di Raffaello De Masi

La programmazione Object Oriented

Continuiamo a parlare di applicazioni e programmazione O.O., e continuiamo a parlarne proprio affrontando il problema di creare una interfaccia utente amichevole, specifica dell'ambiente Mac e simile a quella che vediamo nelle applicazioni già pronte

Facciamo un esempio, utilizzando una applicazione già pronta Macintosh, ad esempio Mac Draw; ci aspettiamo che cliccando un rettangolo appaiano le maniglie, e che afferrando e trascinando il rettangolo stesso per un punto del lato questo si sposterà. Analogamente ci si aspetterà che una volta eseguito lo scrolling attraverso le barre laterali o chiudendo una finestra o rilasciando un menu, venga eseguito il refreshing-redraw dello schermo. Sembra una cosa semplice, da poco, ma nessuno si immaginerebbe davvero quale terribile fatica ci sia nel rendere così agevoli e veloci le operazioni. Certamente chi ha tentato di programmare una finestra di input senza un occhio all'oggetto, sarà ricorso col solito Basic o Pascal, a una serie di input che, una volta seguiti dal Return, sono divenuti granitici e del tutto refrattari a tentativi di correzione attraverso l'istintivo doppio click sullo scritto. Com'è che invece ci viene così facile fare la stessa cosa anche con il più banale w.p.?

La cosa che fa più rabbia, per la verità, è che un banalissimo word processor, con un centinaio di kbyte a disposizione, abbia una interfaccia così curata ed efficiente; reagisce allo scroll, al click, alle chiamate ai menu con una velocità, una immediatezza e soprattutto un ordine che ci pare lontanissimo dal misero risultato del nostro pur valido (a livello concettuale) programma. Perché in quello basta cliccare due volte per correggere un errore mentre nel nostro siamo dovuti ricorrere al vecchio e stantio «Hai sbagliato?», che, in risposta affermativa, ci ripropone i valori immessi chiedendoci l'eventuale modifica? Roba da Apple II di dieci anni fa o da TRS/80 del 1979!

Calma e sangue freddo. Tenete conto di una cosa, di un principio che non ho mai abbandonato nei miei quarant'anni di vita; le cose impossibii sono universali; qualunque cosa vediate di eccezionale, con la pazienza e l'impegno, se ne avete davvero la voglia, potrete riuscirci pure voi. Perciò se sentite i grandi sapientoni della programmazione Mac discutere di locazioni e procedure di sapore astrofisico, non temete! Molto probabilmente stanno bluffando! Posso solo assicurarvi che, nella maggior parte dei casi, i loro capolavori sono pieni della farina del sacco di Inside Macintosh, cui hanno attinto a piene mani per arricchire di piume di pavone le loro deboli creature; l'unico loro merito è che si sono letti i volumoni e sanno dove cercare; comprate anche voi i vostri bei libroni; o magari fatevelo regalare dal vostro distributore Apple; non è necessario che disponiate delle ultime edizioni; la maggior parte delle cose utili sta già nelle prime 64K di ROM, tanto è vero che fin dall'avvento delle 128 e delle 256, per mancanza di materiale da inserire, le ROM contengono le immagini digitalizzate di Fieldman, Sculley, Atkinsons & C.

Perciò, niente paura. Un esempio? Ebbi l'ingenuità, un paio d'anni fa, di chiedere a un collaboratore di Apple

