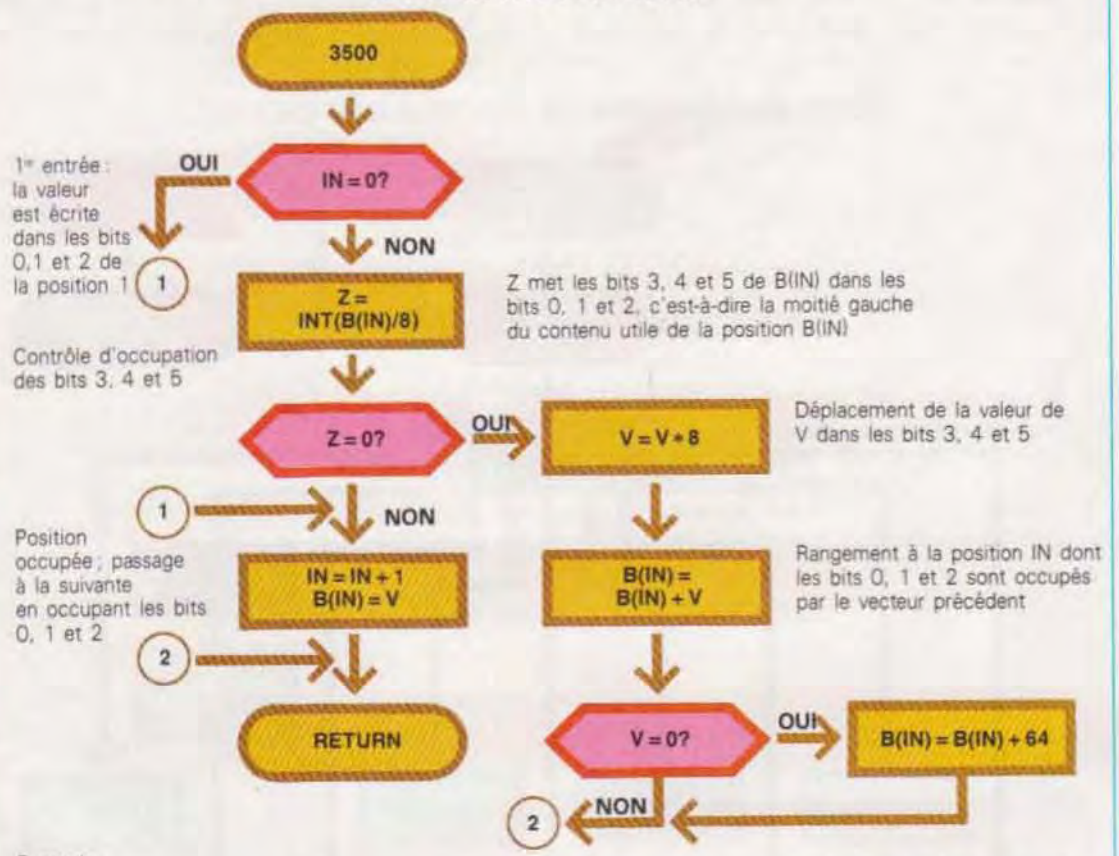
ment » des valeurs prises successivement par la variable V (voir ci-dessous l'organigramme correspondant).
L'indice IN contient la dernière position occupée dans le tableau B(100). Sa valeur initiale est de zéro. De ce fait, au premier déplacement, le sous-programme saute les instructions intermédiaires et mémorise + 1 dans IN, c'est-à-dire dans la partie droite de B(1).
Si l'on pose B(1) = V, la valeur V occupe la position lui correspondant, soit les bits 0 à 2. Lors du deuxième appel, IN ne vaut plus zéro mais il est activé par la partie centrale.
Ces instructions analysent le contenu de B(IN) divisé par 8. Si la donnée contenue dans B(IN) occupe seulement les bits 0, 1 et 2, la valeur entière de la division est zéro ; dans les autres cas, elle est différente de zéro.
Si le résultat de la division est 0, c'est que la partie droite (bits 0 à 2) est déjà occupée (il est impossible que la mémoire ne contienne que des zéros, car IN est incrémenté avant

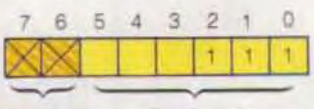
SOUS-PROGRAMME DE MEMORISATION SOUS FORME DE TABLE DE FIGURES



Exemple :

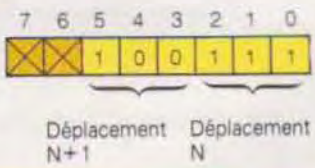
V=7 (déplacement visible à droite)

La mémoire est vide. La représentation binaire de 7 est 111 (écrite dans les bits 0, 1 et 2)



V=4 (déplacement visible vers le haut)

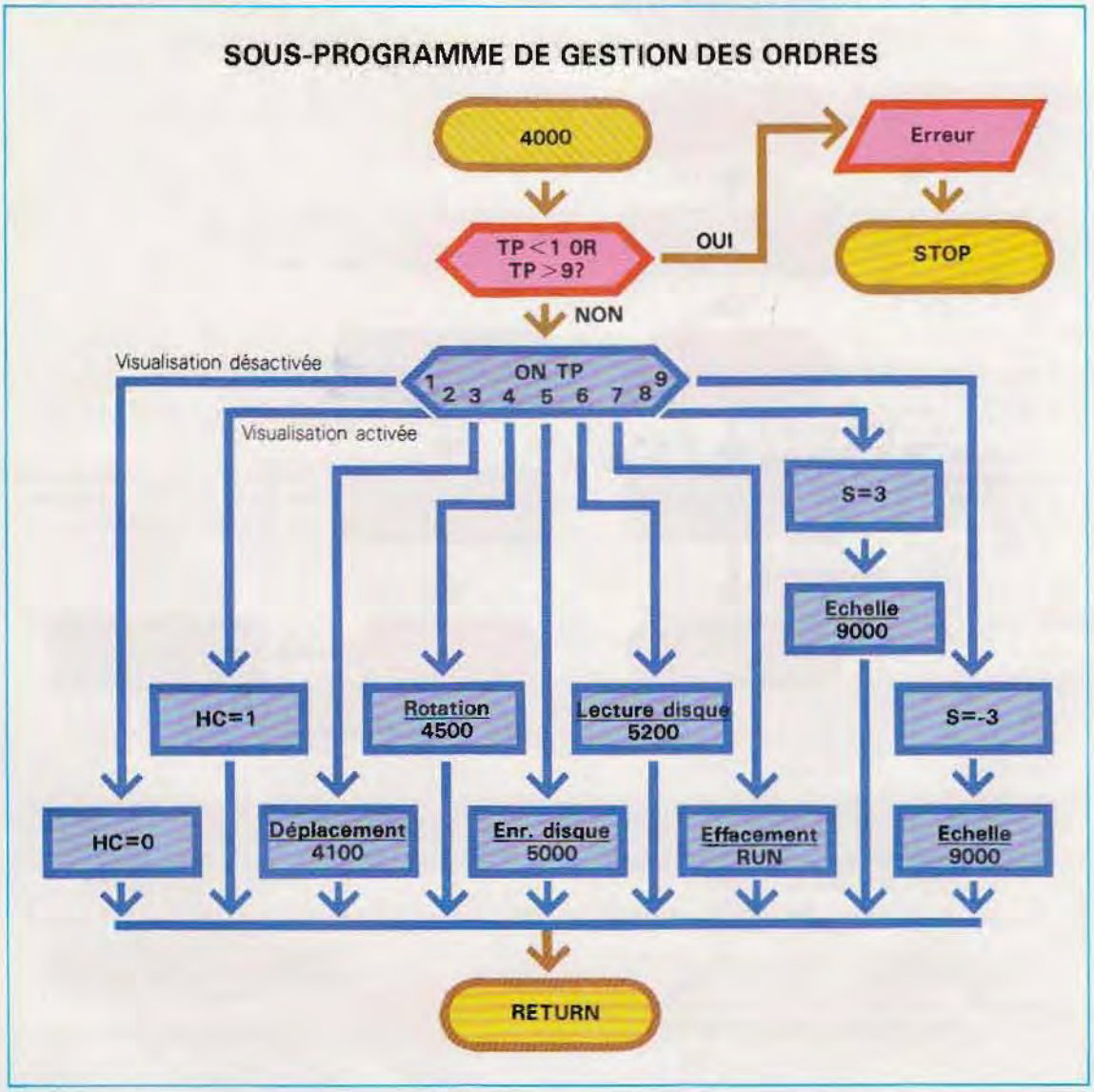
La mémoire utile est occupée dans sa partie droite (INT(B(.)/8=INT(7/8)=0) ; V doit alors être déplacé de 3 bits : V=V*8=32=1 0 0 0 0 0 ($32=2^5$). Le contenu de la mémoire correspond au tracé.

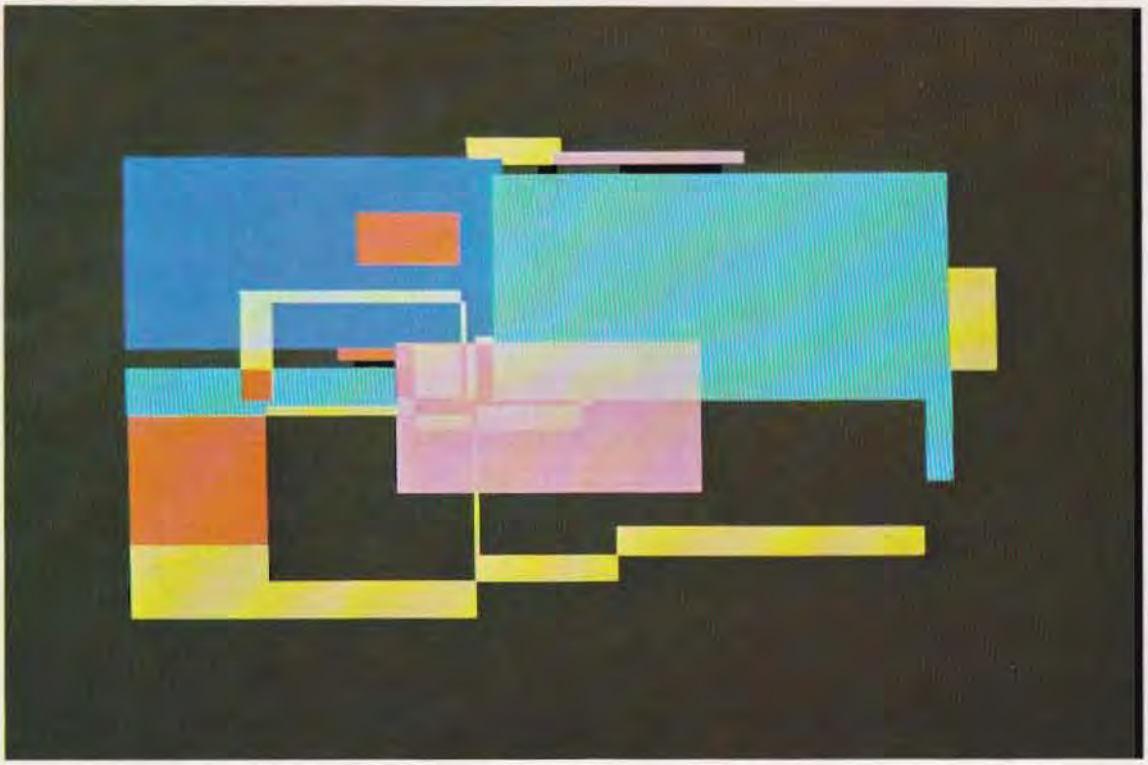


l'écriture). Il faut alors mémoriser V dans les bits 3, 4 et 5, ce qui s'obtient en multipliant sa valeur par 8 et en ajoutant le résultat au contenu précédent. Remarquons toutefois que, de ce fait, le déplacement entré à une adresse en premier se trouve à droite, alors que le suivant est à gauche.

On trouvera ci-dessous l'organigramme du sous-programme 4000 (gestion des ordres) qui se compose uniquement d'appels à d'autres sous-programmes dont chacun est consacré à un des neuf ordres possibles. De nouvelles fonctions seront éventuellement implémentées en introduisant d'autres instructions dans l'appel adressé (ON TP...) et, bien entendu, les sous-programmes nécessaires à leur exécution. Les fonctions de déplacement visible ou invisible nécessitent seulement l'attribution de la valeur 0 ou 1 à l'indicateur HC ; elles sont exécutées dans le sous-programme principal. Cette méthode de subdivision poussée à l'extrême est intéressante. En effet, le sous-programme 4000 aurait pu contenir l'ensemble des autres instructions et éliminé la nécessité de désigner les indicateurs à transmettre aux sous-programmes suivants. Ce sous-programme aurait été cependant beaucoup plus complexe et difficile à modifier. De plus, l'utilisation de sous-programmes consacrés à une tâche bien délimitée permet d'économiser des instructions en réutilisant le même sous programme en plusieurs points du programme.

SOUS-PROGRAMME DE GESTION DES ORDRES





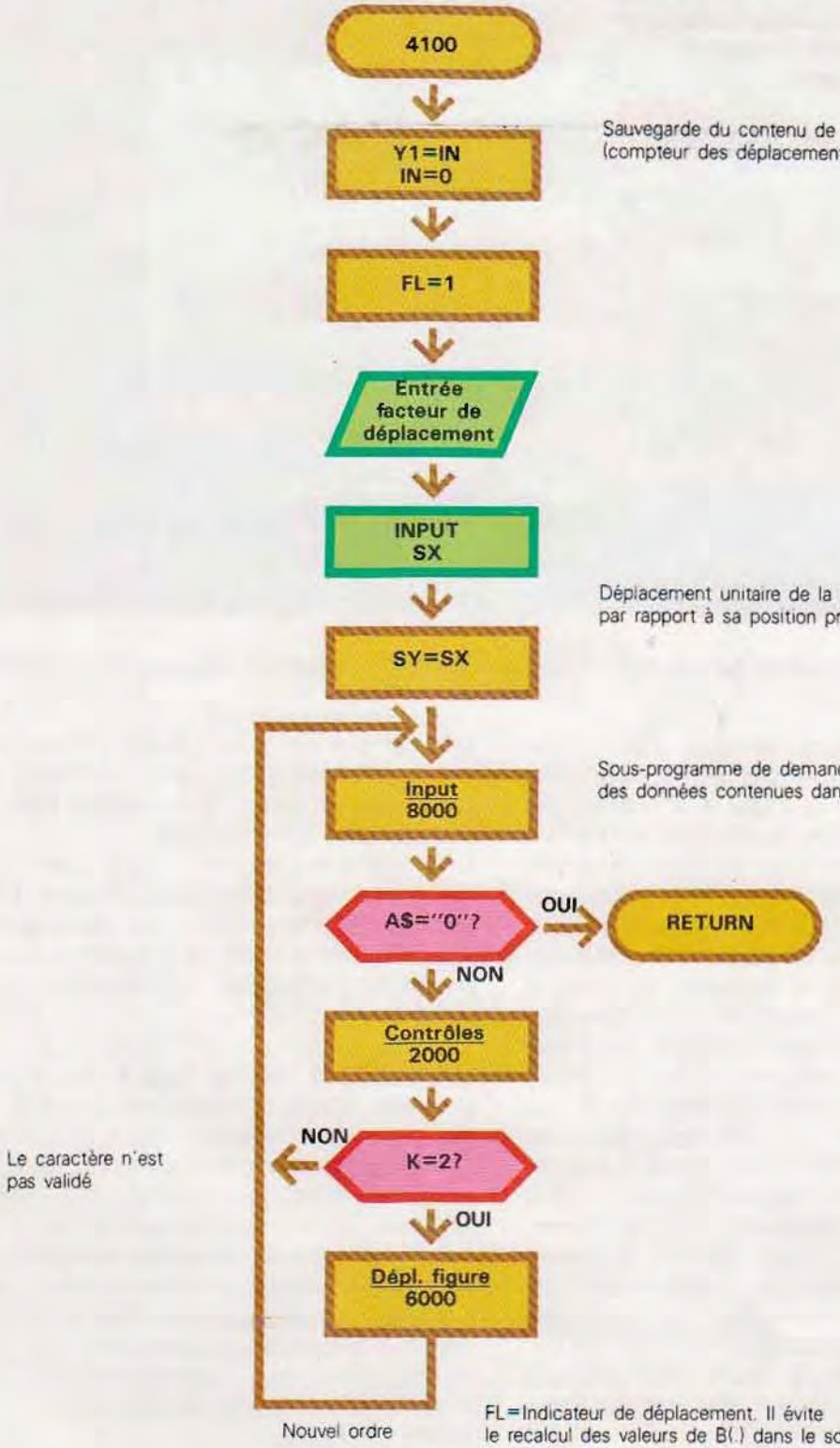
Centre THC

Composition graphique obtenue à l'aide d'instructions simples de haut niveau.

