusta prosecuzione del programma

ESEMPIO: se in INPUT viene richiesto di inserire un NUMERO per indicare un MESE dell'ANNO occorrerà controllare immediatamente quanto segue:

1 – che sia un numero maggiore di ZERO e MINORE di 13
2 – che sia un NUMERO INTERO (vedremo poi come fare questo controllo)

La POSSIBILITÀ di SCELTA è importante perchè permette all'utente di scegliere una delle ALTERNATIVE offerte dal programma (un programma complesso propone di solito una serie di ALTERNATIVE).

L'istruzione IF ... THEN nel BASIC STANDARD, è completata anche dagli OPERATORI LOGICI.

Gli OPERATORI LOGICI più comuni sono i seguenti:

NOT che significa NEGAZIONE
AND che significa CONTEMPORANEITÀ
OR che significa ESCLUSIONE

Il loro utilizzo in FRASI IF ... THEN permette di effettuare numerosi altri CONFRONTI aumentando notevolmente le possibilità del linguaggio, facilitando la programmazione.
ESEMPIO: per controllare SE un numero è COMPRESO tra 1 e 12 si può scrivere la seguente FRASE:

IF N>0 AND N < 13 THEN ...

oppure quest'altra: IF N<1 OR N>12 THEN ...

le due frasi eseguono lo stesso controllo, anche se in due modi diversi:

la prima controlla se i due confronti N>0 ed N<13 sono CONTEMPORANEAMENTE VERI, mentre l'altra controlla se UNO dei due confronti è VERO ESCLUDENDO tutti i casi in cui è VERO uno di tali confronti.

Il BASIC STANDARD permette di completare una FRASE IF ... THEN ... con l'istruzione ELSE (trad= ALTRIMENTI). Questa istruzione offre un'altra possibilità di scelta in opposizione a quella data alla istruzione THEN.

IF..... THEN ..... ELSE.....

È possibile formulare la seguente frase: IF ... THEN ... ELSE che può essere tradotta nel seguente modo:

SE (il confronto è VERO) ALLORA (esegui 1) ALTRIMENTI (esegui 2) dove "esegui 1" ed "esegui 2" possono essere una o più istruzioni BASIC (se sono più di una devono essere separate con i due punti), anche se solitamente sono istruzioni di SALTO che in questo caso prendono nome di ISTRUZIONI di SALTO CONDIZIONATO (cioè sottoposto a condizione).

Il programma prosegue alla riga successiva nel caso in cui tra le istruzioni previste in "esegui 1" o in "esegui 2" non c'è un'istruzione di salto.

Tutte le possibilità di deviare il programma DEVONO essere considerate in fase PRELIMINARE dal programmatore, in quanto il computer è una macchina, e come tale, è in grado solamente di ESEGUIRE i comandi che gli sono impartiti dall'uomo.

Il fatto di poter eseguire CONFRONTI e SCELTE sia tra i NUMERI, sia tra parole, offre innumerevoli possibilità, limitate soltanto dal livello delle conoscenze ed esperienze del programmatore.

È quindi opportuno esercitarsi ad affrontare i ragionamenti logici, considerando le possibilità offerte dal linguaggio (BASIC o altro) del proprio computer, per valutare quali e quanti confronti o scelte siano necessari, ed in relazione a ciò, prevedere le soluzioni per TUTTE le scelte e/o confronti.

6 SPIEGARE COSA SONO LE FUNZIONI MATEMATICHE PREDEFINITE

Il computer è in grado di eseguire calcoli matematici per mezzo degli OPERATORI MATEMATICI.

Le operazioni matematiche possibili però non si limitano alle operazioni elementari * / + – già viste ed alla operazione (più complessa) di ELEVAMENTO ad ESPONENTE.

Il linguaggio di programmazione solitamente mette a disposizione una SERIE di OPERAZIONI complesse che sono chiamate FUNZIONI MATEMATICHE.

FUNZIONI MATEMATICHE

Per esempio: è possibile eseguire la RADICE QUADRATA di un numero, grazie alla funzione apposita.

Prima di considerare alcune delle FUNZIONI più usate, vediamo COME si deve indicare al computer di ESEGUIRE una certa FUNZIONE.

In BASIC le FUNZIONI sono identificate con un NOME che solitamente è costituito da tre lettere. Subito dopo il nome c'è una PARENTESI rotonda APERTA che DEVE seguire IMMEDIATAMENTE il NOME (NON si devono lasciare SPAZI VUOTI tra il NOME della FUNZIONE e la parentesi).