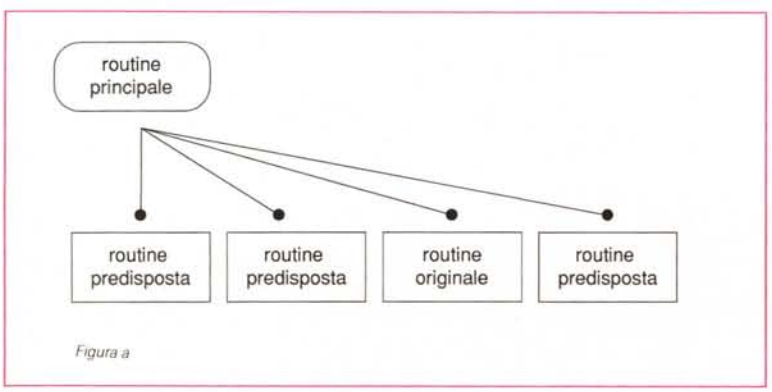


Figura a

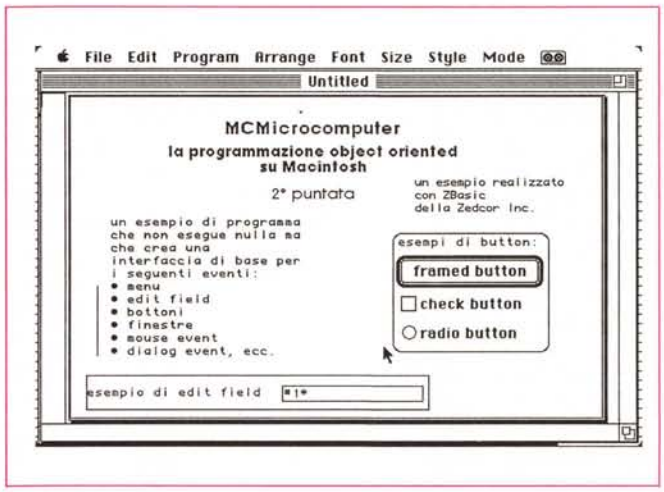
Center una informazione. Molto riassunta era di questo genere: «Come è possibile stampare due pagine successive, una orizzontale e una verticale, senza passare attraverso il driver di stampa?». Ho avuto anche l'improntitudine di dire che il mio programma era scritto in Basic. Il mio interlocutore, di cui non ho poi mai ben capito l'idioma informatico, dopo un gemito di raccapriccio e uno sguardo da top-manager a «vù cumprà», mi dice che certe cose non si possono nemmeno pensare, in Basic.

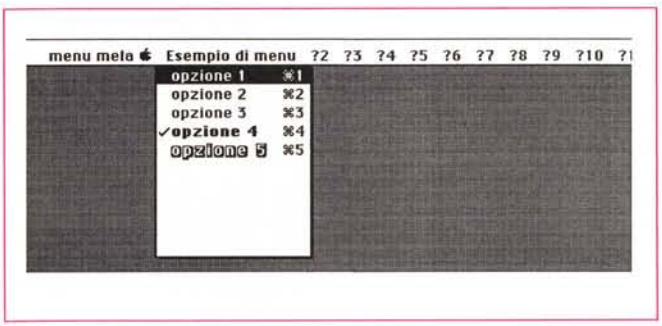
Con la coda (queue) tra le gambe mi sono ritirato nella mia capanna, ho aperto il mio buon Macintosh Revealed (una edizione meno complessa e più commentata dell'«Inside») e dopo una mezz'oretta avevo risolto il problema; la soluzione, guarda un po' l'avevo trovata poi consultando tra le righe il manuale dello Z Basic della Zedcor che, a pagina E-122 parla delle chiamate alle tabelle dei parametri di stampa; una ricerca sistematica con le funzioni PEEKLONG e PRHANDLE mi permetteva di esplorare le locazioni di memoria in un certo intervallo dopo il puntatore PRHANDLE, e settando il print manager una volta in un modo, una volta nell'altro, il gioco (anzi la differenza) era fatta.

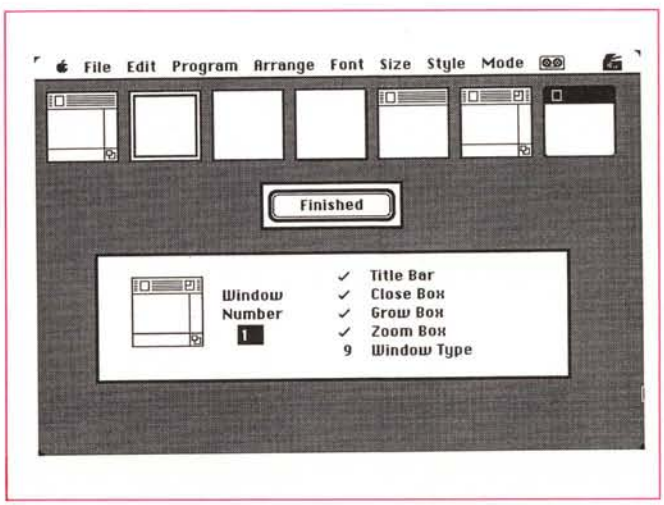
Vi posso assicurare che è possibile fare di tutto nella maggior parte dei linguaggi; forse in C la cosa è più vicina al cuore della vostra macchina, in Pascal vi farà sentire più nobili e «in», ma in Basic e nello splendido ExpertLogo potrete avere gli stessi risultati. D'altro canto non temete! Una passata di compilatore e potrete dire (se ci tenete) che magari l'applicazione l'avete scritta in ADA o in PCL.