Gestion des départements d'ensemble (sous-programme 4100). Ce sous-programme a pour fonction de déplacer la totalité d'une figure dans une zone quelconque de l'écran. (Ne pas confondre avec le sous-programme 2000, qui se charge de la gestion des déplacements lors de la génération du dessin.) Il prévoit la sauvegarde de la table des déplacements B(100) avant la translation du dessin. Il suffit d'utiliser un tableau de correspondance pour que l'original reste inchangé. Il est toujours possible de retrouver son état précédent si les résultats ne sont pas satisfaisants. La première fonction à exécuter est la lecture du type de déplacement désiré. Il s'agit d'une partie du programme semblable à celle utilisée pour la génération de la table, la seule variante étant l'identification de la valeur 0, qui indique la fin de la translation. Les touches activant les déplacements du dessin sont les mêmes que celles qui servent à produire les tables des déplacements formant la figure : c'est pourquoi ils sont également contrôlés par le sous-programme 2000.
Dans ce cas, toutefois, il n'y a pas d'ordre : si la valeur de K est différente de 2 à la sortie du sous-programme 2000, elle est ignorée (K = 2 signifie qu'une des touches prévues pour les déplacements a été enfoncée).
Si la touche est identifiée (et s'il s'agit d'une des quatre prévues), le sous-programme 6000, qui déplace la figure, est activé. Au début du sous-programme 6000, le programme se met en attente d'un nouveau déplacement ou de la valeur 0.

Déplacement de la figure (sous-programme 6000). Ce sous-programme lit les déplacements constituant la figure dans B(100) et les dessins à partir d'un point dont les coordonnées sont modifiées par rapport à l'origine par défaut X0, Y0, d'une quantité SX ou SY, en fonction du type de translation demandée. La fonction est obtenue en extrayant, de la table des déplacements, l'attribut (visible ou non visible) et le type de déplacement (0, 1, 2, 3) et en utilisant les sous-programmes prévus pour la génération de la table. Il s'agit, en fait, d'une entrée simultanée.
Pour ne pas surcharger l'organigramme et le

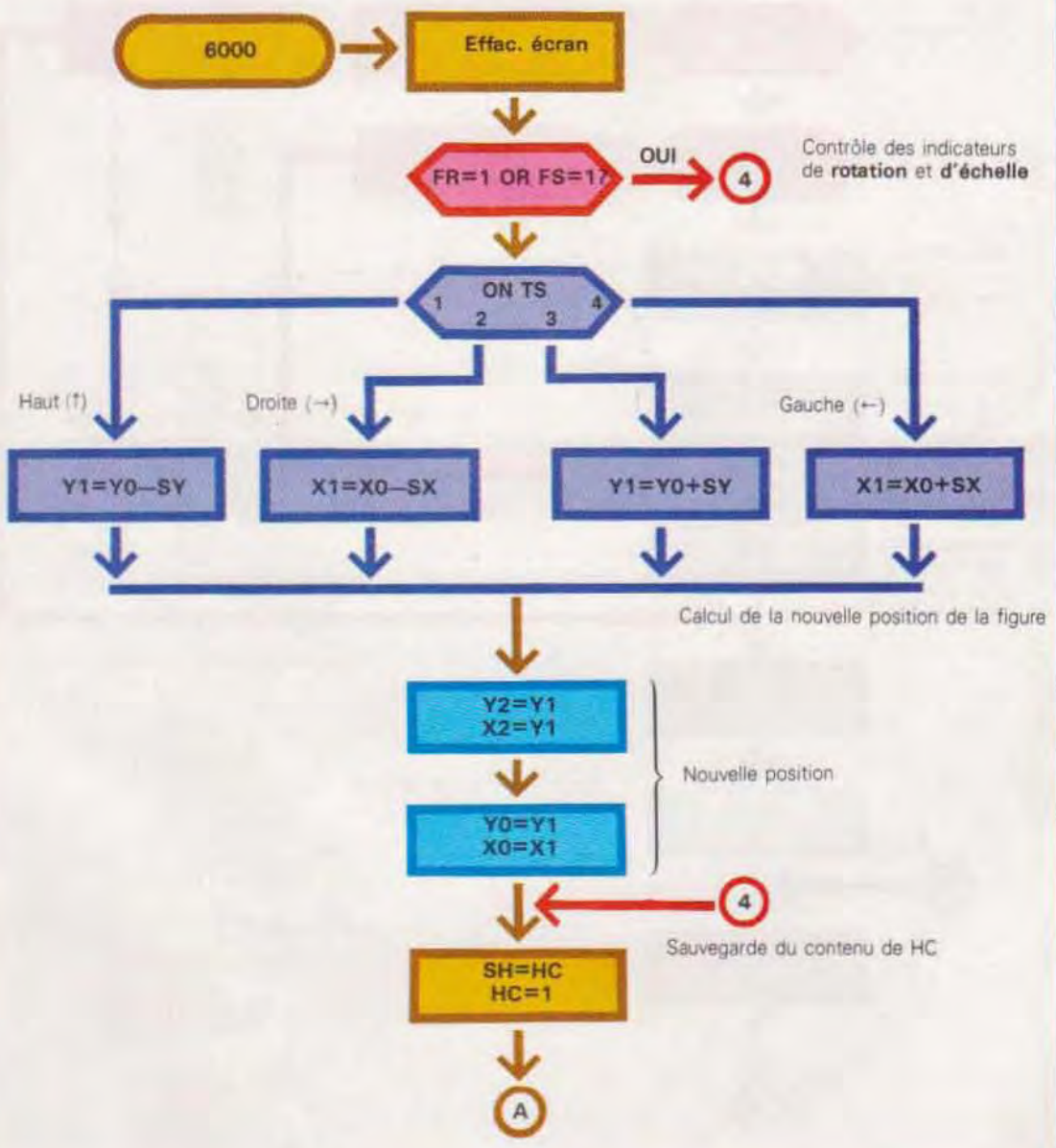
GESTION DES DEPLACEMENTS D'ENSEMBLES

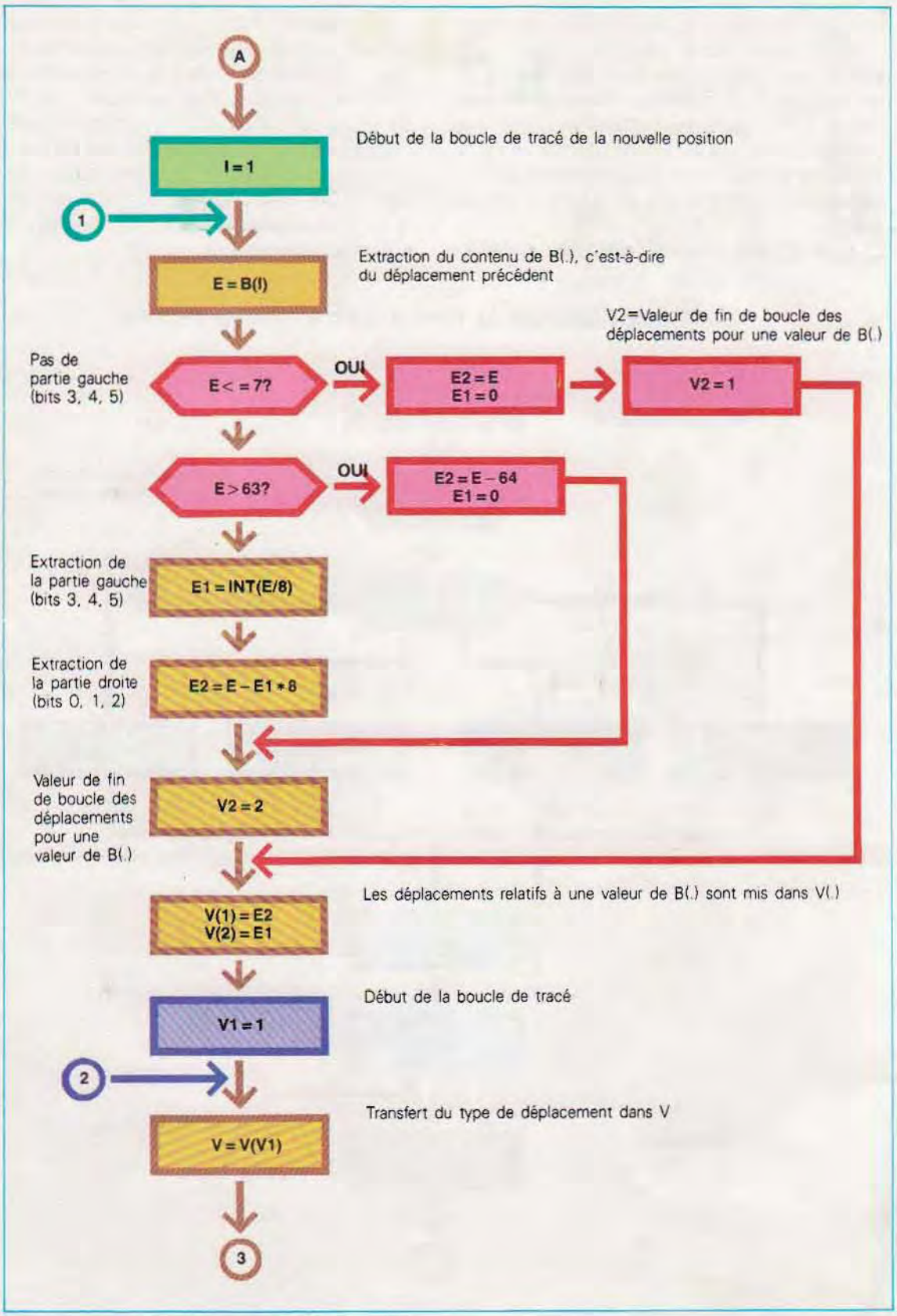


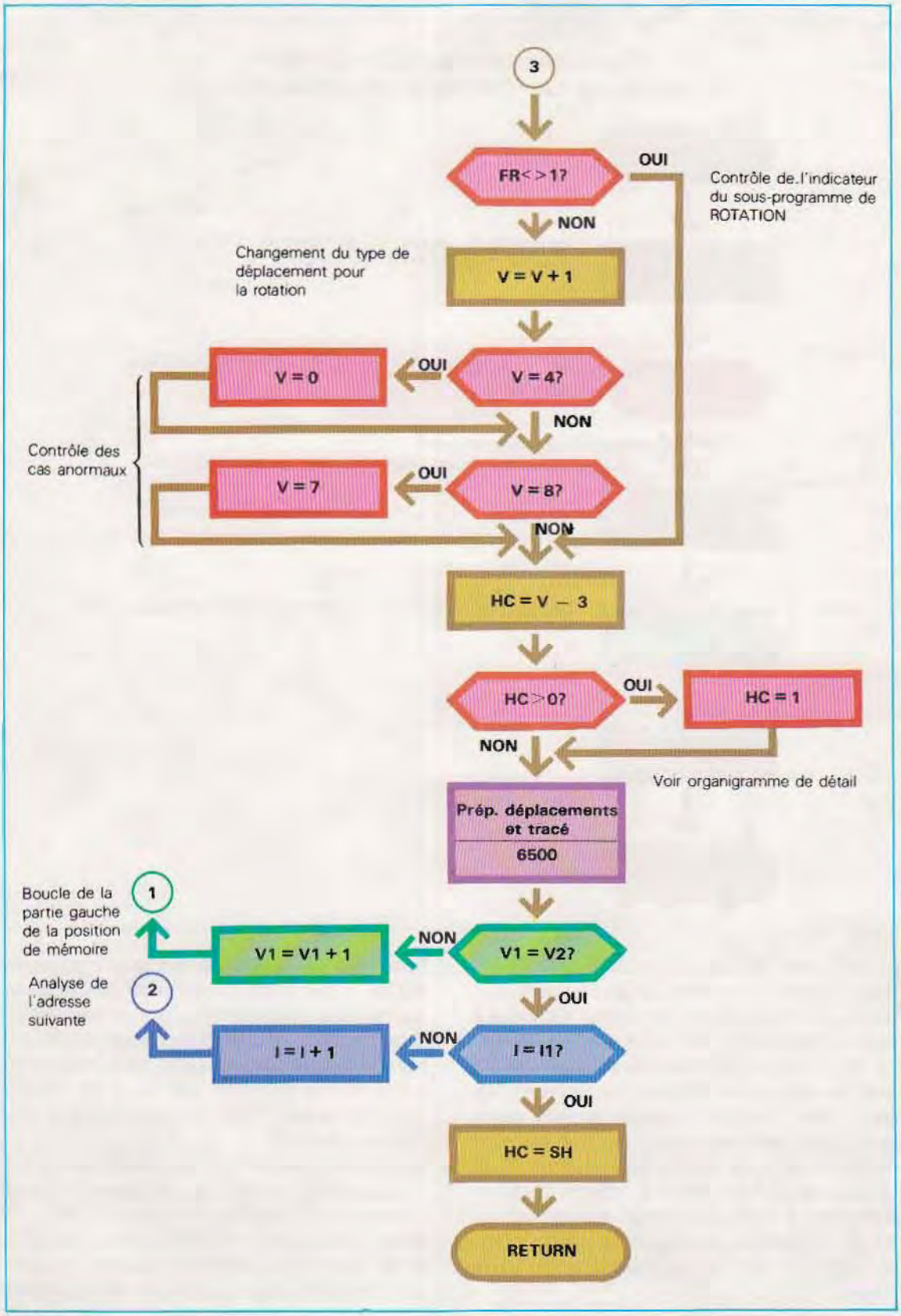
listing, l'effacement du déplacement précédent a été omis. Quand la translation d'une figure est activée, cette même figure est répétée en un autre point. Si la position finale est obtenue (après 10 passages intermédiaires par exemple), il y aura autant de répétitions de la figure. Chacune d'elles sera déplacée de 1/10 du déplacement final par rapport à la précédente.

Deux méthodes peuvent permettre de remédier à cet inconvénient. La première consiste à effacer toute la page graphique avant d'effectuer le déplacement (mais la figure initiale est également effacée). Pour remédier à cet état de fait, il suffit d'introduire l'instruction d'effacement de la mémoire vidéo avant l'appel du sous-programme 6000. La reconstruction de la figure à la position originale peut être activée à la fin du sous-programme 6000, quand il n'y a plus d'écran à vider.

