# Un corso di formazione informatica per insegnanti di fisica Lucca 1984*

*B. Barsella, E. Fabri, U. Penco*

Istituto di Astronomia dell'Università – Pisa

——— o ———

Il corso si è svolto nel periodo Marzo–Maggio 1984 presso l'Istituto Tecnico Commerciale di Lucca; si è articolato in 10 lezioni settimanali per complessive 30 ore, di cui una metà circa di attività pratica svolta su personal computer APPLE–IIe; per garantire questa possibilità il numero dei partecipanti era limitato a 20, divisi in 5 gruppi (la scuola era dotata di 5 esemplari completi di drive per minifloppy e stampante).

Si riporta in sintesi il calendario del corso.

*Calendario*

1. Presentazione del corso
2. Problema dei 2 corpi – Prove di uso del programma
3. Discussione
4. Integrazione numerica – Visualizzazione grafica
5. Schema a blocchi del programma – Lavoro di gruppo
6. Hardware, architettura – Lavoro di gruppo
7. Linguaggi – Lavoro di gruppo
8. Grafica – Lavoro di gruppo
9. Programma completo – Discussione
10. Commento generale - Appl. didattica

Come è stato detto in altra comunicazione [1], nel corso si è fatto uso soltanto del linguaggio BASIC, e non si sono tenute lezioni d'istruzione sul linguaggio. Per questo ci siamo limitati a fornire ad ogni partecipante una copia del manuale di macchina su cui si trovano le descrizioni dei comandi BASIC e semplici applicazioni, ciò che, secondo noi, era sufficiente allo scopo. Da notare che la maggior parte dei partecipanti non conosceva il BASIC né aveva alcuna nozione di programmazione.

Riprendendo e migliorando il tema di una precedente esperienza fatta con la sezione di Pisa, il corso è stato incentrato sull'elaborazione del programma ORBITE: integrazione numerica del moto di una sonda spaziale in orbita attorno

* Comunicazione al XXIII Congresso AIF, Gaeta 27–10–1984; *Atti del XXIII Congresso AIF*, 230 (1986).

alla Terra, manovrabile con un motore. I dati di partenza sono la posizione e la velocità iniziale; il programma integra passo-passo il problema dei due corpi e ne deriva quindi un'orbita kepleriana. Durante il moto si può intervenire (usando la tastiera) per azionare un "motore": questo è schematizzato come una variazione percentuale della velocità, per cui si tratta di un motore capace di generare solo accelerazioni tangenziali e senza la possibilità di invertire il moto. Altri comandi controllano la lunghezza di una "scia" per la visualizzazione dell'orbita, permettono di bloccare temporaneamente il moto e di cambiare le condizioni iniziali.

ORBITE dunque fa parte della classe dei programmi detti "di simulazione," sui quali sarebbe necessario avere idee chiare: infatti in questa categoria possiamo trovare applicazioni interessanti ed altre che lasciano molto perplessi.

ORBITE è stato ritenuto particolarmente idoneo poiché:

- simula una situazione non riproducibile in laboratorio, al quale non pretende quindi di sostituirsi;
- è semplice sotto l'aspetto matematico, ma richiede una quantità di calcolo tale da rendere indispensabile l'uso di un calcolatore;
- offre l'opportunità di usare in modo significativo le possibilità grafiche degli attuali personal computer; una tabella numerica in questo caso, pur avendo un contenuto di informazioni certamente maggiore, non riesce a trasmetterle in modo cos immediato ed espressivo (specialmente quando, facendo uso del motore, si modifica l'orbita);
- infine, da un punto di vista più strettamente didattico, fomisce un mezzo interessante per familiarizzare lo studente con la cinematica e la dinamica del moto su traiettorie curve, perché visualizza l'effetto di una forza centrale e di variazioni di velocità. La simulazione, che — come detto prima — ha in questo caso piena validità sperimentale, può essere usata per ricavarne la terza legge di Keplero e quindi la dipendenza della forza dalla distanza.

Nella prima parte del corso il programma ORBITE è stato mostrato già funzionante, ed è stato fatto usare. Si è potuto constatare che, per quanto sufficientemente descritto, il programma non è risultato di uso immediato: in particolare si sono registrate difficoltà nella scelta di ragionevoli condizioni iniziali. Di conseguenza la programmata discussione sull'applicabilità didattica di questa simulazione è stata abbastanza limitata. Durante questa prima parte i partecipanti hanno proceduto in proprio allo studio del linguaggio BASIC.