Le basi di una applicazione in ambiente Mac

Date un'occhiata al programma che pubblichiamo nelle pagine successive; potete non crederci, ma esso non fa assolutamente nulla tranne presentare la finestra che vedete nella figura, costruire un menu con 5 elementi (non definiti), e fare un poco di scena. Questo codice, neppure tanto complesso e intricato, serve solo a far reagire la macchina ai desideri dell'utente secondo l'interfaccia Mac, vale a dire essere sensibili al tocco del mouse, poter scegliere tra menu diversi, aprire un edit field (si chiama così il campo di input cui ci hanno abituati le applicazioni Macintosh, campi che sono cliccabili e correggibili attraverso il mouse). Ma attenzione, si tratta solo di chiamate a routine estremamente complesse, presenti nel cosiddetto Toolbox; se volessimo analizzare effettivamente cosa fanno queste routine, ci sarebbe da sentir vacillare la mente per lo sforzo.







```
00071 "OPZIONE 2" RETURN
00072 "OPZIONE 3" RETURN
00073 "OPZIONE 4" RETURN
00074 "OPZIONE 5" RETURN
00075 '*************************************************************
00076 '                  • ROUTINE DI BREAK DA TASTIERA •
00077 '*************************************************************
00078 "HANDLE BREAK"
00079 END
00080 '*************************************************************
00081 '                        • ROUTINE DI MOUSE •
00082 '*************************************************************
00083 "HANDLE MOUSE"
00084 :
00085 MACT=MOUSE(0):MX=MOUSE(1):MY=MOUSE(2)
00086 :
00087 RETURN
00088 '*************************************************************
00089 '                      • COSTRUZIONE MENU •
00090 '*************************************************************
00091 "SET MENU"
00092 APPLE MENU "riservato al DA/1"
00093 MENU 1,0,1,"Esempio di menu"
00094 MENU 1,1,1,"opzione 1/1"
00095 MENU 1,2,1,"opzione 2/2"
00096 MENU 1,3,1,"opzione 3/3"
00097 MENU 1,4,2,"<Bopzione 4/4"
00098 MENU 1,5,0,"<Sopzione 5/5"
00099 RETURN
00100 '*************************************************************
00101 '                      • BORDO DI PUSH BUTTON •
00102 '*************************************************************
00103 "FRAME BUTTON"
00104 CALL INSETRECT(T,-4,-4)'                          Setup di T,L,B,R
00105 PEN 3,3,1,8,0'                                  Setup spessore penna
00106 CALL FRAMEROUNDRECT(T,16,16)'                     costruzione bordo
00107 CALL INSETRECT(T,4,4):CALL PENNORMAL              Reset di T,L,B,R
00108 RETURN
00109 '*************************************************************
00110 '                    • ROUTINE DI CREAZ. WINDOW •
00111 '*************************************************************
00112 "BUILD WINDOW"
00113 IF WND=1 THEN WINDOW 1,"",(5,40)-(480,290),9
00114 GOSUB"BUILD EDITS":GOSUB"FILL EDITS"
00115 RETURN
00116 '*************************************************************
00117 '                  • GRAFICA E TESTO NELLA PAGINA •
00118 '*************************************************************
00119 "FORMAT WINDOW"
00120 CALL PENNORMAL
00121 :
00122 LONG IF WINDOW(0)=1
00123    T=3:L=3:B=278:R=453:PEN 1,1,1,8
00124    CALL FRAMERECT(T)
00125     TEXT 130,14,64,0:T=24:L=133:B=39:R=285:TEMP$="MCMicrocomputer"
00126     CALL TEXTBOX(VARPTR(TEMP$)+1,LEN(TEMP$),T, 1)
00127     TEXT ,12,:T=45:L=74:B=71:R=345:TEMP$="la programmazione object oriented"+CHR$(13)
      +"su Macintosh"
00128     CALL TEXTBOX(VARPTR(TEMP$)+1,LEN(TEMP$),T, 1)
00129     TEXT 33,,:T=81:L=173:B=94:R=246:TEMP$="2° puntata"
00130     CALL TEXTBOX(VARPTR(TEMP$)+1,LEN(TEMP$),T, 1)
00131     TEXT 1,9,:T=105:L=37:B=226:R=204:TEMP$="un esempio di programma che non esegue nu
      lla ma che crea una interfaccia di base per i seguenti eventi:"+CHR$(13)+"• menu"
      +CHR$(13)+"• edit field"+CHR$(13)+"• bottoni"+CHR$(13)+"• finestre"+CHR$(13)+"• m
      ouse ev
00132
00133     Ent"+CHR$(13)+"• DIALOG Event, Ecc."
00134     CALL TEXTBOX(VARPTR(TEMP$)+1,LEN(TEMP$),T, 0)
00135     T=165:L=26:B=224:R=26
00136     CALL MOVETO(L,T):CALL LINETO(R,B)
```