SOUS-PROGRAMME DE DEPLACEMENT D'UNE FIGURE

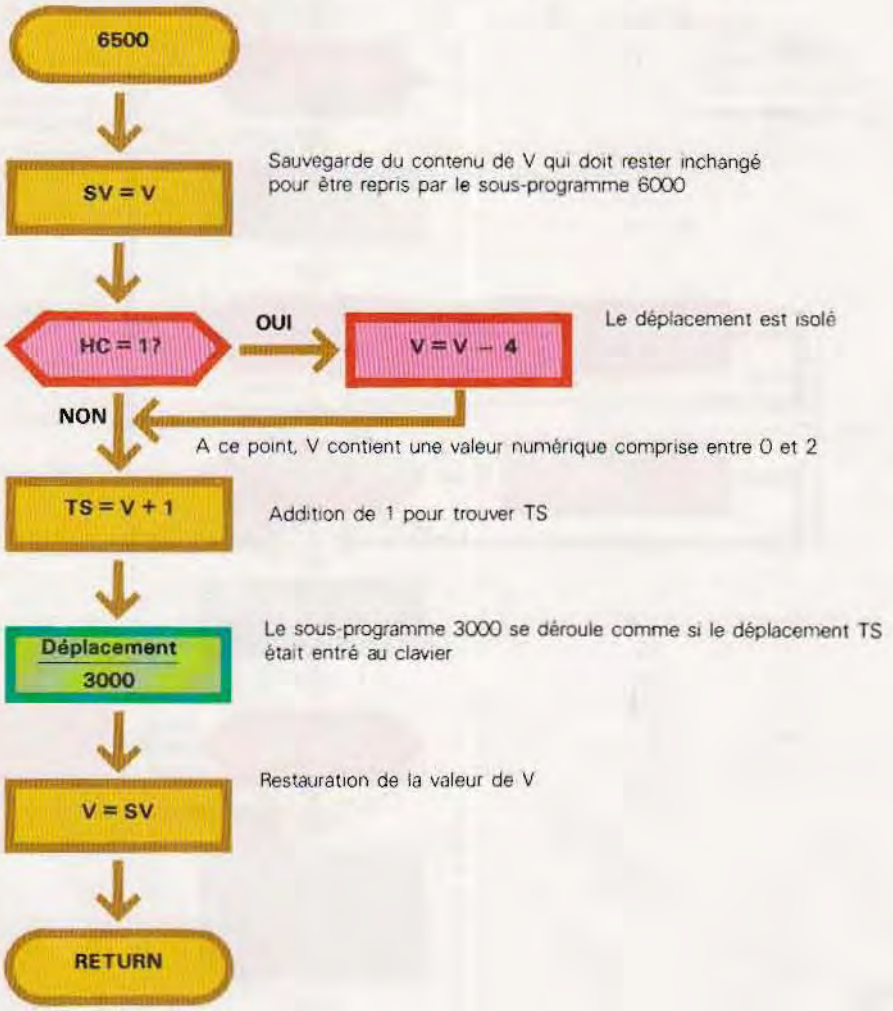



A
I = 1
Début de la boucle de tracé de la nouvelle position
1
E = B(I)
Extraction du contenu de B(.), c'est-à-dire du déplacement précédent
V2 = Valeur de fin de boucle des déplacements pour une valeur de B(.)
Pas de partie gauche (bits 3, 4, 5)
E < = 7?
OUI
E2 = E
E1 = 0
V2 = 1
E > 63?
OUI
E2 = E – 64
E1 = 0
Extraction de la partie gauche (bits 3, 4, 5)
E1 = INT(E/8)
Extraction de la partie droite (bits 0, 1, 2)
E2 = E – E1 * 8
Valeur de fin de boucle des déplacements pour une valeur de B(.)
V2 = 2
Les déplacements relatifs à une valeur de B(.) sont mis dans V(.)
V(1) = E2
V(2) = E1
Début de la boucle de tracé
V1 = 1
2
Transfert du type de déplacement dans V
V = V(V1)
3


3
FR<>1?
OUI
Contrôle de l'indicateur du sous-programme de ROTATION
NON
Changement du type de déplacement pour la rotation
V = V + 1
V = 4?
OUI
V = 0
NON
Contrôle des cas anormaux
V = 8?
OUI
V = 7
NON
HC = V – 3
HC > 0?
OUI
HC = 1
NON
Voir organigramme de détail
Prép. déplacements et tracé
6500
Boucle de la partie gauche de la position de mémoire
1
V1 = V2?
NON
V1 = V1 + 1
Analyse de l'adresse suivante
2
OUI
I = I1?
NON
I = I + 1
OUI
HC = SH
RETURN

1609

PREPARATION DU DEPLACEMENT ET APPEL DE LA FONCTION DE VISUALISATION



La seconde méthode, beaucoup plus compliquée, consiste à repasser sur les contours de la figure précédente avec une couleur identique à celle du fond, ce qui provoque son effacement. En fait, il existe une troisième solution, valable pour certains systèmes comme Siprel et Apple. Quand l'ordinateur dispose de deux pages graphiques, on peut utiliser la première pour conserver la figure dans sa position d'origine et la seconde pour les positions intermédiaires en procédant à chaque fois à un effacement. En fin de translation, les figures (position initiale et déplacée) sont regroupées dans une seule page.

Rotation de la figure (sous-programme 4500). Cette fonction se révèle peu adaptée aux déplacements dans les quatre directions : le programme ne permet pas, en effet, de tracer des lignes inclinées. La logique de rotation à 90° dans le sens des aiguilles d'une montre n'est fournie qu'à titre d'exemple dans le graphique ci-contre.

La rotation s'obtient simplement en ajoutant 1 à chacun des vecteurs de déplacement et en revenant à 0 quand la somme atteint 4.

Dans l'exemple donné, la rotation est de 90° à droite. Selon une méthode analogue, la position initiale de la figure basculée est la position

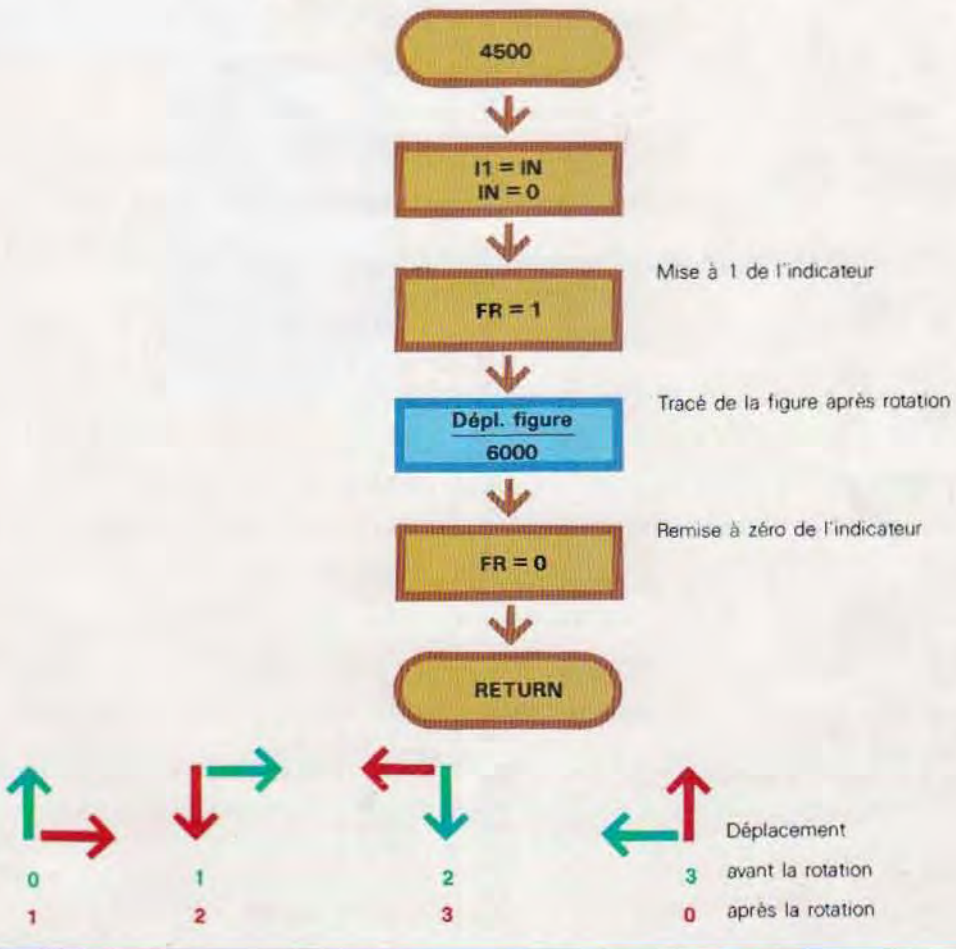
par défaut (X0, Y0) plus une constante (10). Pour extrapoler à partir de ce programme, il faut y ajouter la possibilité, pour l'utilisateur, d'introduire sa sélection.

Enregistrement sur le disque (sous-programme 5000). Le sous-programme comprend uniquement l'entrée du nom du fichier qui devra contenir B (100) et son enregistrement sur le disque. C'est pourquoi nous n'en donnerons que le listing (pages 1612 et 1614).

Lecture du disque (sous-programme 5200).
Ce sous-programme exécute les fonctions inverses du précédent : une fois entré le nom du fichier contenant les données du dessin à visualiser, il transfère son contenu dans B (100) et appelle le sous-programme de présentation 6000. (Voir listing page 1612 à 1614).

Une des phases du fonctionnement du programme (voir listing p. 1612 à 1614). C'est la page graphique 1, réservant 4 lignes pour le texte, qui a été sélectionnée. Ici, la fenêtre de texte sert à l'affichage du menu des commandes (voir ligne 118).

SOUS-PROGRAMME DE ROTATION



CREATION ET GESTION DES TABLES DE FIGURES

```
1 REM ================
2 REM *  CREATION ET  *
3 REM *    GESTION    *
4 REM * FIGURES HAUTE *
5 REM *   RESOLUTION  *
6 REM ================
7 :
8 :
100 GOSUB 1000
110 :
111 REM ==============
112 REM *  LECTURE   *
113 REM * CARACTERES *
114 REM ==============
115 :
117 HGR:HOME
118 VTAB 21:PRINT "I=HAUT/M=BAS
    /K=DROITE/J=GAUCHE/F=INV/V
    =VIS":PRINT "S=DEPL/R=ROT/D
    =DISQUE/L=CHARG/C=EFFAC"
120 GOSUB 8000
140 GOSUB 2000
150 ON K GOTO 160,170,190
160 GOTO 120
170 GOSUB 3000
180 GOTO 120
190 :
192 GOSUB 4000
200 GOTO 120
300 STOP
1000 :
1001 REM ================
1002 REM * INITIALISATION *
1003 REM ================
1004 :
1010 DIM B(100)
1020 DX=10:DY=10:Y0=50:X0=100
     :XM=270:YM=159:Y1=Y0:X1=X0

1025 X2=X1:Y2=Y1:S=1
1030 S(1)=73:S(2)=75:S(3)=77
     :S(4)=74
1035 D(1)=70:D(2)=86:D(3)=83
     :D(4)=82:D(5)=68:D(6)=76

1040 D(7)=67:D(8)=43:D(9)=4
     5
1045 D$=CHR$(4)
1100 RETURN
2000 :
2001 REM ==============
2002 REM * CONTROLE  *
2003 REM * CARACTERE *
2004 REM ==============
2005 :
2010 K=1:TS=0:TP=0:I=1
2020 IF K<>1 THEN RETURN
2030 IF S(I)=0 THEN 2050
2040 IF ASC(A$)=S(I) THEN K=2
     :TS=I:GOTO 2200
2050 IF D(I)=0 THEN 2070
2060 IF ASC(A$)=D(I) THEN K=3
     :TP=I:GOTO 2200
2070 IF I=10 THEN RETURN
2080 I=I+1:GOTO 2020
2100 :
2200 :
2210 GOTO 2070
3000 :
3001 REM ================
3002 REM *    GESTION   *
3003 REM * DEPLACEMENTS *
3004 REM ================
3005 :
3010 ON TS GOTO 3020,3030,3040
     ,3050
3011 :
3012 REM  < HAUT >
3013 :
3020 Y2=Y1-DY:V=0:GOTO 3100

3021 :
3022 REM  < DROITE >
3023 :
3030 X2=X1+DX:V=1:GOTO 3100

3031 :
3032 REM  < BAS >
3033 :
3040 Y2=Y1+DY:V=2:GOTO 3100

3041 :
3042 REM  < GAUCHE >
3043 :
3050 X2=X1-DX:V=3
3060 :
3100 :
3105 IF HC=1 THEN V=V+4
3110 GOSUB 3400
3115 IF FL=1 THEN RETURN
3150 GOSUB 3500
3160 RETURN
3400 :
3401 REM ================
3402 REM *    TRACE     *
3403 REM * DEPLACEMENT *
3404 REM ================
3405 REM :
3420 IF HC=1 THEN HCOLOR=3
     :GOTO 3440
3430 HCOLOR=0
3440 IF Y2>YM OR X2>XM THEN
     RETURN
3450 IF Y2<=0 OR X2<=0 THEN
     RETURN
3460 HPLOT X1,Y1 TO X2,Y2
3470 X1=X2:Y1=Y2
3480 RETURN
3500 :
3501 REM ================
3502 REM * MEMORISATION *
3503 REM *    TABLES    *
3504 REM ================
3505 :
3510 IF IN=0 THEN 3540
3520 Z=INT(B(IN)/8)
3530 IF Z<>0 THEN 3540
3535 V=V*8:B(IN)=B(IN)+V:IF
     V=0 THEN B(IN)=B(IN)+64
     :GOTO 3550
3537 GOTO 3550
3540 IN=IN+1:B(IN)=V
3550 RETURN
4000 :
4001 REM ============
4002 REM * GESTION *
4003 REM * ORDRES  *
4004 REM ============
4005 :
4010 IF TP<1 OR TP>9 THEN STOP
```

```
4011 :
4020 ON TP GOTO 4030,4040,4050
     ,4060,4070,4080,4085,4090
     ,4095
4030 HC=0:RETURN
4040 HC=1:RETURN
4050 GOSUB 4100:RETURN
4060 GOSUB 4500:RETURN
4070 GOSUB 5000:RETURN
4080 GOSUB 5200:RETURN
4085 RUN
4090 S=3:GOSUB 9000:RETURN
4095 S=-3:GOSUB 9000:RETURN

4100 :
4101 REM ================
4102 REM *   GESTION   *
4103 REM * DEPLACEMENTS *
4104 REM ================
4105 :
4106 FL=1
4140 I1=IN
4145 VTAB 23:HTAB 12:INPUT
     "Facteur de déplacement? " ;SX

4147 SY=SX
4148 VTAB=23:HTAB=22:PRINT "
4150 GOSUB 8000
4160 IF A$="0" THEN 4300
4170 GOSUB 2000
4180 IF K<>2 THEN 4150
4190 GOSUB 6000
4200 GOTO 4150
4300 :
4310 :
4320 :
4325 IN=I1
4327 FL=0
4330 RETURN
4500 :
4501 REM =============
4502 REM * ROTATION *
4503 REM =============
4504 :
4510 I1=IN:IN=0
4520 FR=1:GOSUB 6000
4530 FR=0
4540 RETURN
5000 REM ==================
5001 REM * ENREG. DISQUE *
5002 REM ==================
5003 :
5010 VTAB 23:HTAB 12:PRINT "Nom
     :"
5020 VTAB 23:HTAB 19:INPUT " "
     ;NM$
5030 VTAB 1:PRINT D$"Open "NM$
5040 PRINT D$"Write "NM$
5045 PRINT IN
5050 FOR I=1 TO IN: PRINT B(I)
     :NEXT I
5060 PRINT D$"Close "NM$:PRINT
     D$
5070 RETURN
5200 REM ===============
5201 REM * CHARGEMENT *
5202 REM ===============
5203 :
5210 VTAB 23:HTAB 12:PRINT "Nom
     "
5220 VTAB 23:HTAB 19:INPUT " "
     ;NM$
5230 VTAB 1:PRINT D$"Open "NM$
5240 PRINT D$"Read "NM$
5250 INPUT I1
5255 IN=I1
5260 FOR I=1 TO I1:INPUT B(I)
     :NEXT I
5270 PRINT D$"Close "NM$:PRINT
     D$
5273 X0=100:Y0=50:X1=X0:Y1=Y0:X2
     =X1:Y2=Y1:FL=1
5275 GOSUB 6105:FL=0
5280 RETURN
6000 :
6001 REM ================
6002 REM * DEPLACEMENT *
6003 REM *   FIGURE    *
6004 REM ================
6005 :
6007 HGR
6008 IF FR=1 OR FS=1 THEN 6105

6010 ON TS GOTO 6020,6030,6040
     ,6050
6020 Y1=Y0-SY:GOTO 6100
6030 X1=X0+SX:GOTO 6100
6040 Y1=Y0+SY:GOTO 6100
6050 X1=X0-SX
6100 X2=X1:Y2=Y1:Y0=Y1:X0=X1

6105 SH=HC:HC=1
6110 FOR I=1 TO I1
6120 E=B(I)
6130 IF E<=7 THEN E1=0=E2=E:V2=1
     :GOTO 6155
6135 IF E>63 THEN E2=E-64:E1=0
     :GOTO 6153
6140 E1=INT(E/8)
6150 E2=E-E1*8
6153 V2=2
6155 V(1)=E2:V(2)=E1
6160 V1=1
6165 V=V(V1)
6167 IF FR<>1 THEN 6175
6169 V=V+1
6170 IF V=4 THEN V=0
6172 IF V=8 THEN V=4
6175 HC=V-3:IF HC>0 THEN HC=1

6180 GOSUB 6500
6185 VTAB 23:HTAB 35:PRINT TS
6190 :
6200 IF V1<>V2 THEN V1=V1+1
     :GOTO 6165
6210 NEXT I
6220 IF I<I1 THEN I=I1:NEXT I

6230 HC=SH:RETURN
6500 :
6501 REM ================
6502 REM * PREPARATION *
6503 REM * DEPLACEMENT *
6504 REM ================
6505 :
6510 SV=V:IF HC=1 THEN V=V-4
6520 TS=V+1:GOSUB 8000
6530 V=SV:RETURN
8000 :
8001 REM ==========
8002 REM * INPUT *
```

```
8003 REM ==========
8004 :
8010 VTAB 23:HTAB 12:PRINT
     "Entrez (?) commande "
8020 VTAB 23:HTAB 22:GET A$
8021 VTAB 23:HTAB 35:INVERSE
     :PRINT A$:NORMAL
8022 VTAB 23:HTAB 1:PRINT "
                "
8024 VTAB 23:PRINT "X="X1"Y="Y1

8025 IF ASC(A$)=13 THEN TEXT
     :HOME:END
8030 RETURN
9000 REM ===========
9001 REM * ECHELLE *
9002 REM ===========
9003 :
9010 FS=1
9015 DY=DY+S
9020 DX=DX+S:I1=IN:IN=0
9030 GOSUB 6000

9040 FS=0:RETURN
```

Programmes utilitaires pour la création de formes graphiques

L'exemple donné pour l'établissement d'une table archivant les déplacements illustre une méthode de création d'images graphiques avec des systèmes ne disposant pas d'instructions spéciales.