Dopo la parentesi APERTA è possibile scrivere un qualsiasi VALORE. Questo VALORE può essere una COSTANTE NUMERICA, una VARIABILE NUMERICA o anche una ESPRESSIONE di calcolo tra numeri o variabili, o un'altra FUNZIONE.

Naturalmente alla fine occorrerà CHIUDERE la parentesi che era stata aperta dopo il NOME.

Una funzione matematica di solito è usata per ASSEGNARE un valore ad una VARIABILE (con l'istruzione LET), o per eseguire calcoli, (scrivendoli sul video con l'istruzione PRINT).

Vediamo ora due FUNZIONI molto importanti che è possibile trovare nella maggior parte dei LINGUAGGI di PROGRAMMAZIONE:

SQR(x) determina la RADICE QUADRATA del VALORE (indicato con "x")
il nome SQR deriva dalla abbreviazione del termine SQuare Root che significa appunto Radice Quadrata.

INT(x) determina l'INTERO del valore (indicato con "x") cioè considera SOLTANTO la parte INTERA del valore.

Solitamente per indicare ciò che può essere contenuto tra le parentesi di una funzione non si usa il termine VALORE che abbiamo usato nella spiegazione di cui sopra, ma si usa il termine ARGOMENTO.

L'ARGOMENTO di una FUNZIONE MATEMATICA può quindi essere una qualsiasi espressione matematica o un qualsiasi numero (contenuto in una VARIABILE oppure COSTANTE), o un'altra funzione.

ESEMPIO: uso della funzione INT per il calcolo di interessi bancari.

```
10 CLS
20 INPUT "CAPITALE";X
30 INPUT "TASSO DI INTERESSE %";R
40 INPUT "NUMERO ANNI";Y
50 FOR I=1 TO Y
60 B=X+I*X*(R/100)
70 C=INT(10*X*(1+R/100)^I)/10
80 D=C-B
90 CLS
95 PRINT "CAPITALE";X
100 PRINT "TASSO";R
110 PRINT "ANNO";I
120 PRINT "MONTANTE SEMPLICE";B
130 PRINT "MONTANTE COMPOSTO";C
140 PRINT "DIFFERENZA";D : PRINT
150 INPUT "PREMI RETURN PER PROSEGUIRE";Z$
160 NEXT I
170 PRINT : PRINT "FINE PROGRAMMA"
200 END
```


7 SPIEGARE LE FUNZIONI DI EDITING (TAB(x), SPC(x), PRINT @

Molti computer possiedono un EDITING di SCHERMO che permette di muoversi sul video, posizionandosi in un punto prescelto. Questi spostamenti si ottengono per mezzo degli appositi tasti di movimento del cursore.

Scrivendo un programma capita spesso di avere la necessità di scrivere o di cancellare un messaggio sul video in una posizione ben precisa.

Per questo motivo il linguaggio BASIC mette a disposizione di chi programma alcune possibilità per il CONTROLLO dell'OUTPUT, e per il posizionamento del CURSORE sul video.

Tali possibilità naturalmente cambiano da un computer all'altro, per cui analizzeremo solo le più comuni.

Ogni computer possiede le FUNZIONI di EDITING che permettono di scrivere in un punto di tabulazione voluto, oppure di separare tra loro due o più scritte con spazi vuoti. La sintassi di queste funzioni è simile a quella delle FUNZIONI MATEMATICHE già viste: ogni FUNZIONE di EDITING ha un NOME (solitamente di 3 lettere), seguito IMMEDIATAMENTE da una parentesi APERTA. All'interno della parentesi c'è l'ARGOMENTO della funzione, e dopo l'argomento c'è la parentesi CHIUSA.

Vediamo le due funzioni di editing più usuali:

TAB(x) indica il punto di tabulazione (il cui valore è espresso dall'ARGOMENTO "x").
SPC(x) indica il numero di SPAZI vuoti (precisati dall'ARGOMENTO "x").

Queste funzioni sono usate con l'istruzione PRINT per poter scrivere in un punto del video scelto da noi.

ESEMPIO:

PRINT TAB(10) "CIAO" SPC(4) "A TUTTI"

In questa istruzione la costante alfanumerica CIAO verrà scritta a partire dalla colonna 10 del video e dopo 4 spazi vuoti verrà scritta la costante alfanumerica "A TUTTI".

Oltre alle FUNZIONI, alcuni BASIC possiedono ISTRUZIONI speciali per l'editing. Non è possibile in questa sede analizzarle tutte: **consigliamo di controllare quali possibilità offre il proprio computer.** Elenchiamo comunque alcune delle principali ISTRUZIONI:

PRINT AT (x,y): permette di posizionarsi nel punto di coordinate x e y.