Ecco quindi che ci stiamo approssimando piano piano al concetto di programmazione strutturata; le routine di Toolbox sono la prima via per avvicinarsi alla grande tecnica di demandare ad altri il lavoro che toccherebbe a noi. Tutte le buone applicazioni Macintosh, e tra l'altro anche le più facili da svilupare, seguono questo schema, anzi, per essere precisi, seguono lo schema di figura a. qualunque altra applicazione Macintosh, segue il modello della figura, nella quale il programmatore costruisce un set di subroutine (meglio ancora, quando il suo linguaggio lo permette, una serie di macrofunzioni o di procedure), e una routine principale che coordina l'esecuzione di queste; la prima volta che si esegue questo lavoro può trattarsi di un lavoraccio atroce; ma se le cose sono state fatte per dovere e con un minimo di ordine, il programmatore si accorgerà di poter riutilizzare con poche modifiche diversi spezzoni di codice già redatto; da qui a costruirsi una libreria di blocchi e blocchetti riutilizzabili il passo è breve; è quello che fa qualunque meccanico quando si fa costruire dal tornitore l'attrezzino che il rivenditore non gli può procurare e che utilizza volta per volta quando ne ha bisogno.

Ovviamente una parte del programma sarà unica e specifica dell'applicazione, e rappresenterà effettivamente quello che lo rende diverso da tutti gli altri. Qui lo sforzo del programmatore sarà ampio e articolato e, purtroppo non riutilizzabile. Oltre a ciò la tecnica di costruzione di un programma nel modo illustrato in figura a pur traendo vantaggi dall'ampio uso del Toolbox e delle routine di QuickDraw, vantaggi che si rivelano soprattutto in funzione di uniformità

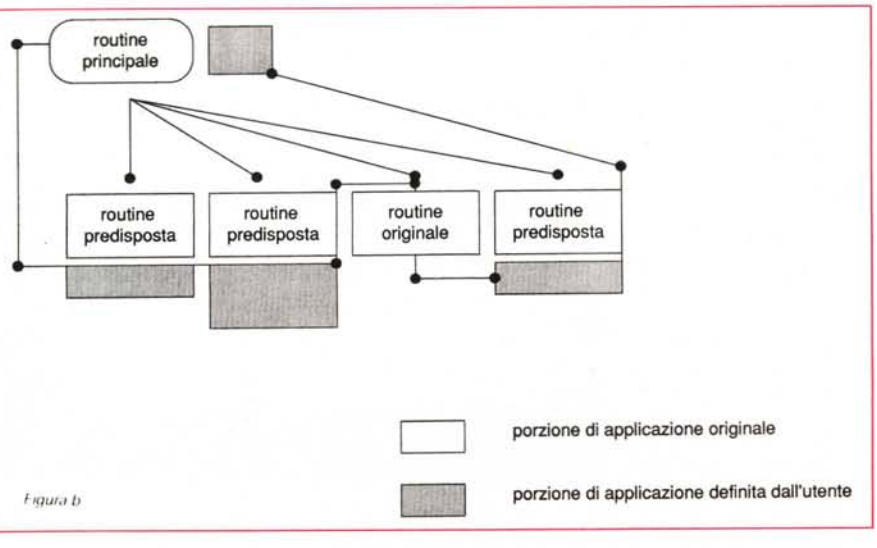


Figura b

del Macintosh Standard User Interface, presenta una grossa difficoltà: proprio l'esteso uso della MSUI tende a rendere da una parte piuttosto complesso il codice, anche solo nella sua stessa lettura (basta dare un'occhiata al misero programmino allegato), dall'altra obbliga sovente a ricostruire (o rimaneggiare) buona parte del codice quando si desidera riutilizzare parte di questo per una nuova applicazione. Siamo solo, quindi, a metà strada; deve cioè esistere una via migliore per risolvere il problema.

