

pom's

La revue francophone des utilisateurs de l'Apple

Editeur de secteurs Mobby-Disk

Gestion de fichiers par RWTS

Cahier Macintosh

Ecriture en haute résolution

Disk check-up

Initiation à l'assembleur (5)



NUMERO 15 • PRIX 40 F

ISSN : 0294-6068

//c

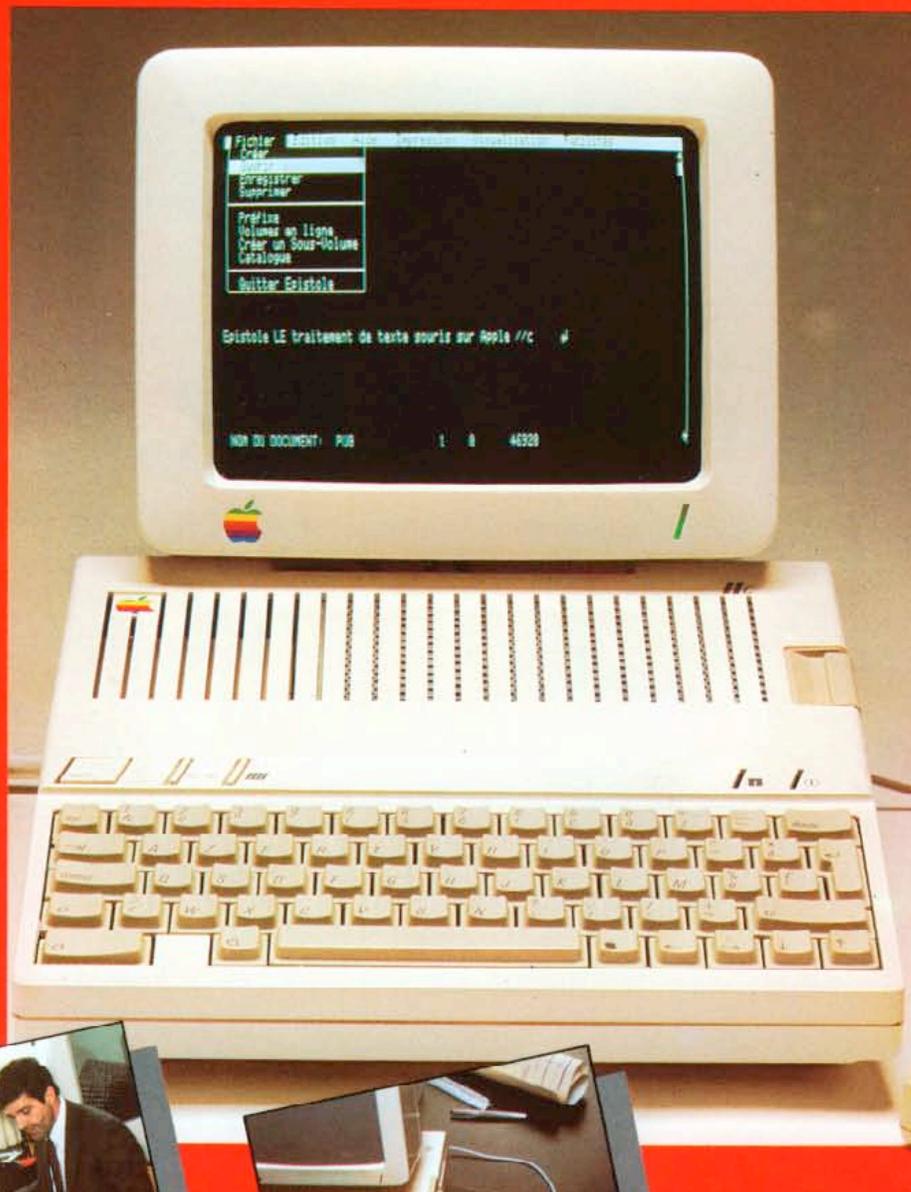
//c

//c

Epistole //c



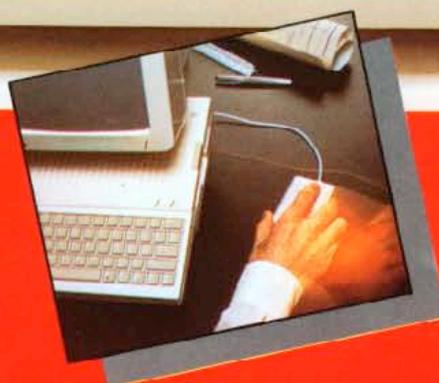
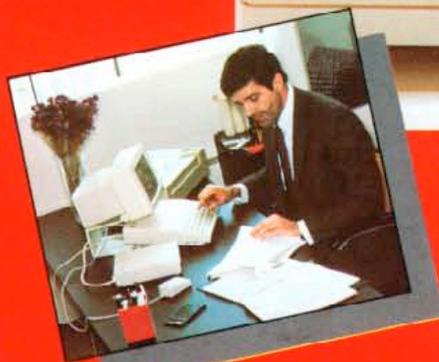
L'écriture souris



Coupez, copiez, collez
tout simplement !

Epistole //c possède
les fonctions de mailing
et calcul intégrés.

Permet de rédiger :
lettres, rapports,
circulaires, mais aussi
factures, devis, budgets,
etc...



**Existe sur Apple IIe.
Démonstration chez votre
revendeur Apple.**

**VERSION
SOFT**

19, rue Ganneron, 75018 Paris - Tél. : (1) 387.94.87

Sommaire	Page	Langage *	Niveau **	Matériel
Editorial par Hervé Thiriez	5			
Mobby Disk par Thierry Le Tallec	7	A	P-T	//e-//c
Fleuves de France par Joseph Pino	17	B	T	//e-//c
Mouse Paint par Guy Lapautre	20		T	//e-//c
Gestion de fichiers par RWTS et DOS 3.3 par Gérard Michel	21	B-A	M-T	//e-//c
Le cahier Macintosh par Hervé Thiriez	32		T	Macintosh
Les routines en ROM du Macintosh par Jean-Luc Bazanegue	33	(B)	M-T	Macintosh
Les caractères programmables sur imprimantes Apple par Apple Seedrin	36		T	
Un éditeur de formes et curseurs par Pom's	43	B	M-T	Macintosh
Les logiciels pour Macintosh	49		T	Macintosh
Omnis 2 à l'essai par Guy Lapautre	50		T	//e-//c
Initiation à l'assembleur (5) par Gérard Michel	53	A	T	//e
HPGRAPH par Nicolas Monsarrat	59	P	M-T	//e-//c
Ecriture en page haute résolution par Erick Ringot	61	B-A	M-T	//e-//c
Disk Check-Up par Alexandre Avrane	66	A	P-T	//e-//c
Bibliographie par Alexandre Duback	71			
Les nouvelles disquettes de Pom's	72			
Micro-Informations par Jean-Michel Gourévitch	73			
Courrier des lecteurs par A. Avrane et A. Duback	75			

* Langage : B(asic) - A(ssembleur) - P(ascal) (B) signifie : relatif au BASIC.

** Niveau : D(ébutant) - M(oyen) - P(rofessionnel) - T(ous).

P-T signifie : programme utilisable par les débutants, mais dont la compréhension est de niveau "Professionnel".

Les annonceurs

Apple : p. 38-39 / ATARI : p. 80 / ATBA Computer III : p. 16 / Club Apple : p. 37 / Dynamite Computer : p. 48 / GMS : p. 70 / Hello : p. 51 / Light : p. 31 / LIST : p. 6 / Maxi : p. 79 / PeachTree : p. 40-41 / P.S.I. : p. 42 / Soft Express : p. 4 / SYBEX : p. 76 / Télécompo : p. 67 / Version Soft : p. 2.

Éditions MEV — 64-70, rue des Chantiers — 78000 Versailles

Directeur de la publication : Hervé Thiriez. Imprimerie Rosay, 94300 Vincennes. Imprimé en France. Dépôt légal : 4^e trimestre 1984.

SOFT EXPRESS

VENTE PAR CORRESPONDANCE

LOGICIELS POUR APPLE*

II (fonctionnent sur II + et //e, le C figurant à côté d'un nom indique que le programme est garanti fonctionner sur //c)

JEUX

Prix T.T.C.

ARCADE

AEE001		DROL (Broderbund)	350,00
AFE001	C	ZAXXON (Datasoft)	380,00
AGE001		MINIT MAN (Penguin)	200,00
AGE002		BOUNCING KAMUNGAS (Penguin)	300,00
AJF001		NIGHT MISSION PINBALL (Sublogic)	350,00
ARF002		AQUATRON (Sierra)	300,00
AEE002		SPARE CHANGE (Broderbund)	350,00
AVE001		WAY OUT (Sirius)	270,00
AWE001	C	THE HEIST (Micro Fun)	410,00
AEE003		GLUMBALL (Broderbund)	310,00
AJE002		FLIGHT SIMULATOR II (Sublogic)	550,00
AUE001	C	SUMMER GAMES (Epyx)	410,00
AEE004		CHOPLIFTER (Broderbund)	340,00
AWE001		MINER 2049ER (Micro Fun)	400,00
ARE003		OIL'S WELL (Sierra)	300,00
AGE004		THE SPY STRIKES BACK (Penguin)	200,00

STRATEGIE

AME001		PROF. TOURNAMENT GOLF (Strategic)	400,00
AAE001		SARGON III (Hayden)	470,00
AME002		BATTLE OF NORMANDY (Strategic)	400,00

AVENTURE

AHE001		MASQUERADE (Phoenix)	350,00
AIE001		LEGACY OF LLYLGAMYN (Sir-Tech)	400,00
ARE001		DARK CRYSTAL (Sierra)	380,00

UTILITAIRES

IMPRESSION EDIT. BASIC TABLEUR COMMUNIC.

ATE001	C	SIDEWAYS (Funk)	620,00
ADE001	C	G.P.L.E. (Beagle Bros)	600,00
ABE002		MULTIPLAN (Microsoft)	1.750,00
AKE001	C	ASCII EXPRESS: THE PRO (United)	1.200,00

//e uniquement : (80 col.)

ORGANISATION TABLEUR+

AQB001		THINK TANK (US) (Living Videotex)	TELEPHONEZ!
ANF001		TK! SOLVER (US) (Software Arts)	3.700,00

//c et //e uniquement : (80 col.)

FICHER IMPRESSION FICHER

ACF001		PFS : FILE (US) (Software Publ.)	1.250,00
ACF002		PFS : REPORT (US) (Software Publ.)	1.250,00
ASF001		HOME CATALOGER (US) (Continental)	520,00

** Macintosh: JEUX

ARCADE

BAG001		MAC VEGAS (US) Roulette-Poker-Blackjack-Keno-Baccarat-Machine à sous-Craps	TÉLÉPHONEZ!
--------	--	--	-------------

STRATEGIE

AAG001		SARGON III (US) (Hayden)	TÉLÉPHONEZ!
AXG001		MILLIONAIRE (US) (Blue Chip)	600,00
AXG002		BARON (US) (Blue Chip)	TÉLÉPHONEZ!
AXG003		TYCOON (US) (Blue Chip)	TÉLÉPHONEZ!
AGG003		TRANSYLVANIA (US) (Penguin)	400,00
AGG005		PENSATE (US) (Penguin)	430,00
AZG001		RUN FOR THE MONEY (US) (Scarb.)	460,00
BAG002		MAC GAMMON (US)	TÉLÉPHONEZ!
BAG003		MAC CHECKERS & REVERSI (US)	TÉLÉPHONEZ!

UTILITAIRES

TABLEUR GRAPHIQUE LANGAGE FICHER IMPRESSION FICHER+

ABG002		MULTIPLAN (US) (Microsoft)	TÉLÉPHONEZ!
ABG005		MICROSOFT CHART (US)	1.300,00
ABG006		MICROSOFT BASIC (US)	1.400,00
ACG001		PFS : FILE (US) (Software Publ.)	1.250,00
ACG002		PFS : REPORT (US) (Software Publ.)	1.250,00
AYG001		FILEVISION (US) (Telos)	2.000,00

Tous nos prix sont valables dans la limite des stocks disponibles. Nous n'encaissons votre règlement que lors de l'expédition de votre commande. Toute commande non expédiée sous un délai maximum de 15 jours sera remboursable sur votre demande. Tous nos envois sont recommandés.

* Apple est une marque déposée Apple Computer Inc.
** Macintosh est sous licence chez Apple Computer Inc.

24, rue d'Armaillé, 75017 Paris - 572 55 15

SOFT EXPRESS

24 rue d'Armaillé 75017 Paris

(1) 572 55 15

Je désire seulement recevoir votre catalogue.
(Joindre deux timbres tarif lettre normal)

P

RÉFÉRENCE	DÉSIGNATION	NOMBRE	PRIX
Participation aux frais de port et d'emballage			25 F
TOTAL			

- Paiement par chèque joint
 CCP
 Paiement par carte bleue VISA

Signature (obligatoire pour paiement par carte de crédit)

N° Date d'expiration

NOM _____ PRÉNOM _____

Rue _____ n° _____

Code postal _____ Ville _____

Téléphone _____

Pour les possesseurs du Macintosh, une bonne et une mauvaise nouvelle à la fois. La bonne nouvelle : le Macintosh 512K est arrivé. La mauvaise nouvelle : il en coûte 10.000 F hors taxe prix public, par rapport au 128K; la mémoire vous revient moins cher en mangeant du poisson... Pour le logiciel itou : la bonne nouvelle est représentée par l'annonce de 80 logiciels, la mauvaise étant que l'on constate dès à présent du retard par rapport aux dates annoncées dans le dossier de presse Apple du Sicob. Il est vrai que la mise au point d'un logiciel sur un équipement possédant les fonctionnalités du Macintosh n'est pas une sinécure. Enfin, fidèles à notre promesse, nous développons dans ce numéro un cahier Macintosh, dans lequel nous essaierons de vous fournir, comme pour l'Apple II, des programmes et des remarques vous apportant plus que la simple lecture des documentations.

Quant à la ligne Apple II, la compatibilité relative du IIc (voir Pom's 13) fait qu'aujourd'hui il se vend encore un IIe pour un IIc, trompant ainsi toutes les prévisions d'Apple. Comme le disent les sages asiatiques, la prévision est un art difficile, surtout quand elle porte sur l'avenir ! Résultat : il faut faire la queue pour acheter un IIe alors qu'on échange sans problème un IIc contre un IIc prêt à emporter.

L'équipe de Pom's vient de s'enrichir de deux personnes. D'ici la fin de l'année, nous espérons donc qu'il nous faudra moins de trois mois pour répondre à votre courrier. Ceci dit, ne prenez pas Pom's pour une entreprise de conseil gratuite : certains lecteurs nous envoient des questions qui ressemblent plutôt à un "cahier des charges" et pour lesquelles la réponse demanderait une véritable intervention de conseil. C'est trop !

Une dernière remarque : certains contributeurs, dont nous admirons la "fluency", nous envoient des programmes dont tous les commentaires sont rédigés en anglais, amenant les lecteurs à croire que nous reproduisons, ô ! honte, des programmes américains. S'il vous plaît, envoyez donc des contributions entièrement en français.

Enfin, ultime bonne nouvelle : Olivier Herz est de retour du Japon avec toujours de l'hexadécimal plein la tête. Gageons que l'auteur de Haifa, du H-Basic et de l'analyseur de syntaxe nous réserve encore quelques surprises !

Hervé Thiriez

Photo de couverture : écran Portac avec batterie incorporée, pour Apple IIc, fabriqué par IEF (voir Micro Informations)
La forme du moniteur n'est pas définitive.

Ont collaboré à ce numéro : Alexandre Avrane - Jean-Luc Bazanegue - Alexandre Duback - Jean-Michel Gourévitch - Guy Lapautre - Thierry Le Tallec - Gérard Michel - Nicolas Monsarrat - Joseph Pino - Erick Ringot - Hervé Thiriez
Rédacteurs : Alexandre Avrane - Gérard Michel
Directeur de la publication, rédacteur en chef : Hervé Thiriez
Dessin : Laurent Bidot
Siège social et abonnements : Editions MEV - 64 70, rue des Chantiers - 78000 Versailles - Tél. : (3) 951.24.43
Rédaction et courrier des lecteurs : 59, bd de Glatigny - 78000 Versailles
Régie publicitaire : Force 7 - Anne Jourdan - 5, place du Colonel Fabien - 75010 Paris - Tél. : (1) 240.22.01
Diffusion N.M.P.P. : Sophie Marniez - Tél. : (1) 240.22.01
Composition : Télécompo - 13-15, avenue du Petit Parc - 94300 Vincennes - Tél. : 328.18.63
Impression : Rosay - 47, avenue de Paris - 94300 Vincennes - Tél. : 328.18.63.

LIST

LIST

LE JOURNAL
DES AMATEURS
DE PROGRAMMATION ^{n°1}

le journal des amateurs de programmation

JUILLET-AOÛT 1984

**A l'essai : le Basic
du nouveau
Thomson MO5**

■ Coup d'œil
sur trois logiciels :
Compactor pour T07
Basic étendu du TI 99/4A
Tool pour Commodore 4

...eurs de poche,
ordinateurs domestiques :
un trésor d'idées
pour mieux programmer

FORUM

Belgique 100 FB - Canada

Si

programmer
un ordinateur est
devenu pour vous
un loisir, un plaisir...

une passion, sachez que **LIST** a été créé
pour vous. **LIST** vous aide à tirer davantage
de votre matériel, à vous perfectionner
dans la conception des programmes
qui "tourneront" sur votre machine.

LIST vous initie aux langages informatiques
et sélectionne les meilleurs livres pour
progresser. **LIST** vous informe de l'actualité
et vous fournit trucs, astuces et idées
pour mieux programmer...

Pour être sûr de ne rater aucun numéro
et pour recevoir **LIST** chez vous,
abonnez-vous!

LIST, LE PLAISIR DE PROGRAMMER

20F chez votre marchand de journaux

FAITES
40 F
D'ECONOMIE!

BULLETIN D'ABONNEMENT

à retourner à **LIST**
(service abonnement)
5, place du Colonel-Fabien,
75491 Paris Cedex 10

Nom : _____

Adresse : _____

Ville : _____

Code postal : [] [] [] [] [] [] Pays : _____

Veillez m'abonner pour 10 numéros au prix
avantageux de **160 F*** au lieu de 200 F. Je fais ainsi
une économie de 40 F sur le prix de vente au numéro.

Je joins mon règlement indispensable libellé
à l'ordre de **LIST**.

* Belgique : 1330 FB ; Suisse : 50 FS ; Canada : 30 \$C ;
autres pays : 210 FF.

Un éditeur de secteurs : MOBBY DISK

Thierry Le Tallec

Le programme le plus utile en cas d'opération malheureuse sur disquette est certainement l'éditeur de secteurs ou "DISK ZAP". Il permet l'affichage à l'écran et la modification des codes hexadécimaux contenus sur la disquette. Il est ainsi possible d'intervenir directement au niveau des secteurs pour recouvrer un fichier "DELETÉ", une disquette qui ne BOOTe plus, ou pour rendre cohérent un enregistrement qui comporte une erreur. L'intérêt de ce type de programme ne se limite pas à la remise en état des secteurs: il devient par exemple possible de suivre précisément le travail du DOS.

Parmi les plus connus, on peut citer INSPECTOR, WATSON et surtout DISKFIXER. Ce dernier aurait ma préférence, mais il lui manque certaines options de WATSON telles que le désassemblage et le choix de la page mémoire à éditer.

Il était intéressant de tenter de faire tenir en moins de 2K les principales fonctions. En effet, le possesseur de carte EPROM peut alors avoir en permanence à portée de la main un utilitaire dont l'usage se révèle fréquent, sans devoir le charger depuis la disquette. A vous de juger si le jeu en valait la chandelle.

Notez que la version publiée ici fonctionne en RAM et n'a donc besoin d'aucune extension particulière.

Après avoir entré et sauvegardé le code-objet, le programme se lance simplement par: BRUN MOBBY-DISK.

Examinons les commandes:

Modes d'affichage

B sélectionne le mode d'affichage mixte, dans lequel le contenu du buffer est affiché simultanément en hexadécimal et en ASCII. C'est le mode d'affichage par défaut. En raison des dimensions de l'écran, seule une moitié du buffer est affichée; on accède à la deuxième moitié en déplaçant le curseur au-delà des limites de la fenêtre.

H affiche le contenu du buffer en hexadécimal seulement.

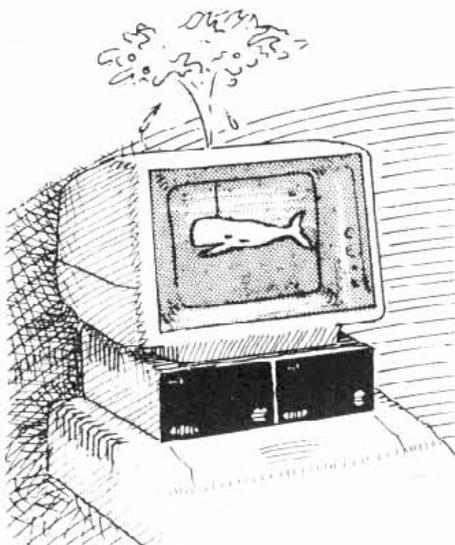
G affiche en ASCII seulement.

Y commute le filtre permettant de n'afficher que les caractères en ASCII normal.

Lecture-écriture

D commande le changement de drive.

R permet la lecture d'un secteur: le curseur se positionne alors sur le numéro de la piste à indiquer en hexadécimal. Si le numéro en cours convient, faire simplement RETURN. Le curseur passe alors au numéro de secteur que l'on contrôle de la même manière. Les flèches gauche et droite commandent la lecture des secteurs précédent et suivant.



W recopie sur la disquette le contenu du buffer. On indiquera les numéros de piste et secteur comme dans l'option lecture.

Edition d'un secteur

I, J, K, M déplacent le curseur respectivement vers le haut, la gauche, la droite, le bas. Le rang de l'octet est affiché sur la ligne supérieure.

: autorise la modification de l'octet où se trouve le curseur. Selon le mode d'affichage, on pourra ainsi entrer un caractère ASCII ou une valeur hexadécimale.

T en mode ASCII commute l'écriture en normal, inverse ou clignotant. Le symbole "#" en bas à droite de l'écran reflète cet état. Les caractères inverses ou clignotants ne seront visibles que si le filtre est supprimé (se reporter à la commande "Y").

RETURN permet la saisie d'un texte ou de valeurs hexadécimales à la volée. Dans les modes mixtes et hexa, tous les octets modifiés apparaissent en inverse jusqu'à l'écriture ou la lecture d'un secteur de la disquette.

ESC sort de ce mode de saisie.

— restitue l'ancienne valeur en cas d'erreur.

X annule toutes les modifications d'un buffer.

P autorise le choix de la page mémoire qui sert de buffer. L'adresse de ce dernier est indiquée en clair au bas de l'écran. On en changera en donnant directement l'adresse au clavier. Par défaut, le buffer est en \$4000.

A noter qu'il est également possible de changer la page du buffer sans passer par la commande P, en utilisant les signes "<" et ">", respectivement pour diminuer ou augmenter de \$100.

Désassembleur

CTRL-D provoque le désassemblage de la page buffer. La barre d'espace fait progresser d'une ligne, la touche RETURN d'une page écran (12 lignes).

ESC sort du mode désassembleur.

Hard-copy d'écran

CTRL-P vous donne une copie imprimante (sortie parallèle) de l'écran, les caractères inverses et clignotants étant recodés avant impression. Cette commande suppose que votre imprimante est connectée dans le slot 1.

Q permet de quitter le programme et rend la main au niveau du moniteur.

NDLR: une version en EPROM est disponible pour les lecteurs possédant une carte RAM/EPROM/TIMER/PORTS PARALLELES. Vous pouvez à cet égard prendre contact avec l'auteur: Mr Le Tallec, 69 rue Sauveur Tobelem - 13007 MARSEILLE.

Si vous n'êtes pas encore abonné à Pom's
prenez le  en " " .

Source Big Mac

```

1 *****
2 *
3 *      ...Mobby-Disk...      *
4 *
5 *      Thierry Le Tallec     *
6 *
7 *****
8
9 * Utilitaire de lecture-(criture de
  disques
10
11 EPROM      KBD          ;0 = RAM,
    1 = EPROM
12          DO      EPROM
13          ORG    $C800
14          ELSE
15          ORG    $8000
16          FIN
17
18 *      constantes:
19
20          LST    OFF
21 FRANCAIS  KBD
22          DO      FRANCAIS
23          LST    ON
24 trackpos =      7
25 sectpos  =     19
26 bytepos  =     29
27 volpos   =     38
28          LST    OFF
29          ELSE
30          LST    ON
31 trackpos =      7
32 sectpos  =     18
33 bytepos  =     27
34 volpos   =     38
35          LST    OFF
36          FIN
37          LST    ON
38 bufpos   =      8
39 CR       =     $8D
40 ctrl_D   =     $84
41 ctrl_P   =     $90
42 ESC      =     $9B
43 *
44 temp     =     $06
45 curCH    =     $08
46 curCV    =     $09
47 dfltval  =     $19
48 val      =     $1A
49 dignum   =     $1B
50 mask     =     $1E
51 byte     =     $E3
52 display  =     $EB          ;0=mixte,
    1=hexa, 2=ASCII
53 filter   =     $EC          ;$00=off,
    $80=on
54 CHtemp   =     $ED
55 base     =     $EE
56 vfst     =     $EF
57 bufptr   =     $FA
58 SBUF     =     $200
59 EBUF     =     $4000
60 *
61 WNDTOP   =     $22
62 WNDBTM   =     $23
63 CH       =     $24
64 CV       =     $25
65 BASL     =     $28
66 BASH     =     $29
67 GBASL    =     $26
68 GBASH    =     $27
69 LENGTH   =     $2F
70 INVFLG   =     $32
71 YSAV1    =     $35
72 PCL      =     $3A
73 PCH      =     $3B
74 RWT3     =     $3D9
75 GETIOB   =     $3E3
76 KBD      =     $C000
77 KBDSTRB  =     $C010
78 SPKR     =     $C030
79 INSTDSP  =     $F8D0
80 PRBLNK   =     $F94A
81 PCADJ    =     $F953
82 INIT     =     $FB2F
83 TABV     =     $FB5B
84 BASCALC  =     $FBC1
85 HOME     =     $FC58
86 CLREOL   =     $FC9C
87 CROUT    =     $FD8E
88 PRBYTE   =     $FDDA
89 COUT     =     $FDED
90 SETNORM  =     $FE34
91 OUTPORT  =     $FE95          ;PR#n
92 *
93 IOB      =     $100
94 SLOT     =     IOB+1
95 DRIVE    =     IOB+2
96 VOLUME   =     IOB+3
97 TRACK    =     IOB+4
98 SECTOR   =     IOB+5
99 USRBUF   =     IOB+8
100 CMDCODE  =     IOB+12
101 ERRCODE  =     IOB+13
102 VOLFND   =     IOB+14
103 SLOTFND  =     IOB+15
104 DRVFND   =     IOB+16
105 *
106
107          TR
108          AST    32
109          LDX    #$FF
110          TXS
111          STX    mask          ;initialis
112          STX    filter       ;caract)re
113          STX    filter       ;filtre en
114          INX
115          STX    byte         ;affichage
116          STX    display      ;adresse d
117          STX    base
118          STX    base         ;$00
119          JSR    INIT
120          JSR    HOME
121          JSR    MSGOUT
122          INV    "GROSSIERS SOFTWARE"
123          ASC    " <Mobby-Disk> "
124          INV    "02/05/84"
125          BRK
126          JSR    MSGAT
127          DFB    0,2
128          LST    OFF
129          DO      FRANCAIS
130          LST    ON
131          ASC    "Piste $ /Secteur $
132          /Octet $ /Vol. $ "
133          LST    OFF
134          ELSE
135          LST    ON
136          ASC    "Track $ /Sector $
137          /Byte $ /Volume $ "
138          LST    OFF

```

```

136     FIN
137     LST  ON
138     BRK
139     LDY  #39
140     LDA  #"-"
141 T1   STA  $480,Y      ;trace 3 1
      lignes de pointilles
142     STA  $580,Y
143     STA  $650,Y
144     DEY
145     BPL  T1
146     JSR  MSGAT
147     DFB  0,21
148     ASC  "Buffer $0000"
149     BRK
150     JSR  MSGAT
151     DFB  33,21
152     ASC  "Drive "
153     BRK
154     JSR  GETIOB
155     STY  temp
156     STA  temp+1
157     LDY  #16
158 MOVIOB LDA  (temp),Y
159     STA  IOB,Y      ;recopie 1
      IOB en zone connue
160     DEY
161     BPL  MOVIOB
162     INY
163     STY  USRBUF
164     STY  bufptr
165     LDA  #)EBUF
166     STA  bufptr+1
167 *
168 WRMSTRT JSR  PRBUF
169     LDA  #$80
170     JSR  PSNBRAC    ;affiche 1
      es "> <"
171 MAIN   JSR  MSGBTM
172     ASC  "->"
173     BRK
174     JSR  GETCMD
175     BCC  MAIN      ;attend co
      mmande suivante
176     BCS  WRMSTRT   ;rafraichi
      t l'cran si carry
177 *
178 GETCMD JSR  GETKEY
179     LDY  #SUBTBL-CMDTBL
180 CMDSRCH DEY
181     BMI  NOTFND
182     CMP  CMDTBL,Y
183     BNE  CMDSRCH
184     TYA
185     ASL
186     TAY
187     LDA  SUBTBL+1,Y
188     PHA
189     LDA  SUBTBL,Y
190     PHA
191     RTS
192 NOTFND CMP  #"0"
193     BLT  BADCMD
194     CMP  #"9"
195     BGE  BADCMD
196     AND  #$0F
197     STA  temp
198     LDA  bufptr+1
199     AND  #$F0
200     ORA  temp
201     STA  bufptr+1
202     SEC
203     RTS
204 BADCMD JSR  SOUND

```

```

205     JMP  GETCMD
206     AST  32
207 CMDTBL ASC  "I" ;monte curseur
208     ASC  "J" ;curseur @ gauche
209     ASC  "K" ;curseur @ droite
210     ASC  "M" ;descend curseur
211     ASC  "Y" ;filtre on/off
212     ASC  "T" ;NRM, INV ou FLS
213     ASC  "G" ;Ascii
214     ASC  "H" ;hexa
215     ASC  "B" ;mixte
216     ASC  "R" ;line un secteur
217     ASC  "W" ;(c)line un secteu
      r
218     ASC  "X" ;restore buffer
219     ASC  "D" ;change de drive
220     ASC  "Q" ;quitter
221     ASC  "P" ;change de page
222     ASC  "<" ;page pr(c)edente
223     ASC  ">" ;page suivante
224     ASC  "-" ;restore un octet
225     ASC  ":" ;saisie d'un octe
      t
226     DFB  CR      ;saisie d'
      une chaine
227     HEX  88      ;secteur p
      r(c)dent
228     HEX  95      ;secteur s
      uivant
229     DFB  ctrl_D  ;d(s)assemb
      le le buffer
230 SUBTBL DA  UP-1
231     DA  LEFT-1
232     DA  RIGHT-1
233     DA  DOWN-1
234     DA  FILTER-1
235     DA  ROLMASK-1
236     DA  SETASCII-1
237     DA  SETHEXA-1
238     DA  SETMIXT-1
239     DA  RDSECT-1
240     DA  WRSECT-1
241     DA  RSTBUF-1
242     DA  CHGDRV-1
243     DA  QUIT-1
244     DA  CHGBUF-1
245     DA  PRUBUF-1
246     DA  NXTBUF-1
247     DA  UNCHNG-1
248     DA  CHGBYT-1
249     DA  CHGDATA-1
250     DA  PRVSCT-1
251     DA  NXTSCT-1
252     DA  DISASM-1
253     AST  32
254 UP     SEC
255     LDA  byte
256     SBC  vfst
257     JMP  PSNBYT
258
259 LEFT   LDY  byte
260     DEY
261     TYA
262     JMP  PSNBYT
263
264 RIGHT  LDY  byte
265     INY
266     TYA
267     JMP  PSNBYT
268
269 DOWN   CLC
270     LDA  byte
271     ADC  vfst
272

```

273	PSNBYT	PHA			
274		LDA	#0		
275		JSR	PSNBRAC	;efface an	
	ciens ">" <"				
276		PLA			
277		STA	byte		
278		JSR	SHOWBYT		
279		LDA	#\$80		
280		JSR	PSNBRAC	;affiche n	
	ouveaux ">" <"				
281		CLC			
282		RTS			
283					
284	PSNBRAC	PHA			
285	NXTBRAC	SEC			
286		LDA	byte		
287		SBC	base	;d(termine	
	la demi-page				
288		STA	temp		
289		LDY	display	;mode mixt	
	e ?				
290		BNE	NXTBR1	;non => sa	
	ute				
291		TAX		;oui => te	
	ste la demi-page				
292		BPL	SAMEPAG		
293		JSR	PRBUF	;change de	
	demi-page si besoin				
294		JMP	NXTBRAC		
295	SAMEPAG	LSR			
296		LSR		;divise pa	
	n 8 (calcule la ligne)				
297		LSR			
298		TAX			
299		LDA	temp		
300		AND	#\$07	;reste = p	
	osition dans la ligne				
301		BPL	NXTBR2	;saute to	
	ujours)				
302	NXTBR1	LSR			
303		LSR			
304		LSR		;divise pa	
	n 16 (modes "G" et "H")				
305		LSR			
306		TAX			
307		LDA	temp		
308		AND	#\$0F	;reste = p	
	osition dans la ligne				
309	NXTBR2	STA	temp		
310		TXA		;r(cup)re	
	ligne				
311		CLC			
312		ADC	#4		
313		STA	curCU		
314		JSR	TABV		
315		LDA	temp	;r(cup)re	
	position dans la ligne				
316		ASL		;double (2	
	caract)res par octet)				
317		CLC			
318		LDY	display	;mode mixt	
	e ?				
319		BEQ	PRBR3	;oui => sa	
	ute				
320		DEY		;mode hexa	
	?				
321		BEQ	PRBR4	;oui => sa	
	ute				
322		ADC	#4	;mode ASCI	
	I : marge = 4				
323		TAY			
324		PLA		;r(cup)re	
	le code				
325		JSR	BRACKETS		
326		STA	(BASL),Y	;met ou en	
	l)ve ">"				
327		INY			
328		STY	curCH		
329		INY			
330		TXA			
331		STA	(BASL),Y	;met ou en	
	l)ve "<"				
332		RTS			
333					
334	PRBR3	ADC	#4	;mode mixt	
	e : marge = 4				
335		ADC	temp	;3 caract)	
	res par octet				
336		TAY			
337		PLA		;r(cup)re	
	le code				
338		JSR	BRACKETS		
339		STA	(BASL),Y	;met ou en	
	l)ve ">"				
340		INY			
341		STY	curCH		
342		INY			
343		INY			
344		TXA			
345		STA	(BASL),Y	;met ou en	
	l)ve "<"				
346		RTS			
347					
348	PRBR4	ADC	#5	;mode hexa	
	: marge = 5				
349		TAY			
350		STY	curCH	;positionn	
	e le curseur				
351		PLA		;r(cup)re	
	le code				
352	FLASHIT	TAX	;\$00=normal, \$80=flas		
	hing				
353		BPL	NOFLASH		
354		LDX	#2		
355	FLASH1	LDA	(BASL),Y	;lit caract	
	t)re (cran				
356		AND	#\$7F		
357		ORA	#\$40		
358		STA	(BASL),Y	;le fait c	
	lignoter				
359		INY			
360		DEX			
361		BNE	FLASH1		
362		RTS			
363	NOFLASH	STY	CH		
364		LDY	byte		
365		JMP	OUTBYTE		
366					
367	SHOWBYT	JSR	MSGAT		
368		DFB	bytepos,2		
369		BRK			
370		LDA	byte		
371		JMP	PRBYTE		
372					
373	BRACKETS	TAX		;A=\$80 =>	
	A=">" X="<"				
374		BMI	BRACK		
375		LDA	"#"	;A=\$00 =>	
	A=" " , X=" "				
376		TAX			
377		RTS			
378					
379	BRACK	LDA	#'>'+\$40		
380		LDX	#'<'+\$40		
381		RTS			
382		AST	32		

```

383 FILTER LDA filter ;met ou en
    l)ve le filtre
384 EOR #$80
385 STA filter
386 SEC
387 RTS
388 AST 32
389 ROLMASK LDA mask
390 LSR
391 CMP #$1F
392 BNE ROLMSK1
393 LDA #$FF
394 ROLMSK1 STA mask
395 RTS
396 AST 32
397 SETASCII LDA #0 ;mode = AS
    CII (G)
398 LDY #2
399 BNE SETMODE
400 SETHEXA LDA #0 ;mode = he
    xa (H)
401 LDY #1
402 BNE SETMODE
403 SETMIXT LDA base ;mode = mi
    xte (B)
404 LDY #0
405 SETMODE STA base
406 STY display
407 SEC
408 RTS
409 AST 32
410 RDSECT JSR GETPRMS
411 RDSECT1 LDA #1
412 SETRWTS STA CMDCODE
413 JSR GORWTS
414 SEC
415 RTS
416
417 GETPRMS LDA #2 ;VTAB 2
418 JSR TABU
419 LDY #trackpos ;HTAB n
420 LDA TRACK
421 JSR GETBYTE ;attend nu
    m(nro de piste
422 CMP #36 ;max 36 pi
    stes
423 BGE GETPRMS
424 STA TRACK
425 GETPRMS2 LDY #sectpos ;HTAB n
426 LDA SECTOR
427 JSR GETBYTE ;attend nu
    m(nro de secteur
428 CMP #16 ;max 16 se
    cteurs
429 BGE GETPRMS2
430 STA SECTOR
431 SEC
432 RTS
433 AST 32
434 WRSECT JSR GETPRMS
435 LDA #2
436 BNE SETRWTS
437 AST 32
438 PRVBUF DEC bufptr+1 ;buffer =
    page pr(c(dente
439 SEC
440 RTS
441 AST 32
442 NXTBUF INC bufptr+1 ;buffer =
    page suivante
443 SEC
444 RTS
445 AST 32
446 PRVSCT DEC SECTOR ;lit le se
    cteur pr(c(dent
447 BPL PRV1
448 LDA #$0F
449 STA SECTOR
450 DEC TRACK
451 BPL PRV1
452 LDA #$22
453 STA TRACK
454 PRV1 JMP RDSECT1
455 AST 32
456 NXTSCT LDY SECTOR ;lit le se
    cteur suivant
457 INY
458 CPY #16
459 BNE NXT1
460 LDY #0
461 LDX TRACK
462 INX
463 CPX #$23
464 BNE NXT0
465 LDX #0
466 NXT0 STX TRACK
467 NXT1 STY SECTOR
468 JMP RDSECT1
469 AST 32
470 CHGDRV LDA DRIVE ;change de
    drive
471 EOR #$03
472 STA DRIVE
473 SEC
474 RTS
475 AST 32
476 QUIT JSR INIT ;efface l'
    (cran et quitte le programme
477 JSR HOME
478 JMP $FF65
479 AST 32
480 CHGBUF LDA #21
481 JSR TABU
482 LDA bufptr+1
483 LDY #bufpos
484 JSR GETBYTE
485 STA bufptr+1
486 SEC
487 RTS
488 AST 32
489 UNCHNG LDY byte ;restore l'
    'octet ol est le curseur
490 LDA SBUF,Y
491 STA (bufptr),Y
492 JMP CHGBYT4
493 AST 32
494 CHGDATA JSR CHGBYT ;permet l'
    (dition du buffer
495 JMP CHGDATA
496 AST 32
497 CHGBYT LDA curCU ;permet l'
    (dition d'un octet
498 JSR TABU
499 LDY byte
500 LDA (bufptr),Y
501 LDY curCH
502 STY CH
503 LDX display ;mode mixt
    e ?
504 CPX #2 ;mode ASCII
    I ?
505 BNE CHGBYT2 ;non => sa
    ute
506 JSR GETKEY
507 JSR DOMASK
508 JMP CHGBYT3

```

509	CHGBYT2	JSR	GETBYTE	
510	CHGBYT3	LDY	byte	
511		STA	(bufptr),Y ;range la	
			nouvelle valeur	
512	CHGBYT4	INY		
513		STY	CHtemp	
514		JSR	PRBUF	
515		LDA	CHtemp	
516		JMP	PSNBYT	
517		AST	32	
518	DISASM	LDA	#4 ;d(sassemb	
			le le buffer	
519		STA	WNDDTOP	
520		LDA	#20	
521		STA	WNDBTM	
522		JSR	HOME	
523		LDA	#18 ;VTAB 18	
524		JSR	TABV	
525		LDA	bufptr	
526		STA	PCL	
527		LDA	bufptr+1	
528		STA	PCH	
529	DIS0	LDX	#16	
530	DIS1	STX	temp	
531	DIS2	JSR	INSTDSP	
532		LDY	LENGTH	
533	DIS3	LDA	(PCL),Y	
534		STA	\$5F3,Y	
535		DEY		
536		BPL	DIS3	
537		JSR	PCADJ	
538		STA	PCL	
539		STY	PCH	
540		DEC	temp	
541		BNE	DIS2	
542	DIS4	LDA	KBD	
543		BPL	DIS4	
544		JSR	GETCP	
545		LDX	#1	
546		CMP	#CR	
547		BEQ	DIS0	
548		BNE	DIS1	
549		AST	32	
550	PRBUF	LDA	#4 ;affiche 1	
			e contenu du buffer	
551		STA	CV ;VTAB 4	
552		LDY	display ;mode mixt	
			e ?	
553		BNE	PRBUF0 ;non => sa	
			ute	
554		SEC		
555		LDA	byte ;oui => dC	
			termine la demi-page	
556		SBC	base	
557		BPL	PRBUF0	
558		LDA	base	
559		EOR	#\$80 ;change de	
			demi-page si besoin	
560		STA	base	
561	PRBUF0	LDA	base	
562		STA	temp	
563	PRBUF1	LDA	CV	
564		JSR	TABV	
565		JSR	PRLINE	
566		INC	CV	
567		LDA	CV	
568		CMP	#20	
569		BNE	PRBUF1	
570	SHOWPRMS	LDA	#2 ;VTAB 2	
571		JSR	TABV	
572		LDA	TRACK	
573		LDY	#trackpos	
574		JSR	DSPBYT ;affiche 1	
			e num(ro de piste	
575		LDA	SECTOR	
576		LDY	#sectpos	
577		JSR	DSPBYT ;affiche 1	
			e num(ro de secteur	
578		LDA	byte	
579		LDY	#bytepos	
580		JSR	DSPBYT ;affiche 1	
			e rang de l'octet	
581		LDA	VOLFND	
582		LDY	#volpos	
583		JSR	DSPBYT ;affiche 1	
			e num(ro de volume	
584		LDA	DRIVE	
585		CMP	DRVFND	
586		BNE	INUDRV	
587		ORA	#\$80	
588	INUDRV	ORA	#\$30	
589		STA	\$6F7 ;affiche 1	
			e drive en service	
590		LDA	#21 ;VTAB 21	
591		JSR	TABV	
592		LDA	bufptr+1	
593		LDY	#bufpos ;affiche 1	
			'adresse du buffer	
594	DSPBYT	STY	CH	
595		JMP	PRBYTE	
596				
597	PRLINE	LDY	#0	
598		STY	CH	
599		STY	temp+1 ;position	
			dans la ligne	
600		LDA	#"#"	
601		JSR	COUT	
602		LDA	temp	
603		JSR	PRBYTE ;affiche "	
			#", l'adresse	
604		JSR	OUTSPACE	
605		JSR	OUTSPACE	
606	HEXASC	LDY	temp	
607		LDA	(bufptr),Y	
608		LDX	display ;teste le	
			mode d'affichage	
609		BEQ	MIXTE	
610		DEX		
611		BEQ	HEXA	
612		JSR	OUTCHAR	
613		JSR	OUTSPACE	
614		JMP	ASCII	
615	HEXA	JSR	OUTBYTE	
616	ASCII	INC	temp	
617		INC	temp+1	
618		LDA	temp+1	
619		CMP	#16	
620		BNE	HEXASC	
621		STA	vfst	
622		JMP	CLREOL	
623	MIXTE	PHA		;** MODE M
	IXTE **			
624		JSR	OUTBYTE	
625		JSR	OUTSPACE	
626		LDY	CH ;sauve CH	
627		LDA	#32	
628		CLC		
629		ADC	temp+1	
630		STA	CH	
631		PLA		
632		JSR	OUTCHAR	
633		STY	CH ;restore C	
			H	
634		INC	temp ;pointe l'	
			octet suivant	
635		INC	temp+1 ;compte ju	

```

      squ' @ 8 fois
636 LDA temp+1
637 CMP #8
638 BNE HEXASC
639 STA vfst
640 LDX #3
641 JMP PRBLNK
642
643 OUTBYTE LDX #$FF
644 LDA (bufptr),Y
645 CMP SBUF,Y
646 BEQ NORMAL
647 LDX #$3F
648 NORMAL STX INVFLG
649 JSR PRBYTE
650 JMP SETNORM
651
652 OUTSPACE LDA #" "
653 OUTCHAR CMP #$A0
654 BGE OUTCH2
655 BIT filter
656 BPL OUTCH2
657 ORA #$80
658 CMP #$E0
659 BLT OUTCH1
660 AND #%10111111
661 OUTCH1 CMP #$A0
662 BGE OUTCH2
663 LDA #"_"
664 OUTCH2 STY YSAV1
665 LDY CH
666 STA (BASL),Y
667 INC CH
668 LDY YSAV1
669 RTS
670 AST 32
671 GETBYTE STA dfltval ;range la
      valeur par d(faut
672 STY CHtemp ;et la pos
      ition dans la ligne
673 GETBYT1 LDA CHtemp
674 STA CH
675 LDA #2
676 STA dignum ;deux cara
      ct)res par octet
677 LDA #0
678 STA val ;initialis
      e la valeur @ z(ro
679 LDA dfltval
680 JSR PRBYTE ;affiche l
      a valeur par d(faut
681 LDY CHtemp
682 STY CH
683 NXTDIG LDY CH
684 LDA (BASL),Y ;lit un ch
      iffre sur l'(cran
685 STA temp+1 ;m(morise
      ce chiffre
686 JSR GETKEY ;attend un
      e touche
687 STA (BASL),Y ;affiche l
      e chiffre tap(
688 CMP #$88 ;backspace
      ?
689 BEQ GETBYT1 ;oui => re
      commence
690 CMP #CR ;return ?
691 BEQ GETBYT3 ;oui => re
      vient tout de suite
692 CMP #"0" ;teste si
      chiffre hexa (0-F)
693 BGE GOOD0
694 BADNUM JSR SOUND ;sinon "so

```

```

      nne",
695 LDA temp+1 ;r(cup)re
      chiffre original,
696 LDY CH
697 STA (BASL),Y ;le remet
      sur l'(cran,
698 JMP NXTDIG ;et va att
      endre autre chose.
699 GOOD0 CMP #"9"+1
700 BLT GOOD1
701 CMP #"A"
702 BLT BADNUM
703 CMP #"F"+1
704 BGE BADNUM
705 SBC #6
706 GOOD1 AND #$0F ;si le chi
      ffre est valable,
707 STA temp+1
708 LDA val
709 ASL
710 ASL
711 ASL
712 ASL
713 ORA temp+1 ;l'introdu
      it @ la droite de val,
714 STA val
715 DEC dignum
716 BNE GETBYT2 ;et passe
      au chiffre suivant.
717 LDA val
718 SEC
719 RTS
720 GETBYT2 INC CH
721 LDY CH
722 LDA #" "
723 STA (BASL),Y
724 JMP NXTDIG
725 GETBYT3 LDA dignum
726 CMP #2
727 BNE GETBYT4
728 LDA dfltval
729 STA val ;revient a
      vec la valeur par d(faut
730 GETBYT4 LDA CHtemp
731 STA CH
732 LDA val
733 JSR PRBYTE
734 LDA val
735 SEC
736 RTS
737 AST 32
738 GORWTS LDA #0
739 STA VOLUME
740 LDA bufptr+1
741 STA USRBUF+1
742 LDY #IOB
743 LDA #>IOB
744 JSR RWTS
745 BCC SAVBUF
746 JSR SOUND
747 LST OFF
748 DO FRANCAIS
749 LST ON
750 JSR MSGBTM
751 ASC "Erreur d"
752 BRK
753 LDA CMDCODE
754 CMP #2
755 BEQ WRERR
756 JSR MSGOUT
757 ASC "e lecture"
758 BRK
759 RETRY JSR MSGOUT