PRINT@n: posizionamento ASSOLUTO, indicato da "n" rispetto al punto 0 (zero) che è l'angolo in alto a sinistra (HOME).

PRINT USING (arg): permette di scrivere numeri o parole in un formato indicato da "arg" (solitamente è una costante alfanumerica).

Vi proponiamo un breve programma che esegue la separazione 3 a 3 delle cifre intere di un numero, (per chi non dispone della istruzione PRINT USING).

```
10 REM SEPARAZIONE CIFRE 3 A 3
20 CLS : FD%=0
30 INPUT "INSERIRE UN NUMERO";N$
40 IF VAL(N$)<1000 THEN X$=N$:GOTO 160
50 K=LEN(N$) : K1=K
60 FOR I=1 TO K1
70 IF MID$(N$,I,1)="." THEN FD%=FD%+1 : KX=I
80 NEXT I
90 IF FD%>1 THEN 10
100 FOR X=1 TO KX
110 X$=X$+MID$(N$,X,1) : B=KX+2-X
120 IF INT(B/3)=B/3 THEN X$=X$+","
130 NEXT X
140 IF RIGHT$(X$,1)="."THEN X$=LEFT$(X$,LEN(X$)-2)
150 IF RIGHT$(X$,1)=","THEN X$=LEFT$(X$,LEN(X$)-1)
160 IF FD%=1 THEN X$=LEFT$(X$,KX+1)+RIGHT$(N$,K1+1-KX)
170 PRINT  X$
```

8 INTRODURRE IL CONCETTO DI CASUALITÀ E DI PSEUDO-CASUALITÀ

Considerando lo sviluppo dell'informatica è possibile classificare i computer per GENERAZIONI, riferendosi alle caratteristiche di base offerte dal sistema considerato.

GENERAZIONI DI COMPUTER

1° GENERAZIONE 1943/1953

Computer MARK I
ENIAC
EDSAC
Calcolatori molto lenti, non erano in grado di memorizzare il programma.

2° GENERAZIONE

Calcolatori a tubi elettronici (valvole) e transistors al germanio
UNIVAC I (1954)
IBM

3° GENERAZIONE (1964)

Sistema 360 IBM.
Uso di linguaggi evoluti, e numerose periferiche, standardizzazione delle interfacce, avvento dei transistors al silicio e dei circuiti integrati.

4° GENERAZIONE (1974)

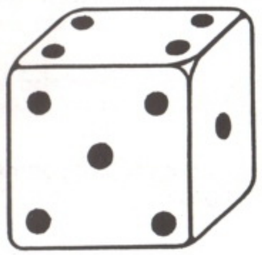
Uso dei MICROPROCESSORI su vasta scala, HARDWARE compatto e a basso costo, periferiche molto veloci, avvento dei PERSONAL ed HOME-COMPUTERS; tecnologia avanzata (stampanti a getto d'inchiostro, dischi Laser).

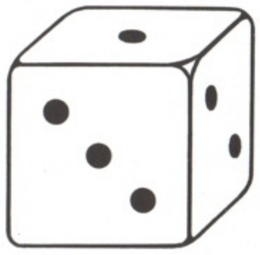
5° GENERAZIONE

Annunciata nell'ottobre del 1981 dal Ministro giapponese dell'Industria.
Attualmente in fase di sviluppo e progettazione: computers ad altissima velocità di elaborazione, enormi capacità di memoria, possibilità di simulare il ragionamento umano. Sistemi Esperti, Intelligenza Artificiale (AI).

I computer prodotti nel decennio 1974/84 appartengono alla QUARTA GENERAZIONE vale a dire che hanno le seguenti caratteristiche: HARDWARE compatto, grazie all'uso dei microprocessori, e tecnologie avanzate per quanto riguarda le periferiche (stampanti molto veloci, utilizzo dei FLOPPY DISK etc.)
La QUARTA GENERAZIONE ha inoltre introdotto il computer in molto settori quali l'AUTOMAZIONE, ed anche nelle abitazioni private (HOME COMPUTER).
Attualmente sono in fase di progettazione i computer della QUINTA GENERAZIONE che sarà caratterizzata da computer programmati in modo tale da essere in grado di formulare "ragionamenti", e di auto-istruirsi, grazie ad una notevole velocità di elaborazione, e ad enorme quantità di memoria per immagazzinare le informazioni.
Si presume che entro il 1990 la QUINTA GENERAZIONE dei computer possa diventare un fatto reale.
Naturalmente questo tipo di Intelligenza Artificiale sarà soltanto una emulazione dell'Intelligenza dell'uomo e dipenderà totalmente da quest'ultima. È molto importante capire che il computer è una macchina costruita dall'uomo, e per questo motivo dipende sempre dall'uomo, cioè non sarà mai in grado di produrre nulla che non gli sia stato insegnato a produrre.
Questo discorso permette di introdurre il concetto di pseudo-casualità.