Nella seconda fase del corso l'algoritmo del programma (in versione semplificata) è stato mostrato in forma strutturata, cioè diviso in moduli. La fig. 1 riproduce lo schema proposto ai partecipanti al corso.

C'è da notare che l'algoritmo di soluzione del sistema di equazioni differenziali rappresenta solo una piccola parte del programma e, da un certo punto di

vista, neppure la più importante; si vuol sottolineare che la scelta del metodo di soluzione è secondaria in questo contesto. Certamente l'errore che si introduce risolvendo il sistema per via numerica deve essere controllato, in modo che non risulti evidente già durante un periodo; ma questo non giustifica l'uso di formule più complicate del necessario, che oltre tutto possono allungare il tempo di calcolo in modo intollerabile.

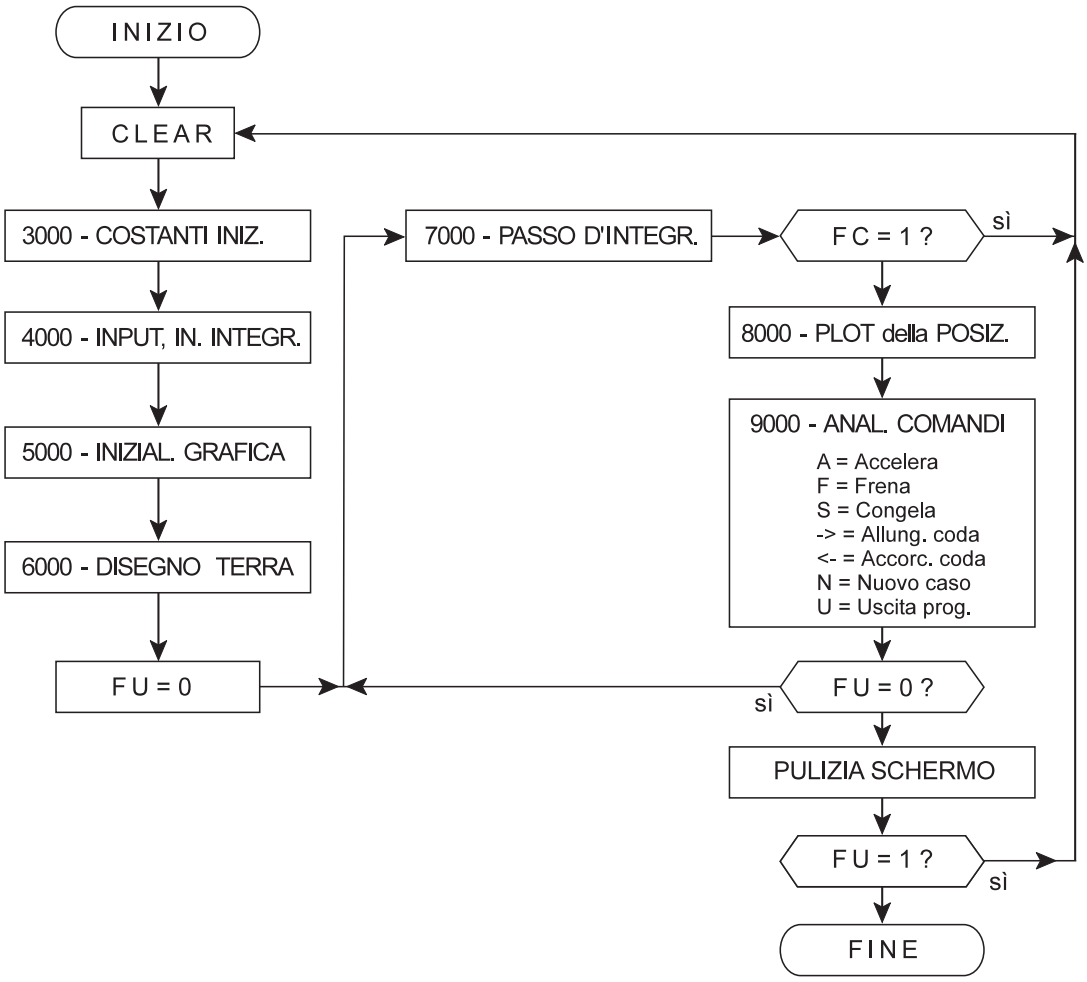


Fig. 1. Schema a blocchi del programma ORBITE

In fig. 2 sono raccolte le equazioni del moto ($a$), il procedimento numerico adottato ($b$), e la relativa traduzione in BASIC ($c$).