```

760	ASC	","nouvel essai ?"			
761	BRK				
762	JSR	GETKEY			
763	CMP	"Y"			
764	BEQ	GORWTS			
765	CMP	"0"			
766	BEQ	GORWTS			
767	SEC				
768	RTS				
769	WRERR	JSR	MSGOUT		
770	ASC	"écriture"			
771	BRK				
772	JMP	RETRY			
773	LST	OFF			
774	ELSE				
775	LST	ON			
776	JSR	MSGBTM			
777	ASC	"can't "			
778	BRK				
779	LDA	CMDCODE			
780	CMP	#2			
781	BEQ	WRERR			
782	JSR	MSGOUT			
783	ASC	"read"			
784	BRK				
785	RETRY	JSR	MSGOUT		
786	ASC	","rety ?"			
787	BRK				
788	JSR	GETKEY			
789	CMP	"Y"			
790	BEQ	GORWTS			
791	SEC				
792	RTS				
793	WRERR	JSR	MSGOUT		
794	ASC	"write"			
795	BRK				
796	JMP	RETRY			
797	FIN				
798	LST	ON			
799	AST	32			
800	SAVBUF	LDY	#0		
801	SAVBUF1	LDA	(bufptr),Y ;recopie l		
			e buffer de travail		
802		STA	SBUF,Y ; dans le		
			buffer de sauvegarde		
803		INY			
804		BNE	SAVBUF1		
805		SEC			
806		RTS			
807		AST	32		
808	RSTBUF	LDY	#0		
809	RSTBUF1	LDA	SBUF,Y ;recopie l		
			e buffer de sauvegarde		
810		STA	(bufptr),Y ; dans		
			le buffer de travail		
811		INY			
812		BNE	RSTBUF1		
813		SEC			
814		RTS			
815		AST	32		
816	GETKEY	JSR	RDKEY		
817	GETCP	BIT	KBDSTRB		
818		CMP	#ctrl_P ;ctrl-P ?		
819		BEQ	HARDCOPY ;oui => co		
			pie l'(cran		
820		CMP	#ESC ;escape ?		
821		BNE	GETRTS		
822		LDX	#\$FF		
823		TXS			
824		JMP	WRMSTRT ;oui => re		
			start		
825	GETRTS	RTS			
826					
827	RDKEY	LDY	CH		
828		LDA	(BASL),Y		
829		PHA			
830		LDA	"#"		
831		JSR	DOMASK		
832		STA	\$7F7		
833	RDK1	LDA	"_"		
834		JSR	BLINK		
835		BMI	KEYFOUND		
836		PLA			
837		PHA			
838		JSR	BLINK		
839		BPL	RDK1		
840	KEYFOUND	PLA			
841		STA	(BASL),Y		
842		TXA			
843		RTS			
844					
845	BLINK	STA	(BASL),Y		
846		LDX	#50		
847		STX	temp		
848	BLINK1	LDA	KBD		
849		BMI	BLINK2		
850		DEX			
851		BNE	BLINK1		
852		DEC	temp		
853		BNE	BLINK1		
854	BLINK2	STA	KBDSTRB		
855		TAX			
856		RTS			
857					
858	DOMASK	CMP	#\$E0 ;minuscule		
			?		
859		BGE	DMSK1 ;oui => ne		
			change pas		
860		LDX	mask ;non: test		
			e si FLS ou INV		
861		CPX	#\$7F		
862		BNE	DMSK2		
863		CMP	#\$C0		
864		BGE	DMSK2		
865		EOR	#\$C0		
866	DMSK1	RTS			
867	DMSK2	AND	mask		
868		RTS			
869		AST	32		
870	HARDCOPY	LDX	#\$2F		
871	HARD0	LDA	0,X		
872		STA	IOB+17,X ;sauve la		
			page z(ro		
873		DEX			
874		BPL	HARD0		
875		LDX	#22		
876		STX	WINDTOP		
877		STX	WINDBTM		
878		LDA	#1		
879		JSR	OUTPORT ;PR#1		
880		LDA	#0		
881		STA	temp		
882	HARD1	JSR	BASCALC		
883		LDA	BASL		
884		STA	GBASL		
885		LDA	BASH		
886		STA	GBASH		
887		LDA	#22 ;VTAB 22		
888		JSR	TABV		
889		JSR	CROUT		
890		LDY	#0		
891	HARD2	LDA	(GBASL),Y		
892		CMP	#\$20 ;inverse ?		
893		BGE	NOINV		
894		ADC	#\$C0 ;oui => no		
			rma1		

```

895 NOINV    CMP    ##60      ;flashing
?
896          BGE    NOFLS
897          DRA    ##80      ;oui => no
      rmal
898 NOFLS    CMP    ##A0      ;caract)re
      de controle ?
899          BGE    NOCTRL
900          ADC    ##40      ;oui => no
      rmal
901 NOCTRL   CMP    ##FF      ;l'imprima
      nte n'aime pas celui-10
902          BNE    PRINT
903          LDA    # " "
904 PRINT    JSR    COUT
905          INY
906          CPY    #40        ;fini lign
      e ?
907          BNE    HARD2      ;non => co
      ntinue
908          INC    temp
909          LDA    temp
910          CMP    #23
911          BLT    HARD1
912          JSR    CROUT
913          LDA    #0
914          JSR    OUTPORT
915          LDX    ##2F
916 HARD3    LDA    IOB+17,X
917          STA    0,X        ;restore 1
      a page z(ro
918          DEX
919          BPL    HARD3
920          JMP    GETKEY
921          AST    32
922 SOUND    LDX    ##80
923 S1       TXA
924          EOR    ##80
925          TAY
926 S2       DEY
927          BNE    S2
928          BIT    SPKR
929          TXA
930          ADC    ##80

```

```

931          TAY
932 S3       DEY
933          BNE    S3
934          BIT    SPKR
935          DEX
936          BNE    S1
937          RTS
938          AST    32
939 MSGBTM   JSR    MSGAT
940          DFB    0,23,0
941          JSR    CLREOL
942 *
943 MSGOUT   PLA
944          STA    temp
945          PLA
946          STA    temp+1
947          LDY    #0
948          BEQ    MSGNXT
949 *
950 MSGAT    PLA
951          STA    temp
952          PLA
953          STA    temp+1
954          LDY    #1
955          LDA    (temp),Y
956          STA    CH
957          INY
958          LDA    (temp),Y
959          JSR    TABV
960 MSGNXT   INY
961          LDA    (temp),Y
962          BEQ    MSGEND
963          JSR    COUT
964          JMP    MSGNXT
965 MSGEND   CLC
966          TYA
967          ADC    temp
968          TAY
969          LDA    temp+1
970          ADC    #0
971          PHA
972          TYA
973          PHA
974          RTS

```

Récapitulation

8000- A2 FF 9A 86 1E 86 EC E8	80E8- 12 C9 B9 80 0E 29 0F 85	81E8- 91 29 C8 CA D0 F4 60 84
8008- 86 E3 86 EB 86 EE 20 2F	80F0- 06 A5 FB 29 F0 05 06 85	81F0- 24 A4 E3 4C 1E 84 20 4F
8010- FB 20 58 FC 20 45 86 07	80F8- FB 38 60 20 22 86 4C CC	81F8- 86 10 02 00 A5 E3 4C DA
8018- 12 0F 13 13 09 05 12 13	8100- 80 C9 CA CB CD D9 D4 C7	8200- FD AA 30 04 A9 A0 AA 60
8020- 20 13 0F 06 14 17 01 12	8108- C8 C2 D2 D7 D8 C4 D1 D0	8208- A9 7E A2 7C 60 A5 EC 49
8028- 05 A0 BC CD CF C2 C2 D9	8118- 45 91 40 81 54 81 5B 91	8210- 80 85 EC 38 60 A5 1E 4A
8030- AD C4 C9 D3 C8 BE A0 30	8120- 0C 82 14 82 20 82 26 82	8218- C9 1F D0 02 A9 FF 85 1E
8038- 32 2F 30 35 2F 38 34 00	8128- 2C 82 36 82 68 82 48 85	8220- 60 A9 00 A0 02 00 0A A9
8040- 20 4F 86 00 02 D0 C9 D3	8130- AB 82 85 82 BE 82 6F 82	8228- 00 A0 01 D0 04 A5 EE A0
8048- D4 C5 A0 A4 A0 A0 AF D3	8138- 73 82 CE 82 DE 82 D8 82	8230- 00 85 EE 84 EB 38 60 20
8050- C5 C3 D4 C5 D5 D2 A0 A4	8140- 77 82 8E 82 0C 83 38 A5	8238- 44 82 A9 01 8D 0C 01 20
8058- A0 A0 AF CF C3 D4 C5 D4	8148- E3 E5 EF 4C 61 81 A4 E3	8240- 06 84 38 60 A9 02 20 5B
8060- A0 A4 A0 A0 AF D6 CF CC	8150- 98 98 4C 61 81 A4 E3 C9	8248- FB A0 07 AD 04 01 20 54
8068- AE A0 A4 A0 A0 00 A0 27	8158- 98 4C 61 81 18 A5 E3 65	8250- 84 C9 24 80 EF 8D 04 01
8070- A9 AD 99 80 04 99 80 05	8160- EF 48 A9 00 20 74 81 68	8258- A0 13 AD 05 01 20 54 84
8078- 99 50 06 88 10 F4 20 4F	8168- 85 E3 20 F6 81 A9 80 20	8260- C9 10 80 F4 8D 05 01 38
8080- 86 00 15 C2 D5 C6 C6 C5	8170- 74 81 18 60 48 38 A5 E3	8268- 60 20 44 82 A9 02 D0 CC
8088- D2 A0 A4 B0 80 B0 B0 00	8178- E5 EE 85 06 A4 EB D0 13	8270- C6 FB 38 60 E6 FB 38 60
8090- 20 4F 86 21 15 C4 D2 C9	8180- AA 10 06 20 51 83 4C 75	8278- CE 05 01 10 0F A9 0F 8D
8098- D6 C5 A0 00 20 E3 03 94	8188- 81 4A 4A 4A AA A5 06 29	8280- 05 01 CE 04 01 10 05 A9
80A0- 06 85 07 A0 10 B1 06 99	8190- 07 10 09 4A 4A 4A 4A AA	8288- 22 8D 04 01 4C 3A 82 AC
80A8- 00 01 88 10 F8 C8 8C 08	8198- A5 06 29 0F 85 06 8A 18	8290- 05 01 C8 C0 10 00 0F A0
80B0- 01 84 FA A9 40 85 FB 20	81A0- 69 04 85 09 20 5B FB A5	8298- 00 AE 04 01 E8 E0 23 D0
80B8- 51 83 A9 80 20 74 81 20	81A8- 06 0A 18 A4 EB F0 14 88	82A0- 02 A2 00 8E 04 01 8C 05
80C0- 3C 86 AD BE 00 20 CC 80	81B0- F0 25 69 04 A8 68 20 01	82A8- 01 4C 3A 82 AD 02 01 49
80C8- 90 F5 B0 EB 20 55 85 A0	81B8- 82 91 28 C9 84 08 C9 9A	82B0- 03 8D 02 01 38 60 20 2F
80D0- 17 88 30 11 09 01 81 D0	81C0- 91 28 60 69 04 65 06 A8	82B8- FB 20 58 FC 4C 65 FF A9
80D8- F8 98 0A A8 89 19 81 48	81C8- 68 20 01 82 91 28 C3 84	82C0- 15 20 58 FB A5 FB A0 08
80E0- B9 18 81 48 60 C9 80 90	81D0- 08 C8 C8 8A 91 28 60 69	82C8- 20 54 84 85 FB 38 60 A4
	81D8- 05 A8 84 08 68 AA 10 0F	82D0- E3 89 00 02 91 FA 4C 02
	81E0- A2 02 B1 28 29 7F 09 40	82D8- 83 20 DF 82 4C D9 82 A5
		82E0- 09 20 58 FB A4 E3 B1 FA

82E9- A4 08 84 24 A6 EB E0 02
 82F0- 00 09 20 55 85 20 A1 85
 82F8- 4C FE 92 20 54 84 A4 E3
 8300- 91 FA C8 84 ED 20 51 83
 8308- A5 ED 4C 61 81 A9 04 85
 8310- 22 A9 14 85 23 20 58 FC
 8318- A9 12 20 58 FB A5 FA 85
 8320- 3A A5 FB 85 38 A2 10 86
 8328- 06 20 00 F8 A4 2F B1 3A
 8330- 99 F3 05 88 10 F8 20 53
 8338- F9 85 3A 84 3B C6 06 D0
 8340- E8 AD 00 C0 10 FB 20 58
 8348- 85 A2 01 C9 8D F0 D6 D0
 8350- D6 A9 04 85 25 A4 EB D0
 8358- 00 38 A5 E3 E5 EE 10 06
 8360- A5 EE 49 80 85 EE A5 EE
 8368- 85 06 A5 25 20 58 FB 20
 8370- B0 93 E6 25 A5 25 C9 14
 8378- 00 F0 A9 02 20 58 FB AD
 8380- 04 01 A0 07 20 86 83 AD
 8388- 05 01 A0 13 20 86 83 A5
 8390- E3 A0 10 20 86 83 AD 0E
 8398- 01 A0 26 20 86 83 AD 02
 83A0- 01 C0 10 01 D0 02 09 80
 83A8- 09 30 8D F7 06 A9 15 20
 83B0- 58 FB A5 FB A0 08 84 24
 83B8- 4C DA FD A0 00 84 24 84
 83C0- 07 A9 A4 20 ED FD A5 06
 83C8- 20 DA FD 20 31 84 20 31
 83D0- 84 A4 06 B1 FA A6 EB F0
 83D8- 1E CA F0 09 20 33 84 20
 83E0- 31 84 4C E8 83 20 1E 84
 83E8- E6 06 E6 07 A5 07 C9 10
 83F0- D0 DF 85 EF 4C 9C FC 48
 83F8- 20 1E 84 20 31 84 A4 24
 8400- A9 20 18 65 07 85 24 68
 8408- 20 33 84 84 24 E6 06 E6
 8410- 07 A5 07 C9 08 D0 8A 85
 8418- EF A2 03 4C 4A F9 A2 FF

8420- B1 FA D9 00 02 F0 02 A2
 8428- 3F 86 32 20 DA FD 4C 84
 8430- FE A9 A0 C9 A0 80 12 24
 8438- EC 10 0E 09 80 C9 E0 90
 8440- 02 29 BF C9 A0 80 02 A9
 8448- DF 84 35 A4 24 91 28 E6
 8450- 24 A4 35 60 85 19 84 ED
 8458- A5 ED 85 24 A9 02 85 18
 8460- A9 00 85 1A A5 19 20 DA
 8468- FD A4 ED 84 24 A4 24 B1
 8470- 28 85 07 20 55 85 91 28
 8478- C9 88 F0 DC C9 8D F0 3F
 8480- C9 80 80 0C 20 22 86 A5
 8488- 07 A4 24 91 28 4C 6D 84
 8490- C9 84 90 0A C9 C1 90 EC
 8498- C9 C7 80 E8 E9 06 29 0F
 84A0- 85 07 A5 1A 0A 0A 0A 0A
 84A8- 05 07 85 1A C6 18 00 04
 84B0- A5 1A 38 60 E6 24 A4 24
 84B8- A9 A0 91 28 4C 6D 84 A5
 84C0- 18 C9 02 D0 04 A5 19 85
 84C8- 1A A5 ED 85 24 A5 1A 20
 84D0- DA FD A5 1A 38 60 A9 00
 84D8- 8D 03 01 A5 FB 8D 09 01
 84E0- A0 00 A9 01 20 09 03 90
 84E8- 54 20 22 86 20 3C 86 C5
 84F0- F2 F2 E5 F5 F2 A0 E4 20
 84F8- A0 0C 01 C9 02 F0 2E 20
 8500- 45 86 E5 A0 EC E5 E3 F4
 8508- F5 F2 E5 00 20 45 86 AC
 8510- A0 EE EF F5 F6 E5 EC A0
 8518- E5 F3 F3 E1 E9 A0 8F 00
 8520- 20 55 85 C9 D9 F0 AF C9
 8528- CF F0 AB 38 60 20 45 86
 8530- A7 E5 E3 F2 E9 F4 F5 F2
 8538- E5 00 4C 0C 85 A0 00 B1
 8540- FA 99 00 02 C8 D0 F8 38
 8548- 60 A0 00 89 00 02 91 FA
 8550- C8 D0 F8 38 60 20 6A 85

8558- 2C 10 C0 C9 90 F0 56 C9
 8560- 98 D0 06 A2 FF 9A 4C B7
 8568- 80 60 A4 24 B1 28 48 A9
 8570- A3 20 A1 85 80 F7 07 A9
 8578- DF 20 8A 85 30 07 68 48
 8580- 20 8A 85 10 F2 68 91 28
 8588- 8A 60 91 28 A2 32 86 06
 8590- AD 00 C0 30 07 CA D0 F8
 8598- C6 06 D0 F4 8D 10 C0 AA
 85A0- 60 C9 E0 80 0C A6 1E E0
 85A8- 7F D0 07 C9 C0 80 03 49
 85B0- C0 60 25 1E 60 A2 2F 85
 85B8- 00 9D 11 01 CA 10 F8 A2
 85C0- 16 86 22 86 23 A9 01 20
 85C8- 95 FE A2 80 85 06 20 C1
 85D0- FB A5 28 85 26 A5 29 85
 85D8- 27 A9 16 20 58 FB 20 3E
 85E0- FD A0 00 B1 26 C9 20 80
 85E8- 02 69 C0 C9 60 80 02 09
 85F0- 8U C9 A0 80 02 69 40 C9
 85F8- FF D0 02 A9 A0 20 ED FD
 8600- C8 C0 28 D0 DE E6 06 A5
 8608- 06 C9 17 90 C1 20 8E FD
 8610- A9 00 20 95 FE A2 2F 8D
 8618- 11 01 95 00 CA 10 F8 4C
 8620- 55 85 A2 80 8A 49 80 A8
 8628- 88 D0 FD 2C 30 C0 8A 69
 8630- 80 A8 88 D0 FD 2C 30 C0
 8638- CA D0 E9 60 20 4F 86 00
 8640- 17 00 20 9C FC 68 85 06
 8648- 68 85 07 A0 00 F0 12 68
 8650- 85 06 68 85 07 A0 01 81
 8658- 06 85 24 C8 B1 06 20 58
 8660- FB C8 B1 06 F0 06 20 ED
 8668- FD 4C 61 86 18 98 65 06
 8670- AB A5 07 69 00 48 98 48
 8678- 60

BONJOUR LES PRIX!!

NOS PRIX SONT F TTC

Carte langage	400	Speech card	320
Carte 128 k ram	1550	Carte horloge	500
Carte 80 colonnes	700	Joystick	165
Interface série	520	Ventilateur	280
Super série	1000	Contrôleur de drive auto switch 13/16	370
Interface parallèle	380	Microdrive 3"	1900
Grappler + buffer 16 k	1350	Moniteur vert 12"	950
Carte modem	2200	Disquettes 5" 1/4 S.F./S.D. par 1 boîte	140/boîte
Carte Z 80	410	Disquettes 5" 1/4 S.F./D.D. par 1 boîte	175/boîte
Wildcard	400		

AU-DESSUS, NOUS CONSULTER.

*Carte bleue et eurocard acceptées
 Vente par correspondance: nous consulter.*

Computer 3

3, rue Papillon 75009 Paris - Tél.: 523.51.15

(métro Poissonnière) ouverture du lundi au samedi de 10 h à 19 h 30



Les fleuves de France

Joseph Pino

L'Enseignement Assisté par Ordinateur (E.A.O.) est à la mode actuellement; Pom's n'échappe pas à la tendance en vous présentant ici un programme d'éducation sur les fleuves et rivières de France.

Le programme E.A.O.FLEUVES est écrit entièrement en Basic, donc facilement modifiable, et propose, à partir d'une carte de France au

1/5 000 000 sur laquelle clignote le tracé d'un fleuve, d'en fournir le nom exact.

Pas d'affolement! Le programme ne vous demande que 10 noms avant de s'arrêter et d'afficher votre score. Pour les imbattables, il restera la possibilité d'augmenter le nombre de fleuves disponibles (60 actuellement).

Le programme est très simple et ne nécessite aucun commentaire particulier. Le plus pénible sera de rentrer toutes les DATA si vous n'avez pas la disquette de Pom's.

Une petite colle avant de commencer: savez-vous où se trouvent le Blavet et la Truyère?

Programme E.A.O.Fleuves

```
100 REM *** E.A.O. GEOGRAPHIE ***
110 HOME : INVERSE : PRINT SPC( 41);:
    NORMAL : PRINT SPC( 38);: INVERS
    E : PRINT " ";: NORMAL : PRINT "
    E.A.O. GEOGRAPHIE CM2 FLEUV
    ES ";: INVERSE : PRINT " ";: N
    ORMAL : PRINT SPC( 38);: INVERSE
    : PRINT SPC( 41): NORMAL : PRIN
    T
120 CLAVIER = 49152:KBDSTB = 49168:TXTCL
    R = 49232:HISCR = 49237: VTAB 10:
    HTAB 1: PRINT " Voulez vous que
    lques explications"; VTAB 12: HTA
    B 18: PRINT "< N >"; VTAB 12: HTA
    B 20: GET WS
130 IF WS < > "O" THEN 500
140 VTAB 9: HTAB 1: PRINT " Apprenez le
    nom des fleuves de FRANCE et de
    leurs affluents en vous amusant.
    "
150 PRINT SPC( 40): PRINT " Vous t
    aperez au clavier le nom du cour
    s d'eau qui clignote sur la carte
    "
160 PRINT : PRINT " Je vous demande
    10 noms et je vous donne les 10
    reponses.": PRINT
170 PRINT " Si vous repondez '?'
    comme nom jereviens en page HGR2
    pour la carte."
180 PRINT : PRINT "Tapez 'C' pour voir
    la carte complete.": GOSUB 10000
    : IF WS = "C" THEN GOTO 1010
500 REM *** DEBUT ***
510 TEXT : HOME : VTAB 1: HTAB 4: INVER
    SE : PRINT " NOMS ";: NORMAL : H
    GR2 : GOSUB 3000: GOSUB 5000: GOS
    UB 4000: FOR T = 1 TO 9: GOSUB 20
    00: GOSUB 10000: GOSUB 9000: GOSU
    B 5000: GOSUB 4000: NEXT
520 GOSUB 2000: GOSUB 10000: GOSUB 9000
    : GOSUB 10000
530 TEXT : HOME : VTAB 2: HTAB 3: PRINT
    "Vous avez bien repondu ";EX;" f
    ois sur 10": PRINT : PRINT : HTAB
    5: PRINT "Voulez vous recommence
    r (O=OUI) ";: GET WS: IF WS = "O
    " THEN RUN
540 VTAB 14: HTAB 15: PRINT "AU REVOIR
    "
1000 REM *** TOUTE LA CARTE ***
1010 HGR2 : GOSUB 3000: FOR N = 1 TO 59
    : GOSUB 8000: NEXT
1020 GET WS: RESTORE :N = 0:N2 = 0: FOR
    I = 0 TO 9:N(I) = 0: NEXT : GOTO
    500
2000 REM *** CORRECTION ***
2010 TEXT : VTAB T * 2 + 1: HTAB 1: PRI
    NT T;: HTAB 3: PRINT " ?....";: H
    TAB 3: INPUT " ";RS: IF RS = "?"
    THEN GOSUB 9000: GOSUB 4000: GOT
    O 2010
2020 TEXT : VTAB T * 2 + 1: HTAB 1: PRI
    NT T;: HTAB 4: PRINT AS; SPC( 20)
    ;: IF RS = AS THEN VTAB T * 2 +
    1: HTAB 19: INVERSE : PRINT " EXA
    CT ";: NORMAL : PRINT SPC( 10);:
    EX = EX + 1: GOTO 2040
2040 VTAB 1: HTAB 28: PRINT "Note : ";E
    X;"/";T;: RETURN
3000 REM *** TRACER LA CARTE ***
3010 GOSUB 9000: HCOLOR= 3:XO = 40:YO =
    - 5: FOR I = 1 TO 9: READ X,Y:
    H PLOT X + XO,Y + YO
3020 READ X,Y: IF X AND Y THEN H PLOT
    TO X + XO,Y + YO: GOTO 3020
3030 NEXT : RETURN
4000 REM *** FAIRE CLIGNOTER ***
4010 GOSUB 8000: HCOLOR= 3: GOSUB 8000:
    IF PEEK (CL) > 127 THEN POKE K
    B,O: RETURN
4020 HCOLOR= 0: GOTO 4010
5000 REM *** TIRER AU SORT ***
5010 NQ = 59: REM
    Nbre Questions
5020 N = INT ( RND (1) * NQ) + 1:N(T) =
    N
5030 IF T = 0 THEN 5060
5040 FOR S = 0 TO T - 1: IF N = N(S) TH
    EN S = T - 1: NEXT : GOTO 5020
5050 NEXT S
5060 VTAB 24: HTAB 10: PRINT SPC( 22):
    RETURN
6000 REM *** QUESTIONS ***
6010 AS = "SOMME": H PLOT 165,24 TO 161,2
    7 TO 160,22 TO 150,24 TO 142,18:
```

RETURN
 6020 AS = "EURE": HPLOT 129,53 TO 138,57
 TO 140,53 TO 137,49 TO 138,46 TO
 134,40 TO 135,37: RETURN
 6030 AS = "LOING": HPLOT 164,74 TO 159,7
 2 TO 157,67 TO 158,64 TO 156,61 T
 O 158,58: RETURN
 6040 AS = "YONNE": HPLOT 175,85 TO 174,8
 4 TO 172,84 TO 169,77 TO 171,74 T
 O 169,67 TO 165,65 TO 165,60 TO 1
 63,57: RETURN
 6050 AS = "ARMANCON": HPLOT 184,82 TO 18
 2,78 TO 183,77 TO 174,66 TO 169,6
 7: RETURN
 6060 AS = "AUBE": HPLOT 193,72 TO 183,61
 TO 181,55 TO 174,52 TO 171,53: R
 ETURN
 6070 AS = "MARNE": HPLOT 196,68 TO 195,6
 8 TO 195,64 TO 192,63 TO 191,52 T
 O 185,51 TO 179,43 TO 169,41 TO 1
 63,45 TO 162,44 TO 154,49 TO 152,
 47: RETURN
 6080 AS = "OISE": HPLOT 177,22 TO 168,24
 TO 165,29 TO 161,31 TO 158,33 TO
 147,42: RETURN
 6090 AS = "AISNE": HPLOT 192,45 TO 190,4
 5 TO 185,33 TO 178,31 TO 173,35 T
 O 164,35 TO 158,33: RETURN
 6100 AS = "ORNE": HPLOT 120,51 TO 117,49
 TO 113,49 TO 110,44 TO 114,37: R
 ETURN
 6110 AS = "RANCE": HPLOT 79,57 TO 83,58
 TO 87,57 TO 88,52: RETURN
 6120 AS = "AULNE": HPLOT 65,53 TO 62,58
 TO 59,58 TO 54,56: RETURN
 6130 AS = "BLAVET": HPLOT 68,55 TO 72,59
 TO 72,63 TO 67,69: RETURN
 6140 AS = "VILAINE": HPLOT 103,60 TO 90,
 62 TO 88,71 TO 79,75: RETURN
 6150 AS = "MAINE": HPLOT 108,75 TO 107,7
 8: RETURN
 6160 AS = "MAYENNE": HPLOT 114,54 TO 113
 ,55 TO 110,54 TO 107,56 TO 105,62
 TO 106,70 TO 108,75: RETURN
 6170 AS = "SARTHE": HPLOT 124,52 TO 122,
 54 TO 116,56 TO 116,60 TO 119,67
 TO 111,69 TO 108,75: RETURN
 6180 AS = "LOIR": HPLOT 133,58 TO 136,60
 TO 134,68 TO 130,70 TO 122,72 TO
 118,72 TO 108,75: RETURN
 6190 AS = "NIEVRE": HPLOT 165,80 TO 162,
 88: RETURN
 6200 AS = "ARROUX": HPLOT 182,86 TO 178,
 93 TO 176,100: RETURN
 6210 AS = "ALLIER": HPLOT 171,135 TO 167
 ,128 TO 165,115 TO 168,109 TO 166
 ,97 TO 161,93 TO 162,88: RETURN
 6220 AS = "CHER": HPLOT 153,112 TO 155,1
 07 TO 155,97 TO 150,92 TO 149,86
 TO 145,82 TO 125,79: RETURN
 6230 AS = "INDRE": HPLOT 150,100 TO 146,
 99 TO 144,94 TO 142,92 TO 135,90
 TO 129,83 TO 120,83 TO 119,81: RE
 TURN
 6240 AS = "VIENNE": HPLOT 147,117 TO 143
 ,116 TO 136,112 TO 134,114 TO 126
 ,112 TO 126,100 TO 124,96 TO 124,
 87 TO 118,84 TO 116,81: RETURN
 6250 AS = "CREUSE": HPLOT 148,113 TO 148
 ,109 TO 140,101 TO 140,99 TO 137,
 97 TO 131,95 TO 127,88 TO 124,87:
 RETURN
 6260 AS = "SEVRE NANTAISE": HPLOT 103,96
 TO 102,93 TO 94,84 TO 93,82: RET
 URN
 6270 AS = "CHARENTE": HPLOT 125,115 TO 1
 19,106 TO 117,116 TO 114,118 TO 1
 05,116 TO 104,112 TO 98,111: RETU
 RN
 6280 AS = "DORDOGNE": HPLOT 159,119 TO 1
 54,119 TO 153,122 TO 139,135 TO 1
 28,135 TO 110,135 TO 105,130: RET
 URN
 6290 AS = "VEZERE": HPLOT 147,120 TO 142
 ,120 TO 137,124 TO 137,129 TO 134
 ,129 TO 128,135: RETURN
 6300 AS = "ISLE": HPLOT 134,117 TO 130,1
 20 TO 127,128 TO 124,128 TO 121,1
 31 TO 111,131 TO 110,135: RETURN
 6310 AS = "LOT": HPLOT 172,142 TO 163,14
 5 TO 155,140 TO 142,144 TO 130,14
 3 TO 128,145 TO 123,144 TO 119,14
 7: RETURN
 6320 AS = "TRUYERE": HPLOT 170,137 TO 16
 7,139 TO 165,134 TO 158,135 TO 15
 5,140: RETURN
 6330 AS = "TARN": HPLOT 173,145 TO 165,1
 47 TO 162,152 TO 157,154 TO 145,1
 56 TO 140,158 TO 135,152 TO 130,1
 52: RETURN
 6340 AS = "AVEYRON": HPLOT 162,147 TO 15
 8,145 TO 154,147 TO 146,146 TO 14
 5,151 TO 135,152: RETURN
 6350 AS = "ARIEGE": HPLOT 141,186 TO 143
 ,185 TO 142,182 TO 139,180 TO 139
 ,171 TO 136,169 TO 136,165: RETUR
 N
 6360 AS = "SAVE": HPLOT 122,173 TO 127,1
 69 TO 134,159: RETURN
 6370 AS = "GERS": HPLOT 120,172 TO 123,1
 67 TO 124,158 TO 123,155 TO 125,1
 50: RETURN
 6380 AS = "BAISE": HPLOT 118,176 TO 117,
 172 TO 120,168 TO 119,148: RETURN
 6390 AS = "ADOUR": HPLOT 115,178 TO 115,
 175 TO 114,172 TO 113,163 TO 108,
 159 TO 95,160 TO 94,164 TO 90,165
 TO 89,164: RETURN
 6400 AS = "AUDE": HPLOT 147,185 TO 151,1
 72 TO 165,173: RETURN
 6410 AS = "HERAULT": HPLOT 170,151 TO 17
 3,156 TO 169,165 TO 169,170: RETU
 RN
 6420 AS = "GARD": HPLOT 173,148 TO 178,1
 49 TO 181,155 TO 187,155 TO 188,1
 57: RETURN
 6430 AS = "ARDECHE": HPLOT 179,139 TO 18
 4,141 TO 182,145 TO 188,149: RETU
 RN
 6440 AS = "SAONE": HPLOT 207,62 TO 205,6
 5 TO 205,72 TO 197,80 TO 196,85 T
 O 191,89 TO 189,92 TO 191,97 TO 1
 88,110 TO 190,112 TO 190,114: RET
 URN
 6450 AS = "DOUBS": HPLOT 210,91 TO 220,7
 8 TO 218,78 TO 218,75 TO 205,84 T
 O 197,89 TO 191,89: RETURN
 6460 AS = "AIN": HPLOT 208,92 TO 203,93
 TO 196,113: RETURN
 6470 AS = "ISERE": HPLOT 221,120 TO 218,
 117 TO 215,118 TO 214,115 TO 211,
 118 TO 207,122 TO 204,126 TO 202,
 124 TO 194,131 TO 191,131: RETURN
 6480 AS = "ARC": HPLOT 223,122 TO 214,12
 5 TO 212,123 TO 211,118: RETURN

```

6490 AS = "DRAC": H PLOT 213,136 TO 211,1
      38 TO 203,133 TO 204,126: RETURN
6500 AS = "DROME": H PLOT 204,142 TO 200,
      140 TO 199,136 TO 195,138 TO 190,
      137: RETURN
6510 AS = "DURANCE": H PLOT 218,132 TO 21
      7,134 TO 217,138 TO 213,143 TO 21
      0,142 TO 208,145 TO 210,150 TO 20
      7,159 TO 201,161 TO 190,155: RETU
      RN
6520 AS = "VERDON": H PLOT 218,147 TO 218
      ,156 TO 210,159 TO 207,159: RETUR
      N
6530 AS = "VAR": H PLOT 221,147 TO 222,15
      4 TO 228,154 TO 228,162: RETURN
6540 AS = "RHIN": H PLOT 240,72 TO 227,72
      TO 227,60 TO 230,48 TO 235,40 TO
      238,33 TO 237,22 TO 233,17 TO 22
      8,19 TO 225,12 TO 222,10 TO 217,4
      TO 216,0: RETURN
6550 AS = "ILL": H PLOT 225,73 TO 224,72
      TO 225,58 TO 227,56 TO 228,50 TO
      230,48: RETURN
6560 AS = "MOSELLE": H PLOT 218,66 TO 216
      ,67 TO 203,50 TO 206,47 TO 205,44
      TO 206,34 TO 209,31 TO 215,24 TO
      220,20 TO 225,12: RETURN
6570 AS = "MEURTHE": H PLOT 218,59 TO 215
      ,58 TO 207,50 TO 206,47: RETURN
6580 AS = "MEUSE": H PLOT 200,65 TO 199,6
      4 TO 200,50 TO 192,35 TO 192,32 T
      O 189,27 TO 187,28 TO 184,23 TO 1
      86,17 TO 187,12 TO 197,6 TO 199,0
      : RETURN
6590 AS = "ESCAUT": H PLOT 165,22 TO 163,
      22 TO 165,15 TO 168,15 TO 168,12
      TO 167,9 TO 165,6 TO 173,0: RETUR
      N
7000 REM *** CARTE DE FRANCE ***

7010 DATA 115,5,100,10,100,20,102,23,99
      ,22,98,25,93,28,86,29,80,33,78,37
      ,83,38,74,42,61,40,59,36,60,34,59
      ,32,54,34,50,32,51,39,54,43,54,54
      ,57,56,51,57,51,54,49,54,48,57,46
      ,55,41,54,37,57,33,50,26,50,24,53
      ,18,52
7020 DATA 7,54,7,58,13,58,14,61,8,61,13
      ,63,13,65,7,65,11,67,12,69,11,71,
      14,72,15,69,25,75,27,74,30,75,28,
      76,29,78,32,78,32,76,36,78,33,79,
      34,81,39,80,38,86,41,86,44,84,49,
      87,44,86,43,89,46,90,43,95,47,103
      ,57,108
7030 DATA 58,116,55,122,60,124,63,128,6
      5,135,61,128,57,124,54,143,57,141
      ,58,144,56,145,55,144,53,146,49,1
      69,46,172,44,172,51,180,63,185,69
      ,185,71,187,82,187,82,184,100,190
      ,100,193,105,195,117,195,120,193,
      125,195
7040 DATA 125,192,123,191,123,180,125,1
      78,129,175,130,176,139,169,141,17
      1,145,172,147,172,147,174,153,173
      ,155,170,159,171,155,173,161,174,
      160,176,168,180,177,178,178,176,1
      80,176,181,171,183,172,188,167,18
      9,164
7050 DATA 191,165,195,162,195,154,185,1
      53,182,147,184,140,179,134,185,13
      0,185,125,180,118,182,115,180,104
      ,175,104,169,109,0,0,180,104,176,
      101,171,104,169,109,169,101,180,8
      8,182,83,181,80,185,80,187,77,188
      ,75,192,55
7060 DATA 197,45,195,45,188,44,185,42,1
      81,44,176,41,174,43,171,36,169,36
      ,164,36,159,34,156,36,147,29,146,
      22,141,27,137,27,137,20,135,17,13
      1,19,130,16,128,14,127,14,125,15,
      123,14,122,9,119,10,116,7,115,5,0
      ,0
7070 DATA 147,82,144,72,137,67,135,60,1
      31,58,123,62,118,63,111,56,112,52
      ,107,48,102,49,95,42,83,38,0,0
7080 DATA 141,140,139,141,136,139,136,1
      32,138,132,141,124,137,115,136,10
      5,122,93,121,88,119,86,120,83,116
      ,78,104,72,94,82,85,84,79,86,76,8
      6,70,82,67,83,65,84,58,83,53,87,4
      9,87,0,0
7090 DATA 89,188,86,189,83,180,86,179,9
      2,176,96,170,94,164,90,157,85,155
      ,79,153,79,152,79,150,74,146,70,1
      46,65,135,0,0,200,101,194,106,191
      ,106,183,110,180,104,0,0,169,109,
      167,110,163,122,154,118,156,118,1
      50
7100 DATA 119,150,125,149,127,151,136,1
      51,138,150,142,148,148,148,154,15
      2,159,150,160,148,162,148,165,144
      ,168,141,171,0,0,144,168,145,172,
      0,0,148,165,153,173,0,0
8000 REM *** AIGUILLAGE ***

8010 IF N > 40 THEN N2 = N - 40: GOTO 8
      040
8020 ON N GOSUB 6010,6020,6030,6040,605
      0,6060,6070,6080,6090,6100,6110,6
      120,6130,6140,6150,6160,6170,6180
      ,6190,6200,6210,6220,6230,6240,62
      50,6260,6270,6280,6290,6300,6310,
      6320,6330,6340,6350,6360,6370,638
      0,6390,6400
8030 RETURN
8040 ON N2 GOSUB 6410,6420,6430,6440,64
      50,6460,6470,6480,6490,6500,6510,
      6520,6530,6540,6550,6560,6570,658
      0,6590
8100 RETURN
9000 REM * VOIR HGR2 SANS EFFACER *

9010 POKE TX,0: POKE HI,0: RETURN
10000 VTAB 24: HTAB 10: PRINT "Tapez un
      e touche :": GET WS: RETURN

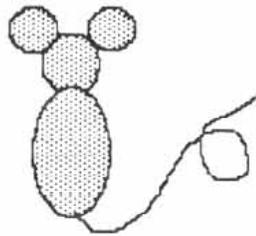
```

Si vous êtes dans la ☞ et oubliez d'acheter Pom's,
envoyez-nous une ✉ et abonnez-vous sans 📧

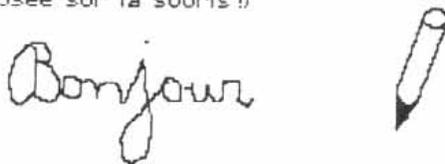
Inutile de vous 📧, les autres numéros de Pom's
sont encore disponibles avec leurs disquettes !

IMPRESSIONS SUR MOUSEPAINT

MousePaint est avant tout un logiciel de réalisation de dessins, fonctionnant sous contrôle de la souris Apple



Un crayon permet de faire du dessin à main levée (si on peut dire ... la main est posée sur la souris !)



ce n'est pas toujours très simple, mais avec un peu d'habitude ...

Et en cas d'erreur, il y a la ressource de la gomme.



On peut également travailler par taches, avec la brosse.

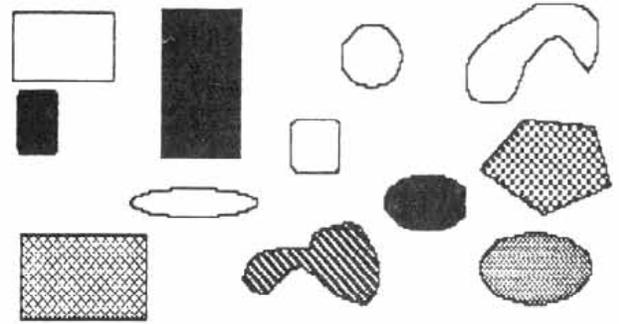


On peut aussi vaporiser des petits nuages

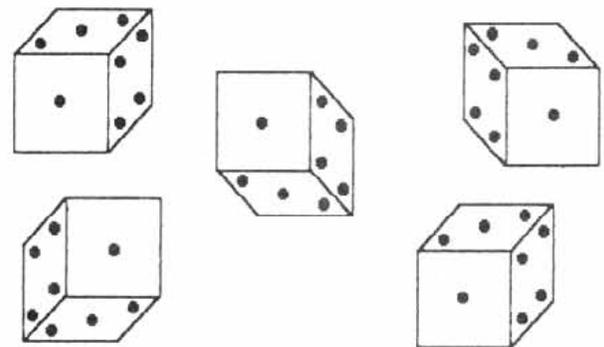


Bien sûr on peut écrire : on dispose de plusieurs types de caractères
Mais pas de caractères accentués

Des formes prédéfinies permettent de tracer des figures géométriques, vides ou pleines, avec décors variés. Mais on ne peut pas remplir après coup



On peut aussi déplacer, dupliquer, faire des symétries, ...



Il existe enfin des fonctions permettant de travailler au niveau du point, ou de contrôler la position du dessin dans la page, d'utiliser une grille pour certains tracés, ..., et bien entendu des fonctions de gestion de disque.

Une lacune:
l'absence d'une fonction "Catalogue" : on ne sait jamais ce qu'il y a sur un disque

La fonction impression EXIGE l'imprimante Image Writer. Pour une autre imprimante, il faut transiter par le disque.



Un produit bien agréable malgré quelques lacunes.

La version étudiée fonctionne sur les modèles II, II+, //e et //c

Gestion de fichiers par RWTS et DOS 3.3

Gérard Michel

Pom's a déjà consacré plusieurs articles aux méthodes de gestion de fichier en DOS 3.3 et, après une longue pose dans cette série, nous vous en proposons un autre, un peu plus complexe dans sa mise en oeuvre mais qui devrait vous fournir, du moins l'espérons-nous, un système relativement rapide et souple pour gérer de gros fichiers de données sur disquettes 140K.

Principes généraux

Il est de notoriété publique que le DOS se charge en mémoire à partir d'une disquette lorsque l'on met l'Apple sous tension ou que l'on "boote" à chaud, par PR#6 par exemple.

L'implantation du DOS sur une disquette résulte, de manière générale, de son initialisation par la commande INIT. A l'issue de cette opération, les pistes 0, 1 et 2 de la disquette contiennent le DOS et ne sont plus accessibles pour la sauvegarde de fichiers ou programmes.

En fait, cette présence du DOS sur toutes les disquettes constitue un luxe inutile. Plus particulièrement, en matière de gestion de fichiers, on peut utiliser fréquemment une disquette "Programmes", qui se place dans le lecteur 1 et assure également le "boot" du système, et une disquette réservée plus spécialement à la sauvegarde des fichiers, placée dans le lecteur 2. Sur cette seconde disquette, le DOS ne représente qu'une perte de place pour le stockage des informations.

De même, si l'on n'emploie qu'une seule disquette pour programmes et fichiers et si l'on ne dispose que d'un seul lecteur, il est toujours possible de charger le DOS en mémoire à partir d'une autre disquette, remplacée immédiatement après par celle nécessaire aux traitements.

Il serait donc possible d'exploiter les 3 pistes protégées du DOS pour y stocker des informations utiles à la gestion de fichiers : nous vous proposons d'y placer la table de référence des enregistrements d'un gros fichier de données. Notons de suite que ceci ne perturbera pas le fonctionnement du DOS pour le maniement de programmes ou fichiers sur la même disquette, puisque les pistes 0, 1 et 2 sont de toute façon considérées comme occupées à l'issue de INIT;

on ne risque donc pas de les "écraser" en cours de traitement. Par un POKE approprié à l'adresse 44723 (voir Pom's 11), on peut même protéger ainsi plus de trois pistes pour la sauvegarde d'une table de référence, et nous y reviendrons ultérieurement.

Organisation de la table de référence

Nous stockerons dans cette table trois types d'informations :

- La clé d'accès, identifiant chaque enregistrement du fichier de données et permettant d'y accéder directement (le nom d'un client, un numéro de dossier...). La longueur de cette clé sera fixe, bornée ou quelconque, au choix du programmeur. De plus, les "homonymes" (plusieurs clés identiques) seront acceptés.
- Des informations "annexes", sous forme d'un certain nombre d'octets pouvant servir de critères pour des recherches rapides dans la table (numéro de département, codification de situation de famille...).
- L'adresse de l'enregistrement correspondant à la clé dans le fichier de données. Ce dernier est un fichier à accès direct DOS 3.3 standard. Dans la table de référence, cette adresse occupera deux octets (ceci autorise des adresses de 0 à 65535, ce qui est plus que suffisant).

Un élément de la table de référence se présentera donc de la façon suivante :

- Longueur totale "clé + informations annexes" : 1 octet
- Longueur de la clé seule : 1 octet
- Clé d'accès : N1 octets
- Informations annexes : N2 octets
- Adresse de l'enregistrement correspondant : 2 octets

Gestion de la table de référence

Pour utiliser les pistes du DOS, il faut y accéder directement par la célèbre routine RWTS, qui permet de lire ou d'écrire un secteur quelconque de la disquette, soit 256 octets.

La table de référence sera donc découpée en un certain nombre de secteurs, 48 (3*16) au maximum si l'on utilise les trois pistes du DOS, numérotés de 0 à 47. Nous appellerons NN le numéro du dernier secteur de clé exploitable.

L'appel à RWTS se fera à partir du programme Applesoft de traitement, tandis qu'une petite routine en assembleur s'occupera de la recherche d'une clé donnée à l'intérieur d'un secteur de clés chargé en mémoire dans un buffer.

Une telle recherche est toujours plus rapide lorsque les clés sur lesquelles elle porte sont classées dans un ordre quelconque, en l'occurrence dans l'ordre des codes ASCII des caractères composant la clé. En revanche, les opérations de classement sont généralement longues en regard des délais d'attente acceptables dans le cours de l'exploitation normale du fichier et il est donc préférable de les séparer de cette dernière.

On peut ainsi se trouver, à un instant donné, avec deux types de secteurs de clés :

- Des secteurs de clés classées, dans lesquels la recherche sera plus rapide. Nous appellerons NZ le numéro du dernier secteur de clés classées, avec NZ compris entre 0 et NN.
- Des secteurs de clés "en vrac", placés à la suite des premiers, dans lesquels les nouvelles clés ont été stockées simplement dans l'ordre chronologique de leur entrée en machine. La recherche sera évidemment plus lente sur ces secteurs, mais on évite ainsi d'imposer à l'utilisateur un délai de classement à chaque nouvel enregistrement. La variable TT contiendra le numéro du dernier secteur de "vrac" : TT est donc compris entre NZ+1 et NN. Dès que TT atteint NN+1, la capacité de stockage des clés est saturée.

Un petit programme, indépendant des programmes de traitement proprement dits, permettra à l'utilisateur de reclasser les clés en vrac à l'intérieur des clés classées et de reconstituer une table de référence composée uniquement de secteurs de clés classées. Cette opération sera par conséquent réalisée sur l'initiative de l'utilisateur, avant de procéder à un listage du fichier, par exemple, ou dès que les temps de recherche sont trop détériorés par le nombre de clés "en vrac" à explorer.

En ce qui concerne les clés classées, et dans la mesure où la recherche porte sur un secteur à la fois, il est bon de savoir si possible quel est

celui qu'il faut charger de préférence en mémoire, c'est-à-dire celui dans lequel on aurait une chance de trouver la clé qui nous intéresse. Pour ce faire, on conservera dans un tableau N\$(NN) la dernière clé de chaque secteur classé : N\$(0) contiendra DUPONT si DUPONT est la dernière clé stockée dans le secteur 0, et ainsi de suite. Si l'on cherche une clé quelconque XXXXXX, on lira en mémoire le dernier secteur S pour lequel XXXXXX <= N\$(S). Si aucun secteur classé ne répond à cette condition, on cherchera alors en séquence dans les secteurs de vrac, et de même si l'on ne trouve pas dans le secteur classé potentiel S.

Fichiers manipulés

Notre système de table de référence ne peut gérer qu'un seul fichier de données par disquette. Par contre, d'autres fichiers de toutes sortes peuvent être utilisés conjointement en DOS 3.3 standard. En particulier, deux petits fichiers annexes sont nécessaires à la gestion du système :

- Un fichier contenant les paramètres de la table de référence : dernier secteur classé (NZ), dernier secteur "en vrac" (TT), dernière adresse attribuée à un enregistrement (NF), premier octet libre dans le secteur de vrac en cours (OO), statut de la table vis-à-vis du classement (IC, qui vaut 0 si tout est classé, 1 si on a à la fois du vrac et du classé, 2 si rien n'est classé). Ce fichier est baptisé FP.
- Un fichier contenant les adresses éventuellement libérées par des annulations d'enregistrements, et baptisé PL.

Analyse du programme de démonstration

Il s'agit d'un programme classique de gestion de fichier, permettant la création, la modification, la consultation, l'annulation et la liste d'enregistrements. Son étude nous aidera à comprendre la mise en oeuvre du système.

Principales variables

- AR : adresse de début de la routine en langage-machine de recherche dans un secteur.
- ADR : adresse de début de la zone où la routine-machine stockera les adresses des enregistrements dont la clé correspond à celle que l'on cherche.
- CLE : adresse du début de la zone où le programme POKÉra la clé recherchée.
- ASY0 : adresse du début de la zone où la routine-machine stockera les positions dans le secteur des clés correspondant à celle que

l'on cherche.

- BU : adresse du début du buffer en mémoire utilisé par RWTS pour lire ou écrire un secteur.
- LC : adresse où sera POKÉE la longueur de la clé cherchée.
- RES : résultat de la recherche, soit 0 si l'on n'a pas trouvé, et le nombre de clés égales à celle cherchée dans le cas contraire.
- ABL : indicateur de fin de secteur, positionné par la routine-machine. Si on trouve effectivement une clé (au moins) correcte, mais que celle-ci est également la dernière d'un secteur, la routine mettra 1 dans l'adresse correspondant à ABL. En effet, dans les clés classées, rien n'interdit que l'on ait un MARTIN, par exemple, à la fin du secteur 3, mais aussi d'autres MARTIN au début du secteur 4. Si ABL vaut 1, il faudra donc examiner le secteur suivant pour vérifier s'il n'y a pas d'autres clés possibles.
- IO : adresse de début de la table IOB pour RWTS.
- AA : longueur des informations annexes qui seront également stockées dans la table.
- DT : adresse de début de la table DCB pour RWTS.
- RWTS : adresse de la routine d'appel de RWTS (voir manuel du DOS 3.3 pour l'utilisation de RWTS).
- N : numéro du secteur courant sur lequel on lit ou écrit.
- C : commande pour RWTS, soit 1 pour une lecture et 2 pour écriture.

Fonctionnement du programme

Le programme manipule un fichier de données DON à accès direct, dont la longueur des enregistrements est fixée à 100. Il commence par une phase d'initialisation, suivie de sous-routines, et l'essentiel des traitements réalisés porte sur la gestion de la table de référence et du fichier Fichiers et programmes sont en outre placés sur la même disquette, qui ne sera donc exploitable que si le DOS a déjà été chargé à partir d'une autre disquette.