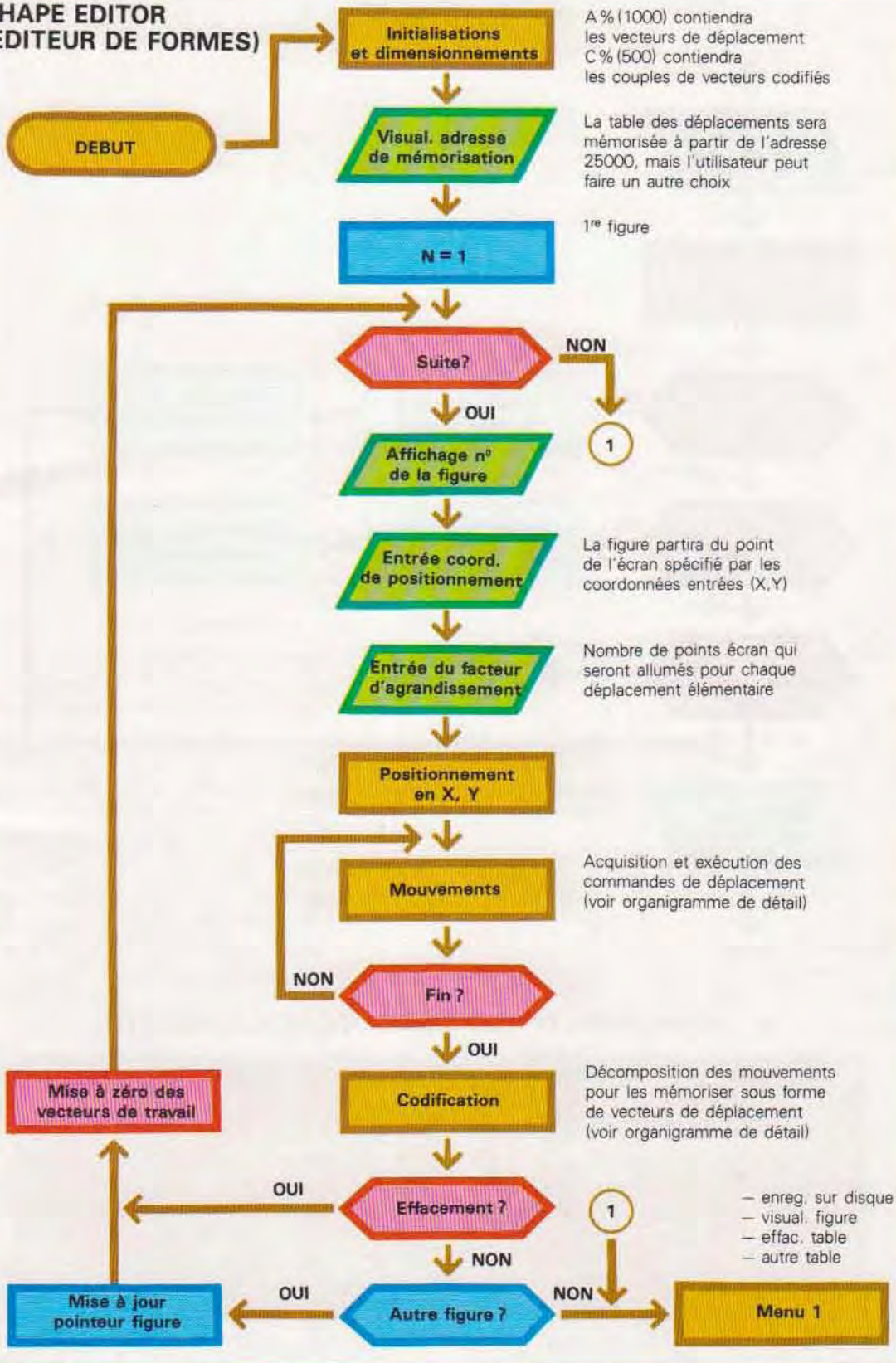
Les micro-ordinateurs — qui prévoient des tables de formes — recourent à deux types d'instructions.
— Le premier (ordinateurs basés sur l'unité centrale 6502 de la famille Apple et compatibles) reprend les déplacements élémentaires, les symboles et les conventions décrits dans notre exemple.
— Le second dispose d'une chaîne de commandes dans laquelle les déplacements sont indiqués les uns à la suite des autres.

Pour générer une table de formes avec le premier type, la meilleure solution consiste à écrire directement en mémoire les vecteurs des déplacements, à l'aide de l'instruction POKE. Il faut donc tout d'abord dessiner la table sur une feuille de papier, en détaillant les déplacements élémentaires, puis la charger en mémoire comme nous l'avons déjà vu.
Mais on peut également simuler les instructions de haut niveau de gestion et de visualisation de formes complexes, même sur des machines dans lesquelles elles n'ont pas été prévues.
Nous présentons ici deux exemples très pratiques.
— Le premier programme, que nous avons appelé « Shape Editor » (éditeur de forme) en reprenant une terminologie très courante, permettra de construire directement à l'écran des formes parfois très complexes et de sauvegarder sur disque, sous un nom conventionnel, les tables des déplacements.
— Le second programme, qui est le « Shape Loader » (chargeur de forme) permettra, à l'inverse, de charger et de visualiser à partir du disque une figure précédemment élaborée.

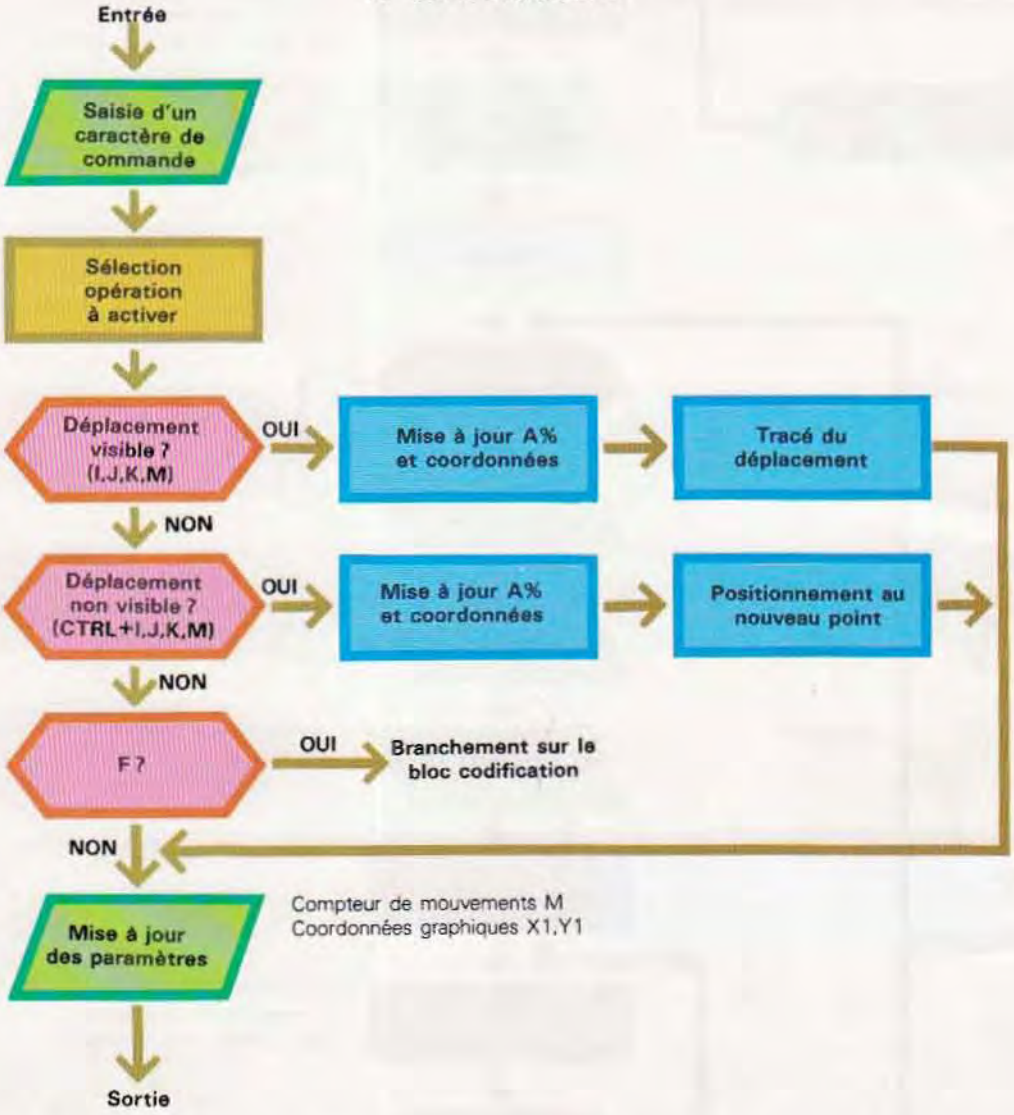
L'éditeur de formes.

L'organigramme du Shape Editor (page ci-contre) prévoit la possibilité de composer un dessin à l'écran à l'aide d'un ensemble de touches, activant le déplacement, visible ou non, d'une pointe, traçante imaginaire. Le bloc 1 détaille l'acquisition et l'exécution des commandes de déplacement (page 1616).
Une fois tracée la forme graphique, on parvient au bloc (3) de codification (page 1617) : les déplacements de la pointe enregistrés dans le tableau A%, sont convertis en un ensemble de vecteurs graphiques et transférés dans le tableau C%. En fin de codification, les vecteurs de C% sont chargés en mémoire. A ce stade, l'utilisateur peut décider de composer une autre figure, qui sera traitée de la même manière par le programme.
Quand le tracé des formes désirées est achevé, on retourne au menu et on commande les enregistrements sur disque sous un nom d'ensemble désignant la table des figures. Dans chaque table, les différentes formes sont numérotées par ordre croissant.
Ce programme est listé dans sa version Siprel, Apple et compatibles, en pages 1618 et 1621.
Les variables employées ont été regroupées dans le tableau de la page 1623.

SHAPE EDITOR (EDITEUR DE FORMES)



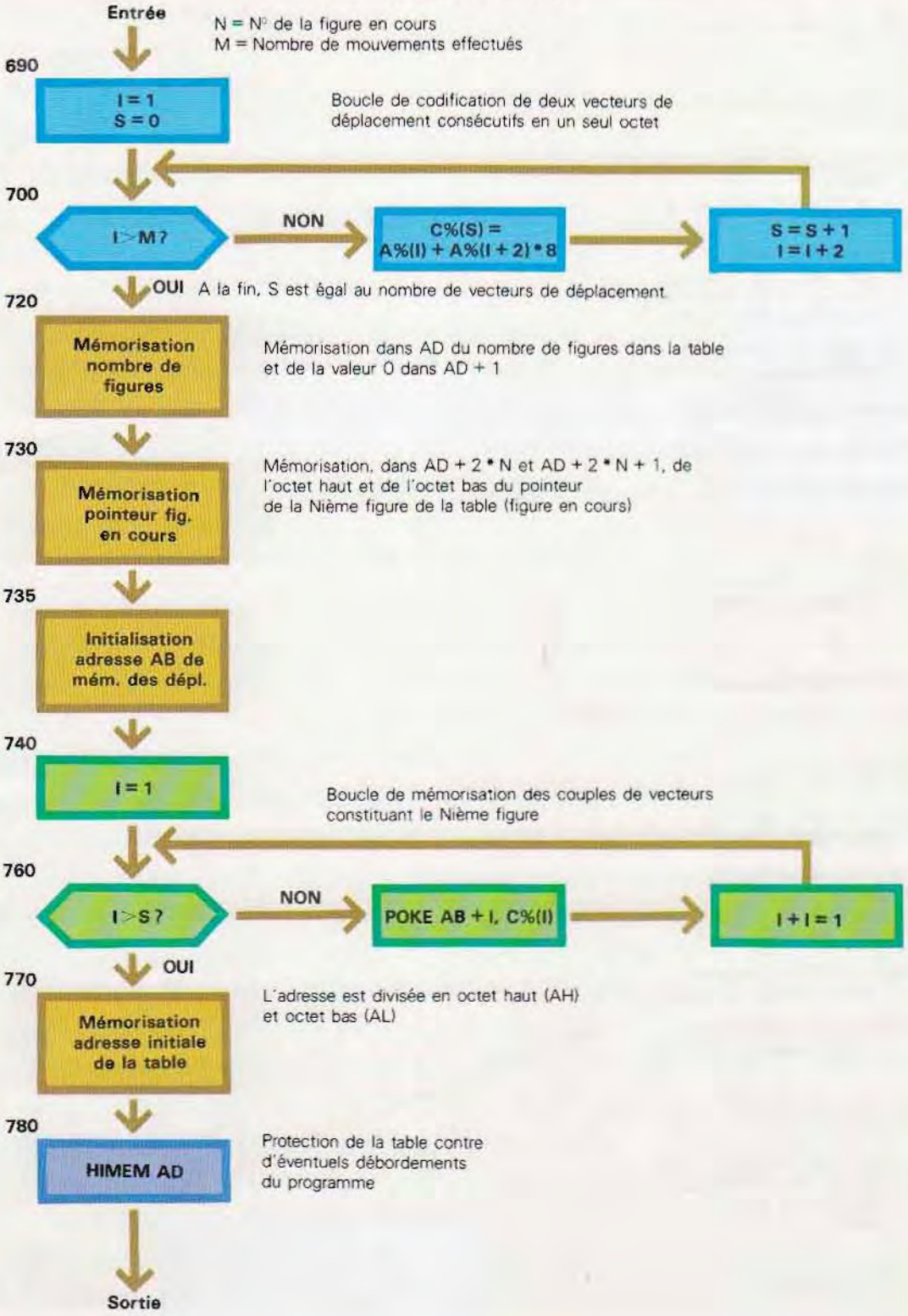
1/ MOUVEMENTS



2/ CARACTERES DE CONTROLE DES DEPLACEMENTS

Caractère	Mode	Direction	Codification dans A%
CTRL + I	Non visible	Vers le haut	0
CTRL + K	Non visible	A droite	1
CTRL + M	Non visible	Vers le bas	2
CTRL + J	Non visible	A gauche	3
I	Visible	Vers le haut	4
K	Visible	A droite	5
M	Visible	Vers le bas	6
J	Visible	A gauche	7