Supponiamo di lanciare un dado da gioco: sappiamo che i numeri scritti sulle facce del dado hanno una uguale possibilità di uscita. Il numero che esce ad ogni lancio è assolutamente casuale in quanto non si può determinare a priori quale sarà questo numero.





La maggior parte dei computer è in grado di "generare" dei numeri casuali, però la generazione di questi numeri NON è totalmente incontrollabile, ma è stata programmata dall'uomo. Qualunque sia il metodo utilizzato, e per quanto sia complesso, è ripetibile. Ciò significa che potremmo sapere a priori quale è il numero generato. Per questo motivo tale casualità viene chiamata PSEUDO-CASUALITÀ, che significa una casualità che non è totalmente casuale.

9 SPIEGARE L'USO DELLA FUNZIONE RND(x).

Vedremo ora come è possibile fare generare al computer dei numeri pseudo-casuali e qualche applicazione:

La maggior parte dei computer che usano il BASIC ha una FUNZIONE che permette la generazione di numeri casuali.
Questa funzione (che ha la sintassi di tutte le altre funzioni viste), di solito è la seguente:

RND(x) (dal termine inglese RaNDom cioè casuale).

Come sempre occorre specificare un ARGOMENTO all'interno delle parentesi.

Sia l'ARGOMENTO, sia il risultato (cioè il numero casuale generato) possono avere interpretazioni diverse da un BASIC all'altro e come al solito occorre consultare il manuale del proprio computer, o fare delle prove, per verificare le possibili interpretazioni.

Solitamente quando l'argomento è 0 (ZERO) il computer genera un numero decimale compreso tra 0 e 0,99

È quindi possibile generare qualsiasi numero utilizzando la funzione RND(0) assieme all'operatore matematico * (MOLTIPLICATORE), ed alla funzione già vista INT(x) cioè INTERO.

ESEMPIO: per simulare il lancio di un dado occorrerà generare un numero pseudo-casuale che sia INTERO, POSITIVO e compreso tra 1 e 6.
Possiamo quindi scrivere il seguente breve programma:

Nota: L'istruzione RANDOMIZE di riga 5 deve essere inserita solo per quei computer che possiedono tale istruzione (come il Laser 500).

```
5  RANDOMIZE
10 LET N=RND(1)
20 LET N=N*6
30 LET N=INT(N)+1
40 PRINT N
50 END
```

Il commento al programma di cui sopra è il seguente:

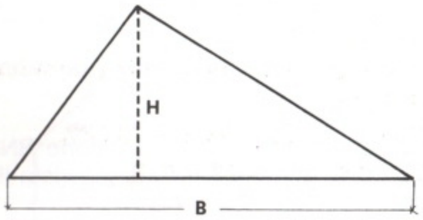
10 generazione di un numero compreso tra 0 e 0.99 ed ASSEGNAZIONE del valore alla variabile N.
20 Moltiplicazione del valore generato (che è contenuto in N) per 6 e RIASSEGNAZIONE del valore alla stessa variabile N.
30 DETERMINA l'INTERO del valore di N (ottenendo un numero tra 0 e 5) gli somma 1 (per avere un numero tra 1 e 6) e riassegna il valore così calcolato alla variabile N.
40 Scrive il numero contenuto in N.
50 FINE

10 FINALIZZARE QUANTO SPIEGATO PER REALIZZARE UN PROGRAMMA

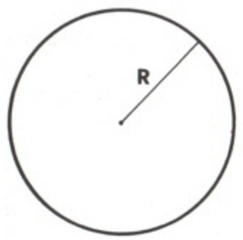
Vedremo ora come è possibile scrivere un programma più complesso, che sia in grado di eseguire calcoli diversi, in relazione alla scelta effettuata dall'utente.

Naturalmente utilizzeremo tutto quanto appreso per dare al programma una impostazione di tipo professionale.

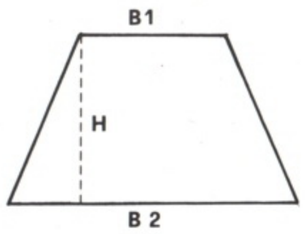
Abbiamo già scritto un programma che calcolava l'area del triangolo, ed ora scriveremo un programma che permetta di calcolare l'area di un figura piana tra le seguenti: TRIANGOLO, CERCHIO, TRAPEZIO.