– **Lignes 2 à 13** : initialisation. On définit la valeur des différentes adresses qui assurent le passage de paramètres entre le programme Applesoft et la routine-machine de recherche de clé (ADR, CLE, ASY0, LC...). On fixe également l'adresse d'implantation de la routine (AR). Cette dernière est relogeable à ceci près qu'elle utilise des tableaux (ADR, CLE et ASY0) dont on veut pouvoir faire varier la taille et l'adresse de début en fonction des besoins, mais dont les adresses ont été données de manière absolue à l'assemblage de la routine. C'est pourquoi la ligne 13

remet dans le code machine, aux endroits où elles doivent apparaître, les adresses réelles correspondant à ADR, CLE et ASY0.

Dans notre exemple, les trois tableaux sont à la suite les uns des autres à partir de l'adresse \$9100, et l'on suppose qu'il n'y aura pas plus de 8 homonymes par secteur (ADR occupe 16 octets, soit 2 par adresse retenue, et ASY0 en prend 8).

A la ligne 4, on fixe l'adresse de la table IOB et on la constitue en recopiant celle du DOS, qui débute en 47080. Pour que la table soit cohérente, il faut donc que le dernier lecteur utilisé juste avant cette copie soit bien celui dans lequel se trouve la disquette du fichier. Ensuite, on indique dans la table IOB l'adresse de notre table DCB et on POKÉ dans cette dernière les valeurs attendues pour nos lecteurs 140K.

En ligne 5, on implante à l'adresse 768 la routine d'appel à RWTS. Les valeurs 148 et 0 qui suivent respectivement 169 (code de LDA) et 160 (code de LDY) correspondent au poids fort et au poids faible de l'adresse de début de la table IOB (IO). Dans notre exemple, le poids fort vaut \$94, soit 148 en décimal, et le poids faible vaut 0. Vous auriez donc à modifier ces deux valeurs si vous vouliez modifier l'adresse de la table IOB.

Il est possible d'employer toute cette partie d'initialisation en modifiant l'implantation selon les besoins. Il suffit pour cela de donner les valeurs qui vous conviennent aux variables concernées : AR, ADR, CLE, ASY0, BU, IO et les deux octets correspondant à l'adresse IO dans les DATA de la routine d'appel à RWTS.

– **Lignes 15 et 20** : lecture des deux fichiers annexes et branchement au programme de traitement qui débute à la ligne 100.

– **Lignes 50 et 60** : recherche du secteur classé dans lequel la clé cherchée pourrait éventuellement se trouver. Si la clé est supérieure à toutes celles déjà classées, N correspond au premier secteur de vrac, soit NZ+1.

– **Lignes 65 et 70** : conversion du numéro de secteur en adresse physique sur disquette (piste P / secteur S), puis lecture du secteur en mémoire.

– **Lignes 80 à 86** : sous-programme d'appel à la routine machine de recherche et exploitation de ses résultats. Lecture du secteur N (GOSUB 65) et appel par CALL AR. En retour, on a le résultat de la recherche (R) et la valeur de l'indicateur de fin de secteur (BL = PEEK(ABL)). Si ZC\$, qui contient la clé cherchée, est égal à la dernière

clé du secteur N\$(N) et que la recherche donne malgré cela un résultat nul (R=0), c'est que cette clé N\$(N) a été annulée depuis le dernier reclassement. Dans la mesure où une clé identique pourrait se trouver au début du secteur classé suivant, on met R à 1 afin d'aller examiner celui-ci (voir plus loin les lignes 300 à 500).

En ligne 82, on élimine les cas où BI. pourrait donner une indication fautive, c'est-à-dire si l'on a examiné le dernier secteur classé alors que tout est classé, ou si l'on a examiné le dernier secteur de vrac s'il y en a (IC>0).

Les lignes 83 et 84 correspondent à la lecture et à l'affichage (GOSUB 90) de tous les enregistrements dont la clé est égale à celle que l'on cherche. A chaque affichage, on demande à l'utilisateur "EST-CE LE BON?"; une réponse affirmative confirme que l'on a bel et bien trouvé, ce que O=1 indiquera par la suite.

Si aucun enregistrement n'est le bon, on prend en compte, comme en 80, l'éventualité d'une annulation de la dernière clé du secteur (ligne 85). A l'issue de la ligne 86, R sera nul ou non selon que BI. vaudra 0 ou non (BL > 0 signifie toujours qu'un homonyme peut encore se trouver au début du secteur suivant celui que l'on vient d'examiner).

– **Ligne 90** : lecture et affichage d'un enregistrement du fichier DON d'adresse A. Ligne 95 : pose une question à réponse "O" ou "N". Ligne 98 : écriture d'un enregistrement du fichier DON d'adresse A.

– **Lignes 120 à 130** : introduction de la clé au clavier, puis POKes aux adresses nécessaires pour la routine-machine. La longueur de la clé est quelconque, à la seule réserve près qu'elle ne doit pas excéder les capacités du système de gestion des clés.

– **Ligne 200** : NZ=-1 si on est à la toute première utilisation du programme (voir plus loin l'initialisation d'une disquette fichier). Inutile donc de chercher dans ce cas, et le premier secteur de vrac à constituer est le secteur 0 de la piste 0.

– **Lignes 300 à 500** : appel et exploitation de la sous-routine 80-86 en fonction du statut du système vis-à-vis du classement (IC). Aux numéros de lignes près, cette partie pourrait être utilisée sans changements dans tout autre programme.

– **Lignes 700 à 740** : mise en place des éléments d'une nouvelle clé dans le secteur de vrac courant. Les trois octets d'informations annexes sont donnés par les trois derniers éléments du tableau ZY\$.

On calcule d'abord la longueur totale LT. Chaque clé occupe LT octets, plus 2 pour les deux longueurs stockées, plus 2 pour l'adresse de l'enregistrement; elle sera en outre suivie d'un 0 lors de son entrée dans le secteur, afin de marquer la fin des données utiles pour ce dernier. Il faut donc LT+5 octets libres dans le secteur courant pour y mettre notre nouvelle clé, ce que l'on vérifie tout d'abord. Si tel n'est pas le cas, il faut "ouvrir" un nouveau secteur de vrac, dans la mesure où le nombre de pistes disponibles le permet (lignes 710 et 715).

On peut ensuite POKer les octets et mettre à jour la position du premier octet libre dans le secteur (OO). Par précaution, le secteur ainsi modifié est aussitôt recopié sur disquette (C=2 : GOSUB 65).

– **Lignes 820 et 840** : modification dans la table de référence. SB donne la position dans le buffer du premier caractère de la clé sélectionnée, position qui a été indiquée par la routine-machine. On remplace donc les informations annexes éventuellement modifiées avant de ré-écrire le secteur.

– **Ligne 980** : pour marquer l'annulation d'une clé, on remplace son premier caractère par un 0, ce qui garantit qu'elle ne sera pas retrouvée par la routine de recherche. Par contre, sa disparition effective de la table ne sera réalisée qu'à l'issue du reclassement.

– **Lignes 2000 à 2999** : fin et sortie du programme.

– **Lignes 3000 à 3070** : liste et affichage des enregistrements dont les clés sont actuellement classées. On lit successivement les secteurs depuis N=0 jusqu'à N=NZ (sauf si l'utilisateur arrête avant). Dans chacun de ces secteurs, on récupère les données de chaque clé non annulée (PEEK (BU+SB) <> 0), on lit l'enregistrement correspondant et on l'affiche à l'écran.

Programmes annexes

Initialisation d'une disquette "fichier"

Vous pouvez utiliser un programme comparable au programme INIT listé ci-après, afin de donner aux paramètres du système les valeurs correctes pour une toute première utilisation. C'est dans un tel programme que vous pouvez par ailleurs réserver plus de 3 pistes pour votre table de référence. Pour disposer de N pistes, faites un "POKE 44723, 4*N" avant la commande INIT, puis "POKE 44723, 12" juste après pour rétablir le standard du DOS. Pour N pistes,

le dernier secteur de clés accessible sera $NN = 16 * N - 1$.

Classement des clés

L'essentiel du travail est réalisé par une routine-machine, appelée (CALL 4096) par le programme Applesoft baptisé CLAS. Ce programme fournit à la routine les paramètres nécessaires :

- AC : adresse mémoire à partir de laquelle seront chargés en séquence tous les secteurs de clés classées.
- AN : adresse à partir de laquelle seront chargés en séquence tous les secteurs de vrac.
- L'adresse du buffer qui sera utilisé pour reconstituer les secteurs classés et les écrire sur disquette est POKée en 24 (poids faible) et 25 (poids fort).
- Le statut par rapport au classement (mixte "classé/vrac" ou "rien de classé") est POKé en 214.
- PC et SC donnent l'adresse sur disquette du dernier secteur de clés classées, tandis que PF et SF donnent le même renseignement pour le dernier secteur de vrac.

Au retour de la routine, le programme récupère les paramètres du système reclassé et constitue notamment le tableau N\$ des dernières clés de chaque secteur.

Tel qu'il vous est présenté, le programme suppose que la disquette fichier se trouve dans le lecteur S6-D2 au moment de son lancement. Si vous voulez l'utiliser avec une disquette dans le lecteur S6-D1, rajoutez "POKE 5357,1" après le chargement de la routine CLAS.OBJ.

A noter que le fonctionnement de la routine-machine suppose que l'on puisse charger tous les secteurs de clés en même temps dans la mémoire centrale. Le buffer utilisé, juste au-dessus de la routine elle-même, va de \$1500 à \$15FF, soit 5376 à 5631. Le nombre maximum de secteurs autorisés est donc : $INT((38400 - 5631) / 256)$, soit 128, de 0 à 127, ou encore 8 pistes complètes, ce qui fait déjà beaucoup, bien qu'il soit difficile de savoir combien d'octets sont perdus à la fin de chaque secteur du fait qu'on ne peut pas toujours y stocker un nombre rond de clés. Si on suppose par exemple que toutes les clés sont formatées à 15 caractères, plus 4 octets d'informations annexes, ce qui représente donc 15+4+4 octets par élément de la table, on pourrait gérer par ce système un fichier de plus de 1400 enregistrements, si la place disponible sur le reste de la disquette le permet. Sur ce dernier point, ajoutons encore que, bien que cela soit sans doute moins pratique, rien ne s'oppose à ce que la disquette du fichier de don-

nées se trouve dans un lecteur et celle de sa table de référence dans un autre. Dans ce cas, il faut songer à appeler le lecteur contenant la disquette "table" avant de recopier la table IOB du DOS dans le programme de traitement (voir ligne 4 du programme de démonstration). En outre, il sera bon de prévoir une boucle d'attente avant l'écriture ou la lecture des secteurs de clés par RWTS, car le passage trop rapide d'un lecteur à l'autre peut générer des erreurs du type "I/O ERROR" (dans notre programme de démonstration, l'ajout d'une boucle "FOR Z=1 TO 100: NEXT" à la fin de la ligne 65 ferait l'affaire).

Par ailleurs, si les secteurs sont nombreux et les clés longues, il peut ne pas y avoir assez de place sous le "HIMEM:4095" pour constituer le tableau N\$ sans écraser le programme de classement. Il faudrait alors modifier ce dernier afin d'écrire les dernières clés de chaque secteur dans le fichier FP au fur et à mesure de leur obtention, plutôt que de les regrouper d'abord dans un tableau en mémoire.

Routines en assembleur

Nous vous laissons le soin d'en conduire une analyse détaillée à l'aide des commentaires qu'elles comportent et nous bornerons ici à indiquer leurs principes généraux de fonctionnement.

Routine de recherche dans un secteur

Afin d'appliquer la même démarche aux clés classées ou en vrac, la routine n'utilise aucun algorithme de recherche. Elle traite en séquence toutes les clés du secteur lu en mé-

moire et compare chaque octet d'une clé à l'octet de même rang de la clé recherchée. Elle ne passe à la clé suivante, en considérant que la comparaison a échoué, que si un octet diffère (test BNE) ou si tous les octets sont identiques mais avec une différence entre la longueur des deux clés comparées.

Routine de classement

Elle se charge de la lecture en mémoire de tous les secteurs de clés, classées ou non. Le principe utilisé est alors le suivant :

- Rechercher la plus petite clé du vrac, la mettre dans une zone à part et signaler le fait en remplaçant sa longueur par \$FF dans son enregistrement.
- Prendre la première clé classée et la comparer à la plus petite du vrac. Les données de la plus petite des deux sont copiées dans le buffer alloué à la constitution de secteurs classés. Dès que ce buffer est plein, on l'écrit sur la disquette en notant la toute dernière clé qu'il contient, et on passe au secteur suivant en repartant au début du buffer.
- Si la plus petite est une clé classée, on refait ensuite la comparaison avec la clé classée suivante. Si c'est la clé "vrac" qui a été mise dans le buffer, on va chercher la nouvelle plus petite clé du vrac et on reprend la comparaison avec la même clé classée.
- A l'occasion de ces opérations, les clés annulées ne sont pas prises en compte et ne sont jamais reportées dans le buffer. Elles disparaissent donc de la table de référence.
- Le classement est terminé lorsque

l'on a ainsi transféré dans le buffer et ré-écrit sur disquette dans des secteurs classés toutes les clés qui étaient déjà classées et toutes les nouvelles clés précédemment stockées en vrac.

Conclusion

Ce système de gestion de fichiers nous semble présenter des points positifs qui peuvent justifier son emploi dans certaines de vos applications :

- Moyennant un reclassement périodique de la table de référence, les temps d'accès aux enregistrements sont très acceptables, même sur des fichiers de taille conséquente.
- Avant de commencer à "travailler", il n'est pas nécessaire d'attendre que se charge en mémoire, à partir d'un fichier important, une table de référence sous forme de tableau Applesoft. La lecture des fichiers annexes FP et PL ne représente plus qu'un délai d'attente minime.
- La mémoire centrale n'est pas encombrée par la table de référence des enregistrements, d'où gain de place pour vos programmes et réduction des problèmes de "nettoyage mémoire".

En revanche, l'exploitation de cette méthode dans vos traitements est sans doute plus complexe qu'une gestion uniquement réalisée sous "DOS 3.3 + Applesoft". Nous espérons cependant que l'analyse du programme de démonstration, dont les parties principales peuvent être reprises à quelques adaptations près pour d'autres programmes, vous permettra de trouver dans cet article une aide efficace pour vos problèmes de fichiers.

Programme TEST

```

1 HIMEM: 9 * 4096 - 1
2 AR = 9 * 4096:ADR = 9 * 4096 + 25
  6:CLE = ADR + 16:ASY0 = CLE +
  256:HS = INT (ASY0 / 256):LS =
  ASY0 - 256 * HS:HD = INT (ADR
  / 256):LD = ADR - 256 * HD:HL
  = INT (CLE / 256):LL = CLE -
  256 * HL
3 LC = 6:BU = 9 * 4096 + 5 * 256:HB
  = INT (BU / 256):LB = BU - 2
  56 * HB: POKE 8, LB: POKE 9, HB
4 RES = 26:ABL = 29:IO = 9 * 4096 +
  256 * 4:AA = 3: FOR I = 0 TO 1
  6: POKE IO + I, PEEK (47080 +
  I): NEXT :DT = 10 + 17:DH = INT
  (DT / 256):DL = DT - 256 * DH:
  POKE IO + 6, DL: POKE IO + 7, D
  H: POKE IO + 17, 0: POKE IO + 1
  8, 1: POKE IO + 19, 239: POKE IO
  + 20, 216
5 POKE IO + 3, 0:RWTS = 768: FOR I =
  0 TO 7: READ OC: POKE RWTS + I

```

```

,OC: NEXT : DATA 169,148,16
0,0,32,217,3,96
10 D$ = CHR$(4):D1$ = CHR$(13) +
  CHR$(4): PRINT D$"BLOAD REC.
  OBJ,A"AR:RU$ = D1$ + "PR#0":NN
  = 47: DIM N$(NN)
13 POKE AR + 21, LL: POKE AR + 22, H
  L: POKE AR + 59, LD: POKE AR +
  60, HD: POKE AR + 66, LD: POKE A
  R + 67, HD: POKE AR + 42, LS: POKE
  AR + 43, HS
15 PRINT D$"OPEN FP": PRINT D$"REA
  D FP": INPUT NZ,NF,TT,OO,IC: FOR
  I = 0 TO NZ: INPUT N$(I): NEXT
  : PRINT D$"CLOSE": DIM PL$(20)
20 TEXT : HOME : PRINT D$"OPEN PL"
  : PRINT D$"READ PL": INPUT PL:
  FOR I = 1 TO PL: INPUT PL$(I)
  : NEXT : PRINT D$"CLOSE": PRINT
  D$"OPEN DON,L100": PRINT RU$: GOTO
  100
50 FOR I = 0 TO NZ: IF Z$ < = N$(
  I) THEN N = I:I = NZ: NEXT : RETURN

```

```

60 NEXT :N = NZ + 1: RETURN
65 P = INT (N / 16):S = N - 16 * P

70 POKE IO + 4,P: POKE IO + 5,S: POKE
  IO + 9,HB: POKE IO + 8,LB: POKE
  IO + 12,C: CALL RWTS: RETURN
75 PRINT : INVERSE : PRINT ZM$;: NORMAL
  : GET Z$: PRINT : RETURN
80 C = 1: GOSUB 65: CALL AR:R = PEEK
  (RES):BL = PEEK (ABL):RI = 0:
  0 = 0: IF R = 0 AND ZC$ = N$(N
  ) AND N < NZ THEN R = 1: RETURN

81 IF R = 0 THEN RETURN
82 IF (N = NZ AND IC < 2) OR (TT =
  N AND IC > 0) THEN BL = 0
83 J = 2 * RI:A = PEEK (ADR + J) *
  256 + PEEK (ADR + J + 1): GOSUB
  90:ZM$ = "EST-CE LE BON": GOSUB
  95: IF 0$ = "0" THEN RI = RI +
  1:0 = 1: RETURN
84 RI = RI + 1: IF RI < = R - 1 THEN
  83
85 IF ZC$ = N$(N) AND N < > NZ THEN
  R = 1: RETURN
86 R = R * BL: RETURN
90 PRINT D$*READ DON,R*A: FOR I =
  0 TO 6: INPUT ZY$(I): NEXT : PRINT
  RU$: PRINT : VTAB 12: FOR I =
  0 TO 6: PRINT " ";ZY$(I);: CALL
  - 868: PRINT : NEXT : PRINT :
  RETURN
95 PRINT : INVERSE : PRINT ZM$;: NORMAL
  : INPUT " ? ";0$: IF 0$ < > "
  0" THEN 0$ = "N"
96 RETURN
98 PRINT D$*WRITE DON,R*A: FOR I =
  0 TO 6: PRINT ZY$(I): NEXT : PRINT
  D$*PR#0": RETURN
100 HOME : PRINT "1 - CREATION": PRINT
  "2 - MODIFICATION": PRINT "3 -
  CONSULTATION": PRINT "4 - ANN
  ULATION": PRINT "5 - LISTE": PRINT
  "6 - FIN": PRINT : INPUT "VOTR
  E CHOIX : ";CH: IF CH < 1 OR C
  H > 6 THEN 100
110 IF CH = 6 THEN 2000
115 IF CH = 5 THEN 3000
120 VTAB 10: HTAB 1: INPUT "CLE ?
  ";Z$: IF Z$ = "" THEN 100
125 ZC$ = Z$: IF LEN (Z$) > (250 -
  AA) THEN 120
130 L = LEN (Z$): POKE LC,L: FOR I
  = 1 TO L: POKE CLE + I - 1, ASC
  ( MID$ (Z$,I,1)): NEXT
200 IF NZ = - 1 THEN R = 0:P = 0:
  S = 0: GOTO 600
210 CI = IC + 1: ON CI GOTO 300,400
  ,500
300 GOSUB 50: IF N < = NZ THEN 31
  0
305 N = TT:C = 1: GOSUB 65:R = 0: GOTO
  600
310 GOSUB 80: IF R = 0 THEN 305
315 IF 0 = 1 THEN 600
320 N = N + 1: GOTO 310
400 GOSUB 50: IF N > NZ THEN 440
410 GOSUB 80: IF 0 = 1 THEN 600
420 IF R = 0 THEN N = NZ + 1: GOTO
  440
430 N = N + 1: GOTO 410
440 GOSUB 80: IF 0 = 1 THEN 600
450 IF N = TT THEN 600
460 N = N + 1: GOTO 440

```

```

500 N = 0: GOTO 440
600 ON CH GOTO 650,800,900,950
650 VTAB 8: HTAB 20: PRINT "CREATI
  ON "; IF 0 = 1 THEN 800
660 VTAB 12: HTAB 1: FOR I = 0 TO
  6: INPUT ZY$(I): NEXT : IF LEN
  (ZY$(4)) > 1 OR LEN (ZY$(5)) >
  1 OR LEN (ZY$(6)) > 1 THEN 66
  0
670 IF PL < > 0 THEN A = PL%(PL):
  PL = PL - 1: GOTO 690
680 NF = NF + 1:A = NF
690 GOSUB 98
700 LT = L + AA: IF 00 + LT + 5 < =
  255 AND TT < NN + 1 THEN 720
710 TT = TT + 1:00 = 0: IF TT > NN THEN
  ZM$ = "PLUS DE PLACE": GOSUB 7
  5:TT = NN + 1: GOTO 100
715 N = TT:C = 1: GOSUB 65
720 ZB = BU + 00: POKE ZB,LT: POKE
  ZB + 1,L: FOR I = 1 TO L: POKE
  ZB + 1 + I, PEEK (CLE + I - 1)
  : NEXT : FOR I = 4 TO 6: POKE
  ZB + L + I - 2, ASC (ZY$(I)): NEXT

725 POKE ZB + LT + 2, INT (A / 256
  ): POKE ZB + LT + 3,A - 256 *
  INT (A / 256): POKE ZB + LT +
  4,0
730 00 = 00 + LT + 4:C = 2: GOSUB 6
  5:IC = 1 +(IC = 2): IF NZ = -
  1 THEN NZ = 0
740 GOTO 100
800 VTAB 8: HTAB 20: PRINT "MODIFI
  CATION "
810 IF R = 0 THEN ZM$ = "N'EXISTE
  PAS": POKE 34,9: HOME : TEXT :
  GOSUB 75: GOTO 100
815 VTAB 12: HTAB 1: FOR I = 0 TO
  6: INPUT ZY$(I): NEXT : IF LEN
  (ZY$(4)) > 1 OR LEN (ZY$(5)) >
  1 OR LEN (ZY$(6)) > 1 THEN 81
  5
820 GOSUB 98:SB = PEEK (ASY0 + RI
  - 1)
840 FOR I = 4 TO 6: POKE BU + SB +
  L + I - 4, ASC (ZY$(I)): NEXT
  :C = 2: GOSUB 65: GOTO 100
900 VTAB 8: HTAB 20: PRINT "CONSUL
  TATION "; IF R = 0 THEN 810
910 VTAB 22:ZM$ = "APPUYEZ SUR UNE
  TOUCHE": GOSUB 75: GOTO 100
950 VTAB 8: HTAB 20: PRINT "ANNULA
  TION "; IF R = 0 THEN 810
955 PL = PL + 1:PL%(PL) = A
960 SB = PEEK (ASY0 + RI - 1)
980 POKE BU + SB,0:C = 2: GOSUB 65
  : GOTO 100
2000 PRINT D1$*CLOSE": PRINT D$*OP
  EN FP": PRINT D$*WRITE FP": PRINT
  NZ: PRINT NF: PRINT TT: PRINT
  00: PRINT IC: FOR I = 0 TO NZ:
  PRINT N$(I): NEXT : PRINT D$*
  CLOSE"
2010 PRINT D$*OPEN PL": PRINT D$*W
  RITE PL": PRINT PL: FOR I = 1 TO
  PL: PRINT PL%(I): NEXT : PRINT
  D$*CLOSE"
2999 END
3000 N = 0
3010 C = 1: GOSUB 65:SB = 2
3020 L1 = PEEK (BU + SB - 2): IF L
  1 > 0 THEN 3050

```

```

3030 N = N + 1: IF N < = NZ THEN 3
      010
3040 GOTO 2000
3050 L2 = PEEK (BU + SB - 1):Z$ =
      **: IF PEEK (BU + SB) = 0 THEN
      SB = SB + L1 + 4: GOTO 3020
3055 FOR I = 0 TO L2 - 1:Z$ = Z$ +
      CHR$ ( PEEK (BU + SB + I)): NEXT
      :A = 256 * PEEK (BU + SB + L1
      ) + PEEK (BU + SB + L1 + 1): VTAB
      10: HTAB 1: PRINT Z$:; CALL -
      868: PRINT :SB = SB + L1 + 4
3060 GOSUB 90: GET Z$: IF Z$ = "S"
      THEN 2000
3070 PRINT : GOTO 3020

```

Programme CLAS

```

5 HIMEM: 4095
7 DIM N$(47)
10 D$ = CHR$ (4): PRINT D$"BLOAD C
      LAS.OBJ": PRINT D$"OPEN FP": PRINT
      D$"READ FP": INPUT NZ,NF,TT,OO
      ,II: FOR I = 0 TO NZ: INPUT N$
      (I): NEXT : PRINT D$"CLOSE":AA
      = 3
20 IF II = 0 THEN END
25 NZ = NZ + 1:TT = TT + 1
30 AC = 38400 - 256 * NZ:AN = AC -
      ((TT - NZ) * 256): POKE 6,0: POKE
      7,AC / 256: POKE 8,0: POKE 9,A
      N / 256: POKE 24,0: POKE 25,21
      : POKE 214,II - 1
33 POKE 26,0: POKE 28,0
35 NZ = NZ - 1:TT = TT - 1
40 PC = INT (NZ / 16):SC = NZ - 16

```

Programme REC.SCE

```

1 ;*****
2 ;*
3 ;* RECHERCHE D'UNE CLE *
4 ;* (LISA 1.5) *
5 ;* CODE = REC.OBJ *
6 ;*
7 ;*****
8 ;
9 ORG $9000
10 OBJ $800
11 LC EPZ $6 ;LONGUEUR
      DE LA CLE
12 INB EPZ $8 ;ADRESSE
      DU BUFFER
13 LC2 EPZ $18 ;LONGUEUR
      DE LA CLE LUE DANS LE BUFFER
14 LT2 EPZ $19 ;LONGUEUR
      TOTALE CLE LUE DANS BUFFER
15 RES EPZ $1A ;RESULTAT
      RECHERCHE/NBRE CLES POSSIBLES
16 SY EPZ $1B ;POSITION
      CLE TROUVEE DANS BUFFER
17 ABL EPZ $1D ;DRAPEAU
      "FIN DE BUFFER"
18 LDX #0
19 STX RES
20 STX ABL
21 LDY #0 ;==> IERE
      CLE DU BUFFER
22 LDA (INB),Y

```

```

* PC:PF = INT (TT / 16):SF =
      TT - 16 * PF: POKE 30,PC: POKE
      31,SC: POKE 206,PF: POKE 207,S
      F
50 CALL 4096: FOR I = 0 TO NZ:N$(I
      ) = **: NEXT
60 P = PEEK (235):S = PEEK (236):
      NZ = (P * 16) + S - 1: FOR I =
      0 TO NZ:B = PEEK (768 + 2 * I
      ):SY = PEEK (769 + 2 * I):B =
      B * 256 + SY:J = 0:L = PEEK (
      B - 2) - AA
70 N$(I) = N$(I) + CHR$ ( PEEK (B +
      J)):J = J + 1: IF J < L THEN 7
      0
80 NEXT :TT = NZ + 1:II = 0:OO = 0
      : PRINT : PRINT D$"OPEN FP": PRINT
      D$"WRITE FP": PRINT NZ: PRINT
      NF: PRINT TT: PRINT OO: PRINT
      II: FOR I = 0 TO NZ: PRINT N$(
      I): NEXT : PRINT D$"CLOSE": PRINT
      D$"PR#0"

```

Programme INIT

```

5 NN = 47
10 D$ = CHR$ (4): PRINT D$"INIT H
      ELLO": PRINT D$"OPEN FP": PRINT
      D$"WRITE FP": PRINT - 1: PRINT
      0: PRINT 0: PRINT 0: PRINT 2:
      Z$ = **: FOR I = 0 TO NN: PRINT
      Z$: NEXT
20 PRINT D$"OPEN PL": PRINT D$"WR
      ITE PL": PRINT 0: PRINT 0: PRINT
      D$"CLOSE": PRINT D$"PR#0"

```

```

23 S2 STA LT2
24 INY
25 LDA (INB),Y
26 STA LC2
27 INY
28 STY SY ;POSITION
      DEBUT DE CLE
29 S0 LDA CLE,X ;COMPARAI
      SON CLE/CLE DU BUFFER
30 CMP (INB),Y
31 BNE S3
32 INX
33 INY
34 CPX LC ;FIN DE L
      A CLE ?
35 BNE S0
36 CPX LC2 ;MEME LON
      GUEUR QUE CLE DU BUFFER ?
37 BNE S3
38 LDX RES
39 LDA SY
40 STA SY0,X ;POSITION
      CLE POSSIBLE DANS BUFFER
41 LDA RES
42 S1 INC RES
43 ASL
44 TAX ;X=2*A
45 LDA SY
46 CLC
47 ADC LT2
48 TAY
49 LDA (INB),Y ;STOCKAGE

```

```

ADRESSE DE L'ENREGISTREMENT
50 STA ADR,X
51 INY
52 INX
53 LDA (INB),Y
54 STA ADR,X
55 LDX #0
56 INY
57 LDA (INB),Y
58 BNE S2 ;SI (<) 0
-> RESTE DES CLES A EXAMINER
59 LDA #1 ;1 DES CL
ES POSSIBLES EST EN FIN BUFFER
60 STA ABL ;AUTRES P
OSSIBLES SUR SECTEUR SUIVANT
61 RTS
62 S3 LDX #0 ;PASSAGE
A LA CLE SUIVANTE
63 LDA SY ;DANS LE
BUFFER
64 CLC
65 ADC LT2
66 ADC #2
67 TAY
68 LDA (INB),Y
69 BNE S2
70 RTS
71 ADR DFS $10 ;ADRESSES
DES ENREGISTREMENTS
72 CLE DFS $FF ;CLE RECH
ERCHEE
73 SY0 DFS $8 ;POSITION
S DES CLES POSSIBLES
74 DCM "INT"
75 END

```

Programme CLAS.SCE

```

1 ;*****
2 ;*
3 ;* ROUTINE DE RECLASSEMENT *
4 ;* (LISA 1.5) *
5 ;* CODE = CLAS.OBJ *
6 ;*
7 ;*****
8 ;
9 ORG $1000
10 OBJ $800
11 ZCC EPZ $6 ;DEBUT Z0
NE CLES CLASSEES
12 ZCT EPZ $8 ;DEBUT Z0
NE CLES "EN VRAC"
13 BUF EPZ $18 ;ADRESSE
DU BUFFER
14 ZCCP EPZ $1A ;DEBUT Z0
NES "DE TRAVAIL"
15 ZCTP EPZ $1C ; "
"
16 PFC EPZ $1E ;PFC/SFC
= PISTE/SECTEUR DU
17 SFC EPZ $1F ;DERNIER
SECTEUR CLASSE
18 PFT EPZ $CE ;PFT/SFT
= PISTE/SECTEUR DU
19 SFT EPZ $CF ;DERNIER
SECTEUR "EN VRAC"
20 IC EPZ $D6 ;STATUT C
LES/CLASSEMENT (0/1)
21 LU EPZ $D7 ;LONGUEUR
"UTILE" DE LA CLE
22 P EPZ $EB ;P/S = PI

```

```

STE/SECTEUR POUR RWTS
23 S EPZ $EC
24 LC EPZ $ED ;LONGUEUR
TOTALE DE LA CLE
25 ZTPP EPZ $EE
26 LCT EPZ $EF ;LONG. TO
TALE + PETIT CLE DU VRAC
27 SZC EPZ $F9 ;POINTEUR
"FIN ZONE CLASSEE"
28 SZT EPZ $FA ;POINTEUR
"FIN ZONE VRAC"
29 SYC EPZ $FB ;POINTEUR
"DEBUT CLE" DANS BLOC CLASSE
30 SYCO EPZ $FC ;POSITION
+ GRANDE CLE DANS SON BLOC
31 LCT2 EPZ $FD ;LONG. TO
TALE D'UNE CLE DU VRAC
32 SYB EPZ $FE ;POINTEUR
DANS BUFFER RECOPIE/DISQUETTE
33 CLEMAX EQU $300 ;STOCK. D
ERNIERE CLE/CHAQUE SECTEUR
34 SXC EPZ $FF ;POINTEUR
DANS LA ZONE "CLEMAX"
35 LDA #1 ;INITIALI
SATION PARAMETRES POUR RWTS
36 LDX #$C
37 STA IOB,X
38 LDA ZCT+1
39 STA ZCTP+1
40 LDA IC
41 BNE S7 ;IC=1 -->
RIEN N'EST CLASSE
42 LDA ZCC+1 ;DEBUT Z0
NE "NON CLASSE"
43 STA ZCCP+1
44 LDA #0
45 STA P
46 STA S
47 LDX #8 ;CMINT(<>0
SERVIRA POUR LA SUITE
48 STX CMINT
49 LDA ZCC
50 STA IOB,X
51 INX
52 S4 LDA ZCCP+1 ;LECTURE
DES PISTES CLASSEES
53 STA IOB,X
54 JSR RWTS
55 LDY P
56 LDX S
57 CPY PFC
58 BNE S1
59 CPX SFC
60 BEQ S50
61 S1 CPX #$F
62 BNE S2
63 INY
64 STY P
65 LDX #0
66 BEQ S3
67 S2 INX
68 S3 STX S
69 INC ZCCP+1
70 LDX #9
71 JMP S4
72 S50 INC ZCCP+1 ;POINTEUR
FIN DE ZONE
73 LDA ZCCP+1
74 STA SZC
75 S5 INX ;PASSAGE
AU PREMIER SECTEUR DU VRAC
76 CPX #$10 ;ET CALCUL
L DES SECTEURS SUIVANTS

```

77	BNE S6	
78	INY	
79	STY P	
80	LDX #0	
81 S6	STX S	
82	JMP S8	
83 S7	LDA #0	;ON ARRIV
	E LA SI RIEN N'EST CLASSE	
84	STA P	;PREMIER
	SECTEUR VRAC = 0/0	
85	STA S	
86 S8	LDA ZCTP+1	;LECTURE
	DES SECTEURS DE CLES EN VRAC	
87	LDX #9	
88	STA IOB,X	
89	JSR RWTS	
90	LDY P	
91	LDX S	
92	CPY PFT	
93	BNE S9	
94	CPX SFT	
95	BEQ S10	
96 S9	INC ZCTP+1	
97	JMP S5	
98 S10	INC ZCTP+1	;M.A.J. P
	ONTEUR FIN DE ZONE	
99	LDA ZCTP+1	
100	STA SZT	
101	LDA #0	
102	STA SXC	
103	STA P	;ECRITURE
	A PARTIR DE PISTE 0/SECTEUR 0	
104	STA S	
105	STA SYB	
106	LDA #2	;COMMANDE
	"ECRITURE" POUR RWTS	
107	STA PFC	
108	LDX ##C	
109	STA IOB,X	
110	LDA IC	
111	BEQ S100	
112	JMP S27	;RIEN N'E
	ST CLASSE	
113 S100	LDA ZCC+1	;EXAMEN A
	PARTIR DU DEBUT DE	
114	STA ZCCP+1	;LA ZONE
	DES CLES CLASSEES	
115 S12	LDY #0	;IERE CLE
	D'UN BLOC (256 OCTETS)	
116	LDA (ZCCP),Y	
117 S11	STA LC	;LONGUEUR
	TOTALE	
118	INY	
119	LDA (ZCCP),Y	
120	STA LU	;LONGUEUR
	UTILE	
121	INY	;SYC POIN
	TE SUR LE 1ER CARACTERE	
122	STY SYC	;DE LA CL
	E	
123	LDA (ZCCP),Y	
124	BNE S13	;SI 0 -->
	CLE ANNULEE	
125 S19	LDA SYC	;ON SAUTE
	A LA CLE SUIVANTE	
126	CLC	
127	ADC LC	
128	ADC #2	
129	TAY	
130	LDA (ZCCP),Y	
131	BNE S11	;SI 0 -->
	LE BLOC EST FINI	
132	INC ZCCP+1	;PASSAGE
	AU BLOC SUIVANT	
133	LDA ZCCP+1	
134	CMP SZC	;DERNIER
	BLOC ?	
135	BCC S12	;NON !
136	INC IC	;OUI -->
	FINIR LE VRAC S'IL EN RESTE	
137	JMP S270	
138 S13	LDA CMINT	;CMINT =
	0 --> PLUS DE VRAC	
139	BEQ S131	
140	LDA PFC	;PFC = 0
	--> ON A DEJA CHERCHE	
141	BEQ S130	;LA PLUS
	PETITE CLE DU VRAC	
142 S129	JSR RPT	
143 S130	LDA CMINT	;PLUS DE
	VRAC SI CMINT=0	
144	BEQ S131	
145	LDA LUT	;LM=1 SI
	LUT<LU	
146	LDX #0	
147	STX LM	
148	CMP LU	
149	BCS S14	
150	INC LM	
151 S14	LDY SYC	;COMPARAI
	SON CLE CLASSE A	
152 S140	LDA (ZCCP),Y	;LA PLUS
	PETITE DU VRAC	
153	CMP CMINT,X	
154	BCC S131	;CLASSEE
	< VRAC	
155	JMP S20	
156 S131	LDA LC	;IL FAUT
	LC+5 OCTETS PAR CLE EN	
157	CLC	;COMPTANT
	LE 0 QUI MARQUE LA FIN	
158	ADC #5	;DU BLOC
159	CLC	
160	ADC SYB	
161	BCC S16	;IL Y A D
	E LA PLACE SUR LE SECTEUR	
162	JSR S132	;ECRIT LE
	SECTEUR/PASSE AU SUIVANT	
163	JMP S16	
164 S132	LDX SXC	
165	LDA SFC	;POIDS FO
	RT ADRESSE DU BLOC	
166	STA CLEMAX,X	
167	LDA SYC0	;POSITION
	DE LA CLE DANS LE BLOC	
168	INX	
169	STA CLEMAX,X	
170	INX	
171	STX SXC	;M.A.J. P
	ONTEUR DE CLEMAX	
172	LDA BUF+1	;ECRIT LE
	SECTEUR CLASSE QUE	
173	LDX #9	;L'ON VIE
	NT DE CONSTITUER	
174	STA IOB,X	
175	JSR RWTS	
176	LDX S	;PASSAGE
	AU SECTEUR SUIVANT	
177	INX	
178	CPX ##10	
179	BNE S15	
180	INC P	
181	LDX #0	
182 S15	STX S	
183	LDY #0	;RAZ POIN
	TEUR DU BUFFER	

184	STY SYB	
185	RTS	
186	S16 LDY SYB	;RECOPIE
	LES DONNEES DE LA CLE	
187	LDA LC	;DANS LE
	BUFFER	
188	STA (BUF),Y	
189	CLC	
190	ADC #2	
191	STA LUT2	;LONGUEUR
	TOTALE + ADRESSE	
192	INY	
193	LDA LU	
194	STA (BUF),Y	
195	INY	
196	STY SYB	;M.A.J. P
	OINTEURS DE CLEMAX POUR	
197	LDY SYC	;LE CAS 0
	U CETTE CLE SERAIT LA	
198	STY SYC0	;DERNIERE
	DU SECTEUR EN COURS	
199	LDA ZCCP+1	;DE CONST
	ITUTION	
200	STA SFC	
201	LDX #0	
202	S17 LDA (ZCCP),Y	;TRANSFER
	T INTERMEDIAIRE	
203	STA CMIN,X	
204	INX	
205	INY	
206	CPX LUT2	
207	BNE S17	
208	LDY SYB	
209	LDX #0	
210	S18 LDA CMIN,X	;TRANSFER
	T DANS LE BUFFER	
211	STA (BUF),Y	
212	INX	
213	INY	
214	CPX LUT2	
215	BNE S18	
216	LDA #0	;MARQUE L
	A FIN PROVISOIRE DU SECTEUR	
217	STA (BUF),Y	
218	STA PFC	
219	STY SYB	
220	JMP S19	;PASSAGE
	A CLE CLASSEE SUIVANTE	
221	S20 BEQ S21	
222	JMP S23	
223	S21 INX	;OCTET "C
	LASSE" = OCTET "VRAC"	
224	INY	
225	LDA LM	
226	BNE S22	;CLE "URA
	C" EST LA PLUS COURTE	
227	CPX LU	;FIN DE L
	A CLE LA PLUS COURTE ?	
228	BEQ S210	
229	JMP S140	;NON -> S
	UITE COMPARAISON	
230	S210 JMP S131	
231	S22 CPX LUT	;FIN DE L
	A CLE LA PLUS COURTE ?	
232	BEQ S23	
233	JMP S140	;NON -> S
	UITE COMPARAISON	
234	S23 LDA LCT	;LA CLE "
	VRAC" EST LA PLUS PETITE	
235	CLC	;STOCKAGE
	DANS BUFFER AVEC CHANGEMENT	
236	ADC #5	;PRELABL
	E DE SECTEURR SI NECESSAIRE	
237	CLC	
238	ADC SYB	
239	BCC S24	
240	JSR S132	
241	S24 LDY SYB	;M.A.J. P
	OINTEURS DE CLEMAX	
242	LDA LCT	
243	STA (BUF),Y	
244	INY	
245	LDA LUT	
246	STA (BUF),Y	
247	LDA SYM	
248	STA SYC0	
249	LDA ZTPP	
250	STA SFC	
251	INY	
252	LDX #0	
253	S25 LDA CMINT,X	;TRANSFER
	T DE LA CLE	
254	STA (BUF),Y	
255	INX	
256	INY	
257	CPX LCT	
258	BNE S25	
259	STY SYB	
260	LDA SYM	
261	CLC	
262	ADC LCT	
263	STA PFT	
264	TAY	
265	LDA (ZCTP),Y	;TRANSFER
	T DE L'ADRESSE	
266	LDY SYB	;DANS LE
	BLOC	
267	STA (BUF),Y	
268	INC SYB	
269	LDY PFT	
270	INY	
271	LDA (ZCTP),Y	
272	LDY SYB	
273	STA (BUF),Y	
274	INY	
275	LDA #0	;FIN DU B
	UFFER	
276	STA (BUF),Y	
277	STY SYB	
278	LDX IC	
279	BNE S26	;IL NE RE
	STE QUE DU VRAC	
280	INX	
281	STX PFC	;SIGNALE
	QU'IL FAUDRA CHERCHER LA	
282	JMP S129	;NOUVELLE
	+ PETITE CLE DU VRAC	
283	S26 LDA CMINT	
284	BNE S27	;IL PEUT
	ENCORE RESTER DU VRAC	
285	S28 JSR S132	;ECRIT DE
	RNIER SECTEUR ET SORT	
286	RTS	
287	S27 JSR RPT	;CHERCHE
	+ PETITE CLE DU VRAC	
288	S270 LDA CMINT	
289	BEQ S28	;PLUS DE
	VRAC -> FIN	
290	JMP S23	;TRAITEME
	NT "BUFFER"	
291	RPT LDA ZCT+1	;IER BLOC
	DU VRAC	
292	STA ZCTP+1	
293	LDY #0	
	CATEUR "RESTE DU VRAC"	;RAZ INDI

294	STY CMINT	;ON CHERC	ETE ANNULEE
	HE UNE 1ERE + PETITE POSSIBLE		351 CMP CMINT,X ;COMPAR.
295	C5 LDA (ZCTP),Y	;DEBUT CL	AVEC + PETITE POTENTIELLE
	E "URAC" DU BLOC		352 BEQ C7
296	STA LCT		353 BCS C1 ;CMINT IN
297	STA LCT2		FERIEUR -> ON LA GARDE
298	INY		354 C11 LDY SY ;ON CHANG
299	LDA (ZCTP),Y		E LE CMINT
300	INY		355 LDA (ZCTP),Y
301	STY SY		356 LDX ZCTP+1
302	CMP #\$FF	;LA CLE A	357 STX ZTPP
	DEJA ETE RECLASSEE ?		358 LDX LUT2
303	BEQ C1		359 STX LUT
304	C1000 STA LUT		360 LDX LCT2
305	STA LUT2		361 STX LCT
306	LDA ZCTP+1		362 JMP C6
307	STA ZTPP	;BLOC DE	363 C7 INX ;OCTET "C
	LA + PETITE POTENTIELLE		MINT" = OCTET "CLE COMPAREE"
308	LDA (ZCTP),Y		364 INY
309	BEQ C1	;LA CLE A	365 LDA LM
	ETE ANNULEE		366 BNE C10
310	C6 STY SYM		367 CPX LUT ;FIN DE L
311	LDX #0	;LA 1ERE	A PLUS COURTE (LUT<LUT2) ?
	CLE NON ANNULEE/NON RECLASSEE		368 BNE C9
312	C0 STA CMINT,X	;QUE L'ON	369 JMP C1 ;OUI -> 0
	TROUVE EST TRANSFEREE		N GARDE CMINT
313	INY	;DANS CMI	370 C10 CPX LUT2 ;FIN DE L
	NT		A PLUS COURTE (LUT2<LUT) ?
314	INX		371 BEQ C11 ;OUI -> 0
315	CPX LCT2		N CHANGE CMINT
316	BEQ C1		372 JMP C9
317	LDA (ZCTP),Y		373 C12 LDA CMINT
318	JMP C0		374 BEQ C13 ;IL N'Y A
319	C1 LDA SY	;PASSAGE	PLUS DE URAC
	A LA CLE "URAC" SUIVANTE		375 LDA ZTPP
320	CLC		376 STA ZCTP+1 ;BLOC DE
321	ADC LCT2		LA + PETITE DEFINITIVE
322	ADC #2		377 LDY SYM ;MARQUE L
323	TAY		E RECLASSEMENT DE LA CLE
324	C2 LDA (ZCTP),Y		378 DEY
325	BNE C3		379 LDA #\$FF
326	INC ZCTP+1	;FIN BLOC	380 STA (ZCTP),Y
	-> PASSE AU BLOC SUIVANT		381 C13 RTS
327	LDA ZCTP+1		382 RWTS LDX #4
328	CMP SZT	;DERNIER	383 LDA P
	BLOC DEJA TRAITE ?		384 STA IOB,X
329	BCC C4	;NON !	385 INX
330	JMP C12	;OUI->ON	386 LDA S
	A LA PLUS PETITE S'IL EN RESTE		387 STA IOB,X
331	C4 LDY #0		388 LDA /IOB
332	BEQ C2		389 LDY #IOB
333	C3 LDA CMINT		390 JSR \$3D9
334	BEQ C5	;ON A TOU	391 RTS
	JOURS LA PLUS PETITE		392 CMIN DFS \$FF
335	LDA (ZCTP),Y		393 CMINT DFS \$FF ;CONTIENT
336	STA LCT2		TOUJOURS + PETITE CLE "URAC"
337	INY		394 LM DFS \$1
338	LDA (ZCTP),Y		395 SYM DFS \$1 ;POSITION
339	INY		+PETITE URAC DANS SON BLOC
340	STY SY		396 LUT DFS \$1 ;LONG. UT
341	CMP #\$FF	;CLE DEJA	ILE + PETITE CLE DU URAC
	RECLASSEE		397 LUT2 DFS \$1 ;LONG. UT
342	BEQ C1		ILE D'UNE CLE DU URAC
343	STA LUT2		398 SY DFS \$1 ;POINTEUR
344	LDX #0		DANS UN BLOC DE URAC
345	STX LM	;LM=1 SI	399 IOB HEX 016002000000
	LUT2>LUT		400 ADR DCB
346	CMP LUT		401 HEX 000000000100FE6002
347	BCS C9		402 DCB HEX 0001EFD8
348	INC LM		403 DCM "INT"
349	C9 LDA (ZCTP),Y		404 END
350	BEQ C1	;LA CLE A	

Récapitulation REC.OBJ

```

9000- A2 00 86 1A 86 1D A0 00
9008- B1 08 85 19 C8 B1 08 85
9010- 18 C8 84 1B BD 6F 90 D1
9018- 08 D0 35 E8 C8 E4 06 D0
9020- F3 E4 18 D0 2B A6 1A A5
9028- 1B 9D 6E 91 A5 1A E6 1A
9030- 0A AA A5 1B 18 65 19 A8
9038- B1 08 9D 5F 90 C8 E8 B1
9040- 08 9D 5F 90 A2 00 C8 B1
9048- 08 D0 BF A9 01 85 1D 60
9050- A2 00 A5 1B 18 65 19 69
9058- 02 A8 B1 08 D0 AC 60
    
```

Récapitulation CLAS.OBJ

```

1000- A9 01 A2 0C 9D EB 14 A5
1008- 09 85 1D A5 D6 D0 53 A5
1010- 07 85 1B A9 00 85 EB 85
1018- EC A2 08 8E E7 13 A5 04
1020- 9D EB 14 E8 A5 1B 9D EB
1028- 14 20 D3 12 A4 EB A6 EC
1030- C4 1E D0 04 E4 1F F0 15
1038- E0 0F D0 07 C8 84 EB A2
1040- 00 F0 01 E8 86 EC E6 1B
1048- A2 09 4C 24 10 E6 1B A5
1050- 1B 85 F9 E8 E0 10 D0 05
1058- C8 84 EB A2 00 86 EC 4C
1060- 68 10 A9 00 85 EB 85 EC
1068- A5 1D A2 09 9D EB 14 20
1070- D3 12 A4 EB A6 EC C4 CE
1078- D0 04 E4 CF F0 05 E6 1D
1080- 4C 53 10 E6 1D A5 1D 85
1088- FA A9 00 85 FF 85 EB 85
    
```

```

1090- EC 85 FE A9 02 85 1E A2
1098- UC 9D EB 14 A5 D6 F0 03
10A0- 4C 03 12 A5 07 85 1B A0
10A8- 00 B1 1A 85 ED C8 B1 1A
10B0- 85 D7 C8 84 FB B1 1A D0
10B8- 19 A5 FB 18 65 ED 69 02
10C0- A8 B1 1A D0 E6 E6 1B A5
10C8- 1B C5 F9 90 DA E6 D6 4C
10D0- 06 12 AD E7 13 F0 27 A5
10D8- 1E F0 03 20 0E 12 AD E7
10E0- 13 F0 1B AD E8 14 A2 00
10E8- 8E E6 14 C5 D7 B0 03 EE
10F0- E6 14 A4 FB B1 1A D0 E7
10F8- 13 90 03 4C 7F 11 A5 ED
1100- 18 69 05 18 65 FE 90 32
1108- 20 0E 11 4C 3A 11 A6 FF
1110- A5 1F 9D 00 03 A5 FC E8
1118- 9D 00 03 E8 86 FF A5 19
1120- A2 09 9D EB 14 20 D3 12
1128- A6 EC E8 E0 10 D0 04 E6
1130- EB A2 00 86 EC A0 00 84
1138- FE 60 A4 FE A5 ED 91 18
1140- 18 69 02 8D E9 14 C8 A5
1148- D7 91 18 C8 84 FE A4 FB
1150- 84 FC A5 1B 85 1F A2 00
1158- B1 1A 9D E8 12 EB C8 EC
1160- E9 14 D0 F4 A4 FE A2 00
1168- 8D E8 12 91 18 EB C8 EC
1170- E9 14 D0 F4 A9 00 91 18
1178- 85 1E 84 FE 4C B9 10 F0
1180- 03 4C 9D 11 E8 C8 AD E6
1188- 14 D0 0A E4 D7 F0 03 4C
1190- F4 10 4C FE 10 EC E8 14
1198- F0 03 4C F4 10 A5 EF 18
11A0- 69 05 18 65 FE 90 03 20
11A8- 0E 11 A4 FE A5 EF 91 18
11B0- C8 AD E8 14 91 18 AD E7
11B8- 14 85 FC A5 EE 85 1F C8
11C0- A2 00 BD E7 13 91 18 E8
11C8- C8 E4 EF D0 F5 84 FE AD
11D0- E7 14 18 65 EF 85 CE A8
    
```

```

11D8- B1 1C A4 FE 91 18 E6 FE
11E0- A4 CE C8 B1 1C A4 FE 91
11E8- 18 C8 A9 00 91 18 84 FE
11F0- A6 D6 D0 06 E8 86 1E 4C
11F8- D8 10 AD E7 13 D0 04 20
1200- 0E 11 60 20 0E 12 AD E7
1208- 13 F0 F4 4C 9D 11 A5 09
1210- 85 1D A0 00 8C E7 13 B1
1218- 1C 85 EF 85 FD C8 B1 1C
1220- C8 8C EA 14 C9 FF F0 21
1228- 8D E8 14 8D E9 14 A5 1D
1230- 05 EE B1 1C F0 13 8C E7
1238- 14 A2 00 9D E7 13 C8 E8
1240- E4 FD F0 05 B1 1C 4C 3B
1248- 12 AD EA 14 18 65 FD 69
1250- 02 A8 B1 1C D0 0F E6 1D
1258- A5 1D C5 FA 90 03 4C C1
1260- 12 A0 00 F0 ED AD E7 13
1268- F0 AD B1 1C 85 FD C8 B1
1270- 1C C8 8C EA 14 C9 FF F0
1278- D0 8D E9 14 A2 00 8E E6
1280- 14 CD E8 14 B0 03 EE E6
1288- 14 B1 1C F0 BC DD E7 13
1290- F0 18 80 85 AC EA 14 B1
1298- 1C A6 1D 86 EE AE E9 14
12A0- 8E E8 14 A6 FD 86 EF 4C
12A8- 36 12 E8 C8 AD E6 14 D0
12B0- 08 EC E8 14 D0 D3 4C 49
12B8- 12 EC E9 14 F0 D6 4C 89
12C0- 12 AD E7 13 F0 0C A5 EE
12C8- 85 1D AC E7 14 88 A9 FF
12D0- 91 1C 60 A2 04 A5 EB 9D
12D8- EB 14 E8 A5 EC 9D EB 14
12E0- A9 14 A0 EB 20 D9 03 60
    
```

TABLES

```

14EB- 01 60 02 00 00
14F0- 00 FC 14 00 00 00 01
14F8- 00 FE 60 02 00 01 EF D8
    
```

VOUS PROGRAMMEZ SUR APPLE II VOUS VOULEZ GAGNER DU TEMPS VOUS CHERCHEZ A AMELIORER VOS CAPACITES LIGHT EST FAIT POUR VOUS

Depose APP "France Logiciel"

LIGHT est un ensemble de puissants logiciels d'aide à la programmation conçu pour :

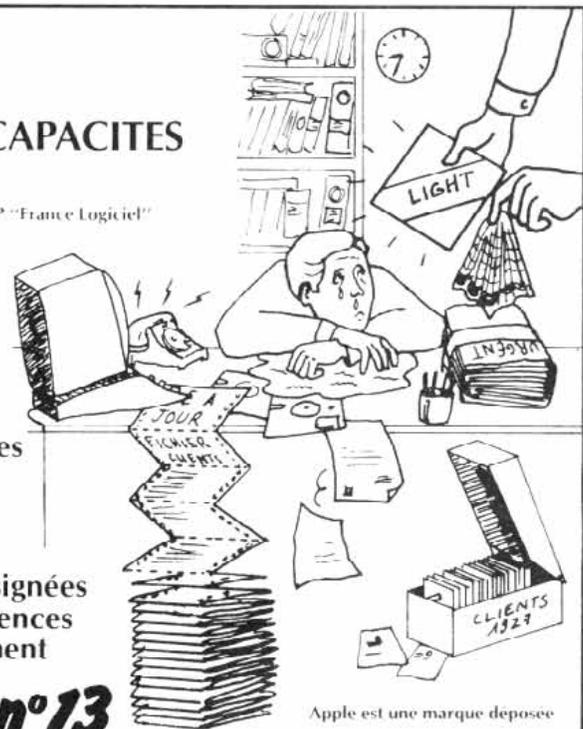
- mieux gérer les fichiers et les disquettes
mise à jour d'un catalogue général
état des secteurs libres sur un disque
- faciliter l'exploration et l'exploitation des disquettes
accès direct, lecture, écriture des secteurs
transfert et recherche des séquences d'octets
- accélérer l'analyse de vos programmes
en Basic : recherche des tokens et des suites désignées
en Binaire : impression en désassemblé de séquences
d'octets ; pistage en tout ou partie du déroulement

banc d'essais dans Pom's n°13

POUR EN SAVOIR PLUS SUR LIGHT :

M. _____
adresse _____

demande une documentation complète, sans engagement de sa part, aux Années Nouvelles, 70, rue du Javelot, 75013 Paris.