Il principio e il nodo da risolvere è: «Esiste un metodo per poter riutilizzare o almeno condividere parte o tutto il codice di una applicazione Macintosh?». Chiaramente le routine di Toolbox risolvono parte di questi problemi, ma proprio per le loro caratteristiche spinte di universalità cui non possono sfuggire,

```
00001  '*************************************************************
00002  '                        MCMicrocomputer
00003  '        La programmazione Object Oriented su Macintosh
00004  ' Il programma che segue rappresenta un set minimo di istru-
00005  ' zioni di base destinate al maneggio di window, button, edit
00006  ' field, grafica da QuickDraw, ecc. (vedasi il testo dell'art.)
00007  ' Il programma è scritto in ZBasic della Zedcor Inc, con l'aiuto
00008  ' della utility "Program Gen." , inserita nel package.
00009  '*************************************************************
00010  '
00011  '*************************************************************
00012  '                • CONFIGURAZIONE DEL SISTEMA •
00013  '*************************************************************
00014  '
00015  '              Tipo di variabile di default : Integer
00016  '       Configurazione dinamica degli array : SI
00017  '                                Option base : 1
00018  '
00019  '*************************************************************
00020  '                • SET UP DELLE VARIABILI •
00021  '*************************************************************
00022  WINDOW OFF:COORDINATE WINDOW:DEF MOUSE=-1:WIDTH -2
00023  DIM T,L,B,R,MY,MX'          Variabili destinate al toolbox
00024  '-------------------------------------------------------------
00025  GOSUB "SET MENU"
00026  WND=1:GOSUB"BUILD WINDOW"
00027  GOTO"EVENT QUEUE"
00028  '*************************************************************
00029  '                          • THE QUEUE •
00030  '*************************************************************
00031  "EVENT QUEUE"
00032  ON DIALOG GOSUB "HANDLE DIALOG":DIALOG ON
00033  ON MENU   GOSUB "HANDLE MENU"  :MENU   ON
00034  ON BREAK  GOSUB "HANDLE BREAK" :BREAK  ON
00035  ON MOUSE  GOSUB "HANDLE MOUSE" :MOUSE  ON
00036  :
00037  "LOOP"
00038  GOTO"LOOP"
00039  :
00040  DIALOG OFF:BREAK OFF:MOUSE OFF:MENU OFF
00041  '*************************************************************
00042  '                       • ROUTINE DI DIALOGO •
00043  '*************************************************************
00044  "HANDLE DIALOG"
00045  :
00046   ACT=DIALOG(0):REF=DIALOG(ACT)
00047  :
00048  IF ACT=3 THEN WINDOW REF:RETURN
00049  IF ACT=4 THEN GOSUB"CAPTURE":WINDOW CLOSE REF:RETURN
00050  IF ACT=5 THEN "FORMAT WINDOW"
00051  IF ACT=11 THEN EDIT FIELD REF,""
00052  :
00053  LONG IF WINDOW(0)=1
00054  END IF
00055  :
00056  RETURN
00057  '*************************************************************
00058  '                          • ROUTINE DI MENU •
00059  '*************************************************************
00060  "HANDLE MENU"
00061  :
00062   MENUID=MENU(0):ITEMID=MENU(1):MENU
00063  :
00064  IF MENUID=255 THEN "RISERVATO AI DA"
00065  ON MENUID GOTO "ESEMPIO DI MENU"
00066  "RISERVATO AI DA" RETURN
00067  '-------------------------------------------------------------
00068  "ESEMPIO DI MENU"
00069  ON ITEMID GOTO "OPZIONE 1","OPZIONE 2","OPZIONE 3","OPZIONE 4","OPZIONE 5"
00070  "OPZIONE 1" RETURN
00137      T=250:L=17:B=261:R=166:TEMP$="esempio di edit field"
00138      CALL TEXTBOX(VARPTR(TEMP$)+1,LEN(TEMP$),T, 0)
00139      T=241:B=271:R=307
00140     CALL FRAMERECT(T)
00141      T=122:L=280:B=133:R=401:TEMP$="esempi di button:"
00142      CALL TEXTBOX(VARPTR(TEMP$)+1,LEN(TEMP$),T, 0)
00143      T=143:L=283:B=163:R=399:GOSUB"FRAME BUTTON"
00144      T=120:L=276:B=222:R=408
00145     CALL FRAMEROUNDRECT(T,16,16)
00146  END IF
00147  :
00148  CALL PENNORMAL:RETURN
00149  '*************************************************************
00150  '                  • CREAZIONE DI EDIT FIELD E BOTTONI •
00151  '*************************************************************
00152  "BUILD EDITS"
00153  TEXT ,,0,0
00154  :
00155  LONG IF WINDOW(0)=1
00156      TEXT 4,9,64:EDIT FIELD 1,"",(182,251)-(302,262),1,1
00157      BUTTON 1 ,1,"framed button",(283,143)-(399,163),1
00158      BUTTON 2 ,1,"check button",(283,173)-(389,188),2
00159      BUTTON 3 ,1,"radio button",(283,198)-(385,213),3
00160    EDIT FIELD 1
00161  END IF
00162  :
00163  RETURN
00164  '*************************************************************
00165  '                • ASSEGNAZIONE DI INPUT A EDIT FIELD •
00166  '*************************************************************
00167  "FILL EDITS"
00168  :
00169  LONG IF WINDOW(0)=1
00170  END IF
00171  :
00172  RETURN
00173  '*************************************************************
00174  '              • LETTURA DEGLI EDIT FIELD E DEI BUTTON •
00175  '*************************************************************
00176  "CAPTURE"
00177  :
00178  LONG IF WINDOW(0)=1
00179  END IF
00180  :
00181  RETURN
```