Formula A = B*H/2
Dati: A = area
B = base
H = altezza



Formula A = R*R*3.14
Dati: R = raggio
A = area



Formula A = (B1+B2)*H/2
Dati: A = area
B1 = base minore
B2 = base maggiore
H = altezza

Analizziamo i punti principali:

1 - INIZIO: pulire lo schermo
2 - SCRIVERE il titolo del programma sul video
3 - Scrivere sul video le SCELTE POSSIBILI
4 - RICHIESTA di inserimento della SCELTA
5 - CONTROLLO di VALIDITÀ della scelta
6 - ESECUZIONE della parte di programma richiesta dall'utente
7 - FINE

Naturalmente il punto 6 deve poi essere analizzato per ognuna delle possibili scelte che nel nostro esempio sono 3.

SCELTA 1: CALCOLO AREA TRIANGOLO

1 - INIZIO : Pulire lo schermo.
2 - SCRIVERE IL TITOLO (cioè la scelta fatta dall'utente).
3 - RICHIESTA della BASE ed ASSEGNAZIONE alla variabile B.
4 - RICHIESTA dell'altezza ed ASSEGNAZIONE alla variabile H.
5 - CALCOLO dell'Area ed ASSEGNAZIONE alla variabile A.
6 - Scrittura sul video del risultato e sua identificazione.
7 - Richiesta per controllare se l'utente desidera eseguire un altro calcolo dello stesso tipo (calcolare l'area di un altro triangolo).
8 - CONTROLLO della validità della SCELTA (altro calcolo ? SI/NO).
9 - Esecuzione in relazione alla SCELTA dell'utente (SI o NO).

Tutti i punti indicati sopra con numerazione da 1 a 9 dovranno essere ripetuti per tutte le scelte possibili, cioè per il calcolo dell'area del cerchio, che ha come DATO in INPUT il Raggio, e per il TRAPEZIO che ha come DATI in INPUT la Base Minore, la Base Maggiore e l'Altezza.

Anche la parte relativa al calcolo dovrà naturalmente essere quella della figura piana prescelta.

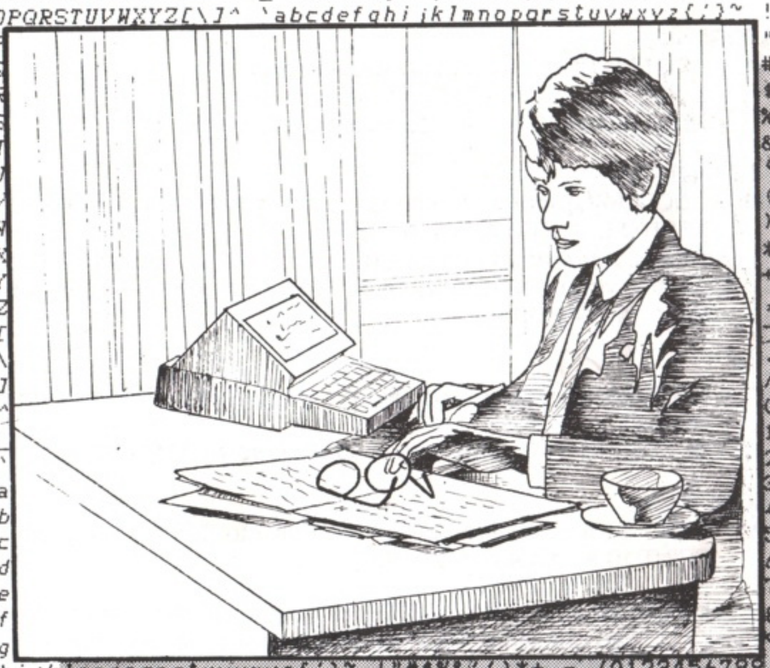
Vediamo quindi che per rappresentare la nostra risoluzione del problema abbiamo utilizzato ancora lo Pseudo-linguaggio, ma abbiamo dovuto scomporre il problema in BLOCCHI diversi.

Usando in ogni BLOCCO le tecniche di programmazione già viste, e le istruzioni che conosciamo finora, possiamo scrivere il programma.