Apple est une marque déposée

MAC CAHIER

Nous vous avons promis un cahier Macintosh dans chaque numéro de Pom's. Voici le premier de ces cahiers. Vous y trouverez, outre les remarques de cette page, un article sur les ROMs du Mac, un article sur la programmation de ses curseurs et enfin (page 49) un tableau avec les renseignements les plus récents sur les logiciels du Macintosh.

Politique éditoriale

Nous nous sommes longuement interrogés sur la politique à suivre pour nos articles dans le cahier du Mac. Nous avons finalement décidé de maintenir, au niveau du Macintosh, la politique de qualité de Pom's, qui a toujours refusé le nivellement vers le bas. Nous partirons donc du principe que les lecteurs de Pom's auront lu les modes d'emploi du Mac et des logiciels qu'ils utilisent et qu'il est donc inutile de leur proposer des explications qui ne feraient que résumer la documentation.

Contenu de la disquette Mac

Nous vous proposons en outre dès ce numéro une disquette Macintosh, dans laquelle vous trouverez :

- le programme Disk Copy, qui permet de réaliser une copie de disquettes en quatre passages

avec seulement le lecteur intégré du Mac;

- la police de caractères Cairo, dont vous pouvez voir les signes en bas de page;
- les programmes du Mac publiés dans les numéros 14/15 de Pom's.

Nous ajouterons à la disquette, au fur et à mesure, les programmes des numéros suivants, ainsi que quelques surprises supplémentaires que nous vous laissons découvrir...

Multiplan sur Macintosh

1. Tableau plus compact

Pour avoir plus de lignes et de colonnes à l'écran, écartez à l'aide du Font Mover les polices Seattle de la disquette Multiplan, en les mettant dans un fichier Caractères afin de pouvoir les récupérer plus tard. Vous obtiendrez alors 20 lignes et 7 colonnes à l'écran, au prix d'une baisse de la lisibilité.

2. Important presse-papier

Quand vous sortez de Multiplan après avoir copié plus de 50 cellules (par exemple pour les transférer à MacWrite), le message "Enregistrement d'un important presse-papier" s'affiche. S'il y a plus de 100 cellules à copier, il faut cliquer pour demander l'enregistrement des va-

leurs formatées et des valeurs non formatées (c'est automatique pour moins de 100 cellules). Dans un cas comme dans l'autre, validez l'enregistrement avec OK.

3. Tabulation numérique

Lorsque vous copiez plusieurs colonnes de valeurs de Multiplan vers MacWrite, vous pouvez réaligner les valeurs avec la tabulation numérique (le triangle pointé). Bien entendu, si vous possédez un Multiplan U.S. (point de décimalisation) avec un MacWrite français (virgule de décimalisation), le cadrage ne se fera pas avant que vous ayez changé les "." en ",".

Si la première colonne contient des valeurs numériques, celles-ci ne seront pas tabulées. Pour atteindre cet objectif, il conviendra d'ajouter préalablement une colonne blanche à la zone à copier vers MacWrite.

4. Autres trucs d'utilisation

De nombreux autres trucs d'utilisation vous sont proposés dans "Multiplan pour Macintosh", qui doit paraître aux Editions du PSI à la fin décembre. Cet ouvrage a été réalisé à partir de la version 1.02 de Multiplan en version française (août 1984).

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
!	@	#	\$	%	&	'	()	*	+	=	-		_	{	}	~	^	~	-	=	+		_	{
1	2	3	4	5	6	7	8	9	0	!	@	#	\$	%	^	&	*	()	~	-	=	+		_
l	}	\		;	:	'	"	,	.	<	>	/	?	Option A	Option i										
															from the font		CAIRO								

Police Cairo - Tableau provenant de Mac the Knife™

Les Routines en ROM du Macintosh

Jean-Luc Bazanegue

Le Basic Microsoft donne accès à 41 routines situées dans la mémoire à lecture seule du Macintosh. Malheureusement, le manuel d'utilisation de l'interpréteur indique seulement la syntaxe des instructions, ce qui est loin d'être suffisant pour une utilisation optimale. De plus, le manuel propose, pour obtenir de plus amples informations, une consultation du "QuickDraw Programmer's Guide", qui n'est pas accessible à l'utilisateur du Macintosh !

Devant cet état de fait, nous avons pensé devoir vous fournir la documentation qui aurait normalement dû être jointe au manuel d'utilisation du Basic Microsoft.

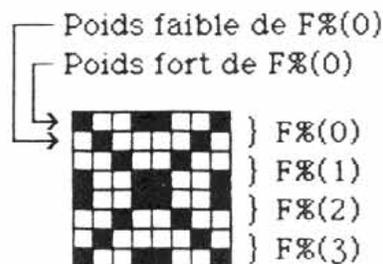
Avant d'entrer dans le vif du sujet, il convient de signaler que cet article n'est pas un "remake" du "QuickDraw Programmer's guide": nous n'avons jamais eu entre les mains ce document.

Call BackPat (VARPTR(F%(0)))

Cette instruction permet de définir une matrice de 8*8 points (ou pixels) qui détermine l'apparence du "fond de l'écran" (sur lequel sont superposés les caractères). Le principe est à peu près le même que celui utilisé dans le "tableau de bord", pour choisir l'aspect du bureau.

Les points "on" (noir) et "off" (blanc) sont déterminés par l'état des 64 bits contenus dans un tableau de 4 variables entières, que nous avons arbitrairement baptisé F% dans notre exemple. L'octet de poids fort de la variable d'indice 0 représente la première ligne de 8 points, l'octet de poids faible représente la seconde ligne de 8 points, l'octet de poids fort de la variable d'indice 1 représente la troisième ligne, etc... Notons que dans le cas d'une utilisation de l'instruction *OPTION BASE 1*, les indices doivent être compris entre 1 et 4 inclus, au lieu de 0 à 3.

Prenons un exemple concret: nous voulons que l'écran (ou plutôt la "fenêtre de sortie", pour employer la terminologie "Mac"!) soit rempli du motif suivant.



Pour obtenir ce motif, nous devons affecter à nos quatre variables les valeurs binaires:

```
10011001 01000010
00100100 10011001
10011001 00100100
01000010 10011001
```

soit, en hexadécimal:

```
&H9942
&H2499
&H9924
&H4299
```

Dans un programme, parmi les multiples solutions possibles, nous pourrions trouver:

```
10 DIM F%(3)
20 DATA &H9942,&H2499,&H9924,
    &H4299
30 FOR I%=0 TO 3
40 READ F%(I%)
50 NEXT
60 CALL BACKPAT(VARPTR(F%(0)))
70 CLS
```

L'instruction *CLS* est indispensable car, si elle est omise, l'écran demeure inchangé. Pour replacer l'écran dans son état initial, il suffit de relancer le programme après modification de la ligne 20:

```
20 DATA 0,0,0,0
```

Gestion du Curseur

Call SetCursor (VARPTR(C%(0)))

Si vous êtes las de voir le curseur en forme de flèche, vous pouvez créer le vôtre. Pour cela, il vous faudra d'abord définir un tableau de 34 variables entières (indices 0 à 33, ou 1 à 34 avec *OPTION BASE 1*) que nous appellerons C%.

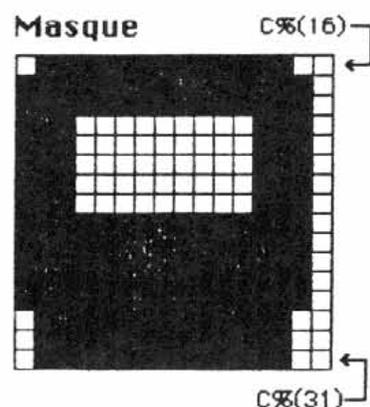
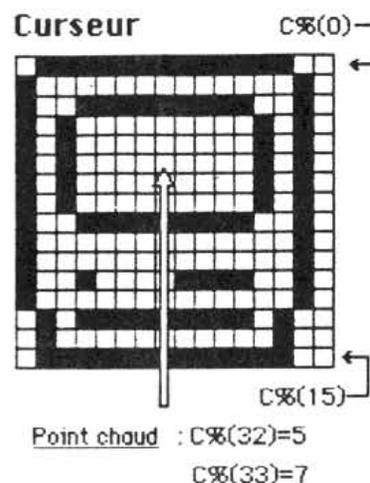
Les variables C%(0) à C%(15) sont utilisées pour définir la forme du curseur dans une grille de 16*16 points, chaque ligne horizontale de 16 points étant affectée à une variable.

Les variables C%(16) à C%(31) définissent le masque du curseur, qui permet une superposition à ce qui est affiché à l'écran. En effet, si un des points du curseur n'est pas masqué, l'opération logique *XOR* ("OU" exclusif, ou encore "l'un ou l'autre mais pas les deux") est effectuée entre le point concerné du curseur et celui occupant la même position sur l'écran.

La variable C%(32) indique la position verticale du point chaud par rapport au coin supérieur gauche de la grille 16*16. Le point chaud correspond aux valeurs retournées par *MOUSE(1-6)*.

La variable C%(33) indique la position horizontale du point chaud, toujours par rapport au coin supérieur gauche.

Pour illustrer le fonctionnement de cette routine, nous allons créer un curseur ayant la forme d'un "Mac"



```
100 DIM C%(33)
110 REM
120 REM - Data pour la définition du
    curseur
130 REM
140 DATA &H7FFC,&H8002,&H9FF2,
    &HA00A
150 DATA &HA00A, &HA00A,&HA00A,
    &HA00A
160 DATA &H9FF2,&H8002,&H8002,
    &H90F2
170 DATA &H8002,&H5FF4,&H4004,
    &H7FFC
180 REM
190 REM - Data pour la définition du
    masque
200 REM
210 DATA &H7FFC,&HFFFE,&HFFFE,&HE00E
220 DATA &HE00E,&HE00E,&HE00E,
    &HE00E
230 DATA &HFFFE, &HFFFE, &HFFFE,
    &HFFFE
```

```

240 DATA &HFFFE, &H7FFC, &H7FFC,
    &H7FFC
250 REM
260 REM - Data pour la position du point
    chaud
270 REM
280 DATA 5,7
290 REM
300 FOR I%=0 TO 33:READ C%(I%):NEXT
310 CALL SETCURSOR(VARPTR(C%(0)))

```

Note : la routine SetCursor n'est pas protégée contre les passages de paramètres invalides: il faut donc être très prudent, surtout en ce qui concerne la position du point chaud, si l'on ne veut pas "planter" le Macintosh.

Call InitCursor

Cette instruction, qui ne requiert pas d'argument, réinitialise le curseur en forme de flèche. Si le résultat de l'exemple précédent est toujours en mémoire, vous pouvez rétablir le curseur d'origine en entrant cette instruction en mode immédiat.

Call HideCursor

Après exécution, le curseur est invisible, bien que toujours présent.

Call ShowCursor

Visualise à nouveau le curseur

Call ObscureCursor

Le curseur devient invisible (comme avec Call HideCursor) mais réapparaît dès qu'un déplacement de la souris est détecté.

Gestion du "crayon"

Les routines décrites dans ce paragraphe conditionnent le résultat obtenu après exécution de toutes les instructions du Basic Microsoft propres à créer des graphismes (*LINE*, *CIRCLE*...), mais aussi après exécution des routines graphiques en ROM (Call FrameRect, Call FillRect, etc...).

Call PenSize (largeur, hauteur)

Détermine les dimensions du crayon (ou traceur). Les valeurs par défaut sont égales à 1 pour la largeur et la hauteur.

Lorsque l'on trace une ligne, la largeur du trait est reportée vers la droite par rapport aux coordonnées horizontales, alors que la hauteur est reportée vers le bas par rapport aux coordonnées verticales. En revanche, pour les figures fermées (un rectangle, par exemple) ou un arc de cercle, les dimensions sont respectées et l'épaisseur du trait est reportée à l'intérieur de la figure.

Call PenPat (VARPTR(F%(0)))

Cette instruction fonctionne exactement de la même façon que Call

BackPat: une nouvelle explication serait donc superflue.

Call Penmode (mode)

Call Penmode détermine le comportement des instructions ou routines graphiques, par rapport à ce qui est déjà affiché à l'écran.

mode = 8

Il s'agit du mode normal: les points constituant la ligne ou la figure, qu'ils soient "on" ou "off", masquent ce qui se trouve sur l'écran.

mode = 9

L'opération logique *OR* (OU) est réalisée entre la ligne (ou la figure) et le fond. Autrement dit, les points "on" du fond et de la ligne sont affichés. Dans le petit tableau placé ci-dessous, la première colonne de chiffres représente les points de la ligne à afficher (1 = point "on", 0 = point "off"), la seconde colonne représente les points déjà présents à l'écran et la troisième indique le résultat de l'opération.

```

1 OR 1 = 1
1 OR 0 = 1
0 OR 1 = 1
0 OR 0 = 0

```

mode = 10

Ce mode autorise une opération logique de type *XOR* (OU exclusif). Les points "on" du fond et de la ligne sont affichés, sauf si un point "on" de la ligne correspond à un point "on" du fond. Dans ce cas, le point résultant est "off" (blanc).

```

1 XOR 1 = 0
1 XOR 0 = 1
0 XOR 1 = 1
0 XOR 0 = 0

```

mode = 11

Opération logique *AND* (ET). Les points résultant de l'opération sont "on" uniquement si les points correspondants de la ligne et du fond sont "on".

```

1 AND 1 = 1
1 AND 0 = 0
0 AND 1 = 0
0 AND 0 = 0

```

mode = 12

Tous les points constituant la ligne sont d'abord inversés (*NOT*), puis sont superposés aux points définissant le fond.

```

NOT 1 = 0
NOT 0 = 1

```

mode = 13

Double opération logique *NOT* et *OR*. Les points constituant la ligne sont d'abord inversés (comme avec 12), et sont ensuite traités comme avec "mode = 9".

```

NOT 1 OR 1 = 1

```

```

NOT 0 OR 1 = 1
NOT 1 OR 0 = 0
NOT 0 OR 0 = 1

```

mode = 14

La méthode employée est la même que pour l'argument précédent, mise à part l'opération logique *OR*, ici remplacée par *XOR*.

```

NOT 1 XOR 1 = 1
NOT 0 XOR 1 = 0
NOT 1 XOR 0 = 0
NOT 0 XOR 0 = 1

```

mode = 15

Le second opérateur logique est cette fois un *AND*.

```

NOT 1 AND 1 = 0
NOT 0 AND 1 = 1
NOT 1 AND 0 = 0
NOT 0 AND 0 = 0

```

Afin de mettre en évidence une des possibilités de cette routine, nous vous proposons un petit exemple qui affiche un texte estompé, à la manière des menus du Mac.

```

10 CLS
20 DIM F%(3)
30 FOR I%=0 TO 3
40 F%(I%)=&H55AA
50 NEXT
60 CALL PENPAT(VARPTR(F%(0)))
70 CALL TEXTFONT(0)
80 CALL PENMODE(11)
90 CALL PENSIZE(1,12)
100 PRINT "Caractères estompés"
110 LINE(2,2)-(140,2)
120 GOTO 120

```

Call GetPen (VARPTR(P%(0)))

Cette routine retourne la position courante du crayon dans P%(0) (coordonnée verticale) et P%(1) (coordonnée horizontale). Call Getpen permet aussi de savoir où sera affiché (avec PRINT ou WRITE) le prochain texte ou nombre. Avec *OPTION BASE 1*, il faut utiliser les indices 1 et 2 au lieu de 0 et 1.

Call HidePen

Inhibe le crayon. Tout se passe comme si les instructions graphiques étaient ignorées.

Call ShowPen

Redonne son efficacité au crayon.

Call PenNormal

Le crayon reprend les caractéristiques qui lui sont données au moment de l'initialisation (valeurs par défaut: Call PenSize(1,1), Call PenMode(8), etc...).

Call MoveTo (X,Y)

Déplace le crayon, sans tracer de ligne, à la position horizontale X et à la position verticale Y.

Call Move (XR, YR)

Déplacement relatif de XR points par rapport à la position horizontale actuelle du crayon et de YR points par rapport à la position verticale. Ainsi, si la position courante est : X = 100 et Y = 100, l'instruction Call Move(10, -10) reporte le crayon à l'emplacement : X = 110, Y = 90.

Les instructions Call MoveTo et Call Move peuvent aussi être utilisées pour déterminer la position des prochains caractères affichés par PRINT ou WRITE.

Call LineTo (X, Y)

Trace une ligne entre la position courante du crayon et les coordonnées X (horizontale) et Y (verticale).

Call Line (XR, YR)

Trace une ligne entre la position courante du crayon et les coordonnées relatives XR et YR (voir Call Move).

Gestion du texte

Call TextFont (police)

Autorise le choix de la police de caractères. Les indications fournies ci-dessous sont valables avec un "dossier système" standard, les arguments pouvant donner des résultats différents avec les versions futures ou avec un dossier "bidouillé".

- 0 : Chicago
- 1 : New York
- 3 : Geneve
- 4 : Monaco
- 5 : Venice
- 6 : London
- 7 : Athens
- 8 : San Francisco
- 9 : Toronto

Call TextFace (aspect)

Détermine l'aspect du texte. La routine utilise les 7 bits de poids faible de l'argument de la manière suivante :

- bit 0 (1) = gras;
- bit 1 (2) = italique;
- bit 2 (4) = souligné;
- bit 3 (8) = relief;
- bit 4 (16) = ombre;
- bit 5 (32) = espacement normal des caractères;
- bit 6 (64) = espace plus important entre les caractères.

Ainsi, l'argument 7 (1 + 2 + 4) correspond à : gras + souligné + italique. 0 affiche des caractères standards.

Call textSize (taille)

La hauteur des caractères affichés est, par défaut, de 12 points. Cette instruction affecte aux caractères une hauteur en fonction de l'argument.

- Une valeur négative inhibe l'affichage.
- 0 redonne aux caractères leur taille par défaut.
- 1 "plante" le Mac !
- 2 et plus indiquent la hauteur des caractères.

Il est possible de donner aux caractères une hauteur de 3 ou 300 points, mais la lisibilité n'est pas très bonne (elle est même franchement mauvaise !). De manière générale, il vaut mieux s'en tenir aux tailles recommandées par MacPaint et MacWrite.

Call TextMode (mode)

Le but de cette instruction est sensiblement le même que celui de Call PenMode pour le crayon, avec des possibilités moins étendues.

mode = 0 : les caractères sont superposés à ce qui se trouve sur l'écran (identique à Call PenMode(8)).

mode = 1 : opération logique OR (identique à Call PenMode(9)).

mode = 2 : opération logique XOR (identique à Call PenMode(10)).

mode = 3 : les caractères sont affichés en blanc, et ne sont donc parfaitement lisibles que sur un fond noir.

Rectangles

Call FrameRect (VARPTR(R%(0)))

Trace le contour d'un rectangle dont les coordonnées sont situées dans les quatre variables entières du tableau R% :

- R%(0) = côté supérieur;
- R%(1) = côté gauche;
- R%(2) = côté inférieur;
- R%(3) = côté droit.

L'aspect du tracé est conditionné par les instructions Call PenMode, Call PenSize et Call PenPat. Si, pour ces instructions, nous conservons les valeurs par défaut, Call FrameRect donne le même résultat que LINE(X1, Y1)-(X2, Y2), B.

Call PaintRect (VARPTR(R%(0)))

Remplit un rectangle dont les coordonnées se trouvent dans le tableau R%, avec le motif initialisé par l'instruction Call PenPat.

Call EraseRect (VARPTR(R%(0)))

Efface la partie de l'écran comprise dans un rectangle dont les coordonnées se trouvent dans le tableau R%.

Call InvertRect (VARPTR(R%(0)))

Inverse l'état des points situés à l'intérieur d'un rectangle (NOT).

Call FillRect (VARPTR(R%(0)), VARPTR(F%(0)))

Cette instruction combine Call Pain-

tRect et Call PenPat. Ceci permet de remplir un rectangle sans modifier le motif affecté au crayon. Les coordonnées du rectangle sont dans le tableau R% et le motif se trouve dans le tableau F%.

Rectangles aux angles arrondis

Call FrameRoundRect (VARPTR(R%(0)), AX, AY)

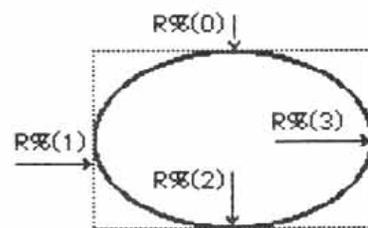
Trace le contour d'un rectangle aux angles arrondis. Les coordonnées du rectangle sont dans le tableau R%. Les caractéristiques de l'arrondi sont déterminées par AX (axe horizontal d'une ellipse) et AY (axe vertical). Si AX et AY sont égaux, l'arrondi est un arc de cercle.

Les instructions PaintRoundRect, EraseRoundRect, InvertRoundRect et FillRoundRect fonctionnent de la même façon que PaintRect, EraseRect, etc... avec en plus les valeurs AX et AY pour les arrondis.

Ellipses

Call FrameOval (VARPTR(R%(0)))

Trace le contour d'une ellipse inscrite dans un rectangle fictif dont les coordonnées se trouvent dans le tableau R%.



Si R%(2) - R%(0) est égal à R%(3) - R%(1), l'instruction trace le contour d'un cercle.

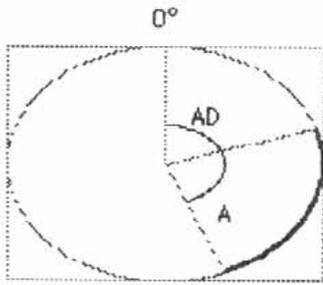
Les instructions PaintOval, EraseOval, InvertOval et FillOval sont comparables à PaintRect, EraseRect, etc... mise à part leur action sur des ellipses au lieu de rectangles.

Arcs de cercles

Call FrameArc (VARPTR(R%(0)), AD, A)

Trace l'arc d'un cercle (en fait, il s'agit plutôt d'un "arc d'ellipse") inscrit dans un carré fictif (ou un rectangle) dont les coordonnées se trouvent dans le tableau R%.

AD correspond à l'angle de départ et A à l'angle de l'arc par rapport à AD, les valeurs étant données en degrés. Les arguments supérieurs à 360 sont ignorés.



PaintArc, EraseArc, InvertArc et FillArc fonctionnent comme PaintRect, etc... tout en utilisant le "système angulaire" de FrameArc. ■

Au sujet de l'article "Les routines en ROM..."

Comme nous le signalons dans le

paragraphe "Gestion du texte", la routine "Call TextFont" peut donner des résultats différents selon la version du dossier système. Les choses allant très vite, nous sommes déjà amené à vous indiquer les modifications à apporter pour le nouveau dossier système (Finder version 1.1) utilisé pour la disquette Pom's.

- 0 = Chicago
- 1 = Geneve
- 2 = New York
- 4 = Monaco
- 5 = Venice
- 6 = London
- 7 = Athens

- 8 = San Francisco
- 9 = Toronto
- 11 = Cairo
- 12 = Los Angeles

Pour la même raison, le programme "Editeur de formes et curseurs" enregistré sur la disquette a subi une petite modification, par rapport à la version listée, afin qu'il soit compatible avec le nouveau système. Pour nos lecteurs qui utilisent la version 1.1 du Finder et ne désirent pas acquérir la disquette, signalons qu'il suffit de remplacer le "Call TextFont(1)" de la ligne 170 par un "Call TextFont(2)".

Caractères programmables sur imprimantes Apple

Apple Seedrin

Sur les imprimantes série Image Writer ou parallèle DMP, la définition de caractères programmables pose des problèmes (caractères soulignés) liés à la façon dont l'Applesoft envoie les codes à l'imprimante. En effet, l'Applesoft positionne toujours le bit de poids fort des octets à 1, ce qui interdit d'envoyer des caractères de contrôle inférieurs à 128.

Il est donc nécessaire de passer par

le langage machine pour envoyer tous les caractères voulus. Le code concerné sera POKé à une adresse donnée par le programme Basic, puis repris par une petite routine du type :

```
300-AD0703 LDA $307
303 20EDFD JSR $FDED
306-60 RTS
```

Dans cet exemple, le code aura été préalablement POKé à l'adresse \$307 (775 en décimal), et l'appel de

la routine se fera ensuite par un CALL 768.

Vous trouverez ci-après un petit programme illustrant cette méthode.

De plus, certaines interfaces interceptent le code "CTRL-I", soit CHR\$(9), comme code de commande, ce qui empêche de l'utiliser dans la définition d'un caractère programmable. Dans ce cas, il est donc nécessaire de neutraliser l'interface (par un PRINT CHR\$(9)"Z" sur la carte Super Série, par exemple) afin de recouvrer l'usage du code 9. On prendra soin alors d'envoyer les instructions de neutralisation en tout début de programme et de ne jamais repasser par elles, pour éviter de les voir s'imprimer de façon intempestive. ■

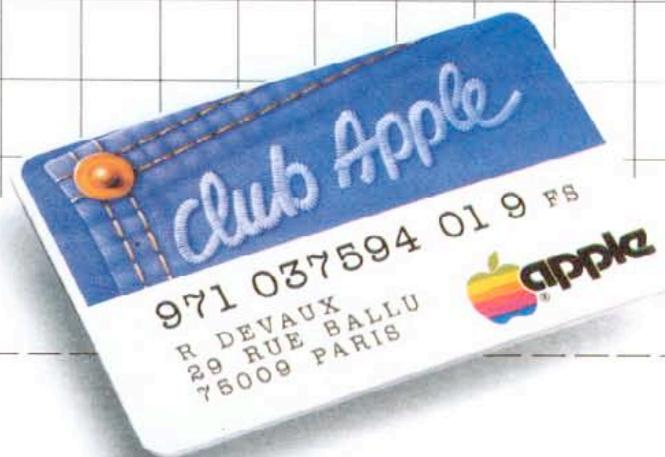
Exemple d'utilisation

```
10 REM CHARGEMENT DE LA ROUTINE EN DEC
   768
20 DATA 173,7,3,32,237,253,96
30 FOR I = 1 TO 7: READ A: POKE 767 + I,
   A: NEXT
40 REM CONNECTE L'IMPRIMANTE
50 PRINT CHR$(4);"PR#1"
60 REM
90 REM CREATION DU CARACTERE
120 REM
130 REM SELECTIONNE LA LARGEUR MAXIMALE
   16 POINTS
140 REM
150 PRINT CHR$(27); CHR$(43)
160 REM
170 REM DEBUT DU CHARGEMENT DU CARACTERE
   E
180 REM
190 PRINT CHR$(27); CHR$(73);
200 REM
230 REM SEQUENCE A EXECUTER POUR CHAQUE
   CARACTERE
260 REM
270 REM DEFINITION D'UN CARACTERE D
   E CODE ASCII DEC 49
280 REM
290 REM Envoi à l'imprimante du code ac
   siii
300 POKE 775,49: CALL 768
```

```
310 REM
320 REM Sp(cification de la largeur du
   caract)re
330 REM
340 PRINT CHR$(80);
350 REM
360 REM ENVOI DU CODAGE DU CARACTERE
370 REM
380 DATA 8,20,34,65,65,65,65,34,34,20,8
   ,20,34,65,65,65
390 FOR I = 1 TO 16: READ A: POKE 775,A:
   CALL 768: NEXT
400 REM
410 REM FIN DU CODAGE
420 REM
430 PRINT CHR$(4)
440 REM
450 REM SELECTION DU JEU DE CAR
   ACTERES PROGRAMMABLES
460 REM
470 PRINT CHR$(27); CHR$(39)
480 REM
490 REM IMPRESSION DU CARACTERE
500 REM
510 PRINT "1 1 1 1 1 1 1 1 1 1 1
   1 1 1 1 1 1 1 1 1 1"
520 REM
530 REM RETOUR AU JEU DE CARACTERES NOR
   MAL
540 REM
550 PRINT CHR$(27); CHR$(36)
560 PRINT CHR$(4);"PR#0"
```

Souris créative cherche amitié passionnée pour fonder club.

Pour tous les passionnés de l'ordinateur personnel,
les branchés et ceux qui le sont moins, Apple lance le Club Apple.
Enfin un club avec des idées et des services
pour comprendre, pour gagner, pour s'évader.
Le Club Apple, c'est l'esprit Apple.
C'est le fruit de la passion.



PROMO 2000

Pour en savoir plus sur le Club Apple - le fruit de la passion - et connaître tous les avantages que nous réservons aux membres du club, découpez dès aujourd'hui le bon et retournez-le à Club Apple, avenue de l'Océanie - ZA de Courtabœuf - BP 131, Les Ulis cedex 91944. Vous recevrez sans engagement de votre part toutes les informations pour devenir membre du club.

Nom _____ Prénom _____

Adresse _____

Code postal _____

Il était temps qu'un capitaliste

MAO
TSE-TUNG

II

ENGELS

LENIN

KARL
MARX

DAS KAPITAL

TROTSKI

fasse une révolution.



N'apprenez plus à devenir une machine, Apple a inventé Macintosh.



PEACHTREE, LA COMP QUE C'EN EST PRESQU



La comptabilité, c'est long, compliqué, fastidieux !
Tout cela était vrai avant Mac Accounting. Avec Mac Accounting, le nouveau logiciel de comptabilité de Peachtree, finis les livres de caisse en deux exemplaires, finis les bons de caisse, finis les carbonnes, terminées les mises à jour qui font perdre du temps. Mac Accounting est un programme comptable développé spécialement par Peachtree pour le Mac Intosh. Et dans le domaine particulier de la comptabilité, la compétence de Peachtree n'est plus à prouver. Avec plus de 100.000 utilisateurs dans le monde entier, Peachtree possède une très large expérience des besoins des petites et moyennes entreprises en matière de comptabilité. Spécialement étudié pour un utilisateur n'ayant aucune connaissance infor-

matique préalable, Mac Accounting est conçu pour être le plus proche des systèmes manuels... avec l'exactitude et la rapidité des solutions informatiques. Un comptable reconnaît immédiatement les pages d'un journal de caisse ou du Grand Livre : c'est exactement ce qu'il retrouvera sur l'écran du Mac Intosh lorsqu'il utilisera Mac Accounting.

Mac Cash est le premier des deux programmes qui constituent Mac Accounting. C'est un livre de caisse sur écran. Idéal pour les PME, ses caractéristiques parlent d'elles-mêmes. La taille du journal, le titre des colonnes, la période comptable et les formats monétaires sont définis par l'utilisateur. Mac Cash accepte jusqu'à cinq taux de taxe, et huit colonnes d'analyse des prix hors-taxes, sa mise à jour est automatique au fur et à mesure de la

TA DEVIENT SI SIMPLE E UN PÈCHÈ.

alliance



saisie des opérations, enfin l'utilisateur spécifie lui-même les modèles des états de sortie et leur contenu. Bref, un outil simple et performant.

Mac Ledger est le second module de Mac Accounting, c'est la partie comptabilité générale. Le format du journal est standard et peut être utilisé aussi bien pour les comptes fournisseurs que pour les comptes clients. L'utilisateur définit lui-même le format des fiches comptables qui se sélectionnent facilement à l'aide de la souris Mac Intosh. Les opérations saisies peuvent être analysées par type de produit, par territoire, par vendeur ou par toute autre variable définie par l'utilisateur...

Les créances sont analysées sur trois périodes, ce qui facilite contrôle et réajustement. En résumé le complément indispensable de Mac Cash.

Dernier avantage : Mac Accounting a été conçu en français et peut être utilisé par n'importe quel utilisateur en moins de trente minutes. Le doute n'est plus permis : avec Mac Intosh et Mac Accounting la comptabilité devient si simple que c'est presque un péché.

Peachtree est distribué par Logiciel PC : 113, bd Pèreire
75017 Paris. Tél. : 763.62.88.



Peachtree

simplement génial

Editeur de formes et curseurs

Pom's

Le Basic Microsoft du Macintosh autorise l'emploi d'instructions graphiques très puissantes. C'est le cas de PUT, qui peut être utilisée de deux manières différentes : transfert d'une matrice de points stockée par GET dans un tableau de variables; transfert d'une matrice de points créée par le programmeur. La seconde solution est sans doute celle qui sera le plus souvent retenue, puisqu'elle permet de concevoir des formes très précises, point par point. Etant donnée la haute résolution du Macintosh, la conception d'une forme à l'aide d'une feuille de papier quadrillée demande beaucoup de patience, ce qui nous a amené à écrire cet utilitaire.

Les formes ou curseurs créés avec l'éditeur sont, bien sûr, destinés à être utilisés par vos propres programmes. Deux cas se présentent :

1) Vous désirez que votre programme contienne la définition des formes ou curseurs sous forme de DATA. Dans ce cas, choisissez l'option "Imprimer", qui vous fournit une copie des fenêtres significatives, ainsi que tous les codes nécessaires à la constitution de la liste de DATA (en décimal et hexadécimal). Le sous-programme de copie d'écran est assez lent, puisqu'il écrit en Basic, mais cette solution nous paraît plus élégante que l'emploi du classique LCOPY. De plus, ce sous-programme pourra être ultérieurement remplacé par son équivalent en langage machine;

2) Vous désirez que le curseur ou la forme soit stockée sur disquette afin de charger sa définition directement depuis votre programme. Pour cela, il faut employer l'option "Enregistrer". Cette dernière solution est à notre avis la meilleure puisqu'il n'est pas nécessaire d'écrire les valeurs sous formes de DATA, ce qui permet un gain de temps non négligeable et économise de la place en mémoire (très important avec un Mac 128Ko !).

Le chargement de la définition d'un curseur à partir de votre programme ne pose pas de problème, puisque le nombre d'octets requis est invariable (voir l'article "Les routines en ROM"). Par exemple, pour charger un curseur baptisé préalablement "curseur" (qui a dit que nous manquons d'imagination ?), on peut utiliser un segment de programme qui ressemblerait à :

10' Chargement et modification du curseur standard

```
20 DIM C%(33)
30 OPEN "i",1,"curseur"
40 FOR I%=0 TO 33
50 INPUT #1,C%(I%)
60 NEXT
70 CLOSE
80 CALL SETCURSOR
(VARPTR(C%(0)))
90 GOTO 90
```

La taille des formes créées avec l'éditeur peut varier de 1*1 à 64*54 points; il faut donc connaître le nombre d'octets nécessaires à leur représentation. On peut utiliser la formule "4 + NY * 2 * INT((NX + 15) / 16)" mais cela oblige à faire le calcul et à noter le résultat sur un papier que l'on ne retrouvera pas forcément. Une autre solution consiste à quitter le programme (retour au bureau) juste après avoir enregistré une forme car, lors de l'enregistrement, le nombre d'octets requis est transféré dans le presse-papier et peut être collé dans la "fenêtre d'information" de la forme. Lorsque l'on utilise des tableaux de variables entières, le nombre indiqué doit être divisé par deux, pour charger une forme de 64 par 54 points (436 octets) baptisée "forme" (toujours autant d'imagination !), on peut utiliser :

10' Chargement et affichage d'une forme 64*54

```
20 OPTION BASE 1
30 DIM F%(218)
40 OPEN "i",1,"forme"
50 FOR I%=1 TO 218
60 INPUT #1,F%(I%)
70 NEXT
80 CLOSE
90 PUT (10,10),F%
100 GOTO 100
```

Nous avons vu précédemment que la taille maximale autorisée par l'éditeur est de 64*54 points, mais il est parfaitement possible de regrouper plusieurs formes pour en constituer une plus grande. Les trois petits programmes suivants réalisent le regroupement de quatre formes baptisées "forme1" à "forme4", et ceci dans différentes configurations :

10' Transformation de 4 formes de 64*54 points en une forme de 64 points

(horizontal) par 216 points (vertical)

```
20 OPTION BASE 1
30 DIM F%(4,218)
40 FOR I%=1 TO 4
50 OPEN "i",1,"forme"+
RIGHT$(STR$(I%),1)
60 FOR J%=1 TO 218
70 INPUT #1,F1%(I%,J%)
80 NEXT
90 CLOSE
100 NEXT
110 OPEN "o",1,"F"
120 PRINT #1,64,216
130 FOR I%=1 TO 4
140 FOR J%=3 TO 218
150 PRINT #1,F1%(I%,J%)
160 NEXT
170 NEXT
180 CLOSE
```

10' Transformation de quatre formes 64*54 points en une forme de 256 points (horizontal) par 54 points (vertical)

```
20 OPTION BASE 1
30 DIM F%(4,218)
40 FOR I%=1 TO 4
50 OPEN "i",1,"FORME"+
RIGHT$(STR$(I%),1)
60 FOR J%=1 TO 218
70 INPUT #1,F1%(I%,J%)
80 NEXT
90 CLOSE
100 NEXT
110 OPEN "o",1,"F"
120 PRINT #1,256,54
130 FOR J%=0 TO 53
140 FOR I%=1 TO 4
150 FOR K%=3 TO 6
160 PRINT #1,F1%(I%,J%+4*K%)
170 NEXT
180 NEXT
190 NEXT
200 CLOSE
```

10' Transformation de 4 formes de 64*54 points en une forme de 128 points (horizontal) par 108 points (vertical)

```
20 OPTION BASE 1
30 DIM F%(4,218)
40 FOR I%=1 TO 4
```



```

16382,16382,32767,32767,28679,28679,28679,28679,
28679,32767,32767,32767,32767,16382,16382,
16382,5,8
70 DATA -32768,16384,8192,4096,2048,1024,512,256,128,
64,32,16,8,4,2,1
80 DATA La position du point chaud n'étant pas,
"étalée, les valeurs assumées sont :",X = 0,Y = 0
90 DATA 0 Effacer,1 Ajouter ligne,2 Supprimer ligne,
3 Ajouter colonne,4 Supprimer colonne,5 Imprimer,
6 Enregistrer,7 Grille,8 Quitter,9 Décaler
100 DATA 8184,16380,32766,-1,-16381,-10021,-14253,
-9509,-12213,-9509,-16381,-1,32766,16380,8184,0,
8184,0,0,32766,32766,32766,32766,32766,32766,
32766,32766,32766,0,0,8184,0,7,7
110 OPEN "I",1,"XL",6:FOR I=1 TO 312:INPUT*1,I(I):NEXT:CLOSE:
OPEN "I",1,"XP",6:FOR I=1 TO 218:INPUT*1,Q(I):NEXT:CLOSE:
RESTORE 70:FOR I=1 TO 16:READ H(I):NEXT:FOR I=3 TO 9:
F(I)=-1:NEXT:WV="Voulez-vous "H=-16:WIDTH "Ipt 1":255
120 CALL TEXTFONT(0):CALL MOVETO(2,12):PRINT "@ Pom's et
JLB".GET(3,3)-(99,12),M:LINE(3,3)-(99,12),30,BF:J=0:
BI=501874:AI=VARPTR(M(1))+4:FOR I=1 TO 10:
FOR J=0 TO 13:POKE BI+J,PEEK(AI):AI=AI+1:NEXT:BI=BI+64:
NEXT
130 GOSUB 2170:RESTORE:FOR I=1 TO 4:P(I)=0:NEXT:
CALL BACKPAT(VARPTR(P(1))):CLS:
LINE(44,30)-(444,86),,B:LINE(148,102)-(340,246),,B:
LINE(150,104)-(338,122),,B
140 FOR I=128 TO 224 STEP 24:LINE(320,I)-(335,I+15),,B:
LINE(335,I)-(320,I+15):LINE(335,I+15)-(320,I):
LINE(322,I+2)-(333,I+13),,B:
LINE(323,I+3)-(332,I+12),30,BF:NEXT:
LINE(150,247)-(340,247)
150 LINE(46,87)-(446,87):CALL PENSIZE(2,2):
LINE(49,35)-(439,81),,B:LINE(341,103)-(341,246):
LINE(445,31)-(445,86)
160 GOSUB 1790:CALL PENPAT(VARPTR(P(1))):
CALL PENSIZE(4,4):P(1)=31:P(2)=45:P(3)=86:P(4)=444:
CALL FRAMERECT(VARPTR(P(1))):CALL PENNORMAL
170 CALL TEXTFONT(1):CALL TEXTSIZE(24):CALL TEXTFACE(1):
CALL MOVETO(64,66):PRINT "Pom's":CALL TEXTMODE(1):
CALL TEXTSIZE(14):CALL TEXTFACE(0):CALL TEXTFONT(5):
CALL MOVETO(192,62):PRINT "Editeur de formes et
 curseurs"
180 CALL TEXTSIZE(12):CALL TEXTFONT(0):FOR I=118 TO 238
STEP 24:CALL MOVETO(152,I):READ W:PRINT W:NEXT
190 R(1)=40:R(2)=54:R(3)=77:R(4)=149:
CALL INVERTROUNDRECT(VARPTR(R(1)),18,18):R(1)=106:
R(2)=152:R(3)=121:R(4)=337
200 FOR I=1 TO 34:READ D(I):NEXT:
CALL SETCURSOR(VARPTR(D(1))):CALL SHOWCURSOR:
PUT(420,226),Q
210 W=INKEY$:IF W<>" " THEN 230 ELSE IF MOUSE(0)<0 THEN
240 ELSE I=I+1:IF I=50 THEN
CALL INVERTRECT(VARPTR(R(1))):I=0
220 GOTO 210
230 N=ASC(W):IF N<49 OR N>53 THEN 210 ELSE N=N-48:
GOTO 260
240 IF MOUSE(1)<321 OR MOUSE(1)>334 THEN 210 ELSE
A=MOUSE(2)
250 IF A>128 AND A<143 THEN N=1 ELSE IF A>152 AND A<167
THEN N=2 ELSE IF A>176 AND A<191 THEN N=3 ELSE IF
A>200 AND A<215 THEN N=4 ELSE IF A>224 AND A<239
THEN N=5 ELSE 210
260 P(1)=105+24*N:P(2)=321:P(3)=P(1)+14:P(4)=335:

```

```

FOR I=0 TO 21:CALL INVERTRECT(VARPTR(P(1))):
FOR D=0 TO 50:NEXT:NEXT:GOSUB 1790:
CALL BACKPAT(VARPTR(P(1))):CLS:
ON N GOTO 290,990,280,970
270 CALL HIDECURSOR:WIDTH "Ipt 1":96:GOSUB 2150:
GOSUB 1860:CALL TEXTFONT(6):CALL TEXTSIZE(18):
CALL MOVETO(132,145):PRINT "Retour au bureau...":
SYSTEM
280 E=0:GOTO 300
290 E=1
300 DIM A(18),B(18):LINE(338,26)-(461,197),30,BF:
LINE(130,210)-(204,262),30,BF
310 FOR I=26 TO 178 STEP 152:LINE(I,26)-(I+132,190),30,BF:
LINE(I,26)-(I+132,190),,B:LINE(I+2,28)-(I+130,52),,B:
LINE(I+2,54)-(I+130,58),,BF:LINE(I+2,191)-(I+134,191):
CALL PENSIZE(2,2):LINE(I+133,27)-(I+133,190):
CALL PENNORMAL:NEXT
320 F=1:GOSUB 1810:CALL MOVETO(68,44):PRINT "Curseur":
CALL MOVETO(220,44):PRINT "Masque"
330 LINE(340,28)-(460,196),,B:LINE(338,26)-(462,198),,B:
FOR I=52 TO 196 STEP 24:LINE(340,I)-(460,I):
CALL MOVETO(348,I-8):READ W:PRINT W:NEXT
340 LINE(340,199)-(464,199):LINE(132,212)-(202,260),,B:
LINE(130,210)-(204,262),,B:LINE(132,263)-(206,263):
CALL PENSIZE(2,2):LINE(205,211)-(205,262):
LINE(463,27)-(463,198):CALL PENNORMAL
350 F(1)=7:F(2)=7:U=0:V=0:G=1:B=0:A(1)=16:A(2)=16:B(1)=16:
B(2)=16:CALL SETCURSOR(VARPTR(C(1))):IF E THEN 410
360 GOSUB 2150:GOSUB 1850
370 Z=""
380 GOSUB 1950:IF A THEN N=1:GOTO 940 ELSE OPEN "I",1,Z,6
390 FOR I=3 TO 18:INPUT*1,A(I):NEXT:FOR I=3 TO 18:
INPUT*1,B(I):NEXT:INPUT*1,V,U:CLOSE:
IF V AND H OR U AND H THEN ERROR 57
400 PUT(108,92),M,PSET:PUT(148,228),A:PUT(172,228),B:
PUT(28,60)-(155,187),A,OR:PUT(180,60)-(307,187),B,OR:
B=1
410 W=INKEY$:IF W<>" " THEN 550 ELSE IF MOUSE(0)<>-1 THEN
410
420 P=MOUSE(1):Q=MOUSE(2):IF P>340 AND P<460 AND Q>28
AND Q<196 THEN N=(Q-4)\24:GOTO 560
430 I=POINT(P,Q) AND 1
440 IF Q<61 OR Q>187 THEN 410
450 IF P>28 AND P<156 THEN 510
460 IF P<181 OR P>307 THEN 410
470 WHILE MOUSE(0)<0:R=(MOUSE(1)-172)\8:
S=(MOUSE(2)-52)\8:IF R=1 AND H OR S=1 AND H THEN 500
480 J=R*8+173:K=S*8+53:S=S+2:IF I THEN PUT(J,K),F,PSET:
B(S)=B(S) AND NOT H(R) ELSE PUT(J,K),F,PSET:
B(S)=B(S) OR H(R)
490 PUT(172,228),B,PSET
500 WEND:GOTO 410
510 WHILE MOUSE(0)<0:R=(MOUSE(1)-20)\8:
S=(MOUSE(2)-52)\8:IF R=1 AND H OR S=1 AND H THEN 540
520 J=R*8+21:K=S*8+53:S=S+2:IF I THEN PUT(J,K),F,PSET:
A(S)=A(S) AND NOT H(R) ELSE PUT(J,K),F,PSET:
A(S)=A(S) OR H(R)
530 PUT(148,228),A,PSET
540 WEND:GOTO 410
550 N=ASC(W)-48:IF N<1 OR N>7 THEN 410
560 R(1)=5+24*N:R(2)=341:R(3)=R(1)+23:R(4)=460:
FOR I=0 TO 21:CALL INVERTRECT(VARPTR(R(1))):
FOR D=0 TO 50:NEXT:NEXT:
ON N GOTO 570,630,700,790,840,880,910