3/ CODIFICATION



L'EDITEUR DE FORMES

Version Siprel, Apple et compatibles

```
70 REM Peut contenir jusqu à
80 REM 30 figures
90 :

100 DIM A%(1000),C%(500)
110 TEXT:HOME

120 INVERSE:PRINT "Editeur de figures"
    :NORMAL

140 PRINT:PRINT

150 AD=25000
160 PRINT "Adresse de la table de
    figures = 25.000":PRINT "Voulez
    -vous changer (O/N) ?";:INPUT Z$

170 IF LEFT$(Z$,1)="O" THEN PRINT
    :INPUT "Nouvelle valeur ?";AD

180 PRINT:PRINT "OK !";AD

190 AH=INT(AD/256):AL=AD-AH*256

200 BL=62:BH=0:BB=62


210 N=1
220 HOME
230 PRINT:PRINT "Numéro de figure ?"
    :INVERSE:PRINT N:NORMAL


240 PRINT:PRINT "<1> Poursuivre"

250 PRINT:PRINT "<2> Stop"

260 PRINT
270 PRINT:INPUT "Choisir";A$

280 IF VAL (A$)=2 THEN N=N-1
    :BB=BB-5:GOTO 1000

290 IF VAL (A$)<>1 THEN 220
300 HGR
310 HOME
320 S=1
330 N=1
340 VTAB 22:INPUT "Coordonnées
    initiales ?";X,Y

350 HOME:VTAB 22:INPUT "Facteur
    d'agrandissement (0-10) ?";IN


360 IF IN<1 OR IN>10 THEN 350
370 HOME:VTAB 21

380 PRINT "<I>, <J>, <K>, <M> pour
    dessiner (avec <CTRL> pour se
    déplacer sans dessiner), <F>
    pour finir.";

390 :
400 REM Point initial
```

```
420 HCOLOR=3
430 HPLOT X,Y
440 X1=X:Y1=Y

450 :

460 REM Mouvements
470 :

480 HTAB 1:VTAB 1:GET A$


490 HTAB 1:VTAB 24:PRINT "Mouvements"
    :INVERSE:PRINT M:NORMAL



500 IF A$="I" THEN A%(M)=4
    :Y=Y-IN:GOTO 600

510 IF A$="M" THEN A%(M)=6
    :Y=Y+IN:GOTO 600

520 IF A$="J" THEN A%(M)=7
    :X=X-IN:GOTO 600

530 IF A$="K" THEN A%(M)=5
    :X=X+IN:GOTO 600

540 IF A$=CHR$(9) THEN A%(M)=0
    :Y=Y-IN:GOTO 610

550 IF A$=CHR$(13) THEN A%(M)=2
    :Y=Y+IN:GOTO 610

560 IF A$=CHR$(10) THEN A%(M)=3
    :X=X-IN:GOTO 610

570 IF A$=CHR$(11) THEN A%(M)=1
    :X=X+IN:GOTO 610

580 IF A$="F" THEN 670
590 GOTO 480
600 HPLOT X1,Y1 TO X,Y:GOTO 630

610 IF IN=1 THEN 630
620 HPLOT X,Y
630 M=M+1
640 X1=X:Y1=Y

650 GOTO 460
660 :

670 REM Codification
680 :

690 FOR I=1 TO M STEP 2
700 C%(S)=A%(I)+A%(I+1)*8
    :S=S+1
710 NEXT
720 POKE AD,M:POKE AD+1,0

730 POKE AD+2*M,BL
    :POKE AD+2*M+1,BH
735 AB=AD+BB-1
740 FOR I=1 TO S
```

```
750 POKE AB+1,C%(I)
760 NEXT
770 POKE 232,AL:POKE 233,AH

780 HIMEM AD
790 :

800 REM Menu 2
810 :

820 HOME:VTAB 21

830 PRINT "Numéro de figure"
    :PRINT
840 PRINT "<1> J'annulle, <2>
     Autre, <3> Stop"
850 PRINT
860 PRINT "Choisir":GET Z$

870 Z=VAL(Z$)
    :IF Z<1 OR Z>3 THEN 800
880 ON Z GOTO 900,910,1000
890 GOTO 910
900 N=N-1:BB=BB-S

910 N=N+1:BB=BB+S

920 BH=INT(BB/256):BL=BB-BH*256)

930 FOR I=1 TO S:C%(I)=0:NEXT


940 FOR I=1 TO N:A%(I)=0:NEXT


950 TEXT:HOME:PRINT "Faire la
    figure numéro :";N

960 VTAB 5:GOTO 230

970 :

980 REM Menu 1
990 :

1000 TEXT:HOME:VTAB 5

1010 FI$="E":IF N=1 THEN FI$="A"

1020 PRINT:PRINT "Vous avez dessiné "
    :N:" figures":FI$

1030 PRINT:PRINT "Adresse de la
     table :";AD

1040 BT=BB+S
1050 IF N=0 THEN BT=0
1060 PRINT "Longueur :";BT
1070 PRINT:PRINT

1080 PRINT "<1> Sauvegarder la table
     sur disque"
1090 PRINT "<2> Visualisation dans l'
     ordre"
1100 PRINT "<3> Annulation de la
     table"
1110 PRINT "<4> Autre table"
1120 PRINT "<5> Fin"
1130 PRINT
1140 PRINT "Choisissez :":GET Z$
1150 Z=VAL(Z$)
```

```
1160 IF Z<1 OR Z>5 THEN 1000
1170 ON Z GOTO 1210,1260,1490,1190
     ,1100
1180 HOME:END

1190 CLEAR:GOTO 100

1200 :

1210 REM Sauvegarde sur disque
1220 :

1230 IF N=0 THEN HOME:VTAB 12:HTAB 10
     :FLASH:PRINT "Il n'y a pas de
      figure":NORMAL:GET Z$:GOTO 1000



1240 PRINT:INPUT "Sous quel nom ";A$

1250 PRINT CHR$(4)"BSAVE";A$", A";AD;
     ",L";BR+S
1260 GOTO 1000
1270 :

1280 REM Présentation de la table
1290 :

1300 FOR I=1 TO N
1310 RO=0:SC=1

1320 HGR:HCOLOR=3:ROT=RO:SCALE=SC



1330 HOME
1340 VTAB 22:PRINT "Figure n° "
     ;I;"Echelle = ";SC;"Rot = ";RO

1350 VTAB 23:PRINT "Pour changer: <S>
      Echelle, <R> Rot, <F> Figure,
      (avec <CTRL><TOUCHE>, on revient
      en arrière)";
1360 DRAW 1 AT 140,80
1370 GET A$
1380 IF A$="S" THEN SC=SC+1
     :IF SC>255 THEN SC=255
1390 IF A$="R" THEN RO=RO+1
     :IF RO>255 THEN RO=255
1400 IF A$=CHR$(18) THEN RO=RO-1
     :IF RO<0 THEN RO=0

1410 IF A$=CHR$(19) THEN SC=SC-1
     :IF SC<1 THEN SC=1

1420 IF A$="F" THEN 1440
1430 GOTO 1320
1440 NEXT
1450 GOTO 1000
1460 :

1470 REM Effacement de la table
1480 :

1490 PRINT:PRINT "Vraiment (O/N) ?"
     ;GET Z$


1500 IF Z$="O" THEN CLEAR:GOTO 100

1510 GOTO 1000
```

SHAPE EDITOR

Ces écrans apparaissent pendant le déroulement du programme Shape Editor.

Le programme commence par calculer l'adresse mémoire initiale à laquelle sera rangée la table créée.
L'utilisateur peut néanmoins la modifier.

Au départ, la table est vide et le programme présente le n° de la forme en cours de traitement.

Une fois entrés les paramètres (coordonnées initiales, facteurs d'agrandissement), le programme passe en mode graphique et présente le menu des commandes. Ici, le symbole d'un condensateur a été dessiné.

La figure est achevée et l'on peut passer à une autre. La table complète est ensuite enregistrée sur disque.

PRINCIPALES VARIABLES DU PROGRAMME SHAPE EDITOR

A%	Tableau contenant les déplacerments codifiés
C%	Tableau des couples de vecteurs graphiques codifiés
AD	Adresse de départ (en mémoire) de la table des formes (partie basse = AL ; partie haute = AH)
BB	Adresse de départ de la nouvelle forme (partie basse = BL ; partie haute = BH)
IN	Facteur d'agrandissement
M	Nombre de mouvements composant la figure
N	Numéro de la figure
RO	Indicateur de rotation d'une figure visualisée
SC	Indicateur d'échelle d'une figure visualisée
X,Y X1,Y1	Coordonnées graphiques

Le chargeur de formes. La mémorisation d'une table des figures sur disque n'a de sens que s'il existe un programme capable de la rappeler et de s'en servir.

La définition d'un certain nombre de figures graphiques, regroupées en une même table, permet, en effet, de disposer d'un menu de symboles qui peuvent être combinés de différentes façons pour former un dessin, même complexe.

Les performances du programme Shape Editor doivent donc être complétées par un programme capable d'effectuer les opérations de lecture d'une table de figures et de composition de ses éléments à l'écran, (voir organigramme 1 page suivante).

L'utilisateur fait son choix à partir d'un menu initial. En réponse, le programme passe la main à la partie du programme appelée. Le sous-programme 300, (organigramme 2, p. suiv.) charge en mémoire une table précédemment enregistrée sur disque. L'écran affiche alors le menu.

Avec le sous-programme 900 (organigramme 3, page 1625), on visualise une des figures de la table. Après ces opérations, le programme réaffiche le menu. Il est ensuite possible d'entrer dans la phase de composition d'une image exécutée avec l'option 3 (sous-programme 500 p. 1625). Le sous-programme 500 (organigramme 3 bis) sélectionne le mode graphique et demande le n° de la figure à afficher ainsi que le point de l'écran à partir duquel elle doit être visualisée.

Une fois ces paramètres entrés, l'image demandée est présentée et le programme passe au bloc 580 de saisie et d'exécution des commandes de composition des figures (voir tableau 4 de la page 1626).

Les fonctions décrites sont complétées par la possibilité de demander le répertoire (directory) du disque et par la sauvegarde de l'image composée.

Le programme de changement des formes est listé pages 1626 à 1630 (les variables utilisées sont résumées dans le tableau 4, page 1630).

Notons que les deux programmes présentés ont été développés sur un ordinateur Siprel. Ils ne fonctionneront sous cette forme que sur les ordinateurs personnels de la famille Apple et compatibles. Pour les transporter sur d'autres machines, il faut recourir aux règles de transposition désormais bien connues.

Instructions Basic de gestion des figures graphiques

Une fois mémorisée, une figure graphique peut, sur certaines machines, être traitée comme une donnée d'ensemble à l'aide d'instructions de haut niveau comme celles-ci.

DRAW	Tracé de la figure
XDRAW	Effacement
ROT	Rotation
SCALE	Changement de facteur d'échelle

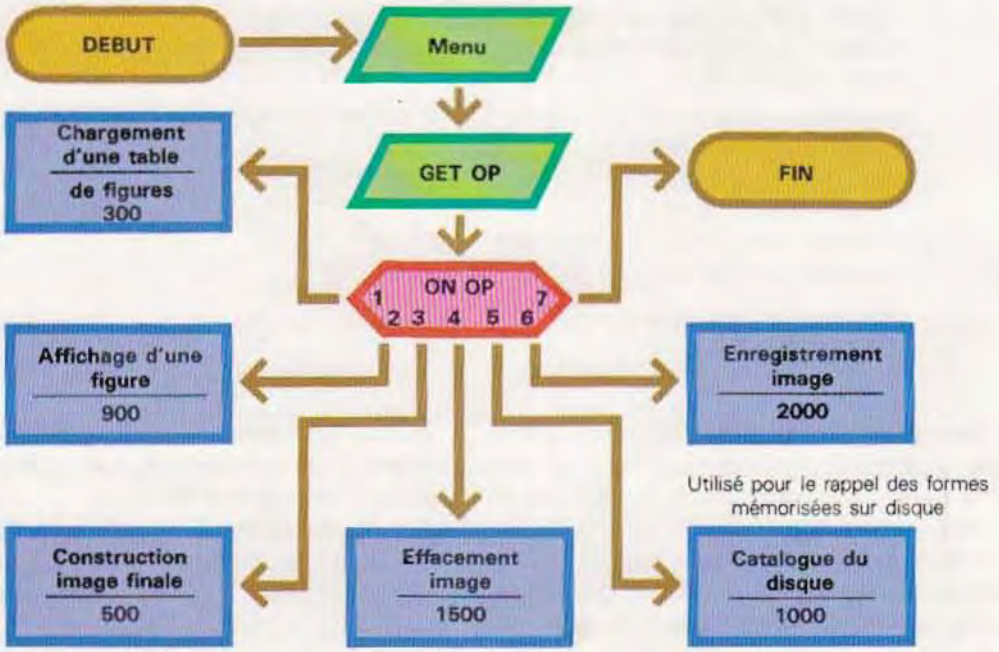
Dans ce qui suit, nous citerons les instructions utilisées par les systèmes Apple.

DRAW. La syntaxe de cette instruction est :

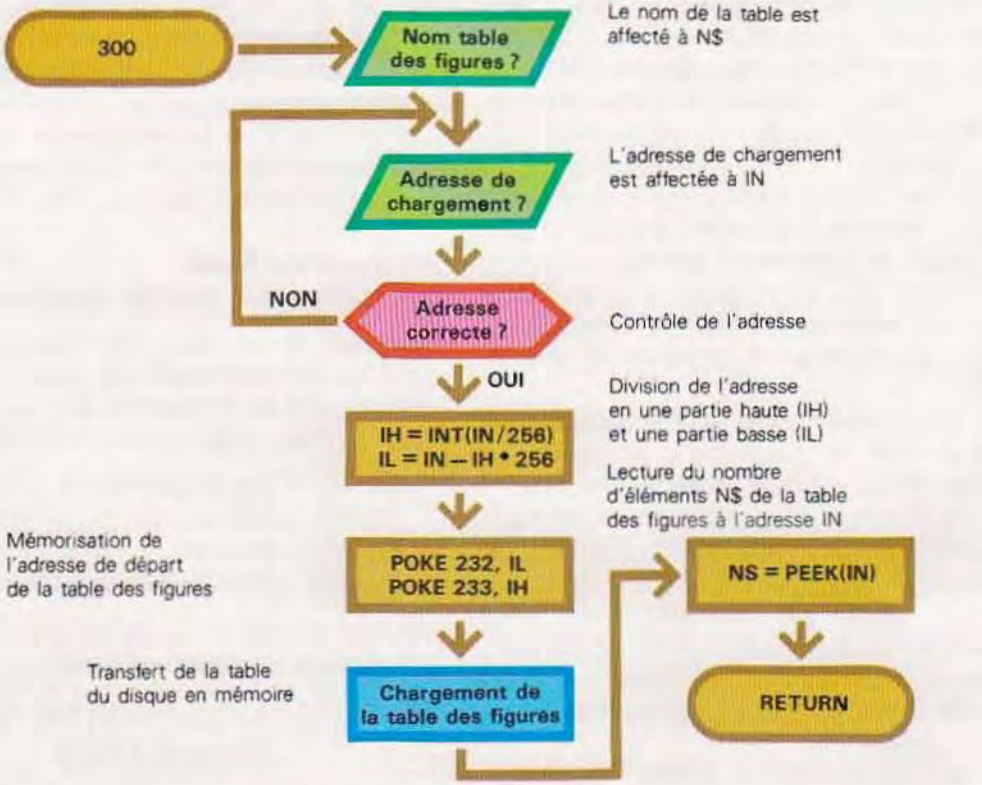
DRAW N AT X,Y

dans laquelle :

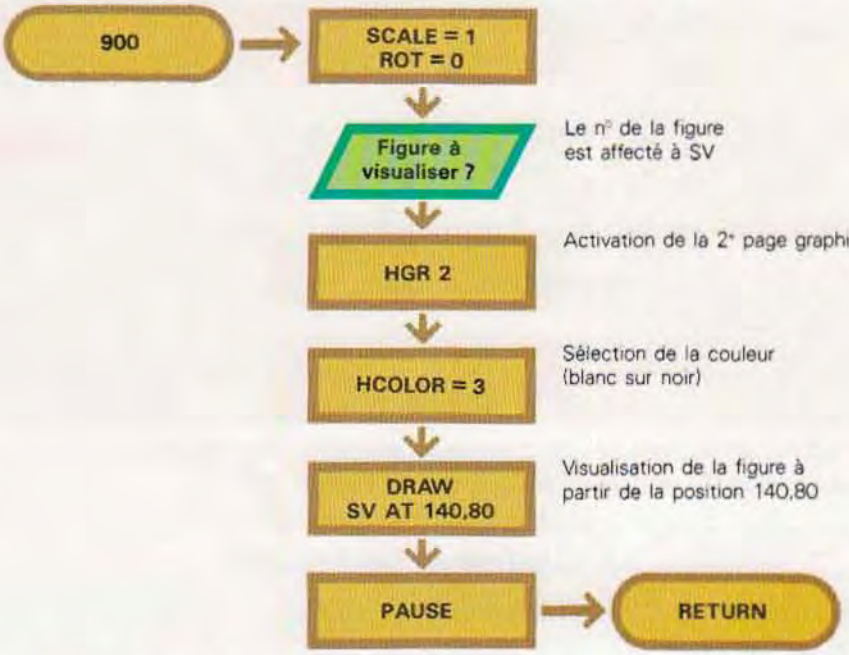
1/ SHAPE LOADER (CHARGEUR DE FORMES)