Il programma per il calcolo di superfici piane, limitato al TRIANGOLO, CERCHIO e TRAPEZIO può essere scritto nel modo seguente:

```
10 REM CALCOLO SUPERFICI PIANE
20 CLS : REM (O ALTRA ISTRUZIONE PER PULIRE LO SCHERMO)
30 PRINT TAB(10);"CALCOLO SUPERFICI PIANE":PRINT
40 PRINT "SCELTE POSSIBILI:":PRINT
50 PRINT "1- TRIANGOLO";SPC(2);"2- CERCHIO":PRINT
60 PRINT "3- TRAPEZIO";SPC(3);"4- FINE":PRINT
70 INPUT "QUALE SCELTA";SC
80 IF SC<1 OR SC>4 THEN 10
90 IF SC<> INT(SC) THEN 10
100 IF SC=4 THEN 400
110 IF SC=3 THEN 300
120 IF SC=2 THEN 200
130 REM ** TRIANGOLO **
140 CLS : REM (O ALTRA ISTRUZIONE PER PULIRE LO SCHERMO)
150 PRINT TAB(13);"AREA TRIANGOLO":PRINT
160 INPUT "BASE";B:INPUT "ALTEZZA";H:LET A=B*H/2
170 PRINT :PRINT "AREA =";A:PRINT
180 INPUT "ALTRO CALCOLO (SI/NO)";R$
185 IF R$="SI" THEN 130
190 GOTO 10
200 REM ** CERCHIO **
210 CLS : REM (O ALTRA ISTRUZIONE PER PULIRE LO SCHERMO)
220 PRINT TAB(14);"AREA CERCHIO":PRINT
230 INPUT "RAGGIO";R:LET A=R*R*3.14
240 PRINT :PRINT "AREA =";A:PRINT
250 INPUT "ALTRO CALCOLO (SI/NO)";R$
260 IF R$="SI" THEN 200
270 GOTO 10
300 REM ** TRAPEZIO **
310 CLS : REM (O ALTRA ISTRUZIONE PER PULIRE LO SCHERMO)
320 PRINT TAB(13);"AREA TRAPEZIO":PRINT
330 INPUT "BASE MIN.";B1:INPUT "BASE MAGG.";B2
340 INPUT "ALTEZZA";H
350 LET A=(B1+B2)*H/2
360 PRINT :PRINT "AREA =";A:PRINT
370 INPUT "ALTRO CALCOLO (SI/NO)";R$
380 IF R$="SI" THEN 300
390 GOTO 10
400 CLS : REM (O ALTRA ISTRUZIONE PER PULIRE LO SCHERMO)
410 END
```


ESERCIZI



ESERCITAZIONI E STUDI DA ESEGUIRE A CASA:

1 Esercitarsi nell'uso delle istruzioni, funzioni, etc. spiegate durante la lezione.

2 Preparare un ELENCO COMPLETO di TUTTO il VOCABOLARIO BASIC appreso, facendo gli opportuni commenti alle diverse voci, e cercando di spiegare quale è l'uso di ogni COMANDO, ISTRUZIONE o simbolo.

TUTTO il VOCABOLARIO BASIC appreso dovrà essere elencato e commentato seguendo il seguente schema:

1) - COMANDI DI SISTEMA
(ESEMPIO: RUN permette di ESEGUIRE il programma esistente in memoria)

2) - VARIABILI
(ESEMPIO: ALFANUMERICHE hanno il segno $ dopo il nome)

3) - COSTANTI
(ESEMPIO: NUMERICHE reali o intere sono NUMERI interi o decimali)

4) - OPERATORI
(ESEMPI: - MATEMATICI * indica moltiplicazione
- LOGICI AND indica contemporaneità
- RELAZIONALI >= indica MAGGIORE o UGUALE)

5) - SEGNI DI SINTASSI
(ESEMPI: $ (DOLLARO) indica il TIPO di variabile (VARIABILE ALFANUMERICA).
; (PUNTO E VIRGOLA) usato con l'istruzione PRINT per scrivere sulla stessa riga oppure dopo una istruzione tipo: INPUT "messaggio"; NOME in cui il punto e virgola ha la funzione di separare il NOME della variabile dal messaggio tra virgolette.

6) - FUNZIONI
(ESEMPI: - MATEMATICHE: INT (arg) determina l'INTERO del valore indicato tra parentesi)
- DI EDITING TAB (arg) stabilisce il punto di TABULAZIONE

7) - ISTRUZIONI BASIC
(ESEMPI: - PRINT scrive sul video ciò che è indicato DOPO l'istruzione stessa (es. PRINT 10 scrive il numero 10)

- INPUT istruzione che CHIEDE di inserire un dato ed ASSEGNA il valore ad una variabile. Il nome della variabile deve essere messo dopo l'istruzione.
Se si una l'istruzione tipo: INPUT "messaggio"; NOME occorre mettere il punto e virgola per separare il NOME della variabile dal messaggio tra virgolette.