```

```

570 RESTORE 60:FOR I=1 TO 34:READ D(I):NEXT R(1)=30:
R(2)=30:R(3)=51:R(4)=155:
CALL INVERTRECT(VARPTR(R(1))):
CALL SETCURSOR(VARPTR(D(1)))
580 IF MOUSE(0)=-1 THEN P=MOUSE(1):Q=MOUSE(2):
U=(P-28)\8:V=(Q-60)\8:
IF (U AND H OR V AND H)=0 THEN 600
590 GOTO 580
600 P(1)=60+8*V:P(2)=28+8*U:P(3)=P(1)+8:P(4)=P(2)+8:
FOR I=0 TO 21:CALL INVERTRECT(VARPTR(P(1))):
FOR D=0 TO 50:NEXT:
CALL INVERTRECT(VARPTR(R(1))):
IF G THEN C(33)=6:C(34)=6 ELSE C(33)=V:C(34)=U
610 B=1:CALL SETCURSOR(VARPTR(C(1)))GOTO 410
620 Enregistrer
630 IF B THEN 650 ELSE GOSUB 2150:GOSUB 1850:
GOSUB 1920:RESTORE 80:FOR I=116 TO 164 STEP 16:
CALL MOVETO(124,I):READ W:PRINT W:NEXT:GOSUB 2070
640 IF D THEN 670
650 GOSUB 2150:IF B THEN GOSUB 1850 ELSE GOSUB 1860
660 GOSUB 1950:IF A THEN 670 ELSE OPEN "O",1,2,6:
FOR I=3 TO 18:PRINT #1,A(I):NEXT:FOR I=3 TO 18:
PRINT #1,B(I):NEXT:PRINT #1,V,U:CLOSE
670 PUT(108,92),M,PSET
680 IF MOUSE(0)<0 THEN 680 ELSE 410
690 Imprimer
700 GOSUB 2150:GOSUB 1850:CALL MOVETO(124,124):
PRINT WV "imprimer ?":GOSUB 1920:GOSUB 2070
710 IF D THEN 670 ELSE PUT(108,92),M,PSET
720 CALL HIDECURSOR:P=270:Q=224:R=470:S=272:
GOSUB 1850:CALL MOVETO(290,251):
PRINT "Impression...":P=26:Q=P:R=158:S=190:
W1="F0001":LPRINT CHR$(24):GOSUB 1880
730 LPRINT CHR$(24) CHR$(27) "r" CHR$(31) "?" CHR$(31) "r"
CHR$(27) "f":P=178:Q=26:R=310:S=190:W1="F0280":
GOSUB 1880
740 LPRINT CHR$(24) CHR$(31) "3":W1="F0001":P=130:Q=210:
R=204:S=262:GOSUB 1880
750 LPRINT CHR$(27) "r" CHR$(31) "?" CHR$(31) "?" CHR$(31)
"5" CHR$(24) CHR$(27) "c"
760 W="000":FOR I=3 TO 18:LPRINT TAB(26);USING"*****";
A(I);Y=HEX$(A(I)):X="$"+LEFT$(W,4-LEN(Y))+Y:LPRINT
TAB(35);X;TAB(73);USING"*****",B(I);Y=HEX$(B(I)):
X="$"+LEFT$(W,4-LEN(Y))+Y:LPRINT TAB(81);X:NEXT
770 LPRINT:LPRINT:LPRINT TAB(26);"Point chaud : x ="
U"/ y =" V:PUT(270,224),M,PSET:CALL SHOWCURSOR:
GOTO 410
780 Curseur
790 IF G=0 THEN 810 ELSE A=1:FOR I=3 TO 18:IF A(I) THEN A=0
800 NEXT:IF A THEN 410
810 IF G THEN G=0:C(33)=V:C(34)=U:FOR I=1 TO 16:
C(I)=A(I+2):NEXT:FOR I=17 TO 32:C(I)=B(I-14):
NEXT ELSE G=1:GOSUB 2170
820 CALL SETCURSOR(VARPTR(C(1)))GOTO 410
830 Effacer curseur
840 GOSUB 2150:GOSUB 1850:CALL MOVETO(124,124):
PRINT WV "effacer ?":GOSUB 1920:GOSUB 2070
850 IF D THEN 670 ELSE PUT(108,92),M,PSET:
LINE(28,60)-(156,188),30,BF:
LINE(180,60)-(308,188),30,BF:
LINE(140,220)-(196,252),30,BF:FOR I=3 TO 18: A(I)=0:
B(I)=0:NEXT:IF F THEN GOSUB 1810
860 U=0:V=0:B=0:GOTO 410
870 Grille curseur

```

```

880 IF F THEN F=0 ELSE F=1
890 GOSUB 1810:GOTO 410
900 Quitter curseur/forme
910 N=1:GOTO 930
920 N=0
930 GOSUB 2150:GOSUB 1850:GOSUB 1920:
CALL MOVETO(124,116):PRINT WV "quitter ?":
GOSUB 2070:IF D THEN IF N THEN 670 ELSE 1340
940 IF N THEN ERASE B
950 ERASE A:CALL HIDECURSOR:GOTO 130
960 Reprendre forme
970 E=0:GOTO 1000
980 Créer forme
990 E=1
1000 DIM A(218)
1010 A(1)=64:A(2)=54:L=64:M=54:RESTORE 100:FOR I=1 TO 34:
READ D(I):NEXT:F(1)=4F(2)=4F-1
1020 LINE(358,3)-(458,93),30,BF:LINE(358,3)-(458,93),,B:
LINE(360,5)-(456,91),,B:LINE(360,94)-(460,94)
1030 LINE(335,110)-(482,274),30,BF:
LINE(335,110)-(482,274),,B:
LINE(337,112)-(480,272),,B:LINE(337,275)-(484,275)
1040 CALL PENNSIZE(2,2):LINE(459,5)-(459,93):
LINE(483,111)-(483,274)
1050 CALL PENNSIZE(2,2):RESTORE 90:FOR I=128 TO 272 STEP
16:LINE(337,I)-(480,I):CALL MOVETO(342,I-4):READ W:
IF I=272 THEN PRINT W; ELSE PRINT W
1060 NEXT:CALL SETCURSOR(VARPTR(C(1)))IF E THEN 1110
1070 GOSUB 2150:GOSUB 1850:Z=""
1080 GOSUB 1950:IF A THEN 950 ELSE OPEN "I",1,2,6
1090 INPUT #1,L,M:IF L<1 OR L>64 OR M<1 OR M>54 THEN ERROR
57
1100 K=(L+15)\16:FOR I=1 TO M:FOR J=1 TO K:
INPUT #1,A((I-1)*4+J+2):NEXT:NEXT:CLOSE:
PUT(108,92),M,PSET
1110 LINE(3,3)-(L*5+7,M*5+7),30,BF:
LINE(3,3)-(L*5+7,M*5+7),,B:GOSUB 1830:
LINE(5,M*5+8)-(L*5+9,M*5+8)
1120 CALL PENNSIZE(2,2):LINE(L*5+8,4)-(L*5+8,M*5+7):
CALL PENNSIZE(2,2):IF E THEN 1140
1130 PUT(5,5)-(324,274),A,OR:PUT(376,21),A
1140 IF M=54 THEN R(1)=129:GOSUB 2130
1150 IF L=64 THEN R(1)=161:GOSUB 2130
1160 IF M=1 THEN R(1)=145:GOSUB 2130
1170 IF L=1 THEN R(1)=177:GOSUB 2130
1180 W=INKEY$:IF W<>" THEN 1270 ELSE IF NOT MOUSE(0)
THEN 1180
1190 P=MOUSE(1):Q=MOUSE(2):IF MOUSE(0)=-1 THEN IF P>337
AND P<481 AND Q>113 AND Q<274 THEN N=(Q-114)\16:
GOTO 1280
1200 I=POINT(P-(P MOD 5=0),Q-(Q MOD 5=0)) AND 1
1210 IF P<5 OR P>L*5+5 OR Q<5 OR Q>M*5+5 THEN 1180
1220 Points forme
1230 WHILE MOUSE(0)<0:R=MOUSE(1)\5:S=MOUSE(2)\5:IF R<1
OR R>L OR S<1 OR S>M THEN 1260
1240 J=R*5+1:K=S*5+1:T=((R+15)\16)+2:S=(S-1)*4+T:
IF I THEN PUT(J,K),F,PRESET:A(S)=A(S) AND NOT
H(R-(16*(T-3))) ELSE PUT (J,K),F,PSET:A(S)=A(S) OR
H(R-(16*(T-3)))
1250 PUT(376,21),A,PSET
1260 WEND:GOTO 1180
1270 N=ASC(W)-48:IF N<0 OR N>9 THEN 1180
1280 IF (N=1 AND M=54) OR (N=2 AND M=1) OR (N=3 AND L=64)
OR (N=4 AND L=1) THEN 1180 ELSE R(1)=113+16*N:

```

```

R(2)=338:R(3)=R(1)+15:R(4)=480
1290 FOR I=0 TO 3:CALL INVERTRECT(VARPTR(R(1))):
FOR D=0 TO 50:NEXT:U=L*5:V=M*5:ON N GOTO 1370,
1410,1440,1480,1510,1610,1650,920,1680
1300 'Effacer forme
1310 GOSUB 2150:GOSUB 1850:CALL MOVETO(124,124):
PRINT WV "effacer ?":GOSUB 1920:GOSUB 2070
1320 IF D THEN 1340 ELSE PUT(108,92),M,PSET:
LINE(5,5)-(5*L+5,5*M+5),30,BF:
LINE(362,7)-(454,89),30,BF:FOR I=3 TO 218:A(I)=0:NEXT:
IF F THEN GOSUB 1830
1330 GOTO 1180
1340 PUT(108,92),M,PSET
1350 IF MOUSE(0)<0 THEN 1350 ELSE 1180
1360 'Ajouter ligne
1370 GET(3,V+1)-(U+9,V+7),M:PUT(3,V+6),M,PSET:
LINE(5,V+6)-(U+5,V+10),30,BF:LINE(5,V+13)-(U+9,V+13):
CALL MOVETO(342,156):PRINT "2 Supprimer ligne"
1380 M=M+1:IF F THEN LINE(5,V+10)-(U+5,V+10):
FOR I=1 TO L+1:LINE(I*5,V+5)-(I*5,V+10):NEXT
1390 GOTO 1140
1400 'Supprimer ligne
1410 LINE(376,20+M)-(439,20+M),30:GET(376,21)-(439,74),A:
GET(3,V+6)-(U+9,V+7),M:PUT(3,V+1),M,PSET:GOSUB 1790
1420 R(1)=V+3:R(2)=3:R(3)=R(1)+6:R(4)=330:
CALL FILLRECT(VARPTR(R(1)),VARPTR(P(1))):
LINE(5,V+3)-(U+9,V+3):M=M-1:CALL MOVETO(342,140):
PRINT "1 Ajouter ligne":GOTO 1160
1430 'Ajouter colonne
1440 GET(U+1,3)-(U+7,V+8),M:PUT(U+6,3),M,PSET:
LINE(U+6,5)-(U+10,V+5),30,BF:CALL PENSIZ(2,2):
LINE(U+13,4)-(U+13,V+7):CALL PENNORMAL:
CALL MOVETO(342,188):PRINT "4 Supprimer colonne"
1450 L=L+1:IF F THEN LINE(U+10,5)-(U+10,V+5):
FOR I=1 TO M+1:LINE(U+5,I*5)-(U+10,I*5):NEXT
1460 GOTO 1150
1470 'Supprimer colonne
1480 LINE(375+L,21)-(375+L,74),30:GET(376,21)-(439,74),A:
GET(U+6,3)-(U+7,V+8),M:PUT(U+1,3),M,PSET:GOSUB 1790:
R(1)=3:R(2)=U+3:R(3)=V+9:R(4)=R(2)+10
1490 CALL FILLRECT(VARPTR(R(1)),VARPTR(P(1))):
CALL PENSIZ(2,2):LINE(U+3,4)-(U+3,V+7):L=L-1:
CALL MOVETO(342,172):PRINT "3 Ajouter colonne":
CALL PENNORMAL:GOTO 1170
1500 'Imprimer forme
1510 GOSUB 2150:GOSUB 1850:CALL MOVETO(124,124):
PRINT WV "imprimer ?":GOSUB 1920:GOSUB 2070
1520 IF D THEN 1340 ELSE PUT(108,92),M,PSET
1530 CALL SETCURSOR(VARPTR(D(1))):P=328:Q=214:R=488:
S=262:GOSUB 1850:CALL MOVETO(333,241):PRINT
"Impression...":P=3:Q=P:R=L*5+7:S=M*5+7:W1="FO000":
LPRINT CHR$(24):GOSUB 1880
1540 LPRINT CHR$(24) CHR$(31) "1":P=358:Q=3:R=458:S=93:
GOSUB 1880
1550 LPRINT CHR$(31):"1":CHR$(24):CHR$(27):"c":
O=(L+15)\16:P=1:R=1:W="000":LPRINT "largeur :";L:
"/ hauteur :";M:4*M*2*((L+15)\16):"octets":LPRINT
1560 FOR I=1 TO M:FOR J=1 TO O:IF R=1 AND P>1 THEN Q=P+5
ELSE Q=P
1570 LPRINT TAB(Q);USING "*****":A((I-1)*4+J+2):
Y=HEX$(A((I-1)*4+J+2)):X="$"+LEFT$(W,4-LEN(Y))+Y:
LPRINT TAB(Q+7):X:R=R+1
1580 IF R=33 THEN R=1:P=P+14:LPRINT CHR$(27) "r" CHR$(31)
"?":CHR$(31) "?":CHR$(31) "2":CHR$(27) "c":

```

```

1590 NEXT:NEXT:PUT(328,214),M,PSET:
CALL SETCURSOR(VARPTR(C(1))):GOTO 1180
1600 'Enregistrer forme
1610 GOSUB 2150:GOSUB 1850
1620 GOSUB 1950:IF A THEN 1340 ELSE OPEN "0",1,7,6:
PRINT#1,L,M0=(L+15)\16:FOR I=1 TO M:FOR J=1 TO O:
PRINT#1,A((I-1)*4+J+2):NEXT:NEXT:CLOSE
1630 OPEN "clip:" FOR OUTPUT AS 1 LEN=6:
PRINT #1,4*M*2*((L+15)\16):"octets":CLOSE:GOTO 1340
1640 'Grille forme
1650 IF F THEN F=0 ELSE F=1
1660 GOSUB 1830:GOTO 1180
1670 'Décaler
1680 CALL INITCURSOR:P=370:Q=80:R=434:S=272:GOSUB 1850:
PUT(391,99),I
1690 IF MOUSE(0)<-1 THEN 1690 ELSE P=MOUSE(1):
Q=MOUSE(2):IF P<386 OR P>418 OR Q<96 OR Q>255 THEN
1690 ELSE N=(Q-96)\32
1700 R(1)=97+32*N:R(2)=387:R(3)=R(1)+31:R(4)=418:
FOR I=0 TO 3:CALL INVERTRECT(VARPTR(R(1))):
FOR D=0 TO 20:NEXT:NEXT:
ON N GOTO 1730,1740,1750,1760
1710 P=376:Q=22
1720 GOSUB 2190:GOTO 1690
1730 P=377:Q=21:GOTO 1720
1740 PUT(370,80),M,PSET:GOSUB 2170:
CALL SETCURSOR(VARPTR(C(1))):GOTO 1350
1750 P=375:Q=21:GOTO 1720
1760 P=376:Q=20:GOTO 1720
1770 'Sous-programmes
1780 'Backpat/Penpat
1790 FOR I=1 TO 4:P(I)=&H55AA:NEXT:RETURN
1800 'Grille curseur
1810 C=30 OR F:FOR I=28 TO 180 STEP 152:
FOR J=1 TO I+128 STEP 8:LINE(J,60)-(J,188),C:NEXT:
FOR J=60 TO 188 STEP 8:LINE(I,J)-(I+128,J),C:NEXT:
NEXT:RETURN
1820 'Grille forme
1830 C=30 OR F:J=L*5+5:K=M*5+5:FOR I=5 TO J STEP 5:
LINE(I,5)-(I,K),C:NEXT:FOR I=5 TO K STEP 5:
LINE(5,I)-(J,I),C:NEXT:RETURN
1840 'Cadre message
1850 GET(P,Q)-(R,S),M
1860 LINE(P,Q)-(R,S),30,BF:LINE(P,Q)-(R,S),B:
CALL PENSIZ(2,2):LINE(P+2,Q+2)-(R-2,S-2),B:
CALL PENNORMAL:RETURN
1870 'Impression
1880 LPRINT CHR$(27) ">" CHR$(27) "n" CHR$(27) "T14":
W=STR$(R-P+1):Y=SPACE$(4-LEN(W))+W:O=S-Q+1:T=0\7:
IF O MOD 7 THEN T=T+1
1890 FOR I=1 TO T:W="":FOR J=P TO R:N=0:FOR K=0 TO 6:
O=(I-1)*7+K+Q:IF O>S THEN 1900 ELSE IF POINT(J,O) AND
1 THEN N=N OR H(15-K)
1900 NEXT:W=W+CHR$(N):NEXT:LPRINT CHR$(27) W:CHR$(27)
"G" Y W:NEXT:RETURN
1910 'OK/Annuler
1920 CALL MOVETO(228,156):PRINT "OK":
CALL MOVETO(308,156):PRINT "Annuler":R(1)=140:
R(2)=196:R(3)=164:R(4)=276:
CALL FRAMEROUNRECT(VARPTR(R(1)),12,12)
1930 R(2)=292:R(4)=372:
CALL FRAMEROUNRECT(VARPTR(R(1)),12,12):
CALL PENSIZ(2,2):R(1)=136:R(2)=192:R(3)=168:
R(4)=280:CALL FRAMEROUNRECT(VARPTR(R(1)),20,20):

```

```

CALL PENNORMAL.RETURN
1940 'Saisie nom fichier
1950 CALL MOVETO(124,124):PRINT "Nom du fichier ?":
CALL MOVETO(308,124):PRINT "Annuler":R(1)=108:
R(2)=292:R(3)=132:R(4)=372:
CALL FRAMEROUNDRECT(VARPTR(R(1)),12,12):
LINE(116,140)-(380,164),,B
1960 E(1)=1:E(2)=16:FOR I=3 TO 18:E(1)--1:NEXT:
CALL TEXTFONT(4):FOR I=1 TO 4:P(I)=0:NEXT:
CALL BACKPAT(VARPTR(P(1))):C=LEN(Z)
1970 CALL MOVETO(124,156):PRINT Z:
1980 X=INKEY$:IF X<>" " THEN 2000 ELSE IF MOUSE(0)<>-1 THEN
A=A+1:IF A<30 THEN 1980 ELSE
PUT((POS(I)*7)-3,144),E:A=0:GOTO 1980
1990 P=MOUSE(1):Q=MOUSE(2):IF P<292 OR P>371 OR Q<108 OR
Q>131 THEN 1980 ELSE A=1:CALL TEXTFONT(0):RETURN
2000 IF X=CHR$(13) THEN A=0:CALL TEXTFONT(0):RETURN ELSE
IF X=CHR$(8) THEN IF C=0 THEN 1980 ELSE C=C-1:
Z=LEFT$(Z,C):LINE(117,141)-(379,163),30,BF:GOTO 1970
2010 IF C=36 THEN 1980 ELSE
PUT((POS(I)*7)-3,144),E,PSET:PRINT X;Z=Z+X:C=C+1:
GOTO 1980
2020 X=INKEY$:IF X<>" " THEN 2040 ELSE IF MOUSE(0)<>-1 THEN
2020
2030 P=MOUSE(1):Q=MOUSE(2):IF P<292 OR P>371 OR Q<140 OR
Q>163 THEN 2020 ELSE 2050
2040 IF ASC(X)<>13 AND ASC(X)<>3 THEN 2020
2050 P=108:Q=92:GOSUB 1860:RETURN
2060 'Saisie OK/Annuler
2070 W=INKEY$:IF W<>" " THEN 2100 ELSE IF MOUSE(0)<>-1
THEN 2070
2080 P=MOUSE(1):Q=MOUSE(2):IF Q<140 OR Q>163 THEN 2070
2090 IF P>195 AND P<276 THEN D=0:GOTO 2110 ELSE IF P>291
AND P<372 THEN D=1:GOTO 2110 ELSE 2070

```

```

2100 IF ASC(W)<>13 AND ASC(W)<>3 THEN 2070 ELSE D=0
2110 RETURN
2120 'Caractères estompés
2130 GOSUB 1790:CALL PENPAT(VARPTR(P(1))):
CALL PENMODE(-1):R(2)=338:R(3)=R(1)+15:R(4)=479:
CALL PAINTRECT(VARPTR(R(1))):
CALL PENNORMAL.RETURN
2140 'Coordonnées cadre message
2150 P=108:Q=92:R=388:S=180:RETURN
2160 'Curseur
2170 RESTORE 50:FOR I=1 TO 13:READ C(I):NEXT:
FOR I=14 TO 32:C(I)=0:NEXT:C(33)=6:C(34)=6:RETURN
2180 'Décalage
2190 GET(P,Q)-(P+L-1,Q+M-1),0:PUT(376,21),0,PSET:
PUT(5,5)-(L*5+4,M*5+4),0,PSET:
GET(376,21)-(439,74),A:GOSUB 1830:RETURN
2200 'Erreurs
2210 CLOSE:GOSUB 1860:CALL MOVETO(324,156):PRINT "OK":
R(1)=140:R(2)=292:R(3)=164:R(4)=372:
CALL FRAMEROUNDRECT(VARPTR(R(1)),12,12):
CALL MOVETO(124,124)
2220 IF ERR=74 THEN PRINT "Volume inconnu !":GOTO 2290
2230 IF ERR=64 THEN PRINT "Nom incorrect !":GOTO 2290
2240 IF ERR=61 THEN PRINT "Disquette saturée !":GOTO 2290
2250 IF ERR=70 THEN PRINT "Disquette protégée !":GOTO 2290
2260 IF ERL=390 OR ERL=1090 OR ERL=1100 THEN PRINT
"Lecture impossible !":GOTO 2280
2270 IF ERR=53 THEN PRINT "Fichier introuvable !"
2280 GOSUB 2020:IF ERL=380 OR ERL=390 THEN RESUME 380
ELSE RESUME 1080
2290 GOSUB 2020:IF ERL=660 THEN RESUME 660 ELSE IF
ERL=380 THEN RESUME 380 ELSE IF ERL=1620 THEN
RESUME 1620 ELSE RESUME 1080

```



54, rue de Dunkerque
75009 PARIS Tél.: 282.17.09
Métro: Gare du Nord (100 m)

SURPRIS ... LES PRIX!!!

PRIX T.T.C.I POUR APPLE ET COMPATIBLES

Diskettes U.S 5" 1/4 SF/SD

Diskettes 5" 1/4 SF/DD

Lecteur diskettes pour APPLE
(mécan. Japonais, entr. direct)
Carte synthétiseur de voix
Carte mémoire/langage 16 K Ram
Carte mémoire 128 K
Carte drive 13/16 sect.
Carte 80 colonnes
Carte imprimante parallèle
Carte imprimante + Buffer 32 K
Carte série
Carte super série
Carte Z80 - CP/M
Joystick de luxe 2 +, 2E, 2C

139 F/boîte 10 (exp. min. 5 boîtes Port 27 F)

170 F/boîte 10 (exp. min. 5 boîtes Port 27 F)



Carte wilcard (déplombage)
Carte communication
Carte IEEE - 488
Carte copieur Eprom
Carte A/D - D/A 12 bit
Carte horloge
Carte musique
Carte RGB + Prise TV Secam
Carte 6522 Via
Port pour une carte
Ventilo 10 W. super silencieux



Port Urgent ajouter 5,50 F

NOUVEAU: Pince spéciale pour diskettes (100 000 trous min.) (port : 13 F) **69 F**

Ordinateur multicompatible Forth, Basic, CP/M, MS-Dos, CP/M86

Écrivez, nous vous enverrons une liste plus complète de nos articles.

Revendeurs, contactez-nous.

Notre devise : "DYNAMIT COMPUTER : MOINS CHER QUE MOI TU MEURS !"

	Logiciels	Distributeur		Prix TTC	
1	PrologII Basic Microsoft Mac Forth Macintosh Pascal Macintosh Basic Pascal UCSD	Prologia Microsoft France Feeder Apple Apple Bus	1/85 06/84 08/84 11/84 12/84 10/84	NC 1694,00 MFI:1589,00 1306,00 1430,00 7000,00	Langage d'intelligence artificielle Le plus répandu Trois versions Pascal interprété Basic pascalien Pascal compilé
2	FileVision CX MacBase File Overvue AD MacFichier ABCbase MacStock Pfs:File/Report MacLedger MacCash Minifact	Apple Contrôle X Microsoft France Apple Answare A.C.I. M.C.S. Sonotec Logiciel P.C. Logiciel P.C. Idées informatique	10/84 10/84 10/84 10/84 08/84 10/84 10/84 10/84 11/84 07/84 11/84	2360,00 2953,00 2396,00 4150,00 2360,00 3320,00 1779,00 2711,00 NC 2609,00 NC	Les informations en images Intègre calculs, graphisme, texte... Structure des fiches variable Tri multi-critères Main street file en français Expression multiforme Gestion de stocks Organisez votre système Comptabilité Livre de caisse Facturation simplifiée.
3	Thinktank Mega Merge Word	Gamic Feeder Microsoft France	07/84 11/84 10/84	1957,00 NC 2336,00	Traitement d'idées Personnalisation par le dessin Sur l'imprimante comme à l'écran
4	Paris Création MacCrypt PolyFiche Mac agenda	A.C.I. C.E.R.I.A. Polygone Gamic	06/84 11/84 09/84 09/84	390,00 NC 790,00 1174,00	Adresses Fichiers codés Carnet d'adresses Emploi du temps
5	Mac Terminal Telemac Telemac Mac Tell Macphone	Apple Mediatec Mediatec Hello C.E.R.I.A.	11/84 11/84 11/84 11/84 11/84	990,00 NC NC NC NC	Emulation TTY, DEC, VT100... Emulation IBM 3270 Emulation minitel Minitel intelligent Composeur téléphonique
6	Mac Space Mac Graph Chart	A.S.A. M.C.S. Microsoft France	12/84 11/84 09/84	NC NC 1411,00	Graphisme en trois dimensions Décisionnel graphique Représentation graphique
7	Agatha MacDent MacDoc DoctorMac Transfu l'A.B.O MacPharm	C.D soft Dag informatique C.M.G JSR Informatique Transfu labo-serv. Inpharmez	11/84 11/84 11/84 NC 12/84 11/84	NC NC NC NC NC NC	Gestion d'un cabinet dentaire Gestion d'un cabinet dentaire Pour médecins généralistes Données médicales Laboratoires d'analyses médicales Gestion d'une pharmacie
8	MacJack MacSlots Jeux de mots Sargon III MacManager Reversi Boule de silicium Run for the money Mac Puzzle Transylvania Le compte est bon	Answare Feeder Hello Sonotec Feeder R.C.I informatique Hyperlog Feeder Feeder Feeder Polygone	08/84 09/84 11/84 11/84 10/84 11/84 09/84 09/84 09/84 09/84 10/84	499,00 824,00 NC NC 700,00 NC 475,00 593,00 522,00 540,00 295,00	Black Jack Deux jeux différents Réservé aux enfants Echecs. 9 niveaux de complexité Jeu de rôle Stratégie Votre autoportrait Aventure Aventure Jeu télévisé
9	MacProject MacExpert Slide 1 Da Vinci	Apple Answare S.L.I. Réalisations Gamic	11/84 11/84 12/84 10/84	1400,00 NC NC 889	Planification des projets Système expert Plans d'architecture Bibliothèques de dessins

1 - langages. 2 - gestion et comptabilité. 3 - traitement de textes. 4 - organisation. 5 - communication.
6 - graphisme. 7 - médecine. 8 - jeux. 9 - divers.

Omnis 2 à l'essai

Guy Lapautre

Cet essai a été réalisé sur un Apple //e muni de deux lecteurs de disques souples. Notre impression générale sur ce logiciel de gestion de fichiers tient en deux mots : PUISSANT - COMPLEXE.

Premier contact

Enfin un bon Guide Pédagogique !

Ne parlons pas de ses qualités pédagogiques proprement dites, on a déjà fait mieux: il manque en particulier certains fac simile d'écran. Mais il sépare TRES NETTEMENT l'essentiel (décrit dans un guide pédagogique simplifié), les options ou fonctionnalités plus complexes (réunies dans un guide pédagogique avancé) et les détails supplémentaires d'usage moins courant (mentionnés seulement dans le manuel de référence). Exemple dont beaucoup pourraient s'inspirer...

De nombreux logiciels proposent aujourd'hui une disquette "Tutorial" pour les tous premiers pas. Compte tenu de la complexité d'Omnis 2, sur laquelle nous reviendrons, cette solution aurait peut-être pu être retenue.

Les disques

Regrettons une ambiguïté dès le départ: après avoir montré le danger de travailler avec les disques originaux, on explique comment faire des copies des disques Program et Utilities (au passage: pourquoi pas Programme et Utilitaires?). Mais on explique aussi que le disque Boot ou System (même remarque), susceptible pourtant des mêmes avanies que les autres, n'est pas copiable. Il y a là de quoi perturber grandement un néophyte. Ce n'est que beaucoup plus tard, en abordant la lecture du Manuel de Référence, qu'on est avisé d'avoir à commander au fournisseur un double de ce disque.

Les disques sont, comme nous venons de le voir, au nombre de trois. Chaque application, si minime soit-elle, nécessite de son côté deux disques: la "librairie" (les explications concernant la dénomination de ce disque ne sont pas très convaincantes) et le disque "data" (données ne serait pas plus mal).

Donc, aucun travail possible sans un minimum de cinq disques avec lesquels on doit jongler dans deux lecteurs. De quoi décourager complètement un débutant. Comprenons bien

que ce n'est pas une critique du logiciel, mais de la façon dont les choses sont présentées. La complexité est la rançon de la puissance, et on peut dire qu'Omnis 2 n'est pas fait pour être utilisé normalement par un amateur sur une configuration Apple //e avec deux disques souples. On peut pourtant lire dans le guide pédagogique (leçon 1): "Ce Guide Pédagogique part de l'hypothèse que vous n'avez pas ou peu d'expérience quant à l'utilisation d'un micro-ordinateur".

Dans cette hypothèse, l'opinion de l'auteur de ces lignes (qui n'engage que lui) est: abstenez-vous de débiter avec Omnis 2, et commencez par vous familiariser avec la micro-informatique en utilisant un logiciel moins performant, mais plus simple.

Les commandes principales

Elles utilisent de façon systématique la touche "ESCAPE". La mode actuelle est plutôt d'utiliser les touches "POMME". Ne prenons pas parti. Les touches "POMME" présentent l'avantage, étant non classiques, de servir assez normalement à tout ce qui n'est pas entrée de données. Mais elles sont spécifiques des versions //e et //c, et d'autres commandes doivent bien être adoptées pour les versions antérieures de l'Apple II.

Il n'existe pas de commande (ou du moins n'en avons nous pas rencontrée) dont on ne puisse pas sortir si on y est entré par mégarde. Voilà qui est rassurant. Seule la frappe inopportune de "Q" dans le menu "system" peut renvoyer hors du programme. Ce n'est pas bien grave, aucune donnée n'étant alors perdue.

La création d'une structure de fichier

La puissance

On peut à peu près tout faire dans ce domaine, qu'il s'agisse:

- de la présentation des écrans;
- de la définition des rubriques;
- des nombreuses options de format qui leur sont associées;
- des modes de contrôle des entrées;
- de la définition des rubriques calculées

On peut aussi (alors que c'est un point faible de nombreux logiciels de gestion de fichiers), modifier la struc-

ture de fichiers existants sans perte d'informations. Au prix toutefois de nombreuses "jongleries" avec les disques...

Regrettons seulement:

- la limitation à une ligne de l'ensemble nom d'écran + contenu;
- la limitation à 7 caractères du nom de rubrique.

La complexité

La ligne de commandes n'est pas très simple à interpréter et le recours à l'écran S.O.S. relativement fréquent.

Mais surtout, TOUTES les caractéristiques des rubriques sont à entrer sous forme de codes à un caractère (sauf la longueur). Les "assez classiques" C pour caractères ou N pour numériques sont simples à interpréter. Mais I ou J pour les index selon leur nature, les codes de protection, de calcul, ... sont pratiquement impossibles à retenir par coeur.

Il n'est par exemple pas évident que l'attribut:

+N2VE

concerne une rubrique calculée, numérique, 2 décimales, acceptant des valeurs négatives et réputée vide si la valeur calculée est égale à 0 (exemple du manuel de référence)!

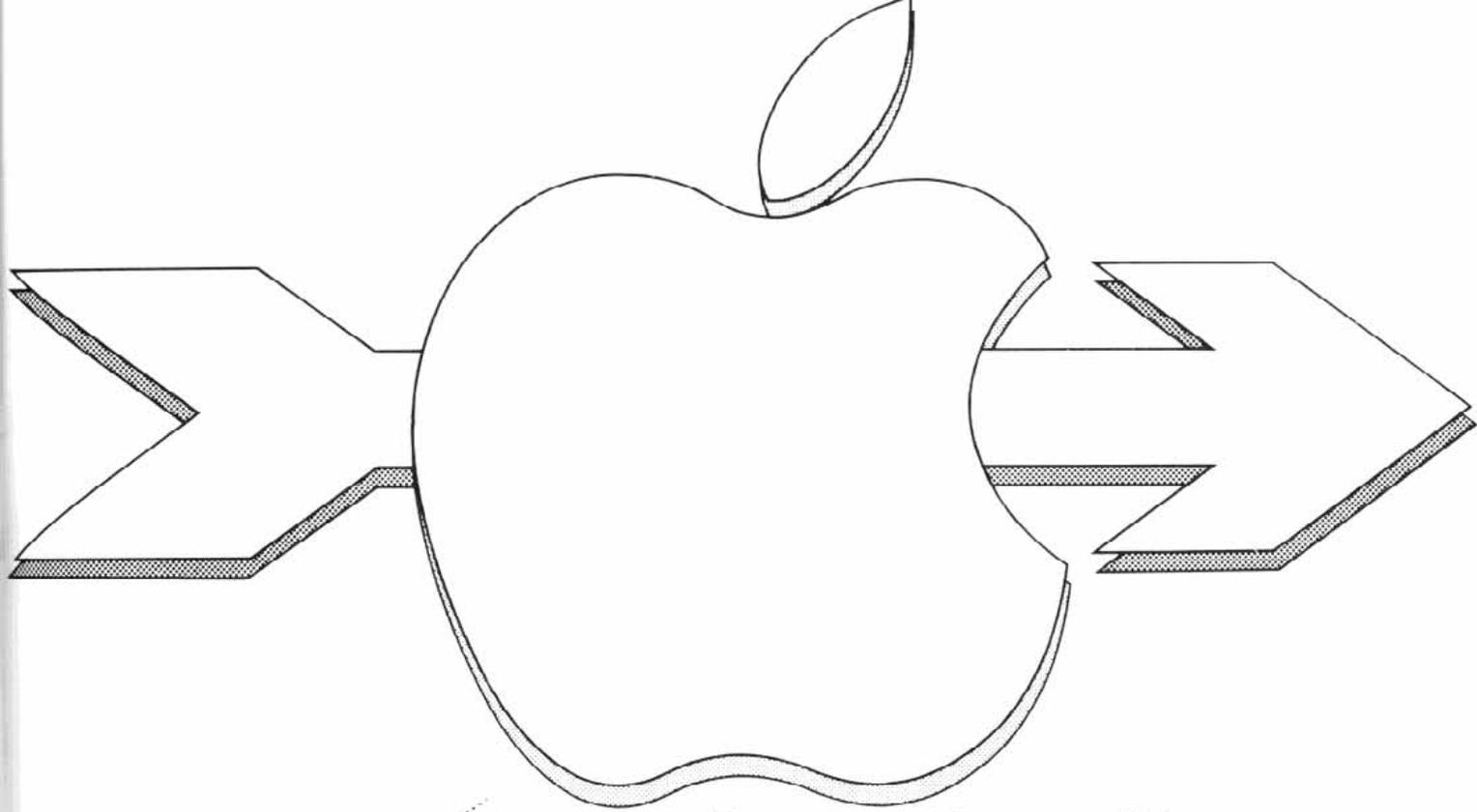
Les formules de calculs et de vérification (très précieuses certes) utilisent de nombreuses fonctions qu'on ne trouve qu'en programmation. Par exemple LEN(...) pour le nombre de caractères d'une chaîne ou CON(...) pour une concaténation de chaînes, ou encore une formulation d'algèbre logique très particulière utilisant les signes ! et &. et ceci dès la leçon 4 du guide pédagogique simplifié! A noter que, dans d'autres cas, on emploie OU et ET en toutes lettres...

La gestion de fichiers

La puissance

La multiplicité des détails utilisables dans la création des structures permet de nombreux contrôles au niveau de la saisie. Le logiciel permet par exemple:

- de ne pas entrer deux fois le même numéro de code, si celui-ci doit être unique;
- de contrôler qu'on entre bien Mr, Mme ou Mlle et rien d'autre;
- de vérifier qu'un code postal comporte 5 chiffres (ni 4 ni 6);



apple-tell

un modem qui fait, d'un simple Apple, un Minitel intelligent

Apple-tell comprend :

- Une carte Modem incluant un décodeur Teletel.
- Un logiciel d'Emulation de Terminal Minitel enrichi de trois éblouissantes fonctions (celles qui faisaient le plus défaut jusqu'à présent sur votre Minitel)

IMPRESSION : l'imprimante de votre Apple est exploitée pour sortir les copies-papier dont vous avez besoin lorsque vous consultez un serveur.

STOCKAGE : les disquettes de votre Apple sont utilisées pour enregistrer les pages dont la consultation vous est nécessaire

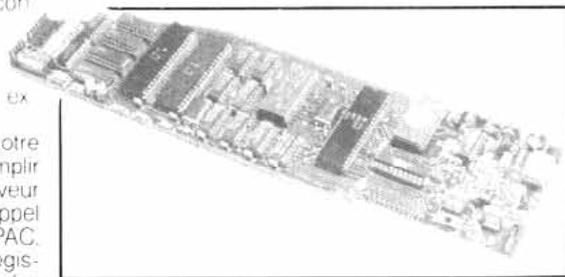
- au format Teletel (c'est à dire telles que vous les avez reçues).
- en mode Texte pur (ASCII) pour exploitation locale ultérieure.

AUTOMATISME : l'intelligence de votre Apple est mobilisée pour accomplir l'interrogation automatique du serveur que vous lui avez désigné (appel téléphonique, orientation TRANSPAC, identification, choix successifs), enregistrer sur papier et/ou sur disque les données consultées, puis pour traiter celles-ci, en les incorporant dans votre application.

(Les procédures d'interrogation sont créées par l'utilisateur sans aucun langage de programmation, grâce au mode d'apprentissage d'Apple-Tell.)

Evénement du dernier SICOB, salué par toute la presse, consacré Pomme d'Or 1983 par le jury Apple, le modem Apple-Tell marque une mutation décisive dans l'évolution des techniques vidéotex en environnement professionnel.

- point d'arrêt à la prolifération des matériels sur votre bureau (effet "mini-Sicob")



- Utilisation optimale des ressources dont vous disposez déjà (disques, imprimante, logiciels, etc...)
- Utilisation possible en mode Terminal autant qu'en mode Serveur (jusqu'à quatre portes)
- Enfin (et c'est sans doute le point le plus important) JONCTION entre le monde extérieur et les outils standards de votre Apple... l'incorporation des données dans Apple-writer, Visicalc, Multiplan, PFS, Quick-File, etc..., et même dans vos applications personnelles (comptabilité, suivi de commandes, fichier...) devient possible.

CARACTERISTIQUES GENERALES :

- Modem multimodes :
 - 1 200/75, (full-duplex)
 - 1 200 (half-duplex),
 - 600 (half-duplex), 300 (full-duplex),
 - standards CCITT et BFL
 (cette caractéristique unique rend accessibles les serveurs nord-américains, y compris par réseau téléphonique commuté).

Sorties : video
HELLO Informatique
1, rue de Metz
75010 PARIS
Tél. : (1) 523 30 34
Telex : GLASH 210500 F

composite (N & B) et Pétrel
couleurs

- compatible Apple 2, Apple 2+, Apple 2e (48 K, une disquette)

- Enrichissable dans n'importe quel slot libre de votre Apple 2

- Transparence totale vis à vis du système

Nom

Société

Adresse

Ville

Code postal Tél.

- Souhaite recevoir une documentation sur le système Apple-Tell
- Commande système(s) Apple-Tell au prix de F. 6.997 T.T.C. (règlement ci joint par chèque bancaire CCP)

HELLO Informatique 1, rue de Metz, 75010 PARIS - Tél. (1) 523 30 34

- de choisir l'ordre d'entrée des rubriques (et pas forcément celui de leur apparition sur l'écran)...

Attention d'avoir sous la main une liste des contrôles effectués, sous peine de courir le risque de ne jamais remplir une fiche, par exemple en s'entêtant à mettre M. pour Monsieur au lieu de Mr.

Très nombreuses possibilités également dans le domaine de la recherche, du tri, de la sélection. Possibilités fort intéressantes de mise à jour et de suppression multiple, de copie de données d'un enregistrement sur un autre. Très utile fonction de "hardcopie" d'écran, etc.

La complexité

Elle est moins grande dans l'exploitation que dans la création. Notons toutefois le même égotisme de la ligne de commande.

L'utilisation des formats de recherche n'est pas toujours évidente. Remarquons au passage que les signes "<" et ">" veulent dire tantôt strictement plus petit (ou plus grand), tantôt inférieur (ou supérieur) ou égal !

Exemple (du guide pédagogique avancé) pour un format de mise à jour multiple :

```
INDCLE = UPP ( CON ( MID (NOM, 1, 3), INITS))
```

Si ce n'est pas de la programmation, qu'est-ce que c'est ?

Les états

On pourrait à ce sujet faire les mêmes types de remarques que dans les paragraphes précédents, tant pour la puissance que pour la complexité. Ne laissons pas le lecteur et contentons-nous de deux remarques importantes :

Puissance : outre les classiques productions d'états et d'étiquettes, Omnis 2 possède un système très astucieux de production de lettres personnalisées.

Complexité : les codes de contrôle d'imprimante tels que le nombre de caractères par pouce, le soulignement,... étant inclus dans la définition de l'état, un changement d'imprimante nécessite une révision complète de toutes les structures d'états qui ont été définies (seules les options générales telles que longueur de la ligne, nombre de lignes par page,... sont traitées à part).

Les mots de passe

Un système ingénieux et très complet de clés d'accès à 4 niveaux (plus le Maître, qui dit bien son nom...) peut être mis en place.

La définition de ce qui est autorisé ou interdit à chacun est assez simple, mais longue à mettre en oeuvre, ne serait-ce que parce qu'on descend (ce qui est parfois précieux) au niveau de la rubrique élémentaire, en lecture et en écriture.

Quand l'ensemble est créé, son utilisation ne pose aucun problème.

Les utilitaires

Ils sont très complets, même s'ils ne sont pas toujours très simples à utiliser; nous avons déjà évoqué ce problème à propos de la modification de structure des fichiers.

Ils permettent, outre les classiques copies, destructions, initialisations, compactages, ..., de faire communiquer Omnis 2 avec d'autres logiciels par l'intermédiaire de fichiers au format DIF, et ce dans les deux sens. Un autre utilitaire permet des conversions entre Pascal et le DOS 3.3.

Capacité et performances

Seules limites de capacité rencontrées : ce sont celles de l'Apple //e avec deux disquettes: en fait, un disque dur s'impose.

Que faire par exemple, avec cette pauvre configuration, de 131068 blocs par fichier ou de 9600 rubriques dans un format d'état ?

Seule limitation inhérente au logiciel, déjà signalée : 79 caractères par rubrique (y compris, le cas échéant, le "nom d'écran").

Ne disposant que d'une version "bridée" au niveau de 50 enregistrements, nous n'avons pu faire aucun essai de performances sur gros fichier. Il semble que les recherches de type T (sur rubriques indexées), soient toujours très rapides, alors que les tris et les recherches de type R ne le soient guère, même pour des fichiers relativement modestes. Pour 50 enregistrements, un tri ou une recherche simple durent environ 40 secondes.

Synthèse

Nous avons écrit en exergue : PUIS-SANT - COMPLEXE.

Les fonctions de création

Création de structure de fichiers, formats d'états, mots de passe, sont totalement hors de portée de l'utilisateur débutant. Même s'agissant d'un utilisateur confirmé de micro-informatique, il n'est pas évident qu'il n'y ait jamais de blocages, au niveau d'une codification très poussée, ou de l'utilisation de l'algèbre logique ou de véritables éléments de programmation.

Mais celui qui aura la capacité (et le courage) d'aller jusqu'au bout sera récompensé par des performances exceptionnelles, surtout avec une configuration "musclée".

Les fonctions de gestion

La gestion de fichier et l'édition d'états sont plus accessibles, moyennant deux précautions dès lors qu'il s'agit de néophytes :

1 - Limiter leur travail aux cas où TOUS les paramètres sont bien définis, y compris tris et sélections, et aux options fondamentales de gestion, en excluant la mise au point de tableaux de sélection par exemple.

2 - Les munir d'une documentation synthétique, d'une part sur les commandes du logiciel qu'ils devront utiliser, d'autre part sur la codification adoptée (ceci n'est pas spécial à Omnis, mais d'autant plus important que les contrôles sont nombreux et sophistiqués).

Nous sommes plus réservés sur la possibilité de faire mettre en oeuvre par n'importe qui les fonctions de gestion de disque, y compris les sauvegardes.

En résumé

Un logiciel de tout premier plan, mieux adapté à une configuration comportant un disque dur, capable de satisfaire les plus exigeants et de constituer un outil complet pour une application dont la gestion de fichier constitue l'essentiel.

Mais un logiciel à ne pas mettre entre toutes les mains. Ce n'est pas un reproche : on peut mettre entre toutes les mains (ou à peu près...) une voiture de série, mais pas une formule 1 ni un 60 tonnes Saharien. Le reproche irait au fait que la documentation (et aussi la publicité) tendent à minimiser exagérément ce fait.

Ne soyez pas un  : si vous possédez un , lisez Pom's !

Initiation à l'assembleur (5)

Gérard Michel

Ce cinquième article marquera la fin de notre revue de détail des instructions de l'assembleur du processeur 6502 qui équipe votre Apple II+ ou //e.

Comme dans tout feuilleton moderne qui se respecte, nous aurons droit à quelques "flash-backs" sur des éléments importants de la vie du héros, la pile et le registre d'état en l'occurrence, dont nous avons jusqu'à présent laissé certains aspects dans l'ombre. Le dictionnaire étant ainsi complété, vous pourrez désormais parler le "6502" couramment...

Les interruptions et le BREAK

Lorsque nous avons abordé le registre d'état du microprocesseur dans le Pom's 13, certains indicateurs n'ont fait l'objet que d'une mention sommaire. Tel fut le cas pour le bit 2 (indicateur baptisé I comme Interruption) et pour le bit 4 (indicateur B comme Break). Réparons dès maintenant cette injustice.

Interruptions du 6502

Nous devons nous limiter sur ce point à un exposé quelque peu théorique, car les signaux d'interruption ne peuvent être envoyés simplement au microprocesseur par le biais d'un programme. Ils viennent en effet des périphériques connectés à l'unité centrale, via les cartes d'interface et les "slots" qui sont reliés au processeur. En résumé, un périphérique demande ainsi au 6502 de stopper l'exécution du programme en cours et de travailler momentanément pour lui, afin de lui transmettre les données dont il a besoin pour s'acquitter de sa propre tâche, ou de recevoir les données qu'il est chargé de fournir à l'ordinateur. Le terme "interruption" ne signifie donc pas que le processeur s'arrête, bien sûr, mais qu'il cesse de travailler pour l'un et se met au service de l'autre.

On distingue deux types d'interruptions, c'est-à-dire deux lignes reliées à deux broches différentes du 6502 : les interruptions masquables (dites IRQ) et les interruptions non masquables, ou prioritaires (dites NMI). Une interruption IRQ ne sera prise en compte par le processeur que si l'indicateur I (bit 2) de son registre d'état est à 0. En mettant cet indicateur à 1, soit à partir d'un programme utilisateur, soit dans une routine d'interface d'une carte de périphérique, on peut donc "masquer"

tous les signaux IRQ qui pourraient arriver au processeur, jusqu'à ce que l'on ait remis le bit I à 0. Par contre, quelle que soit la valeur de ce bit I, une interruption de type NMI sera toujours prise en compte et provoquera l'arrêt de toute autre opération, même s'il s'agit du travail d'un périphérique ayant déjà interrompu le processeur par les lignes IRQ ou NMI.