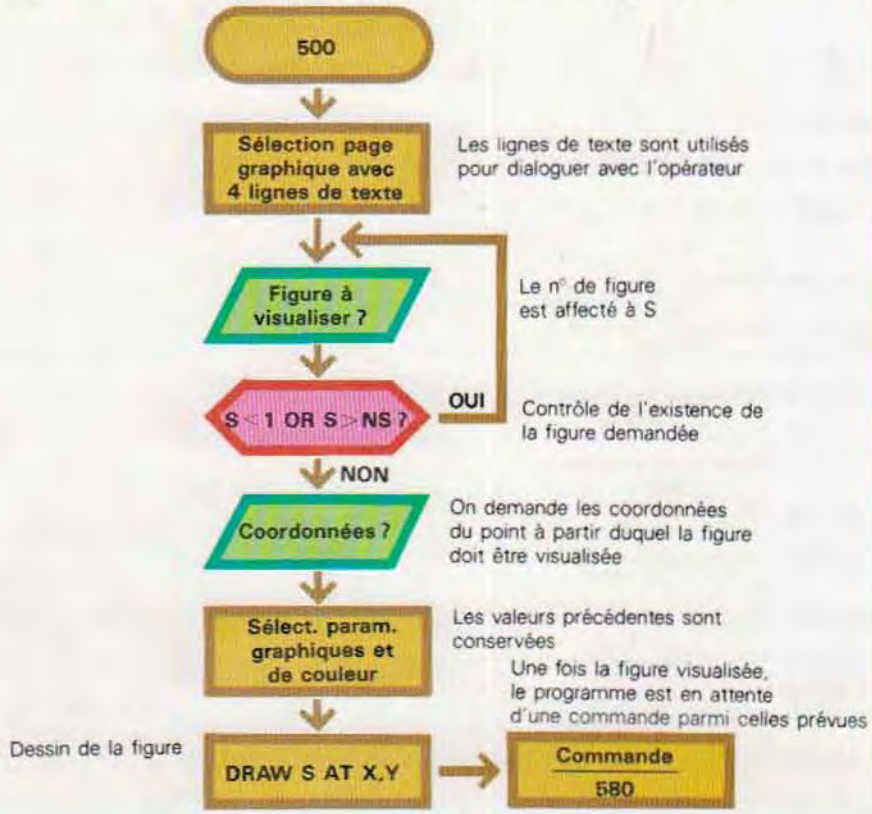
2/ SOUS-PROGRAMME DE CHARGEMENT D'UNE TABLE DE FIGURES



3/ SOUS-PROGRAMME DE VISUALISATION D'UNE FIGURE



3 bis/ Sous-programme de construction de l'image finale



4/ COMMANDES DU SOUS-PROGRAMME 580

Commandes	Signification
/	Effacement de la fenêtre texte
?	Visualisation de la fenêtre texte
I	Déplacement de la figure d'un pas vers le haut
J	Déplacement de la figure d'un pas à gauche
K	Déplacement de la figure d'un pas à droite
M	Déplacement de la figure d'un pas vers le bas
R	Rotation de la figure de 90° à droite
S	Agrandissement de la figure
CTRL + R	Rotation de la figure de 90° à gauche
CTRL + S	Réduction de la figure
C	Effacement
ESC	Retour au programme principal

Le sous-programme 580 permet de composer les différentes figures de la table chargée en mémoire jusqu'à ce qu'un dessin complexe soit formé.

LE CHARGEUR DE FIGURES

```
10 REM ************************
20 REM *                      *
30 REM * CHARGEUR DE FIGURES *
40 REM *                      *
50 REM ************************
60 :

70 REM MENU
80 :

90 TEXT:HOME

100 HI=PEEK(115)+PEEK(116)*256

110 HTAB 18:INVERSE:PRINT "Menu"

120 NORMAL:VTAB 6

130 PRINT "Chargement d'une table de
    figures ... 1":PRINT

140 PRINT "Affichage d'une figure
    ..... 2":PRINT

150 PRINT "Construction image finale
    ..... 3":PRINT
160 PRINT "Effacement image
    ..... 4":PRINT
170 PRINT "Catalogue du disque
    ..... 5":PRINT

180 PRINT "Enregistrement image
    ..... 6":PRINT

190 PRINT "Fin .....

    ..... 7":PRINT
200 VTAB21

210 INPUT "Quelle option choisissez-
    vous ";OP$
```

```
220 OP=VAL(OP$):IF OP<1 OR OP>7
    THEN 90
230 ON OP GOSUB 300,900,500,1500,1000
    ,2000
240 IF OP<>7 THEN 90
250 HOME:END

300 :

310 REM Chargement d'une table de
    figures
320 :
330 TEXT:HOME

340 INPUT "Nom de la table de figures
    ?";n$
345 IF ASC(LEFT$(N$,1))<>65 OR
    ASC(LEFT$(N$,1))>90 THEN 330
350 PRINT:PRINT

360 PRINT "Adresse de chargement :"
    :print "(24576-";HI;")" 370 VTAB
    4:HTAB 26:PRINT SPC(14)



380 VTAB 4:HTAB 26:INPUT " ";IM$


390 IN=VAL(IN$)
400 IF IN<24576 OR IN>HI THEN 370

410 IH=INT(IN/256):IL=IN-IH*256

420 POKE 232,IL:POKE 233,IH

425 VTAB 16:INVERSE:PRINT "Chargez un
    disque et tapez <RETURN>":NORMAL
    :GET R$



427 PRINT CHR$(4)
430 PRINT CHR$(4):"BLOAD";N$;", A";IN

435 MS=PEEK(IN)
440 RETURN
500 :

505 REM Construction de l'image
510 :

515 IF MS=0 THEN HOME:VTAB 12:FLASH
    :PRINT "Il n'y a pas de table en
    mémoire":NORMAL:GET A$:RETURN




520 POKE -16297,0:POKE -16300,0
    :POKE -16301,0:POKE -16304,0
    :POKE 230,32

525 HOME:VTAB 21

530 PRINT "Figure à visualiser (1 - "
    ;MS;")":INPUT " ?";S$
535 IF S$="" THEN RETURN

540 S=VAL(S$)
545 IF S<1 OR S>MS THEN 525
```

```
546 VTAB23:INPUT "Coordonnées (X,Y) "
    ;X$,Y$:X=VAL(X$):Y=VAL(Y$)


547 IF X<0 OR X>279 OR Y<0 OR Y>191
    THEN 546
550 RO=0:SC=1

555 X1=X:Y1=Y:R1=RO:S1=SC


557 HCOLOR=3
560 SCALE=SC:TOR=RO

565 DRAW S AT X,Y
570 HOME:VTAB 23

575 INVERSE:PRINT "Commandes: I, J, K,
     M R, ^R S, ^S /, ? C ESC";
    :NORMAL

580 HTAB 1:VTAB 21:PRINT SPC(79)


585 HTAB 1:VTAB 21

590 PRINT "Figure = ";:INVERSE:PRINT S
    :NORMAL


595 PRINT "Echelle = ";:INVERSE:PRINT
     SC;:NORMAL


600 PRINT "Rot = ";:INVERSE:PRINT RO;
    :NORMAL


605 PRINT "X = ";:INVERSE:PRINT X
    :NORMAL


610 PRINT "Y = ";:INVERSE:PRINT Y;
    :NORMAL


615 VTAB 1:HTAB 1:GET T$


620 IF T$="/" THEN POKE -16302,0
625 IF T$="?" THEN POKE -16301,0
630 IF T$="I" THEN Y=Y-1
    :IF Y<0 THEN Y=191
635 IF T$="J" THEN X=X-1
    :IF X<0 THEN X=279
640 IF T$="K" THEN X=X+1
    :IF X>279 THEN X=0
645 IF T$="M" THEN Y=Y+1
    :IF Y>191 THEN Y=0
650 IF T$="R" THEN RO=RO+1
    :IF RO>255 THEN RO=255
655 IF T$=CHR$(18) THEN RO=RO-1
    :IF RO<0 THEN RO=0

660 IF T$="S" THEN SC=SC+1
    :IF SC>255 THEN SC=255
665 IF T$="C" THEN GOSUB 1000
670 IF T$=CHR$(19) THEN SC=SC-1
    :IF SC<1 THEN SC=1
```

```
675 IF T$=CHR$(27) THEN RETURN

677 HCOLOR=0

680 SCALE=S1:ROT=R1

685 DRAW S AT X1,Y1

687 HCOLOR=3

690 SCALE=SC:ROT=RO:DRAW S AT X,Y

695 S1=SC:R1=RO:X1=X:Y1=Y

700 GOTO 580

900 :

905 REM Affichage d'une figure

910 :
915 IF N$=0 THEN HOME:VTAB 12:FLASH
    :PRINT "Il n'y a pas de table en
    mémoire":NORMAL:GET A$:RETURN

920 TEXT:HOME

925 SCALE=1:ROT=0

930 VTAB 12:HTAB 1

935 PRINT "Figure à visualiser (1 - "
    ;n$;") ";:INPUT SV$:SV=VAL(SV$)

940 IF SV<1 OR SV>N$ THEN 920

945 HGR2:HCOLOR=3

950 DRAW SV AT 140,80

955 GET R$

960 RETURN

1000 :
1010 REM Catalogue

1020 :
1030 HOME:PRINT CHR$(4);"Catalogue"

1040 GET A$:RETURN
1075 PRINT CHR$(4)

1500 :
1510 REM Effacement

1520 :
1530 HOME:VTAB 12:INPUT "Certain (O/N)
     ?";R$
1540 IF R$<>"O" AND R$<>"N" THEN 1530

1550 IF R$="O" THEN HGR
1560 RETURN

1800 :
1810 REM Mode effacement

1820 :
1830 X0=X:Y0=Y:X=140:Y=80
```

```
1835 VTAB 23:HTAB 36:PRINT "C"

1840 HCOLOR=3:HPLOT X,Y

1850 VTAB 1:HTAB 1:GET A$

1855 HCOLOR=0:HPLOT X,Y

1860 IF A$="I" THEN Y=Y-1
     :IF Y<0 THEN Y=191
1870 IF A$="M" THEN Y=Y+1
     :IF Y>191 THEN Y=0
1880 IF A$="J" THEN X=X-1
     :IF X<0 THEN X=279
1890 IF A$="K" THEN X=X+1
     :IF X>279 THEN X=0
1900 IF A$="C" THEN 1920
1910 GOTO 1840
1920 HCOLOR=3:X=X0:Y=Y0

1930 VTAB 23:INVERSE

1940 PRINT "Commandes: I, J, K, M R^R
      S, ^S /, ? C ESC";:NORMAL

1950 RETURN
2000 :

2010 REM Sauvegarde de l'image
2020 :

2030 HOME:VTAB 12

2040 INPUT "Sous quel nom ";NM$
2050 IF ASC(LEFT$(NM$,1))<65 OR
     ASC(LEFT$(NM$,1))>90 THEN 2030

2060 VTAB 14:PRINT "C'est bon !";NM$
     ", PTC"
2070 VTAB 16:INVERSE:PRINT "Chargez un
     disque et tapez <RETURN>":NORMAL
     :GET R$


2080 PRINT CHR$(4)
2090 PRINT CHR$(4);"BSAVE";NM$;", PTC
     , A$2000,L$1FFF"
2100 PRINT CHR$(4)
2110 RETURN
```

VARIABLES EMPLOYEES DANS LE PROGRAMME SHAPE LOADER

HI	= Limite de la mémoire réservée au graphique
IN	= Adresse initiale de la table des figures
N$,NM$	= Nom de la table des figures
NS	= Numéro de la figure dans la table
R1,RO	= Valeurs des ROT (rotations)
S$	= Figure en cours
S1,SC	= Valeurs de SCALE (échelle)
X,Y,XO,YO,X1,Y1	= Coordonnées à l'écran

EXEMPLE D'APPLICATION DU PROGRAMME SHAPE LOADER

Cette série d'écrans illustre une application de composition d'une image utilisant des formes élémentaires (shapes) créées et mémorisées à l'aide du programme Shape Editor

Le programme Shape Loader affiche le menu. L'utilisateur demande le chargement à partir du disque d'une table de figures contenant des symboles électriques

Le sous-programme de composition de l'image finale a été désigné et est exécuté. On peut voir, en bas, la ligne de dialogue et le menu des commandes ; au centre de l'écran, se trouve le dessin en cours de formation. L'élément de gauche est la figure qui va être positionnée.

La construction de la figure composée est terminée. L'utilisateur devra la mémoriser sur disque en lui donnant un nom particulier pour pouvoir la rappeler en cas de besoin

N : C'est le n° de la figure à visualiser
X,Y : Sont les coordonnées de départ de l'affichage de la figure.

Par exemple, l'instruction :

DRAW 2 AT 30,50

présente la figure n° 2 en commençant le dessin à partir du point de coordonnées X = 30 et Y = 50. Si les coordonnées sont omises, la figure commence au dernier point rappelé. Cela signifie que cette instruction peut également se présenter sous la forme DRAW N, qui lance l'affichage de la figure N à partir des dernières coordonnées utilisées dans une précédente instruction DRAW ou HPLOT (pour le positionnement).

XDRAW. Sa forme est identique à celle de l'instruction précédente :

XDRAW N AT X,Y

Elle commande l'effacement de la figure n° N, positionnée au point de coordonnées X,Y.

ROT. La syntaxe est très simple :

ROT M

où M est une valeur numérique exprimant l'angle de rotation, généralement en degrés.
La validité du paramètre M dépend de la valeur définie par l'instruction précédente SCALE. Si le facteur d'échelle est 1, le paramètre M aura une des valeurs suivantes : 0 = 0°, 16 = 90°, 32 = 180°, 48 = 270°.
En revanche, si ce facteur d'échelle est supérieur à 4, il est possible d'avoir 64 rotations différentes à des intervalles réguliers par rapport aux valeurs précédentes (ces valeurs ne sont qu'un exemple lié au matériel considéré et ne constituent donc pas une règle).

SCALE. Cette instruction doit être utilisée en priorité, avant toute autre instruction de ce groupe, puisqu'elle définit l'échelle du dessin.
Sa syntaxe est :

SCALE = Q

dans laquelle le paramètre numérique Q indique le nombre de tracés pour chaque vecteur graphique. Par exemple : SCALE = 1 provoque l'affichage d'une figure dans laquelle chaque vecteur n'est tracé qu'une fois, alors qu'avec SCALE = 2, il le sera deux fois.
Sur certaines machines, il est possible de préparer un dessin composé de déplacements élémentaires à l'aide d'une chaîne de commandes. Le mot-clé reste généralement DRAW, mais sa syntaxe et la signification de ses éléments sont différents, bien que proches.
Cette instruction, peut prendre deux formes :

— Dans la première, les commandes de déplacement sont contenues dans une chaîne associée à l'instruction.

— Dans la seconde, elles sont mémorisées dans une table résidente en mémoire ; l'instruction n'admet alors comme paramètre que le n° correspondant à la figure demandée :

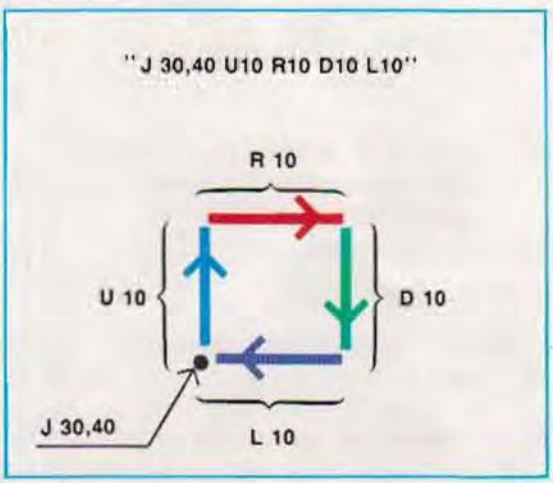
DRAW : « chaîne de commandes »
DRAW : N avec N = n° de la figure à visualiser.