3 Digitare il programma allegato seguendo le regole solite:

1) - Prima di digitare il programma dare il comando NEW (più tasto RETURN).

2) - Inserire le righe ESATTAMENTE come sono scritte, rispettando i segni di sintassi, gli spazi, etc. e ricordarsi di premere il tasto RETURN alla fine di OGNI RIGA.

3) - Listare, controllare ed eventualmente correggere il programma come già spiegato.

4) - Far eseguire il programma e controllarne il funzionamento, apportando le correzioni che dovessero essere necessarie.

5) - Salvare il programma su nastro con il titolo LEZIONE 4 e ricordarsi di portare il nastro ed il listato del programma alla prossima lezione.

```
10 REM **  TABELLE  NUMERICHE  **
11 REM
12 REM          LEZIONE 4
13 REM
14 REM VERSIONE LASER 110/310/500
15 REM
20 PRINT CHR$(31);"TABELLE NUMERICHE":PRINT
30 PRINT "1- PITAGORICA     2- QUADRATI/CUBI":PRINT
40 PRINT "3- NUMERI PRIMI   4- FINE":PRINT
50 INPUT "QUALE SCELTA";SC:IF SC>4 OR SC<1 THEN 10
60 IF SC=4 THEN PRINT CHR$(31):END
70 IF SC=3 THEN  300
80 IF SC=2 THEN  200
100 PRINT CHR$(31);"TAVOLA PITAGORICA":PRINT
110 FOR I=1 TO 10
120 FOR J=1 TO 10
130 N=I*J
135 N$=STR$(N)
140 N$="    "+N$
145 N$=RIGHT$(N$,3)
150 PRINT N$;
155 NEXT J:PRINT
160 NEXT I:PRINT
165 INPUT "VUOI RIVEDERE (SI/NO)";R$
170 IF R$="SI" THEN 100
175 IF R$<>"NO" THEN 165
180 GOTO 10
200 PRINT CHR$(31);"TAVOLA QUADRATI E CUBI":PRINT
205 PRINT "N.  N^2  N^3":PRINT
210 FOR I=1 TO 10
212 I$=STR$(I)
214 I$="  "+I$
216 I$=RIGHT$(I$,2)
220 N2=I*I:N3=I*I*I
230 N2$=STR$(N2)
240 N2$="      "+N2$
245 N2$=RIGHT$(N2$,5)
250 N3$=STR$(N3)
255 N3$="      "+N3$
260 N3$=RIGHT$(N3$,5)
265 PRINT I$;N2$;N3$
270 NEXT I:PRINT
280 INPUT "VUOI RIVEDERE (SI/NO)";R$
285 IF R$="SI" THEN 200
290 IF R$<>"NO" THEN 280
295 GOTO 10
300 PRINT CHR$(31);"NUMERI PRIMI TRA 1 E 500":PRINT
310 FOR I=1 TO 500
320 N=INT(SQR(I)+1)
330 FOR J=2 TO N
335 P=I
340 N$=STR$(P)
345 N$="    "+N$
350 N$=RIGHT$(N$,4)
355 IF INT(I/J)=I/J THEN P=0:J=N
360 NEXT J
365 IF P<>0 OR I<3 THEN PRINT N$;
370 NEXT I:PRINT:PRINT
375 INPUT "VUOI RIVEDERE (SI/NO)";R$
380 IF R$="SI" THEN 300
385 IF R$<>"NO" THEN 375
390 GOTO 10
```



```
10 REM ** TABELLE NUMERICHE **
11 REM
12 REM        LEZIONE 4
13 REM
14 REM VERSIONE COMMODORE VIC 20
15 REM
20 PRINTCHR$(147);"TABELLE NUMERICHE":PRINT
30 PRINT"1-PITAGORA 2-QUAD/CUBI":PRINT
40 PRINT"3-N.PRIMI  4- FINE":PRINT
50 INPUT"QUALE SCELTA";SC : IF SC>4ORSC<1THEN 10
60 IF SC=4 THEN PRINTCHR$(147) : END
70 IF SC=3THEN 300
80 IF SC=2THEN 200
100 PRINTCHR$(147);"TAVOLA PITAGORICA":PRINT