Notare l'eliminazione della costante moltiplicativa allo scopo di minimizzare il tempo di calcolo: ciò è possibile con un'opportuna scelta delle unità di misura, come detto in seguito.

Ogni gruppo aveva il compito di stendere il programma relativo ad uno o due moduli: va osservato che questo fatto impone un modo di procedere decisa-

mente "più pulito" di come abitualmente s'impara scrivendo programmi personali, specialmente in BASIC. In altre parole si è ritenuto opportuno introdurre, sia pure in modo operativo e non formale, una qualche idea di programmazione strutturata.

*Equazioni del moto*

$$\begin{aligned} \ddot{x} &= -\frac{GM}{r^3}\, x \\ \ddot{y} &= -\frac{GM}{r^3}\, y \end{aligned} \qquad (a)$$

*Soluzione numerica*

$$\begin{aligned} x_{k+1} &= 2x_k - x_{k-1} - h^2\, \frac{GM}{r_k^3}\, x_k \\ y_{k+1} &= 2y_k - y_{k-1} - h^2\, \frac{GM}{r_k^3}\, y_k \end{aligned} \qquad (b)$$

*Algoritmo* (posto che $h^2GM = 1$)

$$\begin{aligned} \mathsf{S} &= 2 - 1/\mathsf{R}^3 \\ \mathsf{XP} &= \mathsf{S} * \mathsf{X} - \mathsf{XM} \\ \mathsf{YP} &= \mathsf{S} * \mathsf{Y} - \mathsf{YM} \\ \mathsf{XM} &= \mathsf{X} \\ \mathsf{YM} &= \mathsf{Y} \\ \mathsf{X} &= \mathsf{XP} \\ \mathsf{Y} &= \mathsf{YP} \end{aligned} \qquad (c)$$

Fig. 2. Equazioni, soluzione numerica, algoritmo

Uno dei punti di partenza essenziali di una programmazione strutturata è di avere ben chiara la funzione di ogni modulo; occorre inoltre specificare come passare i dati da un modulo all'altro. Si è quindi fatta una distinzione (che in BASIC è certamente artificiosa) tra le variabili di uso generale e quelle interne al singolo modulo. Analogamente abbiamo distinto le variabili propriamente dette da quelle che in realtà definiscono delle costanti: il loro valore viene assegnato nel modulo di inizializzazione ed in caso di necessità viene cambiato in un solo punto del programma.

Un altro aspetto sul quale è necessario riflettere, e che di solito invece viene completamente trascurato, è la questione delle unità di misura. Come si è visto in fig. 2 ($c$), una scelta opportuna può semplificare e rendere più veloce il calcolo; ciò per altro non deve costringere l'utente a pensare in termini di unità "strane." A questo scopo un modulo del programma è dedicato alle necessarie conversioni, cosicché esternamente le unità possono essere scelte nel modo più conveniente.

Infine la divisione in moduli mostra un altro vantaggio: si impara che è possibile, ed anzi preferibile, verificare il corretto funzionamento di ogni modulo, prima di "montare" l'intero programma. Ciò comporta la necessità di preparare opportuni programmi di prova, ma rende certamente più spedita la messa a punto del programma nel suo insieme.

La stesura dei moduli ha occupato 4 sedute al calcolatore, cui sono state associate lezioni su temi generali:

- Hardware e architettura, Sistemi operativi, Linguaggi e Software, Grafica.
- Architettura.
- Sistemi operativi: Gestione delle periferiche: tastiera e unità disco. Manipolazione logica e fisica delle files.
- Linguaggi: linguaggi interpretati e compilati; BASIC, Assembler.
- Grafica vettoriale e a raster; rapporto tra hardware e software nei problemi di grafica; algoritmi. Problema della risoluzione e occupazione di memoria. Colore: codificazione e memoria necessaria.

[1] B. Barsella, E. Fabri, U. Penco: "Obiettivi e strategie di un corso di formazione informatica per insegnanti di fisica." *Atti del XXIII Congresso AIF*, 226 (1986).