

## 2 - ALGORITMI E PROGRAMMI

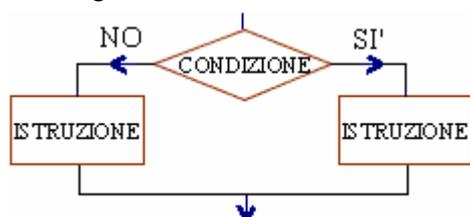
Abbiamo detto che le istruzioni di un algoritmo devono essere concretamente eseguibili e comprensibili da parte dell'esecutore. Ora, ai giorni nostri, l'esecutore per eccellenza di algoritmi, anche molto complessi, è il *computer*, che è precisissimo e rapidissimo nell'eseguire, senza stancarsi e senza protestare, sequenze di istruzioni che gli vengono assegnate nell'ordine corretto da un "programmatore" umano, il quale tenga conto delle limitate capacità che il computer possiede.

Esso non è in grado di pensare, o di prendere autonomamente qualsivoglia decisione:

dipende in tutto e per tutto dall'uomo che, attraverso un "programma", gli ordina, istante per istante, COSA fare.

Essenzialmente, un computer sa soltanto:

- **ASSOCIARE A UNA LOCAZIONE DI MEMORIA UN NOME**  
che la identifichi, come potrebbe essere *n*, o *num*, o *massimodivisoretrovato*, ecc.
- **LEGGERE** un'informazione che venga battuta sulla tastiera (= ricevere un *input* dalla *tastiera*), memorizzando questo input in una locazione nella RAM (oppure, interpretare opportunamente come comandi i movimenti e i "clic" o "doppio clic" fatti col *mouse*)
- **SCRIVERE SUL MONITOR, o MANDARE IN STAMPA, o SALVARE su di una memoria di massa, un'informazione o un'insieme di informazioni**
- **ESEGUIRE CALCOLI E CONFRONTI FRA NUMERI**
- **MODIFICARE IL CONTENUTO DI UNA LOCAZIONE DI MEMORIA**
- controllare **SE** è verificata una certa condizione;  
**IN CASO AFFERMATIVO** eseguire una data istruzione o blocco di istruzioni,  
**ALTRIMENTI** eseguire un'altra istruzione o blocco



Ad esempio:

Nella parte preliminare di un programma scritto nel linguaggio di programmazione **PASCAL**, *dichiarazioni* come **VAR num: integer** oppure **VAR cognome: string[25]** ordinano al computer di riservare locazioni, nella RAM, destinate a contenere un numero intero o, rispettivamente, una "stringa" (sequenza di caratteri) con non più di 25 caratteri, associando a queste "scatolette" i nomi *num*; *cognome*

Nel linguaggio di programmazione **PASCAL**, l'istruzione **READ(num)** ordina al computer, quando l'utente avrà digitato sulla tastiera un numero e premuto il tasto *Invio*, di memorizzare quel numero nella locazione della memoria centrale (RAM), alla quale è stato associato il nome *num*

In linguaggio **PASCAL**, l'istruzione **WRITE(num)** ordina al computer di scrivere sul monitor il valore che in quel momento ha la variabile *num*, ossia il contenuto che c'è in quel momento nella locazione di memoria, nella "scatoletta", alla quale è stato associato il nome *num*

In linguaggio **PASCAL**, la **moltiplicazione** viene indicata con un asterisco \*, la **divisione** con /, la **divisione intera** con **DIV** e il suo **resto** con **MOD**, "**maggiore o uguale**", "**minore o uguale**" si scrivono  $\leq$ ,  $\geq$ , "**diverso da**" si scrive  $\neq$

In linguaggio **PASCAL**, l'istruzione di "**ASSEGNAZIONE**" **c := a+b** (occhio: il simbolo grafico non è =, è :=) ordina al computer di eseguire la somma dei contenuti delle locazioni di memoria associate ai nomi *a*, *b* e di depositare il risultato nella locazione di memoria *c* (assegnare il valore ottenuto alla variabile *c*)

In linguaggio **PASCAL**, l'istruzione (o meglio, l' "istruzione complessa" o "struttura")

**IF a<b THEN max := b ELSE max := a**

ordina al computer di controllare se  $a < b$  (cioè, se il contenuto della casella *a* nella RAM ha un valore numerico minore del contenuto della casella *b*); in caso affermativo, il computer dovrà eseguire l'istruzione  $max := b$  (cioè, dovrà assegnare alla variabile *max* il valore *b* ossia dovrà porre nella "scatoletta" *max* in RAM lo stesso valore che c'è nella "scatoletta" *b*), in caso negativo dovrà eseguire l'istruzione  $max := a$