Dans un cas comme dans l'autre, lorsqu'arrive une interruption, le processeur commence par achever l'instruction en cours. Il dépose ensuite au sommet de la pile le contenu du compteur ordinal, dans l'ordre octet haut / octet bas (rappelons que le compteur ordinal est un registre sur 16 bits qui contient toujours l'adresse de la prochaine instruction à exécuter), puis il empile également le registre d'état, après en avoir mis le bit 4 (indicateur B) à 0.

S'il s'agit d'une IRQ, il saute alors à l'adresse contenue dans les mémoires \$FFFE et \$FFFF (en d'autres termes, il charge le compteur ordinal avec le contenu de \$FFFE-\$FFFF). Si c'est une NMI, en revanche, il saute à l'adresse contenue en \$FFFA-\$FFFB.

Notons bien que ces opérations sont programmées dans le processeur, c'est-à-dire que, quel que soit l'ordinateur étudié, le début du traitement d'une interruption sera celui indiqué ci-dessus si c'est un ordinateur construit autour d'un 6502. Ce qui se passe ensuite dépend par contre de ce que les concepteurs de la machine ont mis aux adresses \$FFFE-\$FFFF et \$FFFA-\$FFFB.

En ce qui concerne les Apple II+ et //e, on trouve en \$FFFE-\$FFFF les valeurs "40 FA". Une IRQ branche donc finalement sur une routine débutant en \$FA40, sur laquelle nous reviendrons un peu plus loin.

En \$FFFA-\$FFFB, on lit "FB 03" et une NMI nous amènera donc en \$3FB. Si vous mettez votre Apple sous tension sans charger le DOS, vous ne trouverez rien de significatif à cette adresse, mais si vous "bootez" le DOS, vous y trouverez "4C 65 FF" ou encore "JMP \$FF65", c'est-à-dire un saut dans le moniteur. C'est évidemment au programmeur de périphériques qui entend utiliser les interruptions NMI qu'il appartient en fait de mettre aux adresses \$3FB-\$3FC-\$3FD un JMP à l'adresse de sa routine de gestion des interruptions.

Instructions de l'assembleur liées aux interruptions

• **SEI (78)** : met l'indicateur I à 1
Les interruptions IRQ qui pourraient arriver après l'exécution de SEI seront donc masquées et ignorées. Ceci peut servir pour garder la priorité dans l'utilisation des ressources du microprocesseur (sauf en cas d'interruption NMI toutefois), que ce soit pour un programme utilisateur ou pour le travail d'un périphérique donné. Dans une routine de gestion des interruptions destinée aux divers périphériques, en particulier, l'emploi de SEI permet de sauvegarder les registres non directement empilés par le processeur (accumulateur et registres X et Y) ou de modifier éventuellement le vecteur IRQ (voir plus loin), sans craindre l'arrivée et la prise en compte d'une nouvelle interruption qui jetterait la confusion dans le système.

En effet, après l'exécution des routines de service du périphérique ayant demandé l'interruption, on doit pouvoir reprendre le programme interrompu exactement à l'endroit où il a été stoppé et dans le même contexte, ce qui suppose que l'on puisse bien sauvegarder tous les registres, au moyen de la pile notamment, afin de les récupérer ensuite en fin de routine de gestion des interruptions. Un autre signal IRQ éventuel doit rester par conséquent ignoré, au moins jusqu'à l'achèvement de cette sauvegarde. Si le signal persiste durant que l'indicateur I est à 1, il sera pris en compte à son tour dès que le "masque" sera levé par l'instruction suivante.

• **CLI (58)** : met à 0 l'indicateur I
Après l'exécution de CLI, tout signal IRQ est pris en compte et lance un nouveau cycle d'interruption. Rien n'interdit de traiter les demandes d'interruption d'un périphérique alors même que le processeur est déjà utilisé par un autre périphérique. Lorsque le second "rend la main", on reprend le travail du premier à l'endroit où il était stoppé et, lorsque celui-ci est également terminé, on retrouve le programme de départ. Ce mécanisme ne doit pas poser de problèmes si le contexte est correctement sauvegardé à chaque interruption.

• **RTI (40)** : retour d'interruption
L'exécution de cette instruction consiste pour le processeur à transférer dans le registre d'état le sommet de la pile, puis à charger le compteur

ordinal avec les deux octets qui se trouvent à leur tour au sommet de la pile. Si l'organisation de cette dernière n'a pas été perturbée durant le traitement de l'interruption, on reprend donc à l'endroit (compteur ordinal) où l'on se trouvait lors de la prise en compte de l'interruption, avec le même registre d'état.

Nous verrons par la suite comment la pile intervient dans la manipulation du compteur ordinal, mais, afin d'illustrer ces mécanismes, voici un exemple simplifié de routine de gestion des interruptions (notez bien que lorsqu'on arrive à cette routine, le registre d'état et le compteur ordinal ont déjà été empilés par le processeur) :

```
SEI
(on masque les interruptions IRQ qui
pourraient survenir)
PHA
TYA
PHA
TXA
PHA
```

(sauvegarde dans la pile des registres A, X et Y)
CLI
(les interruptions IRQ sont de nouveau autorisées)

Détermination du périphérique ayant provoqué l'interruption, en testant par exemple le contenu d'adresses affectées à chaque périphérique connecté.

```
JSR ADR
(ADR est l'adresse de début des routines de service du périphérique, qui lui permettent d'effectuer son travail. Ces routines doivent veiller à restituer, à l'issue de leur exécution par un ultime RTS, un contenu de pile identique à celui que l'on avait avant JSR ADR)
```

```
SEI
PLA
TAX
PLA
TAY
PLA
(restauration des registres A-X-Y, tels qu'ils étaient lors du début de notre routine de gestion des interruptions)
CLI
RTI
(reprise du programme interrompu)
```

Si une nouvelle interruption se produit avant la fin du premier cycle, alors que l'on vient de déterminer le périphérique par exemple (appelons cela le stade S0), le processeur nous ramène au début de la routine ci-dessus et on empile le contexte de S0. On traite ensuite la seconde interruption, puis on récupère le contexte de S0. Le RTI nous renvoie à l'instruction qui suit S0 et l'on reprend ainsi le traitement de la première interruption, avant de récupé-

rer le contexte initial et de retrouver un RTI qui nous reporte cette fois à la suite du programme de départ.

Lorsqu'une routine de ce type est utilisée pour gérer des interruptions NMI, son adresse doit être stockée en \$3FC-\$3FD. S'il s'agit de gérer des IRQ, l'adresse doit être implantée en \$3FE-\$3FF, comme le montrera le paragraphe suivant.

BREAK et simulation d'interruptions

L'instruction de break BRK (00) produit des effets semblables à ceux d'une interruption IRQ, ce en quoi elle ressemble à une interruption "logicielle", mais avec deux différences :

- Le processeur empile la valeur du compteur ordinal augmentée de 2 (comme si l'instruction BRK occupait deux octets).
- Le bit B du registre d'état est mis à 1 (alors qu'il est à 0 dans le cas d'une interruption) avant sauvegarde.

Ensuite, comme dans le cas d'une IRQ, on saute à l'adresse contenue en \$FFFE-\$FFFF, ce qui nous amène donc, sur un Apple II+ ou IIe, à la routine commençant en \$FA40, à qui il appartiendra de faire le distinguo entre BRK et IRQ. Cette routine est listée ci-dessous et nous allons en faire l'analyse.

Listing 1

FA40-	85 45	STA	\$45
FA42-	68	PLA	
FA43-	48	PHA	
FA44-	0A	ASL	
FA45-	0A	ASL	
FA46-	0A	ASL	
FA47-	30 03	BMI	\$FA4C
FA49-	6C FE 03	JMP	(\$03FE)
FA4C-	28	PLP	
FA4D-	20 4C FF	JSR	\$FF4C
FA50-	68	PLA	
FA51-	85 3A	STA	\$3A
FA53-	68	PLA	
FA54-	85 3B	STA	\$3B
FA56-	6C F0 03	JMP	(\$03F0)

Ligne 1 : on stocke l'accumulateur en \$45.

Lignes 2 et 3 : on récupère dans l'accumulateur le sommet de la pile (PLA), c'est-à-dire le registre d'état au moment du BRK ou de l'IRQ, puisque l'on arrive ici à l'issue du cycle d'interruption exécuté par le processeur. On ré-empile aussitôt cette même valeur (PHA) afin de préserver l'organisation de la pile.

Lignes 4 à 6 : les trois ASL amènent le bit 4 de l'accumulateur en position de bit 7. On a donc pour bit de poids fort, ou bit de signe, l'indicateur B du registre d'état au moment du BRK ou de l'IRQ.

Ligne 7 : test du bit de signe de l'accumulateur. Si celui-ci est à 1, cela signifie que l'indicateur B était à 1

(BRK) et BMI branchera en \$FA4C. Si le bit de signe est à 0, l'indicateur B était à 0 (IRQ) : le test BMI échoue et on passe à l'instruction suivante.

Ligne 8 : nous sommes dans le cas d'une interruption IRQ et l'on saute à l'adresse contenue en \$3FE-\$3FF, adresse devant correspondre au début de la routine de gestion des interruptions IRQ si elle existe. Si l'Apple est mis sous tension sans DOS, le vecteur \$3FE-\$3FF ne conduit "nulle part" : si le DOS est "booté", il conduit à \$FF65, soit au moniteur. Là encore, c'est au programmeur désirant exploiter les IRQ qu'il appartient de mettre en \$3FE-\$3FF l'adresse de sa routine de gestion des interruptions (routine du type de celle présentée plus haut).

Lignes 9 à 15 : nous sommes dans le cas d'un BRK. On récupère le registre d'état (PLP), on sauve les registres en \$45 à \$49 (JSR \$FF4C - routine de la ROM du moniteur), on sauve l'octet bas du compteur ordinal empilé lors du BRK en \$3A (PLA STA \$3A) et l'octet haut en \$3B (PLA - STA \$3B), puis on saute à l'adresse contenue en \$3F0-\$3F1. En standard, c'est à dire lorsque vous mettez l'Apple sous tension, \$3F0-\$3F1 contiennent "59 FA" : un BRK conduit donc finalement à la routine commençant en \$FA59, qui affiche le contenu des registres à l'écran et passe en mode moniteur, stoppant ainsi le déroulement du programme en cours.

Lorsque vous utilisez le break sous la forme que lui ont donnée les concepteurs de l'Apple, il peut surtout vous servir lors de la mise au point de vos propres routines en langage-machine. En effet, si tout ne se passe pas conformément à vos espérances, vous insérez un BRK dans le programme source en assembleur (soit un code 00 dans le programme objet) à l'endroit qui vous semble le plus significatif et vous en relancez l'exécution. En arrivant au BRK, le système stoppe le programme et affiche les registres, ce qui vous permet d'en examiner le contenu, ainsi que celui des adresses que vous utilisez, afin de vérifier s'il correspond aux valeurs que vous aviez prévues lors de l'étude de votre programme. Cela facilite bien souvent le diagnostic des erreurs.

Afin d'illustrer le mécanisme des interruptions, et surtout celui des retours d'interruptions, dont le break peut assurer une certaine simulation, nous allons maintenant "revectoriser" le BRK (c'est à dire modifier le contenu de \$3F0-\$3F1), pour qu'il ne conduise plus en \$FA59 mais à une routine que nous implanterons nous-mêmes et qui, puisqu'il s'agit d'un exercice de style, rendra simple-

ment BRK transparent. En d'autres termes, nous ferons en sorte que BRK au milieu d'un programme n'exerce plus aucune action sur son déroulement (sinon au niveau des temps d'exécution) et permette le passage à l'instruction suivante. Cette routine, listée ci-dessous, débute à l'adresse \$300.

Listing 2

```
0300- A6 3A LDX $3A
0302- A4 3B LDY $3B
0304- CA DEX
0305- 10 01 BPL $0308
0307- 88 DEY
0308- 98 TYA
0309- 48 PHA
030A- 8A TXA
030B- 48 PHA
030C- A5 48 LDA $48
030E- 48 PHA
030F- 20 3F FF JSR $FF3F
0312- 40 RTI
```

Lignes 1 et 2 : nous arriverons toujours en \$300 après être passé par \$FA40, qui stocke le compteur ordinal en \$3A-\$3B. On peut donc le récupérer ici (octet bas dans le registre X et octet haut dans le registre Y).

Lignes 3 à 5 : pour reprendre le programme en cours (celui dans lequel on a inséré un break) à l'instruction qui suit BRK il faut rectifier la valeur de ce compteur ordinal en le diminuant de 1 (décrémenter de l'octet bas, puis de l'octet haut si nécessaire).

Lignes 6 à 9 : on empile le compteur ordinal rectifié, dans l'ordre octet haut / octet bas.

Lignes 10 et 11 : le passage en \$FA40 a également provoqué la sauvegarde du registre d'état au moment du break à l'adresse \$48. On va donc le relire et le déposer au sommet de la pile. Cette dernière présente maintenant la même configuration qu'à l'issue d'une interruption.

Ligne 12 : \$FF3F est le début d'une routine du moniteur qui recharge les registres avec les valeurs lues en \$45 à \$48 (le passage en \$FA40 a permis la sauvegarde de ces mêmes registres dans ces mêmes adresses). On retrouve ainsi le contenu des registres au moment du break dans le programme initial.

Ligne 13 : avec l'instruction RTI, on simule un retour d'interruption et, la pile ayant été correctement reconfigurée, on doit retourner au programme principal, juste après l'instruction BRK.

Pour mettre cette routine en mémoire, vous pouvez la charger à partir de la disquette d'accompagnement ou la rentrer directement en mode moniteur en tapant :

```
CALL -151
300: A6 3A A4 ... 40
```

Pour en faire la nouvelle routine de

break, il faut ensuite en donner l'adresse en \$3F0-\$3F1. Passez en mode moniteur si vous n'y êtes pas déjà et tapez :

```
3F0: 00 03
```

La routine listée ci-dessous nous servira de test. En principe, elle doit exécuter un HOME et afficher un A en haut et à gauche de l'écran. Etant donné que nous avons mis des BRK entre chaque instruction, cela ne sera effectivement possible que si notre routine en \$300 rend bien le BRK transparent. Pour le vérifier, chargez cette seconde routine ou rentrez-la directement en mémoire et lancez-en l'exécution par "2000G" à partir du moniteur ou "CALL 8192" à partir du Basic. Sauf erreur de frappe, vous devriez obtenir le résultat escompté. Cela vous donnera une petite idée de la façon dont les interruptions peuvent détourner l'usage du processeur au profit de périphériques et permettre néanmoins une reprise correcte du déroulement des programmes interrompus.

Listing 3

```
2000- 20 58 FC JSR $FC58
2003- 00 BRK
2004- A9 C1 LDA #$C1
2006- 00 BRK
2007- 20 F0 FD JSR $FDF0
200A- 00 BRK
200B- 60 RTS
```

Une instruction qui ne fait rien...

S'il ne s'agit que de disposer d'une instruction transparente dans un programme, il n'est pas vraiment nécessaire (ce serait même plutôt inutile) de revectoriser le BRK comme ci-dessus, car le 6502 vous offre en standard une telle instruction : **NOP** (EA).

NOP ne réalise aucun traitement, ne modifie aucun registre, aucune adresse et se contente de réclamer un peu de temps pour son exécution. Elle n'en est pas totalement inutile pour autant :

- Lorsque vous testez une routine qui vous pose problème, vous êtes amené bien souvent, dans un premier temps du moins, à travailler directement au niveau du code objet en mode moniteur. Si vous remplacez une instruction par une autre plus courte (comportant moins d'octets) ou si vous voulez effacer une petite séquence d'instructions qui vous semble douteuse, vous aurez besoin d'une instruction valide, ne provoquant pas la colère du processeur, mais transparente pour boucher les "trous". Vous remplirez donc ces derniers avec des codes EA (nous en verrons un petit exemple plus loin).

- NOP ne fait rien, certes, mais elle n'en est pas moins traitée par le processeur, ce qui lui prend nécessairement un peu de temps. NOP permet donc d'insérer des délais dans l'exécution d'un programme, aux endroits voulus (pour ralentir un affichage à l'écran, par exemple, ou ne pas "affoler" la mécanique des lecteurs de disquettes...).

Le décimal codé binaire

Autre indicateur du registre d'état laissé de côté dans le Pom's 13, le bit 3 ou indicateur "décimal codé binaire" (DCB en abrégé), mérite maintenant quelques développements.

Le DCB est une technique de représentation des nombres correspondant à une version tronquée de l'hexadécimal, puisque l'on ne représentera que les chiffres décimaux 0 à 9, au lieu de 0 à F en hexadécimal. Comme toujours, revenons un peu au binaire pour clarifier les choses.

Sur quatre bits, on peut coder les valeurs comprises entre 0000 et 1111, soit 0 à 15 en "décimal" ou encore 0 à F selon la notation hexadécimale. Si l'on se borne à coder les chiffres décimaux, on utilise les "combinaisons" 0000 (0) à 1001 (9), mais pas les suivantes (1010 à 1111) qui ne correspondent pas à des chiffres décimaux. En hexadécimal, un octet permet de représenter des valeurs comprises entre 00 et FF (0 à 255), mais en DCB, du fait des combinaisons éliminées, on ne peut représenter que les nombres entre 00 et 99.

Employer la notation DCB permet tout d'abord de retrouver notre système décimal classique, avec lequel nous avons normalement plus de communauté de langage qu'avec l'hexadécimal, ainsi, si l'on ajoute 1 à 09, on trouvera 10 et non 0A. Par contre, cela suppose que soient bien éliminées dans les calculs les combinaisons illicites, sous peine de graves incohérences dans les résultats.

Le processeur 6502 dispose d'un mode de calcul en DCB, dans lequel il se charge lui-même de limiter les combinaisons de bits aux seuls chiffres décimaux. Pour sélectionner ce mode, il suffit de mettre l'indicateur DCB du registre d'état à 1 au moyen de l'instruction **SED** (F8). On revient au système hexadécimal "naturel" en mettant cet indicateur à 0 au moyen de l'instruction réciproque **CLD** (D8).

Vous trouverez ci-dessous un petit exemple d'addition en DCB, dont nous vous laissons conduire l'analyse à l'aide des précisions suivantes :

- Avant de faire appel à des routines du système (moniteur ou Basic),

il est toujours prudent de revenir à l'hexadécimal, car ces routines sont prévues pour fonctionner avec des valeurs hexa et l'on obtient souvent des résultats bizarres si l'on s'y branche en mode DCB. Ceci explique la présence de CLD avant le premier JSR \$FDDA.

– \$FDDA est une routine du moniteur qui affiche le contenu de l'accumulateur sur deux chiffres. \$FD8E envoie un RETURN et \$FD1B attend la frappe d'une touche au clavier.

Si vous lancez cette routine (à partir du moniteur par 300G, par exemple), vous verrez s'afficher 0001 puis le curseur. Après avoir tapé une touche, vous verrez 0002 et ainsi de suite jusqu'à 9999 si vous en avez la patience. Après 9999, on repart à 0000. Notez bien que le calcul passe de 0009 à 0010 puis 0011, de 0019 à 0020 puis 0021...

Listing 4

```

0300- A9 00 LDA #000
0302- 85 07 STA #07
0304- 85 06 STA #06
0306- F8 SED
0307- 18 CLC
0308- A5 06 LDA #06
030A- 69 01 ADC #01
030C- 85 06 STA #06
030E- A5 07 LDA #07
0310- 69 00 ADC #00
0312- 85 07 STA #07
0314- D8 CLD
0315- 20 DA FD JSR $FDDA
0318- A5 06 LDA #06
031A- 20 DA FD JSR $FDDA
031D- 20 8E FD JSR $FD8E
0320- 20 1B FD JSR $FD1B
0323- 4C 06 03 JMP #0306

```

Le mode DCB peut ainsi servir justement à faire un affichage simple d'un compteur en décimal, en vous évitant d'avoir à convertir les résultats avant de les afficher. En ce qui concerne les additions, le processeur peut même vous réserver d'agréables surprises. Remplacez par exemple le 00 de l'adresse \$301 par 0F et relancez la routine, vous verrez apparaître :

1516
1517

Le processeur a corrigé tout seul votre erreur et converti votre \$0F en 15 DCB !

Les soustractions, en revanche, n'offrent pas les mêmes garanties. Transformons notre programme d'addition en programme de soustraction en remplaçant CLC (18) par SEC (38) et les deux ADC (69) par des SBC (E9). Si l'on commence par LDA #00, tout se passe bien et le calcul donne :

9999
9998
9997

Si l'on commence par LDA #\$0F, par contre, on obtient :

0FOE

0F0D
0F0C

Le processeur va calculer ainsi en hexadécimal jusqu'à ce que le contenu de l'octet revienne dans les chiffres décimaux (on a par exemple 0F02 - 0F01 - 0F00 - 0E99 - 0E98). Le résultat d'ensemble n'est donc pas particulièrement cohérent...

Outre les précautions d'emploi qu'il réclame, le DCB présente un inconvénient au niveau de la consommation en mémoire. Deux octets permettent de coder les nombres jusqu'à 65535 (\$FFFF) en hexadécimal, mais jusqu'à 9999 seulement en DCB. Quelle que soit la valeur maximale que l'on désire représenter, il faudra toujours plus d'octets pour le faire en DCB qu'en hexadécimal.

Citons rapidement d'autres petits problèmes relatifs au DCB : l'élimination des combinaisons de bits illicites après des décalages ou des rotations, la disparition de la notion de "bit de signe"...

Finalement, tout cela ne fait pas de la notation DCB un système particulièrement sympathique. Vous pouvez l'utiliser sans grand risque pour l'affichage direct en décimal des résultats de calculs simples, mais au delà, prudence...

La pile : suite et fin

Depuis le Pom's 12, nous avons souvent parlé de la pile du microprocesseur, sans rentrer dans le détail de son organisation car cela n'était pas nécessaire pour la manipulation des instructions présentées à cette occasion. Il nous suffisait de savoir que l'on peut déposer quelque chose à son sommet, l'y reprendre ensuite et que le processeur l'utilisait également pour mémoriser l'adresse de retour après un JSR. Nous avons vu plus haut le rôle qu'elle joue en outre dans les mécanismes de break et d'interruptions. En fait, elle fournit souvent la solution de bien des problèmes de programmation dès lors que l'on sait plus précisément comment elle fonctionne et nous allons donc faire plus ample connaissance avec elle.

La pile est une zone de "stockage" de 256 octets qui se trouve en mémoire aux adresses \$100 (256) à \$1FF (511) - c'est la page 1 de la mémoire RAM de l'Apple. Première remarque : la pile se situe donc dans une zone mémoire qui vous est tout aussi accessible que la zone de travail de l'Applesoft, ou des adresses comme \$6 ou \$7 que nous avons souvent utilisées pour nos routines. Conséquence, ne POKEz jamais quelque chose entre 256 et 511 sans mesurer précisément à l'avance les risques de cette entreprise...

Pour gérer sa zone de stockage, le 6502 dispose d'un pointeur (baptisé S), qui, au contraire de la pile elle-même, est un registre interne au même titre que l'accumulateur ou le compteur ordinal, par exemple. Ce pointeur correspond à un octet, peut donc varier de 0 à 255 (\$0 à \$FF) et indique toujours la première position libre dans la pile, c'est-à-dire son sommet. Particularité du sommet : il s'élève vers le bas car la pile se remplit de l'adresse \$1FF vers l'adresse \$100 et non l'inverse. Lorsque la pile est totalement vide, le pointeur de pile S contient \$FF et le premier octet empilé le sera à l'adresse "\$100 + valeur de S", soit \$1FF. Après cet "empilage", S contient \$FE et la prochaine valeur à empiler le sera à l'adresse \$1FE. Chaque fois que l'on empile un octet, S diminue de 1 et chaque fois que l'on dépile un octet, S augmente de 1.

Il n'existe pas d'instructions pour manipuler directement ce pointeur de pile (du type LDS ou STS), mais on peut néanmoins le faire par l'intermédiaire du registre X grâce aux instructions :

- **TSX (BA)** : copie le contenu du pointeur de pile dans X sans changer la valeur de S
- **TXS (9A)** : copie le contenu de X dans le pointeur de pile

JSR, RTS et pile

Une instruction JSR suivie de l'adresse de la sous-routine concernée occupe toujours 3 octets (par exemple, JSR \$B1 = 20 00 B1 et JSR \$FDED = 20 ED FD). Avant d'exécuter une instruction, le processeur calcule l'adresse de l'instruction suivante et la range dans le compteur ordinal. L'adresse de l'instruction qui suit un JSR est donc "adresse du JSR + 3" et elle correspond à l'endroit où l'on doit revenir après le déroulement d'une sous-routine se terminant par RTS.

Cette adresse de retour est déposée au sommet de la pile, mais elle correspond en fait à l'adresse réelle diminuée de 1 (soit adresse du JSR + 2), car l'exécution de RTS augmente de 1 le compteur ordinal (le processeur prend les deux octets du sommet de la pile pour les mettre dans le compteur ordinal, augmente ce dernier de 1 et passe ensuite à l'instruction dont l'adresse lui est donnée par la nouvelle valeur du compteur).

Nous allons voir comment cette interaction entre JSR et la pile est utilisée par les cartes d'interface pour déterminer le slot dans lequel elles se trouvent et savoir ainsi quelles adresses de travail leur sont réservées par l'Apple.

Prenons l'exemple d'un contrôleur de disquette DOS 3.3, dont le listing

ci-dessous reproduit un court extrait de la ROM. Notez bien que l'adresse de départ \$C621 provient du fait que le contrôleur se trouve dans le slot 6 au moment de cette liste. Les ROMs de l'interface sont conçues de telle façon qu'elles puissent fonctionner quel que soit le slot dans lequel on les installe et, si le contrôleur était dans le slot 5, par exemple, on trouverait exactement la même chose à partir de l'adresse \$C521.

Listing 5

```
C621- 20 58 FF JSR $FF58
C624- BA TSX
C625- BD 00 01 LDA $0100,X
C628- 0A ASL
C629- 0A ASL
C62A- 0A ASL
C62B- 0A ASL
C62C- 85 2B STA $2B
C62E- AA TAX
C62F- BD 8E C0 LDA $C08E,X
C632- BD 8C C0 LDA $C08C,X
C635- BD 8A C0 LDA $C08A,X
C638- BD 89 C0 LDA $C089,X
C63B- A0 50 LDY ##50
C63D- BD 80 C0 LDA $C080,X
```

– \$C621 : l'adresse \$FF58, située dans la ROM du moniteur, contient la valeur #50, c'est-à-dire le code de l'instruction RTS. Un JSR \$FF58 nous ramène donc immédiatement à la suite du JSR mais, entre-temps, l'adresse de retour diminuée de 1 a été déposée au sommet de la pile, dans l'ordre octet haut / octet bas. En l'occurrence, on avait donc "23" au sommet de la pile et "C6" en dessous (adresse de retour = \$C624, soit \$C623 après déduction de 1), avant le RTS de l'adresse \$FF58. Le RTS dépile cette adresse dans le compteur ordinal et augmente le pointeur de pile de 2 mais, aussi longtemps que l'on n'empile rien d'autre, les deux octets du sommet de la pile sont toujours les mêmes. Le registre S pointe sur la position où se trouve "C6" pour indiquer que c'est la première position libre, mais il y a encore pour l'instant à cette position "C6" et "23" à la position supérieure.

– \$C624 : on transfère la valeur du pointeur de pile dans le registre X par TSX.

– \$C625 : l'instruction "LDA \$100,X" permet de charger l'accumulateur avec le contenu de la première position libre de la pile (sans modifier le pointeur de pile, bien sûr) et l'on récupère ainsi "C6" dans l'accumulateur.

– \$C628 à \$C62B : après quatre décalages à gauche (ASL) le contenu de l'accumulateur devient "60". Cette valeur sera ensuite transférée dans le registre X et elle permettra d'accéder aux adresses réservées à une carte d'interface placée dans le slot 6.

Le travail effectué par cette routine

peut vous paraître inutile puisque nous savons depuis le départ que la carte se trouve dans le slot 6 et qu'il suffisait donc de charger X avec #50 pour obtenir le même résultat. Mais le programmeur qui a développé les routines de la ROM du contrôleur ne pouvait savoir dans quel slot vous alliez le mettre et voulait en outre que ces routines fonctionnent indépendamment de ce slot (si vous utilisez le slot 5, le registre X devra contenir #50 au lieu de #60). Etant donné que la ROM est écrite "une fois pour toutes" et que l'on ne peut remplacer "LDX #60" par "LDX #50" lorsqu'on l'utilise, la solution consiste donc à employer un mécanisme "JSR - RTS" pour aller lire ensuite la pile et savoir si l'on a envoyé le JSR à partir d'une adresse C6nn, C5nn, C4nn ou autre.

Pour détailler un peu plus ce système, vous trouverez ci-après la liste et le compte-rendu d'exécution d'une petite routine utilisant le même principe et permettant de vérifier concrètement le contenu de la pile.

Listing 6

```
0300- A2 E0 LDX ##E0
0302- 9A TXS
0303- 20 58 FF JSR $FF58
0306- BA TSX
0307- 86 06 STX $06
0309- BD 00 01 LDA $0100,X
030C- 85 07 STA $07
030E- CA DEX
030F- BD 00 01 LDA $0100,X
0312- 85 08 STA $08
0314- A2 40 LDX ##40
0316- 9A TXS
0317- 00 BRK
0318- 00 BRK
0319- 00 BRK
031A- 00 BRK
031B- 00 BRK
031C- 00 BRK
031D- 00 BRK
031E- 00 BRK
*300G
```

```
0319- A=05 X=40 Y=08 P=30 S=3C
*6.8
```

```
0006- E0 03
0008- 05
*108.1E7
```

```
0108- 6F BE 00 00 30 30 31 05
01E0- 03 30 00 00 00 00 0A
```

On initialise le pointeur de pile à #E0, on exécute le JSR \$FF58 puis on récupère la valeur de S et on la stocke en \$6. On lit ensuite la première position libre et on la stocke en \$7. En décrémentant X (la pile se remplit vers le bas), on lit ensuite l'octet qui se trouve "au dessus" de la première position libre et on le range en \$8. Puis on initialise S à #40, afin de préserver les valeurs que nous avons nous-mêmes empilées (en d'autres termes, on remonte le sommet de la pile loin de l'endroit que nous avons manipulé), et on

stoppe le programme par un BRK standard de l'Apple.

On trouve ainsi aux adresses \$6 à \$8 :

– E0 : le pointeur de pile est bien revenu à sa valeur initiale \$E0 après le RTS de l'adresse \$FF58.

– 03 : octet haut de l'adresse de retour du JSR

– 05 : octet bas de l'adresse de retour du JSR, diminué de 1.

En lisant le contenu de la pile entre les adresses \$1D8-\$1E7, on peut vérifier la présence de "03" à l'adresse \$1E0 (S = \$E0) et de "05" à l'adresse \$1DF (S = \$DF, ce qui constituait bien le sommet de la pile avant RTS).

Manipulation du pointeur de pile

Nous en avons vu un exemple ci-dessus (initialisation à #E0 ou à #40).

En changeant la valeur du pointeur de pile, on modifie l'endroit où sont empilés les octets par la suite, et l'on modifie également les valeurs lues lorsque le processeur dépile quelque chose. Cela est vrai qu'il s'agisse de dépiler dans l'accumulateur par PLA, par exemple, ou qu'il s'agisse de l'adresse de retour d'un JSR.

La routine suivante utilise la manipulation du pointeur de pile pour réaliser l'équivalent d'un POP en langage machine. L'exemple choisi n'a qu'une valeur démonstrative car le traitement effectué ne nécessite nullement pareille complication.

Listing 7

```
0300- 20 58 FC JSR $FC58
0303- A9 C1 LDA #$C1
0305- 20 0C 03 JSR $030C
0308- 20 F0 FD JSR $F0F0
030B- 60 RTS
030C- 2F 7F AND ##7F
030E- BA TSX
030F- E8 INX
0310- E8 INX
0311- 9A TXS
0312- 4C 08 03 JMP $0308
```

La routine effectue un HOME puis affiche un A en mode FLASH. La partie qui nous concerne se situe dans la sous-routine débutant en \$30C.

On met le bit 7 de l'accumulateur à 0 pour modifier le code-écran auquel il correspond et on augmente ensuite le pointeur de pile de 2 (TSX - INX - INX - TXS). Par cette opération, on déplace le sommet de la pile "en-dessous" de l'adresse de retour du JSR \$030C. Le fait de revenir alors de la routine par un JMP \$0308 ne perturbe pas le déroulement du programme car, ayant fait "sauter" une adresse de retour, il n'y a pas réellement plus de JSR que de RTS.

Annulez maintenant la manipulation du pointeur de pile en tapant, en mode moniteur :

30E: EA EA EA EA

ce qui remplace les instructions précédentes par des NOP. Relancez la routine et vous verrez cette fois s'afficher deux A en FLASH au lieu d'un. En voici l'explication :

- Au JSR \$030C, l'adresse \$307 est empilée.

- Comme il n'y pas de RTS dans la routine \$30C, et que l'on revient par JMP \$308, l'adresse \$307 constitue toujours le sommet de la pile.

- La routine débutant en \$FDF0 se termine par un RTS. Le processeur récupère le sommet de la pile, soit \$30A après le JSR \$FDF0, et l'utilise comme base de l'adresse de retour; on revient en \$30B.

- En \$30B, on trouve un RTS. Le processeur reprend encore le sommet de la pile et "tombe" cette fois sur \$307 qui y est remonté lorsque \$30A a été dépilé. On revient de ce fait en \$308, ce qui provoque l'affichage d'un deuxième caractère à côté du premier. Le mécanisme des RTS se fait ensuite normalement puisqu'il n'y a plus d'adresse excédentaire dans la pile.

Gestion de tables

Supposons que vous ayez plusieurs tables d'octets à gérer à l'intérieur d'un programme, telles par exemple des tables contenant les codes des caractères de messages à afficher à l'écran. Malheureusement, vous ne savez pas exactement combien de

ces tables vous seront nécessaires, ni à quels endroits elles se trouveront dans le programme. Comme toujours, plusieurs solutions sont envisageables pour résoudre les problèmes de programmation que pose cette situation. L'une d'elle consiste là encore à utiliser la pile, comme le montre le programme suivant, que nous allons analyser.

Ligne 4, elle correspond aux codes écran du message MESSAGE 1, suivis d'un 00 pour en marquer la fin.

Ligne 5: codes écran de MESS 2, suivis de 00.

L'affichage se fera à chaque fois par un JSR S0 placé avant les codes composant le message.

Lignes 8 à 12: on transfère le pointeur de pile dans X pour lire les deux octets du sommet de la pile, c'est-à-dire l'adresse de retour du JSR S0 diminuée de 1. On prend pour adresses de base \$101 et \$102 au lieu de \$100 et \$101, puisque S pointe vers la première position libre, et donc vers l'octet qui se trouve au dessus de l'adresse qui nous intéresse. On stocke cette adresse en \$6-\$7 pour qu'elle puisse nous servir ensuite d'adresse de base dans un indexage indirect par Y.

- Ligne 13: on initialise Y à 1, pour tenir compte du fait que l'adresse placée en \$6-\$7 ne correspond pas exactement à l'adresse du début de notre table de codes à afficher, mais

à cette adresse diminuée de 1.

- Lignes 14 à 16: incrémentation de l'adresse de retour située au sommet de la pile. Il s'agit en effet de ne pas revenir de notre routine S0 à l'adresse qui suit JSR S0 (début de la table), mais à l'adresse qui suit le 00 marquant la fin des codes.

- Lignes 17 et 18: lecture d'un code dans la table. Si c'est 00, on passe en S1 qui fait afficher un RETURN (JSR \$FC62) et nous amène en fin de routine (RTS). Si l'adresse de retour a bien été augmentée à l'intérieur même de la pile, on doit alors reprendre le programme principal à l'instruction qui suit la table des codes.

- Lignes 19 à 21: affichage du caractère (JSR \$FDF0) et passage au suivant (INY - BNE S3).

Avec un tel système, l'ajout de nouveaux messages en cours de développement du programme ne pose guère de problèmes. Il suffit d'insérer à l'endroit voulu dans le programme-source un "JSR S0" suivi du message à afficher, sans se préoccuper de l'adresse à laquelle commencera la table des codes des caractères à l'issue de l'assemblage final.

Remercions donc tous ensemble Thierry Le Tallec qui, par le biais de l'une de ses contributions, nous a donné l'idée de cette méthode pour illustrer les bienfaits de la gestion de pile. ■

0300		1	ORG \$300
0300	2058FC	2	JSR \$FC58
0303	201B03	3	JSR S0
0306	CDC5D3	4	HEX CDC5D3D3C1C7C5A0B100
0309	D3C1C7		
030C	C5A0B1		
030F	00		
0310	201B03	5	JSR S0
0313	CDC5D3	6	HEX CDC5D3D3A0B200
0316	D3A0B2		
0319	00		
031A	60	7	RTS
031B	BA	8	TSX
031C	BD0101	9	LDA \$101,X
031F	8506	10	STA \$6
0321	BD0201	11	LDA \$102,X
0324	8507	12	STA \$7
0326	A001	13	LDY #1
0328	FE0101	14	INC \$101,X
032B	D003	15	BNE S2
032D	FE0201	16	INC \$102,X
0330	B106	17	LDA (\$6),Y
0332	F006	18	BEQ S1
0334	20F0FD	19	JSR \$FDF0
0337	C8	20	INY
0338	D0EE	21	BNE S3
033A	2062FC	22	JSR \$FC62
033D	60	23	RTS

SOURCE LISA 1.5

Récapitulation

0300-	20	58	FC	20	1B	03	CD	C5
0308-	D3	D3	C1	C7	C5	A0	B1	00
0310-	20	1B	03	CD	C5	D3	D3	A0
0318-	B2	00	60	BA	BD	01	01	85
0320-	06	BD	02	01	85	07	A0	01
0328-	FE	01	01	D0	03	FE	02	01
0330-	B1	06	F0	06	20	F0	FD	C8
0338-	D0	EE	20	62	FC	60		

HPGRAPH est un ensemble de procédures et de fonctions Pascal permettant de simuler une partie des puissantes instructions graphiques du Hewlett-Packard HP 9845. Ces instructions se retrouvent également sur les micros HP, HP 85 et 87. Elles ont été définies sous la forme d'une INTRINSIC UNIT Pascal. Il faut donc la placer dans la SYSTEM.LIBRARY pour pouvoir l'utiliser.

Tout programme utilisant les procédures HPGRAPH doit commencer comme suit :

```
PROGRAM xxxx;  
(*$S+*) (*option Swapping*)  
USES TURTLEGRAPHICS,  
TRANSCEND, HPGRAPH;
```

Voyons maintenant les instructions possibles.

Définition du format de l'écran

SCALE (XMIN, XMAX, YMIN, YMAX :REAL).

Cette procédure définit les bornes de l'écran : le point placé en bas à gauche a pour coordonnées (XMIN, YMIN) et le point situé en haut à droite (XMAX, YMAX). Tout l'écran est utilisé; il y a donc un certain aplatissement des figures. SCALE est fixé par défaut à l'échelle réelle de l'écran, soit SCALE (0, 279, 0, 191).

SHOW (XMIN, XMAX, YMIN, YMAX :REAL).

Cette procédure définit un carré dans lequel seront tracés les dessins. Par exemple, si l'on fait un SHOW (-3.14, 3.14, -1, 1), la zone dans laquelle les dessins seront tracés sera (-3.14, 3.14, -3.14, 3.14). Cette instruction est utile pour obtenir, par exemple, des tracés de cercles qui ne soient pas ovales, c'est-à-dire compressés. L'écran utilisable, en coordonnées réelles, est donc (0, 191, 0, 191).

LIMIT (XMIN, XMAX, YMIN, YMAX :REAL).

Cette procédure est l'équivalent HP de l'instruction VIEWPORT du TURTLEGRAPHICS, mais ici les coordonnées sont dans le format spécifié par SCALE. Par défaut, LIMIT est fixé aux mêmes valeurs que SCALE.

Instructions AXES et FRAMES

FRAME est une instruction sans paramètres qui permet de mettre un cadre autour de l'écran.

AXES (XTIC, YTIC, XINTER, YINTER :REAL).

Comme son nom l'indique, cette instruction permet de tracer des axes sur l'écran. Les axes se croisent au point (XINTER, YINTER) et sont gradués avec une graduation tous les XTIC sur l'axe (x'x) et une tous les YTIC sur l'axe (y'y). Cette procédure sert notamment lors du tracé de courbes.

Instructions de tracé

PEN (COLOR :SCREENCOLOR).

Cette instruction permet de choisir une couleur parmi les couleurs du Pascal. Exemple : PEN (GREEN). Par défaut, la couleur choisie est le WHITE.

PENUP.

Cette procédure, comme son nom l'indique, lève la plume, c'est-à-dire que les prochaines instructions CMOVE, CDRAW, PLOT, RPLLOT et IPILOT seront juste un déplacement, sans tracé.

PDIR (ALPHA :REAL).

Cette procédure est l'une des plus puissantes du Basic graphique HP. Elle réalise une rotation des axes de l'angle ALPHA, suivant le sens trigonométrique. ALPHA est exprimé en radians, et peut être négatif. Toutes les instructions de tracé sont réalisées dans le repère spécifié par SCALE et PDIR. PDIR(0), la valeur par défaut, correspond à un repère orthogonal classique. On trouvera un exemple d'utilisation de cette instruction dans le programme ROSE disponible sur la disquette d'accompagnement.

PLOT (X, Y :REAL; MODE:INTEGER);

Cette instruction réalise un déplacement ou un tracé jusqu'au point de coordonnées (X, Y). Le MODE est un paramètre obéissant aux règles suivantes : s'il est pair, la plume "descend" (PENDOWN) et une ligne sera engendrée. S'il est impair, la plume "monte" (PENUP) et un déplacement sera effectué. Si MODE est positif, la plume changera d'état après l'exécution du PLOT, sinon avant son exécution. Par exemple, MODE = 1 réalise soit un tracé soit un déplacement selon l'état antérieur de la plume, puis baisse la plume (le prochain PLOT sera donc un tracé). MODE = 2 ou MODE = 0 effectue un tracé ou un déplacement, puis lève la plume (le prochain PLOT sera un déplacement). MODE = -2 lève la plume puis effectue le dépla-

cement; MODE = -1 baisse la plume et effectue le tracé.

CMOVE (X, Y :REAL)

Correspond à l'instruction MOVE HP, qui n'a pas pu être transcrite sous le même nom puisqu'une procédure MOVE existe déjà dans le TURTLEGRAPHICS. Elle réalise l'équivalent d'un PLOT(X, Y, -2).

CDRAW (X, Y :REAL)

Correspond à l'instruction DRAW HP. Elle réalise l'équivalent d'un PLOT(X, Y, -1).

IPILOT (X, Y :REAL; MODE:INTEGER).

Cette procédure effectue un déplacement ou un tracé, suivant le MODE, en coordonnées relatives; les coordonnées d'origine sont les coordonnées du dernier PLOT, CDRAW, CMOVE ou RPLLOT. Exemple : tracé d'un carré de centre (0,0) et de côté de longueur 60 :
SCALE (-60, 60, -60, 60);
CMOVE (-30, -30);
IPILOT(60, 0, -1); IPILOT(0, 60, -1); IPILOT(-60, 0, -1); IPILOT(0, -60, -1);

RPILOT (X, Y :REAL; MODE:INTEGER).

Cette instruction est également une des instructions clefs du Basic graphique HP. Elle réalise un tracé ou un déplacement relatif mais, contrairement à IPILOT, le point origine ne varie pas après le RPILOT. Ce point origine est la dernière position de la plume après une instruction PLOT, CDRAW, CMOVE, FRAME, AXES ou IPILOT. Le MODE obéit aux règles habituelles. Un exemple : tracé d'un carré de centre (0,0) et de côté de longueur 60 (dans le repère défini par SCALE) :
SCALE (-50, 50, -50, 50);
CMOVE (0, 0);
RPILOT (-30, -30, -2);
RPILOT(30, -30, -1); RPILOT(30, 30, -1); RPILOT(-30, 30, -1);
RPILOT (-30, -30, -1);

Conclusion

L'unité graphique HPGRAPH est formée de procédures aisément compréhensibles, permettant de traduire facilement en Pascal des programmes graphiques d'origine HP. Ils permettent également d'écrire soi-même facilement tous les programmes nécessitant des formatages d'écran et des rotations d'axes. Il est malheureusement assez lent, par rapport aux instructions Basic d'HP (environ une fois et demi plus lent que

sur le HP 85 pour le tracé de la ROSE). Il serait très facile d'ajouter quelques instructions HP qui lui

manquent, comme GRID (quadrillage de l'écran), les instructions de saisie automatique sur l'écran graphi-

que (DIGITIZE), de sauvegarde sur disque d'écrans graphiques (GLOAD et GSTORE)...

Programme HPGRAPH

```
(*S+*)
UNIT HPGRAPH;
INTRINSIC CODE 16 DATA 17;

INTERFACE
  (*C N.Monsarrat septembre 1984)

  USES TURTLEGRAPHICS,TRANSCEND;
  PROCEDURE FRAME;
  PROCEDURE PENUP;
  PROCEDURE PEN (COLOR :SCREENCOLOR);
  PROCEDURE PDIR (ALPHA :REAL);
  PROCEDURE CMOVE (X,Y:REAL);
  PROCEDURE CDRAW (X,Y:REAL);
  PROCEDURE PLOT (X,Y:REAL;MODE :INTEGER);
  PROCEDURE RPLLOT (X,Y:REAL;MODE :INTEGER);
  PROCEDURE IPLOT (X,Y:REAL;MODE :INTEGER);
  PROCEDURE LIMIT (XMIN,XMAX,YMIN,YMAX :REAL);
  PROCEDURE SCALE (XMIN,XMAX,YMIN,YMAX :REAL);
  PROCEDURE SHOW (XMIN,XMAX,YMIN,YMAX :REAL);
  PROCEDURE AXES (XTIC,YTIC,XINTER,YINTER :REAL);

IMPLEMENTATION
VAR COSAL,SINAL,CURX,CURY,
    PX,PY,DX,DY :REAL;
    HCOLOR :SCREENCOLOR;
    PENDOWN :BOOLEAN;
    LXMIN,LXMAX,
    LYMIN,LYMAX :INTEGER;

FUNCTION FX(X :REAL) :INTEGER;
BEGIN
  FX := ROUND((X-PX)*DX);
END;

FUNCTION FY(Y :REAL) :INTEGER;
BEGIN
  FY := ROUND((Y-PY)*DY);
END;

PROCEDURE FRAME;
VAR OX,OY :INTEGER;
BEGIN
  OX := TURTLEX;OY := TURTLEY;
  PENCOLOR(NONE);MOVETO(LXMIN,LYMIN);
  PENCOLOR(HCOLOR);MOVETO(LXMAX,LYMIN);
  MOVETO(LXMAX,LYMAX);MOVETO(LXMIN,LYMAX);
  MOVETO(LXMIN,LYMIN);PENCOLOR(NONE);
  MOVETO(OX,OY)
END;

PROCEDURE PENUP;
BEGIN
  PENDOWN := FALSE
END;

PROCEDURE PDIR;
BEGIN
  COSAL := COS(ALPHA);
  SINAL := SIN(ALPHA)
END;

PROCEDURE PEN;
BEGIN
  HCOLOR :=COLOR;
  PENDOWN := TRUE
END;

PROCEDURE CMOVE;
BEGIN
  CURX := X; CURY := Y;
  PENCOLOR(NONE);MOVETO(FX(X),FY(Y));
END;

PROCEDURE CDRAW;
BEGIN
  CURX := X; CURY := Y;
  PENCOLOR(HCOLOR);MOVETO(FX(X),FY(Y))
END;

PROCEDURE PLOT;
BEGIN
  CASE MODE OF
    1 :BEGIN
      IF PENDOWN=FALSE THEN CMOVE(X,Y);
      CDRAW(X,Y);PENDOWN := TRUE
      END;
    2,0:BEGIN
```

```
      IF PENDOWN=FALSE THEN CMOVE(X,Y);
      CDRAW(X,Y);PENDOWN := FALSE
      END;
    -2 :BEGIN
      PENDOWN := FALSE;CMOVE(X,Y)
      END;
    -1 :BEGIN
      PENDOWN := TRUE ;CDRAW(X,Y)
      END
  END
END;

PROCEDURE RPLLOT;
VAR OCURX,OCURY,XCART,YCART :REAL;
BEGIN
  OCURX := CURX;OCURY :=CURY;
  XCART := CURX + X*COSAL - Y*SINAL;
  YCART := CURY + X*SINAL + Y*COSAL;
  PLOT(XCART,YCART,MODE);
  CURX := OCURX;CURY := OCURY
END;

PROCEDURE IPLOT;
VAR XCART,YCART :REAL;
BEGIN
  XCART := CURX + X*COSAL - Y*SINAL;
  YCART := CURY + X*SINAL + Y*COSAL;
  PLOT(XCART,YCART,MODE)
END;

PROCEDURE LIMIT;
BEGIN
  LXMIN := FX(XMIN);LYMIN := FY(YMIN);
  LXMAX := FX(XMAX);LYMAX := FY(YMAX);
  VIEWPORT(LXMIN,LXMAX,LYMIN,LYMAX)
END;

PROCEDURE SCALE;
BEGIN
  PX := XMIN;PY := YMIN;
  DX := 279/(XMAX-XMIN);
  DY := 191/(YMAX-YMIN);
  LIMIT(XMIN,XMAX,YMIN,YMAX)
END;

PROCEDURE SHOW ;
BEGIN
  IF YMIN<XMIN THEN XMIN := YMIN
  ELSE YMIN := XMIN;
  IF YMAX>XMAX THEN XMAX := YMAX
  ELSE YMAX := XMAX;
  PX := XMIN;PY := YMIN;
  DX := 191/(XMAX-XMIN);DY := DX;
  LIMIT(XMIN,XMAX,YMIN,YMAX)
END;

PROCEDURE AXES;
VAR I,X,Y,PAS :INTEGER;
BEGIN
  PENCOLOR(NONE);MOVETO(0,ROUND(FY(YINTER)));
  PENCOLOR(HCOLOR);MOVETO(279,TURTLEY);
  PENCOLOR(NONE);MOVETO(ROUND(FX(XINTER)),0);
  PENCOLOR(HCOLOR);MOVETO(TURTLEX,191);
  FOR I := 1 TO 2 DO BEGIN
    PAS := ROUND(XTIC*DX);
    IF I=2 THEN PAS := - PAS;
    X := ROUND(FX(XINTER));
    Y := ROUND(FY(YINTER));
    REPEAT
      PENCOLOR(NONE);MOVETO(X,Y-2);
      PENCOLOR(HCOLOR);MOVETO(X,Y+2);
      X := X + PAS
    UNTIL NOT(X IN [0..279])
  END;
  FOR I := 1 TO 2 DO BEGIN
    PAS := ROUND(YTIC*DY);
    IF I=2 THEN PAS := - PAS;
    Y := ROUND(FY(YINTER));
    X := ROUND(FX(XINTER));
    REPEAT
      PENCOLOR(NONE);MOVETO(X-2,Y);
      PENCOLOR(HCOLOR);MOVETO(X+2,Y);
      Y := Y + PAS
    UNTIL NOT(Y IN [0..191])
  END
END;

BEGIN
  (* INITIALISATIONS*)
  SCALE (0,279,0,191);
  PDIR (0);
  PEN(WHITE);
  PENUP;
  CMOVE(140,95)
END.
```

Ecriture en page haute résolution

Erick Ringot

Introduction

La possibilité qu'offre l'Apple II d'utiliser des pages graphiques sur 280 x 192 points est très souvent mise à contribution pour visualiser, de façon synthétique, des résultats numériques ou autres sous forme de graphes, réseaux, diagrammes, etc...