sono troppo elementari per rappresentare blocchi di programma che eseguono operazioni complesse. Per esempio, nel programma allegato, T, L, B, R, MY, MX rappresentano variabili utilizzate dal Toolbox per l'uso con particolari routine, come la realizzazione di rettangoli o il setup delle famigerate VARPTR; ma si tratta, se ci perdonate la continuazione dell'analogia, di semplici cacciavite, mentre noi cerchiamo la chiave dinamometrica o l'alesatore del cilindro. Come fare?

Invece di indicarvi immediatamente la soluzione, consentiteci di fare un esempio; immaginiamo che questa tecnica esista, vale a dire che esiste una strada che consente di impacchettare molte routine in una routine principale, che può essere riutilizzata. Chiamiamo questa routine col nome di «applicazione espandibile» e cerchiamo di vedere quali caratteristiche dovrebbe avere per essere davvero utile ad uno sviluppatore Macintosh.

L'applicazione espandibile

Una applicazione espandibile, ammesso che possa essere realizzabile, deve essere per definizione quella che fornisce un modo di superare le inefficienze legate allo sviluppo di una applicazione fornendo una applicazione riutilizzabile che implementa quante più possibili routine del MSUI. Questa applicazione di base, che per semplicità può essere, per ora, assimilata a quella della figura a può funzionare solo se modificata secondo quanto si vede nella figura b. Non si tratta di una modifica peregrina. Il principio è che occorre disporre di una struttura di base, di uno scheletro che può essere facilmente modificato; a questo punto il compito del programmatore può essere così riassunto:

- aggiungere del codice ad alcune subroutine
- modificare altre subroutine
- aggiungerne di nuove in vari punti dello sviluppo del programma.

Con queste possibilità, non si è certo ristretti e costretti nello sviluppo di una applicazione. In compenso questa struttura si incaricherà di maneggiare per noi una serie di eventi che sarebbe estremamente tedioso programmare attraverso un approccio tradizionale. Demanderemmo a questi interventi automatici, ad esempio, il ridimensionamento delle finestre e i loro movimenti, la manipolazione di base del testo, la possibilità di lavorare in Multifinder, ecc, in breve, insomma, le caratteristiche comuni a tutte le applicazioni Mac. Questo approccio, ovviamente, leggermente più complesso, si trasforma inoltre (cosa che non guasta) in un notevole decremento della lunghezza del codice, e in una maggiore leggibilità totale del sorgente. Come fare?

Ma lo spazio è tiranno; e dobbiamo fermarci qui, proprio sul più bello. Ne riparleremo la prossima volta.