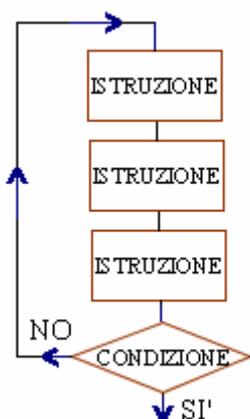
□ **RIPETERE**

un'istruzione,  
o un gruppo  
di istruzioni,

**FINCHÉ**

sarà verificata  
una certa condizione

(CICLO A  
CONTROLLO FINALE)



oppure

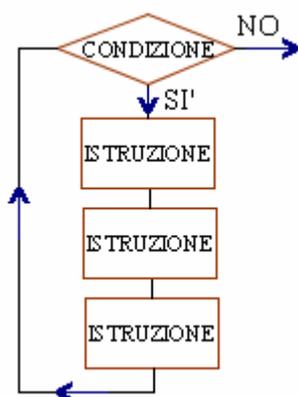
**FINTANTOCHE'**

è verificata  
una certa condizione,

**RIPETERE**

un'istruzione, o un  
gruppo di istruzioni

(CICLO A  
CONTROLLO INIZIALE)



In linguaggio PASCAL, un “pezzo” di programma potrebbe essere il seguente:

```
readln(a);
k:=0;
REPEAT
    k:=k+1;
    mult:=a*k;
    writeln(mult);
UNTIL k = 5
```

Questo programma prevede innanzitutto la *lettura* del numero intero  $a$  (readln è una *variante* di read, e ordina al computer di *andare a capo*, sul monitor, dopo che l'utente ha inserito il valore di  $a$ ) e l'“*inizializzazione*” della variabile  $k$  al valore 0.

Dopodiché, l'istruzione (o meglio, l'“istruzione complessa” o “struttura”)

**REPEAT ... UNTIL ...**

ordina al computer di

RIPETERE il blocco delle tre istruzioni

I) poni nella scatoletta  $k$  il valore che c'era prima, aumentato di 1 (incrementa di 1 il valore della variabile  $k$ : “:=” non indica uguaglianza, indica **assegnazione!**)

II) poni nella scatoletta  $mult$  il valore  $a*k$  (l'asterisco sta per *moltiplicazione*)

III) scrivi sul monitor il contenuto che ha, in quel momento, la scatoletta  $mult$  (writeln è una variante di write, e ordina al computer di andare a capo dopo che ha scritto sul monitor)

FINCHÉ' sarà verificata la condizione  $k = 5$ ;

quando questa condizione si verificherà, il computer dovrà uscire dal ciclo ed eseguire le istruzioni successive del programma.

L'esito sarà quello di mandare in output, sul monitor, i primi 5 multipli di  $a$ .

Ad esempio, se il valore della variabile  $a$  è 9,

il contenuto delle tre “scatolette”  $a$ ,  $k$  e  $mult$ , nella RAM, si evolverà, per effetto del programma, nel modo seguente:

	$k:=0$	$k:=k+1$	$k:=k+1$	$k:=k+1$	$k:=k+1$	$k:=k+1$
$a$	9	9	9	9	9	9
$k$	0	1	2	3	4	5
$mult$		9	18	27	36	45

Sempre in linguaggio PASCAL,

la ripetizione (“iterazione”) di un'istruzione o di un blocco di istruzioni può anche essere comandata da una WHILE ... DO ..., come nel pezzo di programma che segue (dove  $n$  è un intero):

```
readln(n);
WHILE n>=0 DO
    begin
        writeln(n);
        n:=n - 1;
    end;
```

Dopo la lettura, da tastiera, del valore di  $n$ ,

l'istruzione (o meglio, l'“istruzione complessa” o “struttura”)

**WHILE ... DO ...**

ordina al computer di:

scrivere sul monitor il valore di  $n$  (e andare a capo);

diminuire di 1 il valore della variabile  $n$ ;

e CONTINUARE A ESEGUIRE (DO) questa coppia di istruzioni

FINTANTOCHE' (WHILE)  $n$  si mantiene  $\geq 0$  ( $\geq$  in PASCAL sta per  $\geq$ ); quando questa condizione non sarà più verificata, uscire dal ciclo.

L'effetto sarà quello di mandare in output,

sul monitor, un “conto alla rovescia” da  $n$  fino a 0.