Dans la chaîne de commandes, il est nécessaire d'expliquer certains symboles qui représentent des actions élémentaires à exécuter.

Ces paramètres sont :

M DX,DY : Positionnement du curseur au point de coordonnées X = DX et Y + DY en partant du point X,Y.
J,X,Y : Déplacement au point X,Y
U DY : Déplacement de DY vers le haut
L DX : Déplacement de DX vers la gauche.
R DX : Déplacement de DX à droite
C : Indication de la couleur du tracé de la figure ; s'il est omis, le système prend par défaut la dernière couleur utlisée dans une instruction DRAW.

Les paramètres ci-dessus correspondent au système Olivetti M20 ; ils peuvent être différents pour les autres machines, mais si cette forme de l'instruction DRAW existe, la logique reste la même.
La figure ci-dessous montre, à titre d'illustration, la chaîne des commandes nécessaires au tracé d'un carré de côté 10 positionné au point 30,40 (sommet inférieur gauche). La chaîne adresse = 30 et Y = 40 (code J) indique les quatre déplacements nécessaires pour obtenir les côtés.
Telle qu'elle est présentée, l'instruction trace également une ligne entre la position d'origine du curseur et le premier sommet du carré puisque la commande J 30,40 est exécutée en mode visible (dans la couleur spécifiée ou la dernière utilisée). Pour éviter que ce segment (non désiré) soit visualisé, il faut précéder la commande de la lettre B, qui invalide l'affichage, la forme exacte de la chaîne est donc :
"BJ30, 40 U10 R10 D10 L10".



On peut utiliser une des options suivantes avec chacune des commandes :

AND : La couleur résultante est obtenue par intersection logique de 2 couleurs, la précédente et celle spécifiée dans la chaîne

XOR, OR : Même principe, mais avec une union logique (OU exclusif ou inclusif)

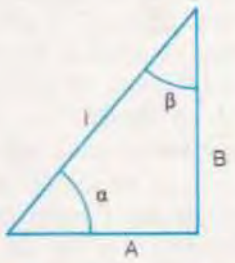
NOT : Indique que la couleur de la figure est le complément de celle utilisée comme fond.

Graphisme tridimensionnel (3D)

La préparation d'un graphique en trois dimensions (3D) exige le recours à des formules mathématiques parfois complexes, qui seront développées plus loin. Les plus employées sont celles de la trigonométrie appliquée aux problèmes de triangles rectangles.

La figure ci-dessous indique, dans le tableau central, comment calculer les éléments inconnus d'un triangle rectangle dans les différents cas de figure.

RESOLUTION DES TRIANGLES RECTANGLES



$B = I * \sin(\alpha)$
$A = I * \cos(\alpha)$

$\frac{B}{A} = \frac{\sin(\alpha)}{\cos(\alpha)} = \mathrm{tg}(\alpha)$

Eléments connus*	Calcul des autres
I, α	Appliquer les formules suivantes
A, B	$\alpha = \mathrm{cotg}\ (B/A)$ $I = A/\cos(\alpha) = B/\sin(\alpha)$
A, α	$I = A/\cos(\alpha)$ $B = I.\sin(\alpha) = A.\frac{\sin(\alpha)}{\cos(\alpha)} = A.\mathrm{tg}(\alpha)$
B, α	$I = B/\sin(\alpha)$ $A = I.\cos(\ ) = B.\frac{\cos(\alpha)}{\sin(\alpha)} = \frac{B}{\mathrm{tg}(\alpha)}$
I, A	$\cos(\alpha) = A/I$ $\sin(\alpha) = \sqrt{1 - \cos^2(\alpha)}$ $B = I.\sin(\alpha)$
I, B	$\sin(\alpha) = B/I$ $\cos(\alpha) = \sqrt{1 - \sin^2(\alpha)}$ $A = I.\cos(\alpha)$

Toutes les formules peuvent être traduites en langage symbolique pour être utilisées dans un programme.
Ne pas oublier que :
— l'angle α doit être en radians
— les noms de fonctions sont généralement :

sin(α) = SIN(ALPHA)
cos(α) = COS(ALPHA)
tg(α) = SIN(ALPHA)/COS(ALPHA)

√ = SQR(.)
COTG (B/A) = ATN(B/A)

*Le cas où c'est l'angle β qui est connu, au lieu de l'angle α, n'est pas pris en considération. En effet, étant donné que α = 90 – β, ce cas peut être résolu à l'aide d'une des figures exposées.

Les fonctions trigonométriques sont (notation mathématique) :

sinus=sin(α)
cosinus=cos(α)
tangente=tg(α)
cotangente=cotg(α)

Pour obtenir les côtés de l'angle droit, à partir de l'hypoténuse I et de l'angle α, il suffit d'appliquer les formules :

A=I*cos(α)
B=I*sin(α)

Les principales fonctions trigonométriques ont été prévues dans les langages de programmation comme le Fortran et le Basic.
Les instructions nécessaires pour effectuer les calculs précédents sont :

```
A=I*COS(ALPHA)
B=I*SIN(ALPHA)
```

à condition de convertir l'angle en radians, selon l'expression :

$$\text{angle en radians} = \frac{\pi * \text{angle en degrés}}{180} = \frac{3{,}14 * \text{angle en degrés}}{180}$$

Certains langages ne comportant pas la fonction tg(α), il faut la calculer à l'aide de l'expression :

```
TG(ALPHA)=SIN(ALPHA)/COS(ALPHA)
```

Dans le programme, la fonction tangente peut être définie par l'utilisateur (au moyen de l'instruction DEF...) et assimilée à une fonction particulière.
La résolution des triangles rectangles est plus difficile, quand les seuls éléments connus sont les côtés, à l'exclusion de tout angle. On peut alors recourir au théorème de Pythagore, mais aussi aux fonctions trigonométriques inversées. Par exemple, si l'on connaît les deux côtés de l'angle droit, on peut calculer l'angle α en utilisant la fonction cotangente (ATN) :

```
ALPHA=ATN(B/A)
```

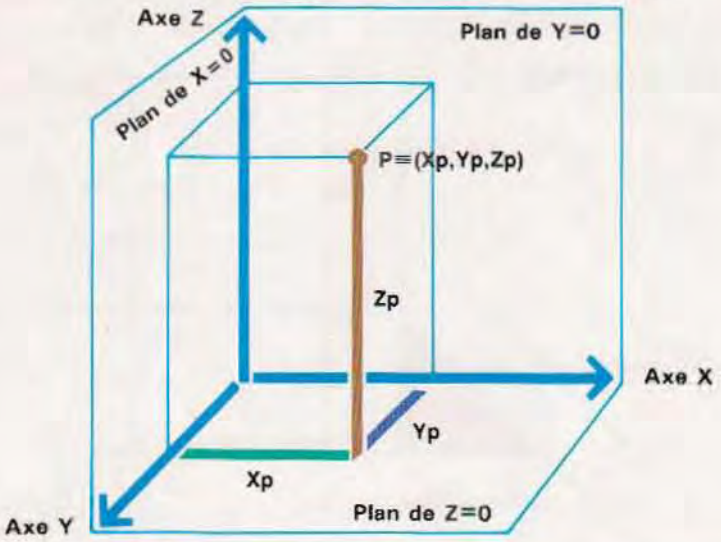
A et B étant les côtés de l'angle droit.

La fonction cotangente fait généralement partie du langage de programmation.

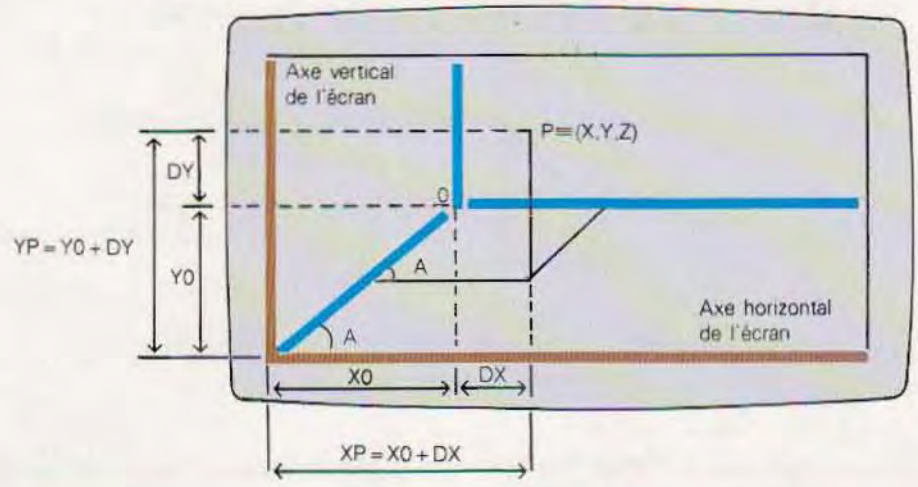
Les fonctions à deux variables

Jusque-là, les fonctions rencontrées se présentaient sous la forme Y=f(X) où Y est fonction d'une seule variable indépendante, X. Les fonctions à une variable sont représentables sur un plan cartésien, puisque chaque point est, en effet, repéré par un couple de valeurs (coordonnées). Les courbes qu'elles représentent sont facilement visualisables en faisant coïncider les axes de référence avec ceux de l'écran.
Mais il existe des fonctions à plusieurs variables et les plus fréquemment utilisées comportent deux variables indépendantes, du type Y=f(X,Z) ou Z=f(X,Y).
Nous reprenons là les symboles habituels : f désigne une fonction générique, c'est-à-dire une loi unissant les valeurs de la variable dépendante à celles des variables indépendantes. Le choix de la forme, ici Y=f(X,Z), est plus proche des formes déjà connues.
Une fonction de ce type ne peut pas être visualisée sur un plan, puisqu'il faut reporter les valeurs d'une 3e coordonnée (Z). Le système de référence adopté doit donc comporter trois axes. La figure 1 (page ci-contre) montre un système de référence à 3 axes X, Y et Z. Chacun des points de l'espace est défini par leurs coordonnées relatives.
De toute évidence, une représentation à deux variables laisse de côté la possibilité d'un 3e axe sur l'écran, pour, par exemple, figurer un objet tridimensionnel sur un plan (feuille de dessin). Pour corriger cette lacune, on a donc imaginé de simuler la profondeur et, pour cela, construire des algorithmes analogues à ceux employés par un dessinateur dans la réalisation d'une perspective ou d'une figure axonométrique.
Les méthodes exposées ici ne sont pas de véritables applications de graphisme tridimensionnel. Il s'agit plutôt de simulations s'adaptant à un grand nombre d'utilisations mais qui sont toutefois limitées par l'environnement matériel.
Pour réaliser un véritable logiciel graphique 3D, il faut disposer de machines très puissantes.

1/SYSTEME DE REFERENCE DANS L'ESPACE



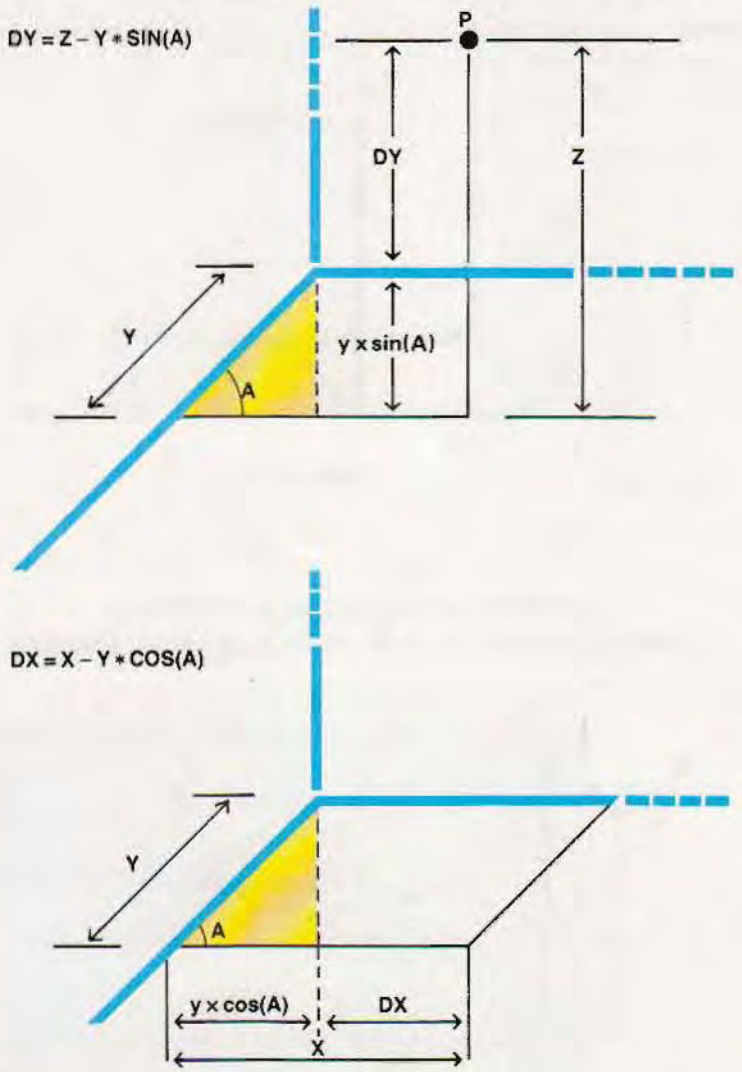
2/REPRESENTATION A L'ECRAN D'UN SYSTEME DE REFERENCE DANS L'ESPACE



Représentation 3D. Le procédé utilisé, pour représenter à l'écran un point quelconque P de l'espace, est schématisé par la figure 2.
L'origine du système de référence, désigné par la lettre O, est positionnée sur un point de l'écran. Pour des raisons de simplicité, nous considérerons que l'origine des axes de l'écran se trouve en bas à gauche (dans certains systèmes, elle est en haut à gauche).
Par rapport à l'écran (du point de vue strictement graphique), le point P (défini par les coordonnées X, Y, Z) a pour coordonnées XP et YP.
Ces coordonnées s'obtiennent en ajoutant respectivement DX et DY aux coordonnées de l'origine des axes.
On a donc :

XP=XO+DX
YP=YO+DY

CALCUL DES COORDONNEES A L'ECRAN



Une fois connues les valeurs DX et DY, on calcule les coordonnées écran de tous les points de la fonction 3D. En dernière analyse, le problème se réduit donc au calcul des quantités DX et DY.
La première phase consiste à calculer l'angle A en utilisant les formules trigonométriques précédentes (pour des raisons pratiques, l'angle est désigné par la seule lettre A, alors qu'il est appelé ALPHA dans les formules). La valeur de A est déterminée par les coordonnées X0, Y0. Celles-ci constituent les côtés de l'angle droit d'un triangle rectangle ayant pour hypoténuse le segment unissant l'origine (écran) au point 0.
Le résultat est: A=ATN(Y0/X0).
Une fois cet angle déterminé, on procède au calcul de DX et DY, toujours à l'aide des règles relatives aux triangles rectangles.
De cette façon, on obtient :

DY=Z-Y*SIN(A)
DX=X-Y*COS(A)

Pour comprendre les formules précédentes, il faut observer les figures ci-dessus, sans prendre en compte l'aspect 3D du système de référence.