Malheureusement, il n'existe pas d'instruction Applesoft permettant d'écrire dans les pages graphiques, et d'apporter simplement des commentaires, titres, valeurs, sur nos travaux. HAIFA (Pom's 5) et DOS TOOL KIT (Apple) sont des utilitaires permettant d'écrire en page graphique, grâce à des POKES, mais au prix d'une tabulation de type TEXT.

Le but de cet article est de montrer comment écrire des mots à l'écran, tout en profitant de la haute résolution.

Principe

L'instruction DRAW de l'Applesoft permet de dessiner des formes contenues dans une table. Les formes sont définies par l'utilisateur et peuvent parfaitement représenter l'alphabet, les chiffres et caractères spéciaux. Ecrire un mot revient donc à dessiner, l'une derrière l'autre, de telles formes.

Une table ASCII.SET (de 755 octets) est proposée ici. Elle contient 63 formes représentant les 62 caractères de codes ASCII 32 à 94, définis dans une matrice classique de 5 x 7, plus un carré plein dont nous verrons l'utilité ultérieurement.

La solution en Basic

Méthode

Soit à écrire en page haute résolution une chaîne de caractères, contenue dans la variable CH\$, à la ligne YS, à partir de la colonne XS. La méthodologie est la suivante :

1. NS = longueur de CH\$
2. Pour IS allant de 1 à NS
3. A\$ = caractère No IS
4. CS = code ASCII de A\$
5. SS = CS - 31
6. Effacer un carré en XS - YS
7. Dessiner la forme No SS en XS - YS
8. Incrémenter XS et IS
9. Si IS <= NS, retourner en 3

Sous-programme Applesoft

Les paramètres d'appel sont :

- CH\$: chaîne à écrire
- YS : ligne d'écriture
- XS : 1ère colonne

Les variables locales sont :

- IS : indice du caractère
- A\$: caractère courant
- CS : code ASCII
- SS : forme courante

En sortie, XS est modifié.

Dans le paragraphe suivant, nous condenserons cette routine.

Utilisation du sous-programme

Elle impose quelques servitudes :

- chargement de la table de formes ASCII.SET à une adresse choisie;
- préciser cette adresse au système;
- préciser : ECHELLE = 1, ROTATION = 0;
- protection de la table et de la page graphique utilisée.

Plusieurs "cartes d'occupation" de la RAM sont possibles, selon la taille du programme Basic et la page graphique utilisée.

Si AD est l'adresse de chargement de la table, alors AH% = (AD / 256) et AL% = (AD - 256 * AH%) désignent respectivement les adresses haute et basse, et il faut préciser : POKE 232,AL% : POKE 233,AH%.

Si l'on utilise une autre table de formes, ces deux instructions doivent être réexécutées à chaque écriture en page HGR.

Il faut donc compléter notre routine, et nous en profitons pour en donner une version compacte :

```
1 AD = PEEK (115) + PEEK (116) *
  256: IF PEEK (AD) = 63 THEN 1
  000
2 AD = AD - 755
3 PRINT CHR$ (4)"BLOAD ASCII .SET
  ,A*AD: HIMEM: AD: GOTO 1000
99 REM --ROUTINE EN BASIC--
100 NS = LEN (CH$): REM LONGUEUR
110 FOR IS = 1 TO NS
120 A$ = MID$ (CH$,IS,1): REM EXTR
  ACTION
130 CS = ASC (A$): REM CODE
140 SS = CS - 31: REM NUMERO SHAPE
150 HCOLOR= 0: DRAW 63 AT XS,YS: REM
  EFFACE
160 HCOLOR= 3: DRAW SS AT XS,YS: REM
  ECRIT
170 XS = XS + 6: REM DECALAGE
```

```
180 NEXT IS
190 RETURN
199 REM --EN UNE LIGNE--
200 SCALE= 1: ROT= 0: POKE 232,AL%
  : POKE 233,AH%: FOR IS = 1 TO
  LEN (CH$): HCOLOR= 0: DRAW 63
  AT XS,YS: HCOLOR= 3: DRAW ASC
  ( MID$ (CH$,IS,1)) - 31 AT XS,
  YS:XS = XS + 6: NEXT : RETURN
```

```
999 REM ==PROG.PPAL==
1000 HGR
1001 AH% = AD / 256:AL% = AD - 256 *
  AH%: POKE 232,AL%: POKE 233,AH
  %
1010 HCOLOR= 3
1020 VTAB 21: CALL - 958: INPUT "
  CHAINE ";CH$: IF CH$ < > "" THEN
  INPUT "LIGNE ";YS: INPUT "COL
  ONNE ";XS: GOSUB 200: GOTO 102
  0
```

La solution en langage machine

Le principe reste le même.

Le programme est articulé autour de la routine DRAW, à l'adresse ROM \$F601, qui dessine une forme contenue dans la zone mémoire dont l'adresse haute est versée dans le registre Y et l'adresse basse dans le registre X, le paramètre de rotation (zéro) étant contenu dans l'accumulateur.

L'utilisation de cette routine nécessite donc de connaître l'adresse de chacune des formes.

Structure des tables de formes

Le calcul de l'adresse absolue de la forme numéro i s'opère de la façon suivante :

- soit SHAPE l'adresse de début de table
- on verse cette adresse en page zéro dans deux octets consécutifs, aux adresses TABLE et TABLE + 1
- on verse dans l'accumulateur le numéro i de la forme à dessiner. L'adresse relative de cette forme est contenue dans les octets "SHAPE + 2i" et "SHAPE + 2i + 1" (que l'on atteint par adressage indirect indexé). On obtient l'adresse absolue en ajoutant cette adresse relative à l'adresse SHAPE.

Le programme HGRECR.LIB

Le programme source est listé ci-après. Le code objet donné dans la récapitulation, relogeable, intègre la table ASCII.SET, celle-ci suivant la routine d'écriture proprement dite (longue de 168 octets).

Commentaires

– Lignes 48-59 : nécessaires au calcul de l'adresse de la table, puisque le programme est relogeable. Ce dernier consulte la pile du microprocesseur pour connaître sa propre adresse. En ajoutant la longueur de la routine, on obtient l'adresse de la table située à sa suite. Cette adresse est versée en `TABLE = $CE` et `TABLE+1 = $CF`, généralement libres.

– Lignes 63-66 : on empile l'échelle et la couleur en cours, afin de les sauvegarder.

– Lignes 72-83 : on saisit dans le texte du Basic le nom de la variable alphanumérique à écrire. Le nombre de caractères est porté en `LONCH`, l'adresse de la chaîne en `ADRCH` (et `ADRCH+1`).

– Lignes 87-92 / lignes 96-97 : saisie des coordonnées du premier caractère. Noter que `X` occupe 2 octets (`PXH` et `PXL`) et `Y` un seul (`PY`).

– Lignes 101-102 : les caractères se dessinent à l'échelle 1.

– Ligne 104 : `Y` sert de compteur de caractères.

– Lignes 106-109 : le caractère courant est empilé.

– Lignes 113-119 : efface avant d'écrire, en dessinant un carré (forme numéro 63) de couleur noire.

– Lignes 124-129 : on récupère le code ASCII du caractère sur la pile et on calcule le numéro de forme correspondant en déduisant `#$1F` (soit 31), puis on la dessine.

– Lignes 133-138 : le décalage du caractère suivant s'obtient en incrémentant l'abscisse `X` de la quantité

`#TRANS`, ici fixée à 6.

– Lignes 142-146 : on récupère le numéro du caractère sur la pile, on l'incrémente et on vérifie si la chaîne a entièrement été écrite ou non.

– Lignes 150-155 : on restitue couleur et échelle en cours. Fin de routine.

– Lignes 159-162 : modification classique du pointeur de pile dans les programmes relogeables.

– Lignes 164-173 : calcul de l'adresse absolue de la forme à dessiner; cette adresse est empilée provisoirement.

– Lignes 175-178 : positionnement du curseur à l'écran.

– Lignes 179-184 : dessine la forme.

– Lignes 186-187 : fin de procédure.

– Lignes 189-190 : c'est l'assembleur qui calcule pour nous la longueur de la routine... sympa !

Remarques

HGRECR.LIB ne modifie ni l'échelle, ni la couleur en cours dans le programme Basic. Il est compatible avec une autre table de formes puisqu'il n'utilise pas les adresses `$E8-$E9` (232-233).

Utilisation

(1) Charger, selon la taille du programme Basic et la page HGR utilisée, la routine HGRECR.LIB en mémoire (adresse AD).

(2) Protéger page graphique et routine par `LOMEM` et/ou `HIMEM`.

(3)-Syntaxe :

`CALL AD, CH$, H, V`

la virgule sert de séparateur entre chaque paramètre.

`CH$` = variable alphanumérique contenant la chaîne à écrire

`H` = colonne du premier caractère.

`V` = ligne d'écriture.

(`H` et `V` sont des expressions numériques)

(4) Limitations :

- `CH$` ne doit contenir que des caractères dont le code ASCII est compris entre 32 et 94 (pas de caractères de contrôle notamment)
- Il n'y a pas de vérification de la longueur du mot : attention !

(5) Possibilités :

On peut changer la police de caractères (utiliser des minuscules, des alphabets étrangers ou spéciaux). Il suffit pour cela de remplacer la table existante par une autre à la suite du programme. La forme numéro 63 doit être un carré plein. Voir HGRECR.DEM pour un exemple d'application.

Etude comparative

Occupation mémoire

La ligne Basic condensée n'occupe que 96 octets, à comparer aux 168 octets de la routine en langage machine. Toutefois, on n'a pas pris en compte les variables auxiliaires que nécessite le Basic.

Vitesse d'exécution

Le test consistant à écrire les 1000 premiers nombres à l'aide des deux procédés donne :

BASIC : 1 minute 14 secondes

MACHINE : 53 secondes

ce qui est éloquent.

Comparaison qualitative

BASIC :

- tient en une ligne
- modifie les paramètres HGR
- nécessite un indice de boucle et des transferts de chaîne

MACHINE :

- syntaxe condensée
- pas de modification des variables Basic
- esthétique du programme
- utilisation possible dans une bibliothèque gérée par l'ampersand (&)

Programme de démonstration

```
10 REM
```

```
-----  
HGRECR.DEM-2  
-----
```

```
15 HM = PEEK (115) + 256 * PEEK (116) : REM HIMEM  
20 IF PEEK (HM) = 32 AND PEEK (HM + 1) = 88 THEN 30
```

```
25 HM = HM - 923 : HIMEM : HM : PRINT  
CHR$(4) "BLOADHGRECR.LIB,A" : HM  
: REM NEW.HIMEM  
30 DATA "EXEMPLE -> GANTT", "EXEMPLE  
E -> HISTOGRAMME", "EXEMPLE ->  
RESEAU", "SYNTAXE INSTRUCTION",  
"POLICE CARACTERES", "DEMO.AUTO",  
"RETOUR BASIC" : NC = 6 : DIM M$(NC) : FOR I = 0 TO NC : READ M$(I) : NEXT  
35 DATA "TERRASSEMENTS", "FONDATION  
S", "RESEAUX", "MURS EXTERIEURS",  
"CLOISONS", "PLANCHER", "COUVERTURE", "FUMISTERIE", "MENUISERIE
```

```

S*, "ENDUITS", "PEINTURES": DIM
G$(10): FOR I = 0 TO 10: READ
G$(I): NEXT
40 DATA 10,30,30,60,60,80,100,150
,170,200,50,100,30,160,40,120,
140,60,80,150: FOR I = 1 TO 10
: READ X(I): NEXT : FOR I = 1 TO
10: READ Y(I): NEXT
45 DATA "PARIS", "MARSEILLE", "LYON"
, "BORDEAUX", "LILLE", "TOULOUSE"
, "STRASBOURG", "RENNES", "ORLEAN
S", "LENS", "LE PUY": FOR I = 1 TO
10: READ R$(I): NEXT
60 GOTO 10000
995 REM ---EXEMPLE---
1000 HCOLOR= 1
1005 FOR I = 0 TO 10
1010 Y = 10 * (I + 1)
1015 CALL HM, G$(I), 10, Y
1020 X1 = 100 + 15 * I + RND (1) *
10: X2 = 200 + RND (1) * 79
1025 FOR J = - 3 TO 3: H PLOT X1, J
+ Y TO X2, J + Y: NEXT
1030 NEXT
1035 A$ = "UN EXEMPLE D'EMPLOI ": CALL
HM, A$, 50, 130
1040 A$ = " DIAGRAMME-GANTT ": FOR
I = 10 TO 170 STEP 6: CALL HM,
A$, I, 150: NEXT
1045 RETURN
1995 REM ---SYNTAXE---
2000 A$ = "SYNTAXE DE L'INSTRUCTION
": CALL HM, A$, 100, 20: H PLOT 90
, 30 TO 250, 30
2005 A$ = "CALL ADR, CH$, C, L": CALL
HM, A$, 50, 50
2010 A$ = "ADR = ADRESSE DE CHARG
EMENT DE LA ROUTINE": CALL HM,
A$, 10, 100
2015 A$ = "A$ = VARIABLE ALPHANU
MERIQUE A ECRIRE": CALL HM, A$,
10, 110
2020 A$ = "C = COLONNE DU PREMI
ER CARACTERE": CALL HM, A$, 10, 1
20
2025 A$ = "L = LIGNE D'ECRITURE
": CALL HM, A$, 10, 130
2030 RETURN
2995 REM ---POLICE---
3000 REM A$ = "CARACTERES DISPONIB
LES ": CALL HM, A$, 100, 20: H P
LOT 90, 30 TO 250, 30
3005 HCOLOR= 3: FOR I = 0 TO 8: H PLOT
30 * I, 0 TO 30 * I, 191: NEXT :
H PLOT 0, 0 TO 240, 0: H PLOT 0, 2
0 TO 240, 20: H PLOT 0, 191 TO 24
0, 191
3010 A$ = "CARACTERES DISPONIBLES":
FOR I = 1 TO LEN (A$): C$ = MID$(
A$, I, 1): CALL HM, C$, 250, 8 * I
: NEXT
3015 A$ = "NO.": FOR I = 0 TO 3: CALL
HM, A$, 10 + 60 * I, 10: NEXT : A$
= "CAR": FOR I = 0 TO 3: CALL
HM, A$, 40 + 60 * I, 10: NEXT
3020 FOR I = 1 TO 62
3025 N$ = STR$(I): A$ = CHR$(I +
31): Q% = (I - 1) / 4: R% = (I -
1) - 4 * Q%: X = 10 + 60 * R%: Y
= 30 + 10 * Q%
3030 CALL HM, N$, X, Y: CALL HM, A$, X +
30, Y
3035 NEXT

```

```

3040 RETURN
3995 REM --HISTOGRAMME--
4000 HCOLOR= 3: H PLOT 10, 10 TO 10,
150 TO 260, 150
4010 FOR I = 0 TO 4: Y% = 150 - RND
(1) * 140: Y$ = STR$(150 - Y%
): X1 = 10 + 50 * I: X2 = X1 + 5
0: H PLOT X1, 150 TO X1, Y% TO X2
, Y% TO X2, 150: CALL HM, Y$, 50 *
I + 20, Y% - 5
4015 ST% = (I + 1): FOR L = Y% TO 1
50 STEP ST%: Z% = NOT Z%: FOR
C = X1 + Z% * ST% TO X2 - Z% *
ST% STEP ST% + 1: H PLOT C, L: NEXT
: NEXT : NEXT
4020 A$ = "H I S T O G R A M M E": CALL
HM, A$, 50, 160
4700 RETURN
4995 REM --RESEAU--
5000 HCOLOR= 3
5005 FOR I = 1 TO 10: FOR J = I TO
10
5010 R = RND (1): IF R < .8 AND R >
.2 THEN H PLOT X(I), Y(I) TO X(
J), Y(J)
5020 NEXT : NEXT
5030 FOR I = 1 TO 10: CALL HM, R$(I
), X(I), Y(I): NEXT
5040 A$ = "R E S E A U": CALL HM, A$
, 180, 10
5100 RETURN
5995 REM --DEMO.AUTO--
6000 B = 1
6005 HGR : POKE - 16302, 0: ON B GOSUB
1000, 4000, 5000, 2000, 3000
6010 FOR I = 1 TO 1000: IF PEEK (
K) > 127 THEN RETURN
6011 NEXT
6015 B = B + 1: IF B > 5 THEN B = 1
6020 GOTO 6005
9999 REM ---MENU.PPAL---
10000 TEXT : NORMAL : HOME : S = -
16336: KS = - 16368: K = - 163
84
10010 GOTO 11000
10020 FOR I = 1 TO 40: PRINT "-";:
NEXT : RETURN
10030 INVERSE : X = PEEK (S) - PEEK
(S) + PEEK (S) - PEEK (S)
10040 VTAB 10 + I: PRINT M$(I): NORMAL
: RETURN
11000 GOSUB 10020: PRINT "DEMONSTR
ATION ECRITURES": PRINT "EN PA
GE GRAPHIQUE HAUTE-RESOLUTION"
: GOSUB 10020
11010 VTAB 10: FOR I = 0 TO NC: PRINT
M$(I): NEXT
11020 I = 0
11030 GOSUB 10030: POKE KS, 0: WAIT
K, 128: POKE KS, 0: A = PEEK (K)
: GOSUB 10040
11040 IF A = 21 THEN I = I + 1: IF
I = NC + 1 THEN I = 0
11050 IF A = 8 THEN I = I - 1: IF
I = - 1 THEN I = NC
11060 IF A < > 13 THEN 11030
11070 HGR : POKE - 16302, 0: ON I +
1 GOSUB 1000, 4000, 5000, 2000, 30
00, 6000, 12000: GET A$: GOTO 10
000
12000 HOME : TEXT : END

```

Assembleur Toolkit

```

SOURCE FILE: HGRECR
0000:      1 ;E.RINGOT LE 2/6/84
0000:      2 ;-----
-----
0000:      3 ;ECRITURES DE CHAINES
0000:      4 ;EN PAGE HAUTE-RESOLUT
ION
0000:      5 ;
0000:      6 ;HGRECR.LIB RELOGEABL
E
0000:      7 ;-----
-----
0000:      8 ;
0000:      9 ;MODE D'EMPLOI
0000:     10 ;
0000:     11 ;SYNTAXE : CALL ADR,
CH$,X,Y
0000:     12 ;
0000:     13 ;CH$=VARIABLE ALPHANU
MERIQUE
0000:     14 ;X  ABSCISSE
0000:     15 ;Y  ORDONNEE
0000:     16 ;
----- NEXT OBJECT FILE NAME IS HGRECR.OBJ
0
1000:     17      ORG  $1000
      ;POUR ASSEMBLAGE SEULEMENT
1000:     18 ;
1000:     19 ;<1>-ROUTINES ROM
1000:     20 ;
0083:     21 VARPNT EQU  $83
      ;POINTE SUR LA LONGUEUR
DEBE:     22 CHKCOM EQU  $DEBE
      ;VERIFIE PRESENCE VIRGULE
DD6C:     23 CHKSTR EQU  $DD6C
      ;VERIFIE LA PRESENCE D'UNE CHAIN
E
DFE3:     24 PTRGET EQU  $DFE3
      ;SAISIT UNE VARIABLE
E105:     25 GETINT EQU  $E105
      ;SAISIT UN ENTIER SUR 2 OCTETS
E6F5:     26 GTBYTC EQU  $E6F5
      ;SAISIT UN ENTIER SUR 1 OCTET
F601:     27 DRAW  EQU  $F601
F411:     28 HPOSN  EQU  $F411
FF58:     29 RETURN EQU  $FF58
      ;INSTRUCTION RTS
1000:     30 ;
1000:     31 ;<2>-PARAMETRES
1000:     32 ;
00A0:     33 FACMD  EQU  $A0
00A1:     34 FACLO  EQU  $A1
00E0:     35 PXL   EQU  $E0
00E1:     36 PXH   EQU  $E1
00E2:     37 PY    EQU  $E2
00CE:     38 TABLE EQU  $CE
      ;ET $CF
00E4:     39 HCOLOR EQU  $E4
00E7:     40 SCALE EQU  $E7
0006:     41 ADRCH EQU  $06
0008:     42 LONCH EQU  $08
003F:     43 CARRE  EQU  $3
0006:     44 TRANS  EQU  $6
      ;TRANSLATION CURSEUR
1000:     45 ;
1000:     46 ;<3>-ADRESSE TABLE DE
FORMES
1000:     47 ;

```

```

1000:20 58 FF  48 DEBUT  JSR  RETURN
1003:BA      49      TSX
1004:CA      50      DEX
1005:CA      51      DEX
1006:9A      52      TXS
1007:18      53      CLC
1008:68      54      PLA
1009:69 A6    55      ADC  #LONG
1008:85 CE    56      STA  TABLE
100D:68      57      PLA
100E:69 00    58      ADC  #0
1010:85 CF    59      STA  TABLE+1
1012:        60 ;
1012:        61 ;<3'>-SAUVEGARDE ENVIR
ONNEMENT HAUTE RESOLUTION
1012:        62 ;
1012:A5 E4    63      LDA  HCOLOR
1014:48      64      PHA
1015:A5 E7    65      LDA  SCALE
1017:48      66      PHA
1018:        67 ;
1018:        68 ;<4>-SAISIE ARGUMENTS
1018:        69 ;
1018:        70 ;4/1/EXPRESSION A EDIT
ER
1018:        71 ;
1018:20 BE DE  72      JSR  CHKCOM
1018:20 E3 DF  73      JSR  PTRGET
101E:20 6C DD  74      JSR  CHKSTR
1021:A0 00    75      LDY  #0
1023:B1 83    76      LDA  (VARPNT),
Y ;LONGUEUR
1025:85 08    77      STA  LONCH
1027:C8      78      INY
1028:B1 83    79      LDA  (VARPNT),
Y ;ADRESSE BASSE
102A:85 06    80      STA  ADRCH
102C:C8      81      INY
102D:B1 83    82      LDA  (VARPNT),
Y ;ADRESSE HAUTE
102F:85 07    83      STA  ADRCH+1
1031:        84 ;
1031:        85 ;4/2/ABSCISSE
1031:        86 ;
1031:20 BE DE  87      JSR  CHKCOM
1034:20 05 E1  88      JSR  GETINT
1037:A5 A1    89      LDA  FACLO
1039:85 E0    90      STA  PXL
1038:A5 A0    91      LDA  FACMO
103D:85 E1    92      STA  PXH
103F:        93 ;
103F:        94 ;4/3/ORDONNEE
103F:        95 ;
103F:20 F5 E6  96      JSR  GTBYTC
1042:86 E2    97      STX  PY
1044:        98 ;
1044:        99 ;<5>-ECRITURE DE LA CH
AINE
1044:        100 ;
1044:A9 01    101      LDA  #1
1046:85 E7    102      STA  SCALE
      ;ECHELLE
1048:        103 ;
1048:A0 00    104      LDY  #0
      ;1ER CARACTERE
104A:        105 ;
104A:98      106 WRITE TYA
      ;EMPILAGE
104B:48      107      PHA
      ;NUMERO CARACTERE
104C:B1 06    108      LDA  (ADRCH),Y
104E:48      109      PHA

```

```

104F:      110 ;
104F:      111 ;5/1/EFFACE UN CARRE
104F:      112 ;
104F:A9 00 113      LDA  #500

1051:85 E4 114      STA  HCOLOR
      ;COULEUR NOIRE
1053:A9 3F 115      LDA  #CARRE
1055:20 58 FF 116    JSR  RETURN
1058:50 26 117      BVC  DESSIN
105A:A9 7F 118      LDA  #7F
      ;COULEUR BLANCHE
105C:85 E4 119      STA  HCOLOR
105E:      120 ;
105E:      121 ;5/2/DESSINE LE CARAC
      TERE
105E:      122 ;
105E:      123 ;
105E:68 124      PLA
105F:38 125      SEC

1060:E9 1F 126      SBC  #1F
      ;CALCUL NUMERO FORME
1062:      127 ;
1062:20 58 FF 128    JSR  RETURN
1065:50 19 129      BVC  DESSIN
1067:      130 ;
1067:      131 ;5/3/TRANSLATION DU C
      URSEUR
1067:      132 ;
1067:18 133      CLC
1068:A5 E0 134      LDA  PXL
106A:69 06 135      ADC  #TRANS
106C:85 E0 136      STA  PXL
106E:D0 02 137      BNE  SUITE
1070:E6 E1 138      INC  PXH
1072:      139 ;
1072:      140 ;5/4/CARACTERE SUIVAN
      T
1072:      141 ;
1072:68 142      SUITE  PLA
      ;DEPILE LE NUMERO CARACTERE
1073:A8 143      TAY
1074:C8 144      INY
      ;CARACTERE SUIVANT
1075:C4 08 145      CPY  LONCH
      ;EST-CE-FINI ?
1077:D0 D1 146      BNE  WRITE
      ;NON ! ON CONTINUE
1079:      147 ;
1079:      148 ;<6>-RESTITUTION ENVIR
      ONNEMENT HAUTE RESOLUTION
1079:      149 ;

```

```

1079:68 150      PLA
107A:85 E7 151      STA  SCALE
107C:68 152      PLA
107D:85 E4 153      STA  HCOLOR
107F:      154 ;
107F:60 155      RTS
      ;OUI, RETOUR BASIC
1080:      156 ;
1080:      157 ;<7>-PROCEDURE DE TRAC
      E DE FORME
1080:      158 ;
1080:BA 159      DESSIN  TSX
      ;PROGRAMME
1081:CA 160      DEX
      ;RELOGEABLE
1082:CA 161      DEX
      ;CF.POM'S 7
1083:9A 162      TXS
      ;PAGE 27
1084:      163 ;
1084:0A 164      ASL  A
      ;DOUBLE LE NUMERO
1085:A8 165      TAY
1086:B1 CE 166      LDA  (TABLE),Y
      ;LSB INDEX FORME
1088:18 167      CLC
1089:65 CE 168      ADC  TABLE
108B:48 169      PHA
108C:C8 170      INY
108D:B1 CE 171      LDA  (TABLE),Y
      ;MSB INDEX FORME
108F:65 CF 172      ADC  TABLE+1
1091:48 173      PHA
1092:      174 ;
1092:A6 E0 175      LDX  PXL
1094:A4 E1 176      LDY  PXH
1096:A5 E2 177      LDA  PY
1098:20 11 F4 178    JSR  HPOSN
109B:68 179      PLA
109C:A8 180      TAY
109D:68 181      PLA
109E:AA 182      TAX
109F:A9 00 183      LDA  #0
      ;ROTATION NULLE
10A1:20 01 F6 184    JSR  DRAW
10A4:      185 ;
10A4:2C 58 FF 186    BIT  RETURN
10A7:60 187      RTS
10A8:      188 ;
10A8:      189 SHAPE  EQU  *
00A6:      190 LONG  EQU  SHAPE-DEB
      UT-2

```

HGRECR.LIB

```

1000- 20 58 FF BA CA CA 9A 18
1008- 68 69 A6 85 CE 68 69 00
1010- 85 CF A5 E4 48 A5 E7 48
1018- 20 BE DE 20 E3 DF 20 6C
1020- DD A0 00 B1 83 85 08 C8
1028- B1 83 85 06 C8 B1 83 85
1030- 07 20 BE DE 20 05 E1 A5
1038- A1 85 E0 A5 A0 85 E1 20
1040- F5 E6 86 E2 A9 01 85 E7
1048- A0 00 98 48 B1 06 48 A9
1050- 00 85 E4 A9 3F 20 58 FF
1058- 50 26 A9 7F 85 E4 68 38
1060- E9 1F 20 58 FF 50 19 18
1068- A5 E0 69 06 85 E0 00 02
1070- E6 E1 68 A8 C8 C4 08 D0
1078- D1 68 85 E7 68 85 E4 60
1080- BA CA CA 9A 0A A8 B1 CE

```

```

1088- 18 65 CE 48 C8 B1 CE 65
1090- CF 48 A6 E0 A4 E1 A5 E2
1098- 20 11 F4 68 A8 68 AA A9
10A0- 00 20 01 F6 2C 58 FF 60
10A8- 3F 00 80 00 84 00 8A 00
10B0- 90 00 9D 00 AA 00 85 00
10B8- C1 00 C5 00 CD 00 05 00
10C0- E2 00 EA 00 EE 00 F3 00
10C8- F6 00 FD 00 08 01 13 01
10D0- 1D 01 28 01 32 01 3E 01
10D8- 48 01 51 01 5C 01 67 01
10E0- 68 01 70 01 79 01 81 01
10E8- 8A 01 94 01 A1 01 AD 01
10F0- 89 01 C4 01 D0 01 D0 01
10F8- E7 01 F2 01 FE 01 07 02
1100- 0F 02 1C 02 25 02 31 02
1108- 3D 02 49 02 53 02 5F 02
1110- 6C 02 77 02 80 02 88 02
1118- 96 02 A2 02 AE 02 87 02
1120- C2 02 CA 02 D1 02 D9 02

```

```

1128- 5B 49 02 00 52 22 20 24
1130- 2C 00 08 24 1F 36 06 00
1138- 3A 67 3C 0C 6C BE 2D 1E
1140- 2E 1E FE 2C 00 E7 0C 25
1148- 15 F5 AB 15 1F 15 3F 77
1150- 29 00 0C 0C DC 38 2E 96
1158- 17 4D 2E 24 00 60 1C BF
1160- AE 17 76 65 1C 0D 16 07
1168- 00 08 24 05 00 92 1C 1C
1170- 24 0C 0C 06 00 92 0C 0C
1178- 24 1C 1C 06 00 3C 1C 4C
1180- 6E 1E 16 3F 17 00 0D DE
1188- 07 00 20 8D 3A 3F 77 31
1190- 05 00 89 F6 04 00 3F 4C
1198- 11 35 00 12 05 00 0C 0C
11A0- D6 DA 1E 06 00 0C 25 1C
11A8- 3F 17 36 2E 1E 0E 2D 0C
11B0- 24 07 00 24 BC 96 31 17
11B8- 2D 04 00 65 E4 3F 17 95
11C0- BA 2E 2D 25 00 25 0C 3C

```

11C8- 3F B7 92 15 2D 0C 24 00
 11D0- 3A 27 0C 0C 0C 36 36 F5
 11D8- 3E 00 38 27 2C 2D F5 AA
 11E0- 36 1E 3F 1C 04 00 AD F6
 11E8- 3F 1C 24 25 0C 0C 35 00
 11F0- 0C 0C 3C 3F 77 92 36 05
 11F8- 00 E7 64 2D 15 BE 15 F6
 1200- 3F 1C 2C 00 E7 64 2D 15
 1208- 36 77 1E 17 3F 04 00 08
 1210- 16 06 00 08 16 BE 05 00
 1218- 91 E2 1C 1C 0C 0C 0C 06
 1220- 00 38 67 89 B5 3F 3F 04
 1228- 00 93 62 0C 0C 1C 1C 1C
 1230- 06 00 0C 0C 1C 3F 17 95
 1238- 0A 16 05 00 30 2E 2C 24
 1240- 1C 3F 17 36 36 0E 2D 25
 1248- 00 3A 37 6E 09 24 67 E4
 1250- 1C 1E 1E 2E 00 3F 24 2C
 1258- 2D 15 BE 0E BE 3F 27 2C
 1260- 00 89 F2 3F 1C 24 24 0C

1268- 2D 15 06 00 09 36 1E 3F
 1270- 27 24 24 2C 2D 15 3E 00
 1278- 39 B7 3A 24 24 2D 2D
 1280- 96 92 3F 04 00 39 B7 1A
 1288- 24 24 24 2D 2D 06 00 11
 1290- 35 3E 3F 1C 24 24 0C 2D
 1298- 35 00 2B 2D 24 FC 18 36
 12A0- 36 36 4D 21 24 00 52 3A
 12A8- 67 24 24 3C 2D 06 00 9B
 12B0- 72 2D 0C 24 24 3C 00 73
 12B8- 0E 15 DF 23 24 24 6C 09
 12C0- 1E 1E 06 00 89 12 3F 3F
 12C8- 24 24 24 05 00 E0 1C 36
 12D0- 36 36 4D 21 24 24 BC 06
 12D8- 00 0E 56 24 24 24 DF 33
 12E0- 2E 1E 36 2E 00 92 E7 24
 12E8- 24 0C 2D 15 36 36 17 05
 12F0- 00 65 3C 38 3F 36 2E 1E
 12F8- 36 05 00 AA 15 1F 3F 20
 1300- 24 64 2D 15 36 36 00 77

1308- 15 15 DF 23 24 24 2C 2D
 1310- 15 BE 06 00 E7 64 2D 15
 1318- 97 15 F6 3F 1C 04 00 24
 1320- 1F 28 2D F5 92 33 2E 00
 1328- 92 E7 24 24 6C 09 36 36
 1330- BE 05 00 92 1C 1C 24 24
 1338- 4D 31 36 BE 06 00 F6 1E
 1340- 24 24 24 4D 31 36 BE 35
 1348- 07 00 0C 0C FC 18 76 16
 1350- 17 6E 09 E4 04 00 1C 1C
 1358- 6C 09 F6 06 36 05 00 0C
 1360- 0C 3C 3F 77 92 17 2E 2D
 1368- 25 00 24 2C B5 02 33 2E
 1370- 2D 00 1C 1C 56 4A 0E 06
 1378- 00 24 3C B7 52 31 3E 3F
 1380- 00 25 3F 36 2D 25 24 3F
 1388- 3F 36 36 2D 2D 25 24 24
 1390- 3F 3F 3F 36 36 36 2D 2D
 1398- 2D 05 00 85

*

DISK CHECK-UP

Alexandre Avrane

Mieux vaut prévenir que guérir, paraît-il, et que celui d'entre vous qui n'a jamais vu le message I/O ERROR apparaître sur son écran me jette la première pierre...

Les disquettes courantes sont (théoriquement) certifiées pour une durée utile d'une soixantaine d'heures. Mais comme personne n'a le courage de déclencher un chronomètre à chaque appel au DOS, il est bien difficile d'en évaluer l'état d'usure.

Le programme DISK CHECK-UP vérifie les disquettes initialisées par DOS 3.3, Pascal, ProDOS, CP/M, MEM/DOS et autres systèmes d'exploitation de l'Apple reposant sur la PROM 16 secteurs et son standard de formatage physique. Il indique l'état physique de vos disquettes. Mais, à la différence des utilitaires qui fournissent la liste des secteurs illisibles et éventuellement quelques méthodes pour les récupérer en partie, ce programme donne, à l'avance, une indication sur l'intégrité physique de vos disquettes et leur usure. En revanche, l'intégrité logique, en particulier les pointeurs utilisés par le système d'exploitation pour relier le directory aux fichiers, n'est bien entendu pas contrôlée.

Fonctionnement du DOS

Le DOS de l'Apple est un animal patient et bien dressé : pour calmer la tension nerveuse de son utilisateur, il ne lui fournit le dramatique message d'I/O ERROR qu'après avoir vainement tenté 96 fois de lire le secteur demandé. Plus exactement, il essaie d'abord 48 fois puis, s'il n'est pas encore parvenu à obtenir correctement le secteur, il recalibre le bras de lecture (c'est le bruit vibrant et très désagréable car souvent annonciateur

de catastrophe) pour effectuer à nouveau 48 tentatives.

Contrôles effectués

Le programme demande au DOS d'observer l'ensemble de la disquette et relève, pour chaque secteur, le nombre de tentatives infructueuses. Normalement, ce nombre doit être très faible; il s'affiche en hexadécimal (0, 1, ...F), en mode normal pour la plage de valeurs 1 à 16, en inverse pour 17 à 32, en flash pour 33 à 48. Les valeurs 1 et 2 sont affichées par un simple point (.) afin d'aérer l'écran. Toute autre valeur affichée en mode normal (donc inférieure à 16) n'est pas inquiétante immédiatement. En revanche, si le programme affiche des secteurs en inverse ou en flash, il devient urgent de faire une copie de la disquette. Le programme ne se préoccupe pas des valeurs supérieures à 48 car celles-ci ont déclenché le recalibrage du bras de lecture et il est alors fortement déconseillé de continuer à utiliser la disquette. Si la tentative de lecture se solde par un échec complet, un "?" est affiché en flash.

Théoriquement, un secteur est obtenu avec succès après 8 tentatives en moyenne; en effet, une piste contient 16 secteurs et la tête de lecture peut se trouver à n'importe quelle position sur la piste. Néanmoins, et afin d'accélérer le temps d'exécution, le programme est minuté de manière à minimiser le nombre de lectures, en gérant le secteur demandé selon la position de la tête. En conséquence, seul le secteur 0 de la piste 0 doit, théoriquement, présenter un nombre non négligeable d'accès.

Les valeurs obtenues sur les autres secteurs de la piste zéro, et sur les

autres pistes, doivent normalement être très faibles; elles dépendent non seulement de l'usure physique de la disquette, mais également de la clé de traduction entre numéro de secteur logique et secteur physique (c'est un paramètre interne utilisé lors de l'initialisation d'une disquette). Dans tous les cas, ces valeurs doivent être inférieures à 16 pour indiquer une qualité correcte.

Exécution du programme

Le programme se lance par un simple BRUN DISK CHECK-UP.M et se charge en \$900. Il nécessite que le DOS 3.3 soit chargé en mémoire (éventuellement sur la carte langage). L'utilisateur indique simplement les numéros de slot et de drive contenant la disquette à vérifier, puis le nombre de tentatives de lecture pour chaque secteur s'affiche. Dans tous les cas, RETURN valide la saisie, ESC annule l'opération en cours. Pour mieux apprécier l'exécution du programme, ouvrez légèrement et quelques instants la porte du lecteur et observez le résultat!

Le fichier DISK CHECK-UP.S contient le source en Big Mac. Le fonctionnement interne du programme repose sur l'examen de l'adresse \$578 (RETRYCNT) utilisée par RWTS.

Quelques derniers conseils

Ce programme est destiné à prévenir les problèmes d'usure. Il n'a malheureusement pas la faculté de savoir que vous poserez un électro-aimant à côté de vos disquettes demain matin, ou que celles-ci vont être empruntées par votre petit dernier pour construire son château de cartes...

D'autre part, si vous décidez de contrôler, à l'aide de ce programme, l'ensemble de votre bibliothèque, ne vous arrêtez pas après avoir vérifié les quelques disquettes les plus utili-

sées: si celles-ci étaient réellement usées, il est probable que vous auriez déjà eu des ennuis. Les problèmes, comme disait Murphy, ne surviennent que sur les disquettes que vous n'utilisez quasiment jamais et, bien

sûr, au moment où vous en avez rapidement besoin.

Conclusion

Ce programme ne doit pas induire une impression de sécurité qui serait

illusoire: il détecte les ennuis latents mais n'empêche pas leur apparition: ce n'est ni un anti-moustiques, ni un traitement contre la rouille...

Assembleur Big Mac

```

1          LST OFF
2
3 *****
4 *          DISK CHECK-UP          *
5 *****
6
7 * Copyright [C] 1984 A. Avrane
8
9 * M.A.J. : 14/07/84
10 * Creation:09/05/84
11
12 * Ce programme vérifie
13 * l'intégrité physique de toute
14 * disquette formatée 16 secteurs,
15 * en affichant le nombre d'essais
16 * requis par RWTS pour lire
17 * chacun de ses secteurs.
18 *
19 * Affichage des résultats en
20 * normal si <16, inverse si >16
21 * et <32, flash sinon, pour le
22 * premier calibrage.
23 * Code ? et S si i/o error.
24
25 LOWCASE = 1

```

```

26
27 CR          =    $8D
28 ESC        =    $9B
29 BLANK      =    " "
30
31 BASL       =    $28
32 TRAP       =    $48
33 GORWTS     =    $3D9
34 DOS        =    $3EA
35 RETRYCNT   =    $578
36 SYMBOL     =    $680
37 KEYBOARD   =    $C000
38 STROBE     =    $C010
39 PRBL2      =    $F94A
40 BASCALC    =    $FBC1
41 HOME       =    $FC58
42 WAIT       =    $FCA8
43 COUT0      =    $FDF0
44 SETVID     =    $FE93
45           ORG   $900
46
47 * Initialise format ecran
48 * =====
49 START      JSR   SETVID
50           JSR   HOME
51           LDX   #14
52           JSR   PRBL2          set x=0

```

LA PHOTOCOMPOSITION EN PROLONGEMENT DE LA MICRO-INFORMATIQUE



TRANSMETTEZ-NOUS VOS TEXTES
PAR TÉLÉPHONE

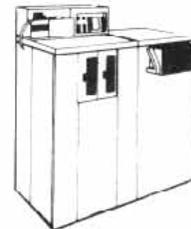
ou

DONNEZ-NOUS VOTRE DISQUETTE



Les textes de vos articles, catalogues, annuaires ou brochures saisis sur votre APPLE sont envoyés directement sur notre photocomposeuse.

Nous vous évitons ainsi, le coût et le temps de la saisie supplémentaire que nécessite le traitement traditionnel de la photocomposition avant l'impression des documents, si vous le désirez nous pouvons également nous charger de l'impression et du brochage.



NOTRE RÉFÉRENCE... LA REVUE POM'S

TELECOMPO 328.18.63

PHOTOCOMPOSITION
BUREAUTIQUE
TRANSMISSION DE DONNÉES

GESTION DE FICHIERS
MATÉRIEL DE
TRAITEMENT DE TEXTES

13 et 15 avenue du Petit Parc
94300 VINCENNES

```

53 DV11^2 LDA MSG1,X
54 BEQ DV12
55 JSR COUT0
56 INX
57 BNE DV11^2 =jmp
58
59 * Demande slot, drive, format
60 * =====
61 DV12 = *
62 LDX #8+$80
63 JSR GETKEY
64 BCS GOEXIT
65 BEQ DV12^1
66 PHA
67 ORA #$80
68 STA MSGSLOT
69 PLA
70 ASL
71 ASL
72 ASL
73 ASL
74 STA IOBSLOT
75
76 DV12^1 LDX #23+$80
77 DV12^2 JSR GETKEY
78 GOEXIT BCS EXIT
79 BEQ DV2^0
80 CMP #3
81 BCS DV12^2
82 STA IOBDRIIVE
83 ORA #$80
84 STA MSGDRIVE
85
86 * Boucle sur les pistes
87 * =====
88 DV2^0 LDA #0
89 STA IOBTRACK
90 STA IOBVOL
91
92 * Boucle sur les secteurs
93 * =====
94 DV2^1 LDY #$FF
95 STY SECTPTR
96 DV20^2 INC SECTPTR 0-15
97 LDY SECTPTR
98 CPY #16 last?
99 BCS DV31^9
100 LDA SECTABL,Y
101 STA IOBSECTR
102 ADC #8 carry=0
103 JSR BASCALC set bas1
104
105 LDA KEYBOARD
106 BPL DV20^3
107 CMP #ESC
108 BNE DV20^3
109 STA STROBE
110 BEQ EXIT =jmp
111 GOSTART BEQ START
112
113 DV20^3 LDA #>IOB
114 LDY #<IOB
115 JSR GORWTS
116 BCS DV31^3
117 LDY RETRYCNT $01-$30
118 DEY
119 LDA TBLVALUE,Y
120 DV31^2 LDY IOBTRACK
121 INY
122 INY
123 STA (BASL),Y
124 BNE DV20^2 =jmp
125

```

```

126 DV31^3 = *
127 LDA IOB^STAT
128 LSR
129 LSR
130 LSR
131 LSR
132 TAY
133 LDA TBLSTAT,Y
134 BNE DV31^2 =jmp
135
136 DV31^9 = *
137 INC IOBTRACK
138 LDA IOBTRACK
139 CMP #$23
140 BCC DV2^1
141
142 * Autre disque ou quitte
143 * =====
144 EXIT = *
145 LDX #30
146 LOOPEXIT LDA MSG2-30,X
147 BEQ LOOPX^0
148 STA SYMBOL-$80,X
149 INX
150 BNE LOOPEXIT =jmp
151 LOOPX^0 STA TRAP
152 LOOPX^1 LDX #38
153 JSR GETKEY
154 BCS BYE
155 BEQ GOSTART =jmp
156 BYE JSR HOME
157 JMP DOS
158
159 * Saisie touche / curseur special
160 * =====
161 GETKEY STX SCREEN1
162 STX SCREEN2
163 LDA #"("
164 BNE PUTKEY =jmp
165 WAITKEY LDA SYMBOL
166 SCREEN1 = *-2
167 EOR #1
168 PUTKEY STA SYMBOL
169 SCREEN2 = *-2
170 LDA #6
171 JSR WAIT
172 LDA KEYBOARD
173 BPL WAITKEY
174 STA STROBE
175 PHA
176 LDA #BLANK
177 STA SYMBOL-$80,X
178 STA STROBE
179 PLA
180 CMP #ESC
181 BEQ CHECKKEY2 carry=1
182 CMP #CR
183 BEQ CHECKKEY1
184 CPX #$80
185 BCC GETKEY
186 CMP #"1"
187 BCC GETKEY
188 CMP #"3"
189 BCS GETKEY
190 STA SYMBOL-$100,X
191 AND #%00001111 1 thru 7
192 CHECKKEY1 CLC
193 CHECKKEY2 RTS
194
195 *=====
196
197 SECTPTR DFB 0
198 SECTABL DFB 0,$E,$D,$C,$B,$A,9,8,

```

```

7,6,5,4,3,2,1,$F
199
200 TBLVALUE = *
201 FLS "FEDCBA9876543210"
202 INV "FEDCBA9876543210"
203 ASC "FEDCBA98765432.."
204 TBLSTAT = *
205 FLS "_PV_?__S"
206
207 IOB = *
208 DFB 1
209 IOBSLOT DFB 6*$10
210 IOBDRIVE DFB 1
211 IOBVOL DFB 0
212 IOBTRACK DFB 0
213 IOBSECTR DFB 0
214 DA DCT
215 DA $2000
216 DFB 0,0
217 DFB 1
218 IOB^STAT DFB 0
219 DFB 0
220 DFB 6*$10
221 DFB 1
222 DCT DFB 0,1
223 DA $D8EF
224
225 MSG1 = *
226 INV "DISK CHECK-UP"
227 DFB CR,CR
228 DO LOWCASE
229 ASC " Copyright [C] 1984
    Alexandre Aurane"
230 ELSE
231 ASC " COPYRIGHT [C] 1984
    ALEXANDRE AURANE"
232 FIN
233 DFB CR,CR
234 DO LOWCASE
235 ASC " Slot:_"
236 MSGSLOT ASC "6"
237 ASC "   Drive:_"
238 ELSE
239 ASC " SLOT:_"
240 MSGSLOT ASC "6"
241 ASC "   DRIVE:_"
242 FIN
243 MSGDRIVE ASC "1"

```

```

244 ASC " "
245 DFB CR,CR
246 INV " PISTE"
1111111111111111222"
247 DFB CR
248 INV " 0123456789ABCDEF01
23456789ABCDEF012"
249 DFB CR
250 INV "S0"
251 DFB CR
252 INV "E1"
253 DFB CR
254 INV "C2"
255 DFB CR
256 INV "T3"
257 DFB CR
258 INV "E4"
259 DFB CR
260 INV "U5"
261 DFB CR
262 INV "R6"
263 DFB CR
264 INV " 7"
265 DFB CR
266 INV " 8"
267 DFB CR
268 INV " 9"
269 DFB CR
270 INV " A"
271 DFB CR
272 INV " B"
273 DFB CR
274 INV " C"
275 DFB CR
276 INV " D"
277 DFB CR
278 INV " E"
279 DFB CR
280 INV " F"
281 DFB 0
282
283 MSG2 DO LOWCASE
284 ASC "Encore?"
285 ELSE
286 ASC "ENCORE?"
287 FIN
288 DFB 0
289 END

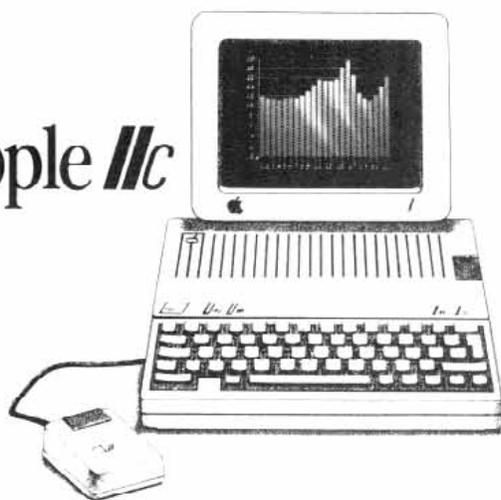
```

Récapitulation

0900-	20	93	FE	20	58	FC	A2	0E	0980-	F5	85	48	A2	26	20	C2	09	0A78-	D9	D2	C9	C7	C8	D4	A0	DB	
0909-	20	4A	F9	8D	65	0A	F0	06	0989-	B0	02	F0	B6	20	58	FC	4C	0A80-	C3	D0	A0	B1	B9	B8	B4	A0	
0910-	20	F0	FD	E8	D0	F5	A2	88	09C0-	EA	03	8E	CD	09	9E	D2	09	0A98-	A0	C1	CC	C5	D8	C1	CE	C4	
0918-	20	C2	09	80	15	F0	0E	48	09C9-	A9	A8	D0	05	A0	80	06	49	0A90-	D2	C5	A0	C1	06	D2	C1	CE	
0920-	09	B0	8D	A3	0A	68	0A	0A	09D0-	01	3D	80	06	A9	06	20	A8	0A98-	C5	8D	8D	A0	D3	CC	CF	D4	
0928-	0A	0A	8D	51	0A	A2	97	20	09D8-	FC	A0	00	C0	10	EE	8D	10	0AA0-	BA	DF	DF	B6	0F	DF	A0	A0	
0930-	C2	09	B0	70	F0	0C	C9	03	09E0-	C0	48	A9	A0	9D	00	06	8D	0AAB-	A0	A0	C4	D2	C9	D6	C5	BA	
0936-	B0	F5	8D	52	0A	09	B0	8D	09E8-	10	C0	68	C9	9B	F0	16	C9	0AB0-	DF	DF	B1	DF	DF	8D	8D	20	
0940-	82	0A	A9	00	3D	54	0A	8D	09F0-	8D	F0	11	E0	80	90	C8	C9	0ABB-	20	10	09	13	14	05	20	20	
0948-	53	0A	A0	FF	8C	06	0A	EE	09F8-	B1	90	C7	C9	B8	80	C3	9D	0AC0-	20	20	20	20	20	20	20	20	
0950-	06	0A	AC	06	0A	C0	10	B0	0A00-	80	05	29	0F	18	60	00	00	0AC8-	20	31	31	31	31	31	31	31	
0958-	41	B9	07	0A	8D	55	0A	69	0A08-	0E	0D	0C	0B	0A	09	08	07	0AD0-	31	31	31	31	31	31	31	31	
0960-	08	20	C1	FB	AD	00	C0	10	0A10-	06	05	04	03	02	01	0F	44	0AD8-	31	32	32	32	32	3D	20	20	30
0968-	0B	C9	9B	DD	07	3D	10	C0	0A18-	45	44	43	42	41	79	78	77	0AE0-	31	32	33	34	35	36	37	38	
0970-	F0	32	F0	8C	A9	0A	A0	50	0A20-	76	75	74	73	72	71	70	06	0AE8-	39	01	02	03	04	05	06	30	
0978-	20	D9	03	80	10	AC	78	05	0A28-	05	04	03	02	01	39	38	37	0AF0-	31	32	33	34	35	36	37	38	
0980-	38	B9	17	0A	AC	54	0A	C8	0A30-	36	35	34	33	32	31	30	C6	0AF8-	39	01	02	03	04	05	06	30	
0988-	C8	91	28	D0	C2	A0	5D	0A	0A38-	C5	C4	C3	C2	C1	B9	B8	B7	0B00-	31	32	8D	13	30	8D	05	31	
0990-	4A	4A	4A	4A	08	B9	47	0A	0A40-	B6	B5	B4	B3	B2	AE	AE	5F	0B08-	8D	03	32	8D	14	33	8D	05	
0998-	D0	EA	EE	54	0A	AD	54	0A	0A48-	50	56	5F	7F	5F	5F	5F	53	0B10-	34	8D	15	35	8D	12	36	8D	
09A0-	C9	23	90	A6	A2	1E	8D	15	0A50-	01	60	01	00	00	00	61	0A	0B18-	20	37	8D	20	38	8D	20	39	
09A8-	0B	F0	06	9D	00	06	E8	D0	0A58-	00	20	00	00	01	00	00	60	0B20-	8D	20	01	8D	20	02	8D	20	
									0A60-	01	00	01	EF	D8	04	09	13	0B28-	03	8D	20	04	8D	20	05	8D	
									0A68-	0B	20	03	0B	05	03	0B	2D	0B30-	20	06	00	C5	CE	C3	CF	D2	
									0A70-	15	10	8D	8D	A0	C3	CF	D0	0B38-	C5	BF	00						

Tout ce que vous avez toujours voulu savoir...