110 FOR I = 1 TO 7
120 FOR J = 1 TO 7
130 N=I*J
135 N$=STR$(N)
140 N$="    "+N$
145 N$=RIGHT$(N$,3)
150 PRINTN$;
155 NEXTJ  : PRINT
160 NEXT I :PRINT
165 R$="":INPUT"ANCORA (SI/NO)";R$
170 IF R$="SI" THEN 100
175 IF R$<>"NO" THEN 165
180 GOTO 10
200 PRINTCHR$(147);"TAVOLA QUADRATI E CUBI":PRINT
205 PRINT " N.  N↑2  N↑3":PRINT
210 FOR I = 1 TO 10
212 I$=STR$(I)
214 I$="  "+I$
216 I$=RIGHT$(I$,2)
220 N2=I*I : N3=I*I*I
230 N2$=STR$(N2)
240 N2$="      "+N2$
245 N2$=RIGHT$(N2$,5)
250 N3$=STR$(N3)
255 N3$="      "+N3$
260 N3$=RIGHT$(N3$,5)
265 PRINT I$;N2$;N3$
270 NEXT I  : PRINT
280 R$="":INPUT"ANCORA (SI/NO) ";R$
285 IF R$="SI" THEN 200
290 IF R$<>"NO" THEN 280
295 GOTO 10
300 PRINTCHR$(147);"N. PRIMI TRA 1 E 500":PRINT
310 FOR I = 1 TO 500
320 N=INT(SQR(I)+1)
330 FOR J = 2 TO N
335 P=I
340 N$=STR$(P)
345 N$="    "+N$
350 N$=RIGHT$(N$,4)
355 IF INT(I/J) = I/J THEN P=0 : J=N
360 NEXT J
365 IF P<>0 OR I<3THEN PRINT N$;
368 IFPOS(X)>18THEN PRINT
370 NEXT I : PRINT
375 R$="":INPUT"ANCORA (SI/NO) ";R$
380 IF R$="SI" THEN 300
385 IF R$<>"NO" THEN 375
390 GOTO 10

READY.
```

```
10 REM ** TABELLE NUMERICHE **
11 REM
12 REM        LEZIONE 4
13 REM
14 REM  VERSIONE COMMODORE 64
15 REM
20 PRINTCHR$(147);"TABELLE NUMERICHE":PRINT
30 PRINT"1-PITAGORICA    2-QUADRATI/CUBI":PRINT
40 PRINT"3-NUMERI PRIMI  4- FINE":PRINT
50 INPUT"QUALE SCELTA";SC : IF SC>4ORSC<1THEN 10
60 IF SC=4 THEN PRINTCHR$(147) : END
70 IF SC=3THEN 300
80 IF SC=2THEN 200
100 PRINTCHR$(147);"TAVOLA PITAGORICA":PRINT
110 FOR I = 1 TO 10
120 FOR J = 1 TO 10
130 N=I*J
135 N$=STR$(N)
140 N$="   "+N$
145 N$=RIGHT$(N$,3)
150 PRINTN$;
155 NEXTJ  : PRINT
160 NEXT I : PRINT
165 INPUT"VUOI RIVEDERE (SI/NO) ";R$
170 IF R$="SI" THEN 100
175 IF R$<>"NO" THEN 165
180 GOTO 10
200 PRINTCHR$(147);"TAVOLA QUADRATI E CUBI":PRINT
205 PRINT " N.  N↑2  N↑3":PRINT
210 FOR I = 1 TO 10
212 I$=STR$(I)
214 I$="  "+I$
216 I$=RIGHT$(I$,2)
220 N2=I*I : N3=I*I*I
230 N2$=STR$(N2)
240 N2$="     "+N2$
245 N2$=RIGHT$(N2$,5)
250 N3$=STR$(N3)
255 N3$="     "+N3$
260 N3$=RIGHT$(N3$,5)
265 PRINT I$;N2$;N3$
270 NEXT I  : PRINT
280 INPUT"VUOI RIVEDERE (SI/NO) ";R$
285 IF R$="SI" THEN 200
290 IF R$<>"NO" THEN 280
295 GOTO 10
300 PRINTCHR$(147);"NUMERI PRIMI TRA 1 E 500":PRINT
310 FOR I = 1 TO 500
320 N=INT(SQR(I)+1)
330 FOR J = 2 TO N
335 P=I
340 N$=STR$(P)
345 N$="    "+N$
350 N$=RIGHT$(N$,4)
355 IF INT(I/J) = I/J THEN P=0 : J=N
360 NEXT J
365 IF P<>0 OR I<3THEN PRINT N$;
370 NEXT I : PRINT
375 INPUT"VUOI RIVEDERE (SI/NO) ";R$
380 IF R$="SI" THEN 300
385 IF R$<>"NO" THEN 375
390 GOTO 10

READY.
```