Apple //c



Macintosh™



Apple et GMS c'est une rencontre. Apple c'est toute une gamme d'ordinateurs personnels pour professionnels.

L'Apple IIe et son jeune frère compact l'Apple IIc. Ils sont très sérieux pour la

gestion, la tenue des stocks ou le traitement de texte...

Et puis il y a Macintosh et sa souris. On clique sur la souris, on appelle le programme. Tout un menu est affiché par symboles, les éléments sont

simples et les combinaisons infinies. Enfin il y a GMS, une équipe de professionnels qui vous accueilleront et vous conseilleront personnellement. Alors tout ce que vous avez toujours voulu savoir... 345.28.52.

INFORMATIQUE GMS

Informatique GMS 212-214 avenue Daumesnil 75012 Paris.


Apple

Bibliographie

Alexandre Avrane et Alexandre Duback

La réputation de Pom's vient-elle de faire un saut quantitatif ? La presse micro-informatique devient-elle folle ? Nous ne savons quelle en est la cause principale, mais nous avons reçu depuis le début du mois de septembre une vingtaine de livres nouveaux, portant pour la plupart d'entre eux explicitement sur l'Apple. Nous avons donc dû effectuer une sélection, les autres ouvrages étant purement cités à la fin de la bibliographie.

Expériences d'Intelligence Artificielle en Basic, de John Krutch, Editions Eyrolles - 119 pages.

Le plus surprenant avec l'Intelligence Artificielle (prononcez "IA" pour faire dans le vent), c'est que personne n'y est indifférent: les uns y voient l'une des plaies d'Égypte, les autres y deviennent le point de départ du futur âge d'or.

Ce livre tente de démythifier ce sujet et propose au lecteur de se lancer immédiatement dans les travaux pratiques, en exécutant quelques programmes en Basic. Comme l'annonce le titre, il ne s'agit pas d'être exhaustif, mais de démontrer qu'il n'est pas nécessaire de disposer d'un ordinateur de grande puissance et de maîtriser un langage spécialisé (type LISP) pour aborder ce thème.

Les sujets présentés sont variés: méthodes d'évaluation des programmes de jeu de dames ou d'échecs (minimax, alpha-beta), résolution d'analogies ("Socrate est un homme; tous les hommes sont mortels, donc..."), générateur de poésies (création de quelques alexandrins, malheureusement souvent très sybillins), voire de romans policiers.

Le livre termine ces expériences en exposant les mécanismes du célèbre programme DOCTOR (parfois diffusé en France sous le nom de FREUD), où l'ordinateur engendre les questions d'un psychanalyste en fonction des réponses précédentes du patient humain.

Les programmes inclus sont très structurés, de manière à faciliter les modifications. Ils sont destinés à l'origine au Basic du TRS-80, mais une annexe permet une traduction en Applesoft. Il est cependant dommage qu'ils n'aient pas été traduits en français.

20 progiciels-outils pour l'Apple II, par Jean-Louis Marx & Alain Thibault, Editions du P.S.I. - 248 pages - 122 FF.

Un des problèmes de l'Apple, c'est

-ne sachiez pas la richesse de sa bibliothèque de logiciels. En effet, le client potentiel a souvent la crainte de choisir un programme qui ne lui conviendrait pas et, devant le nombre de progiciels disponibles sur Apple et donc la possibilité de se "tromper", il choisit parfois un autre matériel où il n'existe qu'un seul programme: "s'il n'y en a qu'un, c'est qu'il est bon!".

Ce livre se propose d'aider le futur acheteur à s'y retrouver parmi la galaxie de progiciels disponibles sur Apple. Seuls les produits de grande diffusion en France sont abordés: ceux-ci sont présentés en quatre familles:

1) les tableurs: Visicalc et Multiplan;
2) les traitements de texte: Apple Writer, Ka-Texte, WordStar MailMerge et Plume II;



3) les gestionnaires de fiches: Visifile, PFS:File Report, Visidex, L'Organisateur, Omnis, dBase II et CX-Base 200;

4) les autres: Factor, AppleWorld, Graphor, Visitrend/Visiplot, PFS:Graph et Typing Tutor.

On remarque donc de grands absents (DB Master, Magic Window, etc.). Les nouveaux produits (apparus depuis quelques mois: Jane, Apple Works, Epistole...) manquent également (mais tout va si vite!).

Chacun des logiciels est évalué de manière identique selon une dizaine de critères (documentation, performance à l'impression, sécurité, etc.) que le lecteur doit, par la suite, pondérer selon l'importance qu'il leur attribue.

Cas rare dans ce type de comparatif, la configuration minimum nécessaire à chaque produit est précisément détaillée (ce qui a tout de même une certaine importance si l'on envisage d'utiliser dBase II par exemple).

Certainement utile aux gestionnaires désireux de choisir rapidement un progiciel, ce livre aurait cependant

mérité d'être étoffé par l'analyse de quelques produits supplémentaires.

Applesoft Basic for the Apple II & //e, par Lois Graff & Larry Joel Goldstein, chez R.J. Brady Co. - 328 pages - \$20.75.

Voici un excellent livre d'initiation à l'Applesoft: clair, précis et avec de nombreux programmes-exercices. Mais son problème, vous l'avez deviné, c'est la langue: l'anglais est rarement le favori des débutants en informatique. D'autre part, il existe déjà de nombreux ouvrages similaires parus en France.

Cependant ce livre n'est pas sans intérêt, même pour les anglophobes, car il apprend au lecteur à analyser un problème et à structurer la solution, au lieu de se précipiter devant le clavier pour taper quelques centaines de lignes mal comprises.

Parallèlement, les instructions du Basic Applesoft et du DOS 3.3 sont étudiées; malheureusement certaines, telles que WAIT, HIMEM, POS et l'&, sont (volontairement?) ignorées. Une carte de référence plastifiée est jointe.

P-Source, a guide to the Apple Pascal System, par Randall Hyde, chez Reston Publishing (Prentice Hall) - 462 pages - \$24.95.

Débutants s'abstenir! Ce livre n'est pas un guide d'initiation au Pascal sur Apple, mais s'adresse aux programmeurs systèmes désirant connaître leur machine sur le bout de leurs 16 doigts.

Une excellente connaissance du Pascal est nécessaire avant d'en aborder la lecture: en particulier, ce livre ne commence pas son exploration là où se terminent les manuels d'Apple, mais bien au-delà. Dès la première page, il part défricher l'allocation mémoire de l'interpréteur p-code, puis enchaîne gaillardement vers une étude ethnologique des variables en Pascal.

Après quelques conseils pour accélérer les programmes, ce manuel propose plusieurs utilitaires en assembleur pour la mise au point des procédures, puis fournit une étude détaillée de chaque instruction machine du p-code. Enfin, les routines de l'interpréteur sont décortiquées, et quelques dernières astuces de programmation achèvent le tout.

Cet ensemble représente, à ma connaissance, le manuel le plus détaillé sur le fonctionnement interne du Pascal. Il est certainement dommage qu'il manque d'approche pé-

dagogique, mais ses 450 pages bien tassées pourront éviter à certains de faire un stage de perfectionnement à l'Université de San Diego.

Visicalc Applications, de Stanley Trost, traduit par Jean-Pierre Loison, chez Sybex - 274 Pages - 148 FF.

Enfin une traduction bien faite: cela vaut le coup d'être souligné, le travail étant trop souvent effectué par des tâcherons mal payés. Ce livre s'adresse aux utilisateurs de Visicalc déjà compétents et leur apporte de nombreux tableaux commentés et accompagnés de listes permettant de les reconstituer.

Clefs pour Visicalc, par Jean-Louis Marx et Alain Thibault, Editions du PSI - 101 pages - 100 FF.

Clefs pour Multiplan, par Jean-Louis Marx et Alain Thibault, Editions du PSI - 127 Pages - 100 FF.

Ces deux ouvrages comptent parmi les seuls que nous connaissons, en langue française, dans lesquels l'utilisateur peut trouver un certain nombre de "trucs" d'utilisation. Il s'agit d'ouvrages de référence, où toutes les commandes sont analysées dans l'ordre alphabétique et commentées en détail.

Nous conseillons l'utilisation de ces

"Clefs" pour se renseigner sur telle ou telle commande, ou trouver la réponse à une question précise.

Memento Multiplan, par P. Bonnet et M.T. Dinh, chez Edimicro - 108 pages - 78 FF.

Nous retrouvons ici les mêmes objectifs que dans les deux ouvrages précédents, mais on a du mal à voir ce que ce livre apporte de plus que la documentation du programme Multiplan. Reportez-vous donc aux ouvrages précédents.

The Visicalc Applications Book for the Apple IIe, par Jack Grushcow, chez Reston Publishing (Prentice Hall) - \$16.45.

Ce livre en anglais comporte une première partie d'apprentissage de Visicalc, suivie de six exemples approfondis: suivi de règlements, analyse financière d'entreprise, prévision, plan, trésorerie et gestion de portefeuille. Ouvrage bien fait; malheureusement, les exemples ne sont pas forcément transposables à l'environnement français.

Autres titres

dBase II sans embûches, par G. Grigorieff - Eyrolles - 176 pages -

115 FF.

Du style avec Wordstar, par R.B. White Jr. - Eyrolles - 232 pages - 150 FF.

Forth pour micros, par J.M. de Geeter - Eyrolles - 192 pages - 90 FF.

L'art des bases de données, par S. Miranda et J.M. Busta - Eyrolles - 248 pages - 180 FF.

Les systèmes de gestion de bases de données, par J. Akota - Eyrolles - 320 pages - 170 FF.

The Apple IIe personal computer for beginners, par Seamus Dunn et Valerie Morgan - Prentice Hall - 251 Pages - \$12.95.

Handbook of Applesoft Basic for the Apple II and IIe, par Roy Earl Myers et David Schneider - Prentice Hall - 321 pages - \$16.45.

Apple programming for learning and teaching (over 50 application programs), par Frederick Bell - Reston (Prentice Hall) - 305 pages - \$ 22.05.

Le Basic bien programmé, par A.P. Stephenson - MicroDunod - 120 pages - 65 FF.

Basic Microsoft et Basic ANSI, par M.Maiman - MicroDunod - 165 pages - 80 FF.

Les nouvelles disquettes de Pom's

Alexandre Duback

Pom's vous présente aujourd'hui trois nouvelles disquettes: un éditeur plein écran, la version 2 du Disk Manager (enfin) et une disquette Macintosh, la première d'une longue série. Saviez-vous que les disquettes de Pom's sont les disquettes les plus vendues en France ?

Disk Manager version 2

Prix: 450 francs. Echange gratuit avec DM1.

La version 2 du Disk Manager remplace enfin à partir de ce numéro la version 1, dont l'expérience a prouvé qu'elle n'était hélas pas exempte de bugs. Afin de ne pas léser les acheteurs de la première version, Pom's offre gratuitement la seconde version à tout acheteur de la première: renvoyez (après en avoir fait une copie de sécurité) la disquette originale reçue de Pom's et une enveloppe au format 23*16 avec votre adresse et timbrée à 3,50 francs. Vous recevrez gratuitement la nouvelle version.

Editeur plein écran

De C. Leyo. Prix: 150 francs.

Cette disquette contient un excellent programme pour éditer un pro-

gramme Basic en plein écran. Vous listez votre programme Basic, le faisant défiler à tout instant en marche avant ou en marche arrière, l'arrêtant quand vous voulez. Vous effectuez les changements directement sur l'ensemble de l'écran, les commandes étant lancées (comme dans le Program Line Editor - Pom's 1) par des touches de contrôle.

Les commandes sont nombreuses: début de programme, fin de programme, marche avant, marche arrière, destruction, insertion, entrée de caractères de contrôle, effacement de fin de ligne, accès aux commandes DOS, recherche de chaînes... Les caractères de contrôle dans les instructions apparaissent en inverse.

Une particularité que nous apprécions beaucoup: un programme spécifique permet à l'utilisateur de choisir lui-même les caractères de contrôle des commandes.

Disquette Macintosh

Prix: 150 francs.

Avec le développement du dossier Macintosh, il était inévitable que nous vous proposons aussi une dis-



quette Macintosh. Cette première disquette comporte les programmes publiés dans les numéros 14 et 15, ainsi que le programme Disk Copy (copie rapide de disquettes avec un seul lecteur) et la police de caractères Cairo, grâce à laquelle vous pouvez vous exprimer en images (voir le cahier Macintosh). Jusqu'à saturation de cette disquette, nous y mettrons automatiquement tous les programmes Mac de Pom's, ainsi que de petites surprises, utilitaires, polices de caractères, ...

Disquette Pascal

Prix: 80 francs

Suite aux requêtes de nombreux lecteurs, nous avons regroupé dans une disquette unique tous les programmes Pascal publiés dans Pom's depuis le début. Cela vous évitera de transférer les programmes du DOS 3.3 vers le Pascal.

Les fichiers TEXT sont au recto de la disquette et les fichiers CODE au verso; ainsi, il n'y aura pas de problème de compilation.

Apple, quoi de neuf ?

Après un Macintosh de 512K, né quatre mois avant terme, que prépare-t-on chez Apple ? Les meilleurs limiers de la presse micro-informatique américaine sont sur le coup. Pour l'heure, le silence, de rigueur à Cupertino, laisse libre cours à toutes les supputations.

Côté Macintosh, on prévoit des lecteurs de disquettes double face; la capacité passerait de 400 à 800K. D'autre part, le plus exaspérant des problèmes de Macintosh, la lenteur, est partiellement dû au système d'exploitation qui figure sur chaque disquette programme. Le fait d'avoir sur la disquette une partie du système est par contre un avantage car on peut aisément en changer pour, par exemple, adapter un programme à un clavier AZERTY. Les projets en cours consisteraient donc à intégrer les systèmes au Macintosh, pour en débarrasser la disquette et obtenir ainsi un accès plus rapide aux informations. Une fois résolus les problèmes de capacité et de vitesse, Apple pourrait, dit-on, se pencher sur l'adaptation de la couleur au Mac...

Côté Apple II, on attend pour la fin de l'année les nouveaux accessoires du //c et, notamment, l'écran à cristaux liquides disposant de 25 lignes d'affichage. Si on ne l'a pas vu plus tôt, c'est simplement que son fabricant, le Japonais Sharp a subi un retard de six mois dans la mise au point de cet accessoire. Pour le //e, on reste muet à Cupertino sur les développements du processeur 65816, qui permettrait de le convertir en une machine 16 bits. Problèmes de mise au point ou approche du dénouement ?

D'autres études portent sur la capacité des lecteurs de disquettes. Au moment où IBM parvient à stocker 1,2 Mo sur une disquette de 5 pouces 1/4, la mini-capacité du lecteur du II (143K), devient problématique. On peut donc s'attendre très prochainement à voir cette capacité augmenter notablement. La difficulté consiste seulement à maintenir la compatibilité avec les disquettes existantes. En attendant, voici des nouveautés pour tous les Apples.

Moniteur couleur

La nouveauté de l'automne, c'est aux Etats Unis le nouveau moniteur couleur sorti par Apple : le moniteur 100. Esthétiquement, il ressemble beaucoup au moniteur monochrome du //e. Son écran est inclinable de la même façon mais, cette fois, un petit moteur se charge de l'opération (le tube est beaucoup plus lourd) et il suffit de presser une touche pour que l'écran s'incline vers le haut ou vers

Micro-informations

Jean-Michel Gourévitch

le bas. Pour faire fonctionner ce moniteur, il faut bien sûr disposer d'une carte couleur (ou d'une carte Chat Mauve).

Ce moniteur est vendu aux Etats Unis un prix respectable : 600 dollars, soit, pas beaucoup moins de 6000 Francs au cours actuel du billet vert. L'explication d'Apple : il s'agit d'un des rares moniteurs couleur à fournir en texte la qualité d'un moniteur monochrome, particulièrement en 80 colonnes. On peut désormais se contenter d'un seul moniteur pour les jeux, les graphiques ou les activités professionnelles. En 80 colonnes, un switch permet de choisir la couleur de l'affichage : bleu, jaune, etc... Un bouton permet de passer du monochrome à la couleur.

Disques durs

Pour disposer d'un stockage de données, ou de programmes accessibles rapidement, le disque dur est la solution. Les prix commencent à baisser sensiblement. Ainsi, le modèle présenté par la firme britannique Dering propose une capacité de 10 Mo pour un prix de 1095 Livres, soit environ 14300 F. Pas cher l'octet...

Côté clavier

Vous pouvez remédier à la faible capacité du buffer du clavier de l'Apple //e: il vous suffit d'intercaler le **Type Ahead Buffer TAB1** de Trac Systems Corp entre votre clavier et sa prise sur la carte mère. Pour seulement 45 dollars.

Et si l'on pouvait se passer carrément du clavier ? Pourquoi ne pas entrer les données en donnant de la voix ? C'est possible avec le **Voice Input Module** de Cascade Graphics. Il suffit de parler pour que l'enregistrement des instructions se fasse. Vous dites "Catalog" et la disquette affiche son contenu à l'écran. Le système se compose d'une carte, d'un casque, d'un micro et de logiciels. Il peut être utilisé sur l'Apple II, mais aussi sur le //e (il faut dans ce cas enlever quelques circuits). Principal défaut : il ne supporte que la VO (c'est à dire l'anglais) et peut se révéler sourcilieux sur la prononciation.

La pince à tiercé

Enfin, un petit instrument vendu 69 francs permet (à ceux qui ont épuisé les charmes de la pince à tiercé et qui ne veulent pas se lancer dans la construction du double switch) de doubler la capacité de leurs disquet-

tes. Il découpe automatiquement au bon endroit l'encoche qui permet au lecteur d'accepter la disquette pour enregistrement. Attention, il vaut mieux vérifier ensuite la surface de la deuxième face avec un utilitaire (genre Locksmith), et ne pas l'utiliser pour des informations capitales: la garantie du fabricant de disquettes ne s'applique pas au verso.

Des yeux pour le Macintosh

BIP distribue la caméra de digitalisation **Microneye**, vendue 5.300 FF HT avec son interface et son logiciel. Cette caméra digitalise une image (qui doit être contrastée) et la transforme en image MacPaint, donc modifiable à loisir. Elle fonctionne aussi à la lumière artificielle et à l'infrarouge. On peut reprendre un document de la taille d'une photo d'identité. Parfait pour une utilisation avec le gestionnaire de fiches MacBase de Contrôle X, qui permet de mettre des images dans les fiches.

Des yeux pour tous

Palette, l'imageur électronique couleur Polaroid, permet d'obtenir des diapositives instantanées ou des photos papier à partir de nombreux logiciels courants (TGS, Décisionnel Graphique, Visiplot, etc...) ou d'images numérisées.

Des cordes vocales pour l'Apple //

Ediciel vient de sortir la première carte de synthèse vocale en langue française pour Apple, le **Porte-Palette**. Non seulement elle parle français, mais pour lui faire dire à peu près correctement "I write on the blackboard", il faut écrire "Aille raille onn ze blakbourde"! Nous reparlerons de cette carte.

Un écran portable pour le //c

Après avoir déjà prouvé son originalité avec le Stadu.P, un Apple //e portable, I.E.F. a présenté au Sicob un moniteur portable pour //c avec une batterie incorporée d'une autonomie d'environ 4 heures. Certes, cet écran est sensiblement plus volumineux que le //c, mais il comporte une batterie, ce qui ne sera pas le cas de l'écran plat du //c toujours attendu en France. Prix 2.590 FF HT. Voir la photo de couverture de ce numéro.

Des programmes pour une souris

Il n'aura pas fallu attendre très longtemps pour voir apparaître des programmes utilisant la souris équipant le //e et le //c.

Version Soft (le créateur d'Epistole) présente ainsi **VersionCalc**, le premier tableur souris pour le //c. Il possède une fonction graphique et permet de cumuler des tableaux et des graphiques. Le tout s'effectuant avec la souris et des menus qui rappellent furieusement le Macintosh. Attention, l'utilisation de ce logiciel sur un //e (même avec une souris) peut produire des effets surprenants.

Dazzle Draw, de Broderbund, présente un écran rappelant lui aussi le Macintosh et permet, à condition de disposer d'une souris (ou d'un joystick, d'une tablette Koala, etc...) et de 128K, de dessiner à l'écran, sur plusieurs fenêtres de surcroît. Le tout pour un prix "canon": 50 dollars seulement.

Enfin, dernière conquête de la souris, **Summergames** d'Epyx, présente tout simplement les jeux olympiques. Epreuves de natation, de saut à la perche, de tir, etc... On choisit la nationalité (18 pays possibles) des concurrents avec la souris, on joue avec le même instrument. Après chaque victoire, l'Apple // vous joue l'hymne national du gagnant, et calcule la moyenne des médailles obtenues. Graphiques et couleurs extraordinaires. Il y a même une cérémonie d'ouverture!

Plus classique

Papyrus est la version francisée d'Homeword, un traitement de textes avec icônes. Certes, ce n'est pas le plus sophistiqué des traitements de textes. Pourtant, sa facilité d'emploi en fait un logiciel hors du commun. D'abord, parce qu'on peut, pour chaque action, choisir un icône sur lequel on se déplace avec le curseur. Mais, heureusement, des codes permettent aux clavistes expérimentés de court-circuiter ces icônes et de gagner en rapidité. Un bon point. Ensuite, ce traitement de textes permet d'afficher à l'écran le texte tel qu'il sera imprimé. C'est suffisamment rare pour mériter un second bon point. Enfin, lorsqu'on lui tape un accent circonflexe suivi d'une voyelle, il vous affiche à l'écran la voyelle avec son accent circonflexe. Bravo!

Si on ajoute que Papyrus permet d'insérer dans un texte un document qui se trouve déjà enregistré sur disquette (Macwrite, le premier traitement de texte du Macintosh, ne le fait pas directement), qu'il permet les caractères gras, soulignés, l'effacement des mots et des lignes, et peut

même automatiquement numéroter des paragraphes indentés, on peut s'estimer satisfait du bilan. Tout au plus peut-on lui reprocher, en saisie de textes, une certaine confusion à l'écran. Lorsqu'on change de caractères (passage en souligné, gras, etc...) et qu'il faut revenir au caractère normal, ce passage s'inscrit en clair à l'écran, ainsi que les RETURNs. Reste que pour taper simplement une petite lettre, ce programme n'a guère de concurrent. D'autant qu'il ne coûte que 850 francs. Fortement recommandé.

Parmi les programmes intégrés, à signaler **Magic Officed** d'Artsci, qui comprend un traitement de textes, un tableur, des graphiques et un dictionnaire de correction, le tout clairement présenté en tiroirs, dossiers, avec des possibilités de couper-coller entre documents. Bien mieux présenté qu'Appleworks, mais il ne lui manque que Quickfile: il n'y a pas de gestionnaire de fiches.

Depuis peu, **the Team** de U-Microcomputers, offre un traitement de textes, un gestionnaire de fiches, des possibilités de calcul et un graphisme. Le tout fonctionnant en Pascal pour Apple II Plus, //e et //c.

A remarquer encore le logiciel de correction d'orthographe automatique **Sensible Speller** de Sensible Software, contenant 80000 mots sur un seul disque (une performance sur 143K!). C'est malheureusement en anglais, et il reste encore à inventer un de ces logiciels de correction en français. Le Sensible Speller fonctionne avec la plupart des programmes de traitement de textes et coûte quelques 125 dollars.

Le programme **Spelling Handler**, de ALS (Sunnyvale, Californie) fait encore mieux avec 90 000 mots!

Jeux d'aventures

Enfin, deux jeux d'aventure de création française: **Epidémie** et **Paranoïak**. Du niveau des jeux d'aventure américains selon les critères du scénario, des graphiques et du son. Edités par Froggy Software et distribués par Shift Editions. Fonctionnent sur II Plus, //e et //c. Prix: 350 FF TTC pièce.

Imprimante à marguerite

Kardex vient de sortir une machine à écrire/imprimante à marguerite économique à 5.000 FF HT prix public conseillé, l'**AE 355**, offrant de nombreuses possibilités. Elle fonctionne en parallèle, mais on peut acquérir un boîtier pour la connexion série avec un buffer de 2K (1.500 FF HT) ou 4K (1.700 FF HT). Une marguerite est fournie d'origine, mais 9 autres sont disponibles. Vitesse d'impression: 14 cps. Trois pas d'écri-

ture: 10, 12 et 15.

Un modèle récent, l'**AE 385**, vous offre pour 1.000 FF HT de plus des caractères gras, un espacement proportionnel et une impression bidirectionnelle.

Deux logiciels un peu particuliers

Le **Biofeedback** de Synapse comporte un bandeau à placer sur le front, contenant des capteurs, et mesure le niveau de stress de l'utilisateur de l'Apple // I. Le résultat s'affiche sur l'écran en courbes et graphiques. Un stress qui commencera avec l'achat: prix 150 dollars.

Le **Smarthome** de Cyberlynx, permet lui d'automatiser un appartement et de le contrôler avec l'Apple //c. En se servant de la souris, et avec des icônes, on contrôle l'électricité et les systèmes d'alarmes antivols. Prix: 499 dollars.

Adresses

Alpha Systèmes - 29, bd. Gambetta - 38000 Grenoble - Tél (76) 43.19.97.

B.I.P. - 13, rue Duc - 75018 Paris - Tél 255 4463.

Broderbund Software - 17, Paul Drive, San Rafael - CA 94903 - USA.

Cascade Graphics Development - 185 lower Richmond Road, Richmond - Surrey - GB.

Contrôle X - Tour Maine Montparnasse - 33, avenue du Maine - 75755 Paris Cedex 15 - Tél 538 9887.

Cyberlynx - 4828 Sterling Drive, Boulder - CO 80301 - USA.

Ediciel Matra Hachette - 22, rue la Boétie - 75008 Paris - Tél 260 0032.

I.E.F. - 193, rue de Javel - 75015 Paris - Tél 828 0601.

Kardex - 201, rue Carnot - 94120 Fontenay sous Bois.

Sensible Software - 24011 Seneca Oak Park - MI 48237 - USA.

Synapse - 5221 Central Avenue Richmond - CA 94804 - USA.

Trac System Corp. - 444 North 3rd Street, suite 201, Sacramento - CA 94814 - USA.

U-Microcomputers - Winstanley Industrial Estate, Long Lane, Warrington - Cheshire WA2 8PR - GB.

Version Soft - 60, rue Castagnary - 75015 Paris - Tél 530 0528.

Froggy Software - 33, avenue Philippe Auguste - 75011 Paris.

Polaroid France - 4, rue J-P Timbaud, B.P. 47 - 78391 Bois d'Arcy Cedex - Tél (3) 460 6166.

Shift Editions - 27, rue du Général Foy - 75008 Paris. ■

Courrier des Lecteurs

Alexandre Duback

Utilisateur d'AppleWriter sur une Oki84, je ne parviens pas à imprimer le "è". L'Oki84 ne semble en effet pas reconnaître le caractère de déplacement à gauche Ctrl-H, utilisable avec les autres imprimantes.

Jean-Marie Hasquenoph - 77500 Chelles

Avec les Oki, le backspace ne fonctionne pas. Par contre, sur mon Oki 92 (ce devrait être la même chose pour une 84), j'obtiens comme vous venez de le voir un "è" en appuyant sur un tréma (non suivi d'une lettre). Pour tester toutes les possibilités de caractères de votre imprimante, créez un fichier avec toutes les touches de votre clavier, en mode normal, puis Shift, puis Ctrl, puis Ctrl-Shift et enfin Esc.

Vous aurez peut-être des surprises avec votre imprimante (dans ce cas, enlevez ce qui "plante" l'imprimante), mais cette recherche systématique peut vous permettre de trouver des solutions à des problèmes tels que le précédent.

Vends Apple II Plus avec carte langage (16K), joystick, paddles, moniteur et deux lecteurs de disquettes. Prix 13.500 F à débattre.

A. Avrane - 124 quai Louis Blériot - 75016 Paris.

Vends Apple //c état neuf encore sous garantie (cause double emploi) avec moniteur et stand moniteur. Prix 11.700 F. Possibilité de souris et second lecteur. Téléphone 956 80 00, poste 643.

Vends Apple II Plus avec moniteur Sanyo vert, deux lecteurs et un contrôleur DOS 3.3, carte communication asynchrone (CCS) et carte langage 16K. Prix 10.000 F TTC.

Vends machine à écrire/imprimante Olympia ESW100-KSR (interface série). Excellent état. Prix 8.000 F TTC (Achat 14.587 F). Bruno Lemaire. Tél (3) 956.80.00.

Vends Apple II Plus avec moniteur Philips ocre, deux lecteurs et un contrôleur DOS 3.3, carte langage Microsoft, carte RVB en option. Prix 8.500 F TTC. Tél (3) 952.59.31.

L'utilitaire APA d'APPLESOFT TOOLKIT comporte un bug. En effet, il est impossible d'obtenir la liste des variables (fonction &XREF) si la programme analysé contient des DATAs. L'addition des deux lignes BASIC ci-dessous au programme LOADAPA autorise la référence croisée dans tous les cas :

```
115 HI = PEEK (115) + 256 * PEEK (116)
      - 6: HIMEM: HI
```

```
315 POKE HI,32: POKE HI + 2, PEEK (HI -
      1170): POKE HI - 1170,HI / 256:
      POKE HI + 1, PEEK (HI - 1171): PO
      KE HI - 1171,HI - 256 * PEEK (HI
      - 1170): POKE HI + 3,201: POKE H
      I + 4,0: POKE HI + 5,96
```

Yvan KOENIG - avenue du stade - 06220 VALLAURIS

Je suis en train d'écrire un programme gestionnaire de fichiers et deux problèmes se posent à moi :

1) Je voudrais que mon programme puisse me dire à tout moment combien de secteurs restent libres sur la disquette : comment faire ?

2) J'aurais également voulu que le RESET agisse comme un RUN. La routine de RUN étant implanté en D566, j'ai converti ce nombre en décimal, et j'en ai poké les poids fort et faible aux adresses 1010 et 1011 : POKE 1010,102 et POKE 1011,213 suivi de CALL - 1169. Ceci provoque bien un RUN à chaque RESET mais les ordres du DOS s'affichent alors à l'écran et ne fonctionnent plus. Comment y remédier ?

Olivier LEDOUX - 135/12 rue du Gal Leclerc - 59790 RONCHIN

1) Dans le Pom's 2 en page 17, figurent différentes modifications du DOS, en particulier le "patch" nécessaire pour obtenir le nombre de secteurs disponibles dans le CATALOG. Vous trouverez ci-dessous le listing qui réalise la modification. Il suffit alors d'initialiser votre disquette avec ce nouveau DOS.

```
10 REM *****
    * SECTEURS LIBRES *
    * A CHAQUE *
20 REM * CATALOG *
    * *****
30 REM
```

```
35 HOME : LIST 10,20
100 Y* = "AB84:5B N ADC3:20 00 B6
      N B600:A2 0C 20 4A F9 A9 00
      85 40 85 41 A0 CB 18 B9 F2
      B3 F0 0E 0A 90 FB 48 E6 40 D
      0 02 E6 41 60 18 90 F0 88 D0
      E9 A6 40 A5 41 AC 00 E0 C0
      20 D0 07 20 1B E5 20 2F AE 6
      0 20 24 ED 20 2F AE 60"
110 GOSUB 63900
120 PRINT "OK": END
63900 REM
ROUTINE MONITEUR
63905 REM APPLESOFT DE S.H.LAM
63910 Y* = Y* + " N D9C60"
63920 FOR I = 1 TO LEN (Y*)
63930 POKE 511 + I, ASC ( MID* (
      Y*,I,1) ) + 128
63940 NEXT I: POKE 72,0: CALL -
      144
63950 RETURN
```

2) Les POKEs que vous utilisez provoquent en effet un RUN lors du RESET. Toutefois, le RESET "déconnecte" les entrées/ sorties du DOS. CALL 1002 au début de votre programme résoudra votre problème. Cependant, il est bon de préciser qu'un RESET dans un programme qui travaille sur disquette n'est pas prudent, surtout en cours d'écriture...

Je recherche un logiciel d'astrologie professionnelle exploitable sur Apple //e. Je sais qu'il en existe un en français. Pouvez-vous me le confirmer ?

Joël Moreau - 7 rue J. Moeeyse - 44100 Nantes

Peut-être un lecteur peut-il nous répondre ?

J'ai fondé à New Delhi une société spécialisée dans la recherche acoustique et l'ethnomusicologie, qui utilise de façon intensive les micro-ordinateurs, Apple entre autres. Je voudrais signaler à tous ceux que ces domaines intéressent que je compte fonder la branche européenne de cette activité d'ici un an environ. Dans cette perspective, j'invite ces personnes à me contacter.

Bernard Bel - ISTAR - 113 Jorbagh - New Delhi 110003 - Inde

Je commercialise un logiciel de décodage des émissions radiotélégraphique (RTTY) au prix de 600 francs. A l'aide d'un récepteur BLU, il est possible de décoder les messages en trois vitesses, les sauvegarder sur disquette et les visualiser sur écran ou imprimante. Les résultats obtenus sont tout à fait satisfaisants et j'invite les personnes intéressées à m'écrire.

Monsieur Guedj Michel - 34 rue Pierre Curie - 93130 Noisy le Sec

10^e

MIAMI
EXPO



10^e CONGRÈS-EXPOSITION DE MICRO-INFORMATIQUE, DU 16 AU 19 FÉVRIER 1985,
PALAIS DES CONGRÈS, CIP, PORTE MAILLOT, PARIS.

EXPOSITION : MICRO-ORDINATEURS / LOGICIELS / DIDACTIQUES / PROGiciels / BUREAUTIQUE / TELEMATIQUE / ROBOTIQUE / INTERCONNEXIONS / PÉRIPHÉRIQUES / ACCESSOIRES / CAO / DAO / EAO / ÉDITION / PRESSE SPÉCIALISÉE / INSTITUTS DE FORMATION / SOCIÉTÉS DE SERVICES / LABORATOIRES DE RECHERCHE. **CONFÉRENCES :** ACHAT D'UN MICRO-ORDINATEUR / LE CONTRAT INFORMATIQUE / LANGAGES : BASIC, PASCAL, MODULA II, C, ADA / SYSTÈMES : VERS UN NOUVEAU STANDARD / COMPRENDRE LA TÉLÉMATIQUE / L'AVENIR DU VIDÉOTEX / INTELLIGENCE ARTIFICIELLE : LES SYSTÈMES EXPERTS / LE LOGICIEL OUTIL DE GESTION : BASES DE DONNÉES - LOGICIELS INTÉGRÉS - TABLEURS - DÉCISIONNELS GRAPHIQUES / MICRO-INFORMATIQUE ET PROFESSIONS. UN PASSEPORT D'UNE VALEUR DE 100 F DONNE ACCÈS À TOUTES LES CONFÉRENCES. CATALOGUE DÉTAILLÉ SUR SIMPLE DEMANDE À **SYBEX**, 6-8, IMPASSE DU CURÉ, 75018 PARIS.

A propos "des trucs pour Apple Writer II et //e" de H. Thiriez dans le Pom's 12, j'ai ajouté quelques mots sur la commande CTRL-V. Il est vrai que le glossaire de la version //e interprète les caractères de contrôle comme en mode direct. Il est donc nécessaire de les faire précéder de CTRL-V dans le glossaire. Or, pour des raisons évidentes, il n'est pas possible d'entrer CTRL-V dans un texte par le moyen habituel. On a donc prévu sur la disquette un fichier (CONTROLV) contenant le seul caractère CTRL-V pour pouvoir l'insérer dans un texte qui peut servir de glossaire. Il y a toutefois un moyen qui évite les accès répétés à la disquette. Il suffit d'utiliser n'importe quel autre symbole à la place de CTRL-V. * par exemple. En fin de travail, on remplacera toutes les "*" par des CTRL-V en utilisant la commande: CTRL-B puis CTRL-F /*CTRL-V. En effet, l'instruction CTRL-F n'interprète pas le CTRL-V mais se contente de l'insérer.

Daniel HIRST - 13090 AIX

Je vous propose un petit programme Basic complémentaire de la routine de comparaison de programmes Basic de Gérard Michel, publiée dans le Pom's 11 (DIF.OBJ).

Ce programme demande les noms des programmes à comparer et crée un fichier EXEC dont il lance l'exécution pour réaliser la comparaison.

Gérard Hisleur - 38320 Eybens

```
10 HOME : HTAB 5: INVERSE : PRINT "COMPARAISON DE FICHIERS BASIC"
20 VTAB 7: HTAB 1: PRINT "FICHIER 1 : "
30 VTAB 10: HTAB 1: PRINT "FICHIER 2 : "
40 NORMAL : VTAB 7: HTAB 13: INPUT "" ; F1#
50 VTAB 10: HTAB 13: INPUT "" ; F2#
60 D# = CHR# (4) : ONERR GOTO 70: PRINT D# ; "DELETE DIF.EXEC"
70 POKE 216,0 : PRINT D# ; "OPEN DIF.EXEC" : PRINT D# ; "WRITE DIF.EXEC"
80 PRINT "BLOAD DIF.OBJ" : PRINT "LOAD " ; F1# : PRINT "POKE 103,PEEK(175) : POKE 104,PEEK(176) : PRINT "LOAD " ; F2# : PRINT "CALL 37500"
85 PRINT "CALL 979" : REM PATCH A. AVRANE
90 PRINT D# ; "CLOSE" : PRINT D# ; "EXEC DIF.EXEC"
```

Disk check-up...



N.D.L.R.

1) Plusieurs lecteurs nous ont signalé qu'ils avaient résolu les problèmes concernant la copie sur imprimante des écrans de présentation des disquettes Pom's. La solution adoptée consiste alors toujours à envoyer à l'imprimante les codes permettant de supprimer l'affichage simultané à l'écran.

Nous n'avions peut-être pas assez insisté sur ce point dans les numéros précédents, mais il est certain que, si l'interface dont vous disposez affiche à l'écran en même temps qu'elle envoie les caractères à l'imprimante, l'écran ainsi recopié est simultanément modifié et ne peut correspondre à l'écran original.

Si tel est votre cas, il suffit donc de rajouter un PRINT du code de contrôle supprimant cet affichage sur votre matériel, dans la ligne du programme MENU des disquettes qui appelle la routine de hard-copy (généralement la ligne 70). Cette instruction PRINT "code de contrôle" doit venir juste avant le CALL 37989.

2) Plusieurs lecteurs nous ont par ailleurs retourné des disquettes "détruites" à la suite d'une fausse manoeuvre sur laquelle il peut être bon d'insister un instant. En l'occurrence, les personnes concernées ont tapé "EXEC Fichier", alors que "Fichier" n'était autre qu'un fichier source en assembleur sauvé sous format TEXT. Précisons donc que tout fichier TEXT (précédé d'un T au catalogue) n'est pas pour autant un fichier EXECutable. En particulier, il ne faut JAMAIS faire un EXEC d'un fichier source d'assembleur. La revue précise toujours quelles sont les opérations permettant de mettre en route un programme en langage-machine, et la présentation des disquettes rappelle également à quoi correspondent les différents fichiers et comment il faut les utiliser. En suivant ces indications, vous éviterez souvent de détruire vos disquettes...

Précisions et errata

Moniteur Etendu du Pom's 8

A l'issue d'un long débat, il semble que la seule solution simple pour résoudre le petit problème du buffer d'entrée dans ce programme passe par une modification du programme-source comme suit :

- Ligne 160 : LDX #0 au lieu de LDX #1 (soit 8D5F: 00 dans le code-objet).

- Ligne 163 : NOP au lieu de INY (soit 8D66: EA dans l'objet).

- Insérer une ligne 897 INC CH avant la ligne 897 originale (JSR UP).

Bloc-notes du Pom's 13

A la ligne 2540, il convient de remplacer ".121" par ".L121" pour que la fonction de destruction des fichiers fonctionne normalement.

ICARE - Pom's 14

La liste des modifications publiée dans le Pom's 14 concernant l'adaptation du programme Tortue du Pom's 6 en vue de son intégration au logiciel ICARE était malheureusement incomplète. Vous trouverez ci-après la récapitulation des opérations à effectuer sur le programme source original pour cette implantation :

- Remplacer la séquence "Suite Instructions" par celle publiée dans le Pom's 14
- Supprimer les lignes 579 à 612
- La ligne 573 devient BNE E.ERR
- La ligne 271 devient PLUME HEX 00
- Dans la ligne 138, T10 remplace T11
- Supprimer les lignes 134 à 137
- Supprimer les lignes 65 à 90
- Supprimer la ligne 55
- Assembler le nouveau source ainsi constitué à l'adresse \$886E

Disque 64K - Pom's 14

Il faut rajouter un code D9 à la fin de la récapitulation du programme AUTO RWAUXINIT.OBJ en 50 FF.

Disques virtuels 16K et 64K

Si vous possédez deux lecteurs de disquettes, vous pouvez en "fabriquer" un troisième en évitant de mettre le disque virtuel en lieu et place de votre lecteur 2. Il suffit pour cela de désigner le disque virtuel comme "lecteur 3" et de faire admettre au DOS l'existence de ce lecteur. La routine de gestion du lecteur virtuel étant chargée, les POKES suivants réalisent l'opération.

- 1) Disque 16K :
 - POKE 48825,3
 - POKE 43355,3
- 2) Disque 64K :
 - POKE 48372,3
 - POKE 43355,3

pom's

DISQUETTES

HAIFA	(cf. Pom's n° 5)	à	55,00 F
H-BASIC	(cf. Pom's n° 8)	à	150,00 F
MUSIC	(cf. Pom's n° 10)	à	80,00 F
DISK-MANAGER	(cf. Pom's n° 11)	à	450,00 F
DBSTAG (CP/M)	(cf. Pom's n° 11)	à	450,00 F
JEUX A	(cf. Pom's n° 12)	à	80,00 F
JEUX B	(cf. Pom's n° 12)	à	80,00 F
BASICIUM	(cf. Pom's n° 13)	à	150,00 F
DEMO CX SYSTEME 64K	(cf. Pom's n° 13)	à	55,00 F
DEMO JANE 64K	(cf. Pom's n° 13)	à	55,00 F
E.P.E.	(cf. ce numéro)	à	150,00 F
MACINTOSH	(cf. ce numéro)	à	150,00 F
PASCAL	(cf. ce numéro)	à	80,00 F

RECUEILS

N° 1, recueil des revues 1 à 4	à	130,00 F
Disquettes d'accompagnement des numéros 1 à 4	à	150,00 F
N° 2, recueil des revues 5 à 8	à	130,00 F
Disquettes d'accompagnement des numéros 5 à 8	à	190,00 F

ANCIENS NUMEROS

REVUES	<input type="checkbox"/> 4	<input type="checkbox"/> 7	<input type="checkbox"/> 8	à	35,00 F												
REVUES	<input type="checkbox"/> 9	<input type="checkbox"/> 10	<input type="checkbox"/> 11	<input type="checkbox"/> 12	<input type="checkbox"/> 13	<input type="checkbox"/> 14	<input type="checkbox"/> 15	à	40,00 F								
DISQUETTES	<input type="checkbox"/> 1	<input type="checkbox"/> 2	<input type="checkbox"/> 3	<input type="checkbox"/> 4	<input type="checkbox"/> 5	<input type="checkbox"/> 6	<input type="checkbox"/> 7	<input type="checkbox"/> 8	<input type="checkbox"/> 9	<input type="checkbox"/> 10	<input type="checkbox"/> 11	<input type="checkbox"/> 12	<input type="checkbox"/> 13	<input type="checkbox"/> 14	<input type="checkbox"/> 15	à	55,00 F

ABONNEMENTS

POUR 6 NUMEROS à partir du n°

ABONNEMENT SANS DISQUETTES	à	200,00 F
ABONNEMENT AVEC DISQUETTES	à	480,00 F

TOTAL TTC

Supplément expédition par avion

MONTANT DU REGLEMENT

Ces tarifs comprennent l'envoi postal en France métropolitaine, CEE et Suisse (voie aérienne exceptée)
Supplément avion : 10 F par numéro et/ou disquette

Envoyez ce bon et votre règlement à :

Editions MEV - 64-70, rue des Chantiers - 78000 VERSAILLES

Nom :

Adresse :

MACSI INFORMATIQUE

125, rue Amelot 75011 PARIS
M^o Filles du Calvaire et Oberkampf
Tél. 355.07.01

Ouvert tous les jours sauf dimanche
de 9 h 30 à 13 h et de 14 h à 19 h 30

pour



PROMOTION 1^{er} ANNIVERSAIRE

Z 80
+
1 bte de
disquettes
500 F

	Prix TTC
Lecteur de disquette 5" 1/4 half size	1950
Carte contrôleur	400
Disquettes grande marque (les 10)	220
CARTE LANGAGE + 1 bte disq. .	500
CARTE 80 COLONNES (pour II +)	750
(avec kit inverse et minuscules accentuées)	
CARTE 128 K	1600
Interface parallèle pour EPSON av cable.....	400
Microbuffer 32 K	1400
Carte Série.....	600
Carte Communication	650
VENTILATEUR EXTERNE	300
JOYSTICK LUXE (précisez II + ou IIe).....	200
Accelerator, Applicard, Carte 8088, etc... nous consulter	
MONITEUR PHILIPS 12" Vert	1000
ASCII Express Professionnal	1200
& beaucoup d'autres programmes	
PROMOTION SUR NOS IMPRIMANTES	
IMPRIMANTE Graphique avec interface parallèle et cable	3500
Prix modifiables sans préavis, stock limité.	
* APPLE II est une marque déposée de APPLE COMPUTER INC.	

**SUPER
SERIE
900 F**

**MODEM
BUZZ BOX
1000 F**

**LECTEUR
COMPATIBLE
2 C - 2.200 F**

**CARTE
PARALLELE
400 F**

**MAINTENANCE
ASSURÉE**

BON DE COMMANDE à retourner à MACSI 125, rue AMELOT 75011 PARIS

NOM, Prénom.....
rue

Code postal Ville

Tél. Matériel possédé

Signature

QU.	DESIGNATION	PRIX

POM'S

REGLEMENT JOINT	+ particip. sur envoi	+ 35.00
Chèque <input type="checkbox"/>	TOTAL	
C.C.P. <input type="checkbox"/>	Port gratuit pour	
Mandat lettre <input type="checkbox"/>	Achat: 3000 F.	

**Le cadeau d'Atari[®]
à tous les possesseurs de VIC 20,
Commodore 64, Apple II et TI 99/4A.**



Avec Atarisoft[®], découvrez 13 des plus grandes stars d'Atari[®].

Même si vous n'avez pas d'ordinateur Atari, Atarisoft vous permet désormais de découvrir les programmes Atari les plus célèbres : Pac-Man, Centipède, Jungle Hunt, Pôle Position, Galaxian, Miss Pac-Man, Joust, Moon Patrol, Dig-Dug, Donkey Kong, Robotron 2084, Defender, Stargate. **ATARISOFT[™]**