Voyons, maintenant, comment tracer une fonction tridimensionnelle :

1/Affectation des valeurs arbitraires aux variables indépendantes X et Z
2/Calcul de la valeur correspondante de Y en utilisant l'expression mathématique de la fonction
3/Calcul des coordonnées XP et YP en points écrans à l'aide des formules :

```
YP=XO+DX
YP=YO+DY
```

ou avec les formules équivalentes :

```
XP=XO+(X-Y*COS(A))
YP=YO+(Z-Y*SIN(A))
```

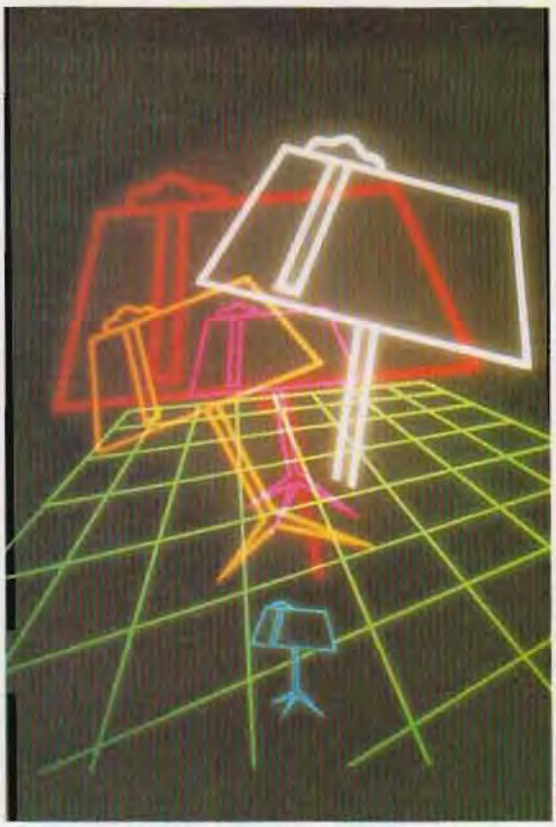
dans lesquelles DX et DY ont été remplacés par :

```
DX=X-Y*COS(A)
DY=Z-Y*SIN(A)
```

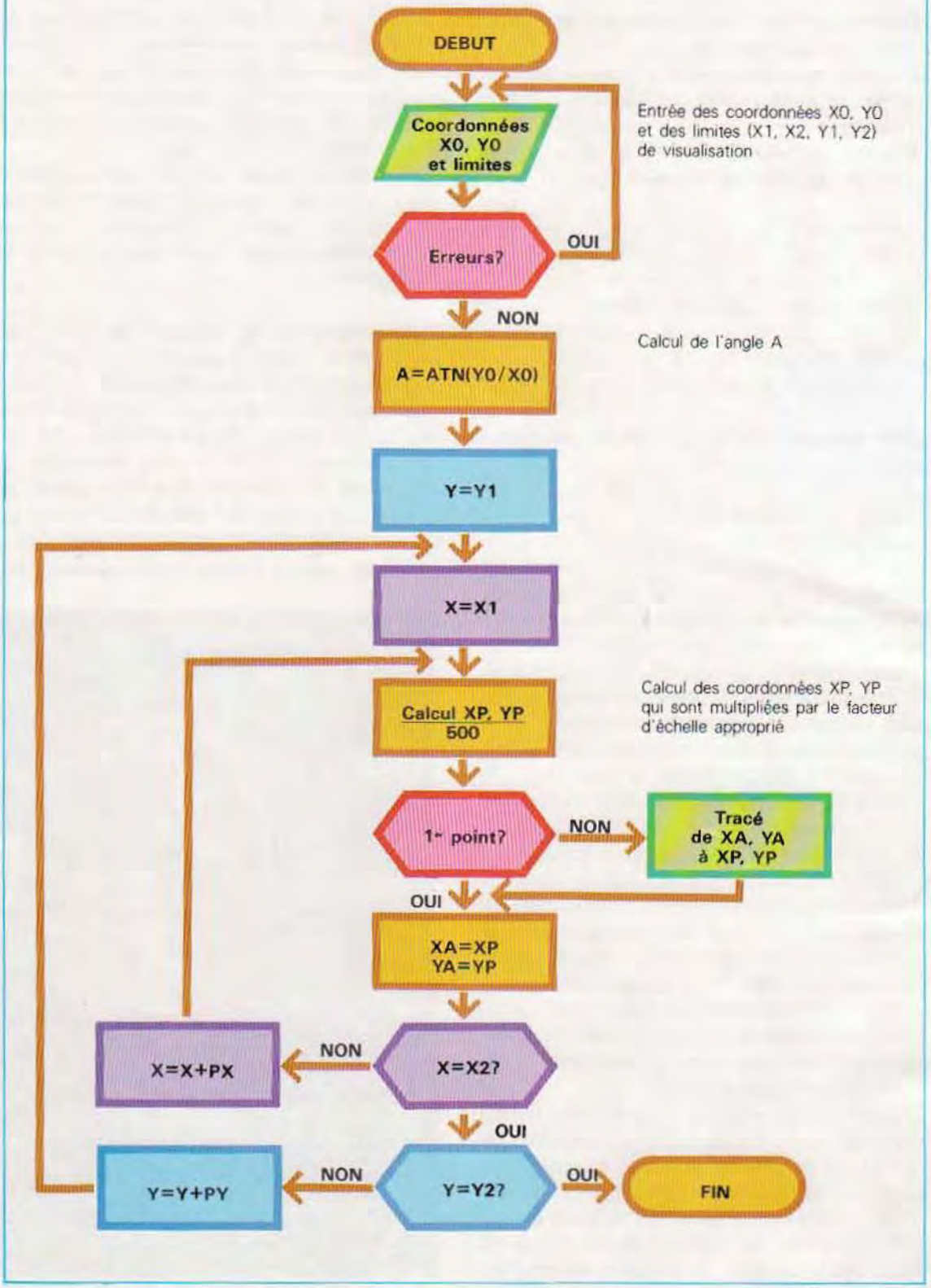
Notons que les coordonnées XP, YP en points écran n'ont rien de commun avec les coordonnées X, Y des points des courbes. Par conséquent, si la fonction est de la forme Z=f(X,Y), et non plus Y=f(X,Z), toutes les formules restent valides, à condition de changer les coordonnées utilisées dans les calculs. Dans ce cas, les variables indépendantes ne sont plus X,Z, mais X,Y ; la fonction permet donc de calculer Z à la place de Y.
La forme Z=f(X,Y) est la plus courante. Pour simplifier, nous avons omis tous les contrôles et les changements d'échelle utilisés dans les programmes de tracé des fonctions dans le plan. La bouche externe est construite à partir de l'incrémentation de Y (considérée comme coordonnée de la fonction), et la boucle interne avec des valeurs de X. Cela équivaut à présenter le tracé de la fonction par sections successives, parallèles au plan XZ.
Supposons, par exemple, que la valeur initiale de la coordonnée Y soit zéro, c'est-à-dire que Y1=0. La bouche interne (sur X) montrerait alors tous les points de la courbe pour lesquels Y=0, c'est-à-dire la partie située dans le plan Y=0. Une fois les valeurs de X pour Y=0 épuisées, la variable de la boucle externe est incrémentée. Y passe donc à une nouvelle valeur, par exemple 1. Dans ce cas, la boucle interne présenterait la courbe de la fonction dans le plan Y=1 (parallèle au précédent). En continuant ainsi à incrémenter Y de 1, on obtient le graphique 3D de la fonction, par superposition d'autant de graphiques plans dont chacun montre la courbe dans un plan particulier (Y=0, Y=1, Y=2...).
Ces représentations sur des plans parallèles sont des sections de la figure solide représentée par la courbe. Le mécanisme décrit est schématisé en page 1640 suivi du listing du programme.

Les problèmes du dessin en 3D. Il faut distinguer le véritable graphisme 3D de la simple représentation tridimensionnelle. En effet il y a là une différence considérable qui se retrouve tant au niveau des équipements, que des programmes et de leurs coûts respectifs. Le graphisme 3D demande la mémorisation des attributs de chacun des points de l'espace que

Image obtenue à l'aide d'un programme 3D.

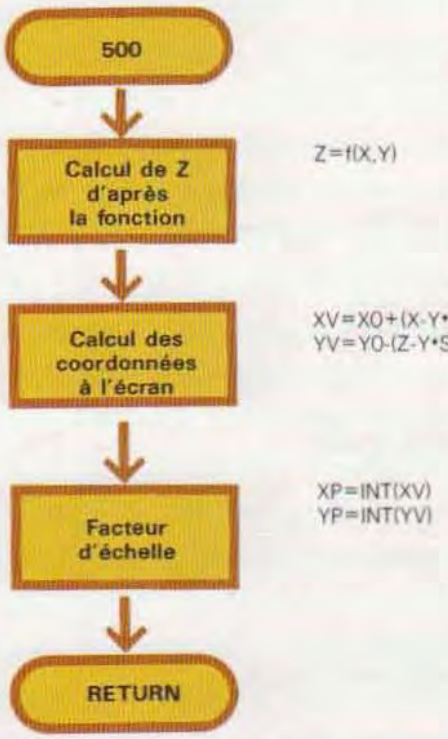


VISUALISATION D'UNE FONCTION A 2 VARIABLES

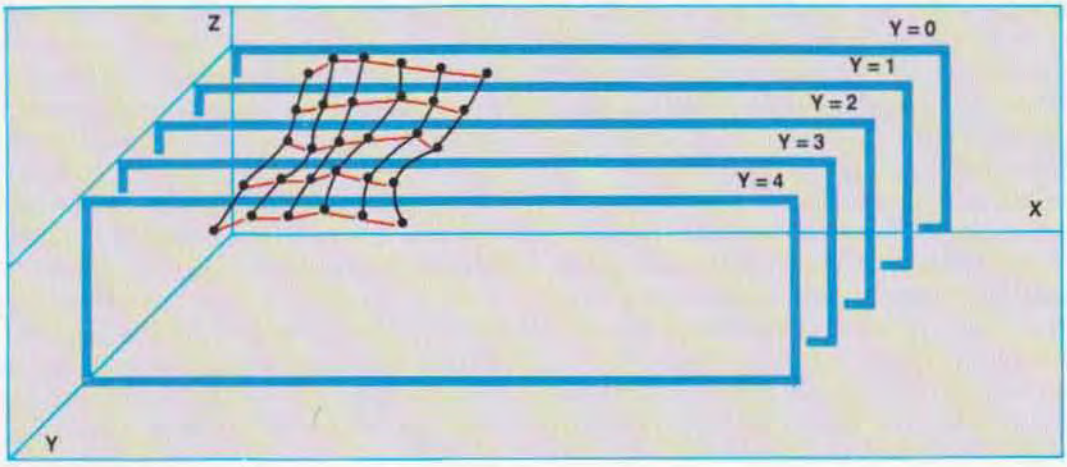


constitue le solide à représenter. On peut alors obtenir des projections, des vues en coupe ou tout autre traitement graphique d'un objet.
Mais un tel logiciel ne peut fonctionner que sur des machines très sophistiquées, notamment en terme de capacité mémoire et en résolution du terminal vidéo ou des autres périphériques (traceur de courbes par exemple).
Dans le cas de la représentation 3D, en revanche, le problème est nettement plus simple et peut être réduit à un simple traitement d'une vue de l'objet donnant une impression de tridimensionnalité, ce qui ne demande pas des machines et des programmes complexes.
Bien entendu, il n'est nullement question de visualiser une vraie coupe, en l'absence des attributs relatifs à la 3e dimension.
Dans la méthode utilisée pour la représentation 3D (voir graphique p. 1642), la 3e dimension est obtenue par translation de la figure de base (vue frontale de l'objet) qui est inclinée de 45° (de ce fait, les déplacements de chaque point sont égaux sur les deux axes). Les sommets de la figure de base étant numérotés de 1 à 6, l'impression d'épaisseur est obtenue en traçant la même figure après avoir incrémenté les coordonnées des sommets de la quantité S ; une fois que le contour déplacé est réalisé, il faut unir les points homologues. Il reste toutefois à résoudre le problème de l'effacement (ou mieux du non tracé) des segments de la vue déplacée qui sont couverts par la figure de base. Nous touchons là à un des problèmes les plus complexes du graphisme 3D : comment déterminer les lignes à éliminer. En effet, les surfaces qu'elles circonscrivent sont cachées par d'autres parties de l'objet représenté.
Il existe de nombreux algorithmes de résolution de cette question, dont la plupart sont utilisables dans le tracé de fonctions. Mais il s'agit là généralement de solutions très spécialisées et difficilement généralisables aux dessins dont les lignes ne sont pas des fonctions mathématiques.
Autrement dit, il faut que le logiciel soit capable de trouver les lignes cachées et d'éviter qu'elles ne soient tracées.
La méthode utilisée dans le programme que

SOUS-PROGRAMME DE CALCUL DES COORDONNEES A L'ECRAN



SCHEMA DE LA METHODE DE REPRESENTATION



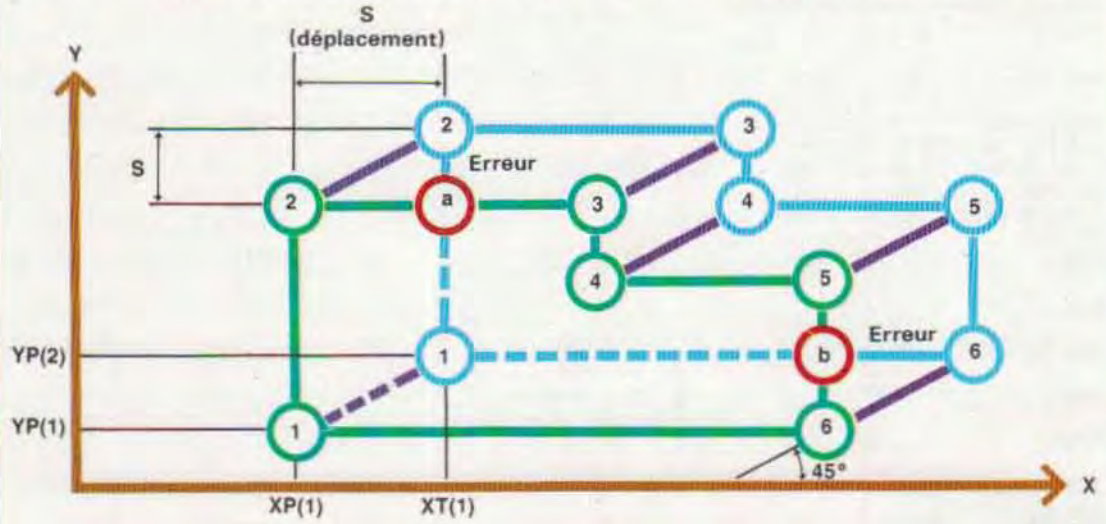
GRAPHIQUE D'UNE FONCTION A DEUX VARIABLES

```
10 REM ******************************
20 REM *                            *
30 REM *                            *
40 REM * GRAPHIQUES A 3 DIMENSIONS *
50 REM *                            *
60 REM *                            *
70 REM ******************************
80 :

90 :

100 REM ------------------
110 REM LIMITES DE L'ECRAN
120 REM ------------------
130 XN=0
      :XM=279
      :YN=0
      :YM=191
140 REM ------------------
150 REM FACTEURS D'ECHELLE
160 REM ------------------
170 FX=5
      :FY=10
180 REM ------------------
190 REM DENSITE DES POINTS
200 REM ------------------
210 PX=.2
      :PY=.2
220 REM ---------------------
230 REM COORDONNEES DU CENTRE
240 REM ---------------------
250 X0=129:
      Y0=61
270 REM ------------------
280 REM ENTREE DES DONNEES
290 REM ------------------
300 TEXT
      :HOME
310 VTAB 12
320 INPUT "Intervalle axe X ";X1,X2

330 VTAB 14
340 INPUT "Intervalle axe Y ";Y1,Y2
```

```
350 A=ATN(Y0/X0)
      :REM ANGLE
360 HGR2
      :HCOLOR=3
370 GOSUB 2000
380 :

390 :

400 REM ************************
410 REM **                    **
420 REM ** BOUCLE PRINCIPALE **
430 REM **                    **
440 REM ************************
450 :

460 :

470 FLAG=1
480 FOR Y=Y1 TO Y2 STEP PY
490 FOR X=X1 TO X2 STEP PX
500 REM ----------------
510 REM CALCUL DE XP, YP
520 REM ----------------
530 :

540 :

550 Z=(SIN(X)/X)*(5*SIN(Y))

560 :

570 :

580 DX=X-Y*COS(A)
590 DY=Z-Y*SIN(A)
600 XV=X0+DX*FX
610 YV=Y0-DY*FY
620 XP=INT(XV)
630 YP=INT(YV)
640 REM ------------------
650 REM TRACE DU GRAPHIQUE
660 REM ------------------
670 IF FLAG THEN FLAG=0
      :GOTO 690
680 HPLOT XA,YA TO XP,YP
690 XA=XP:
      YA=YP
700 NEXT X
710 FLAG=1
720 NEXT Y
730 :

740 :

750 REM **********************
760 REM ** FIN DE LA BOUCLE**
770 REM **********************
780 :

790 :

800 GET D$
810 TEXT
     :HOME
820 END
2000 REM ---------------
2010 REM AXES CARTESIENS
2020 REM ---------------
2030 HPLOT XN,YN TO XM,YN TO XM,YM
      TO XN,YM TO XN,YN
```

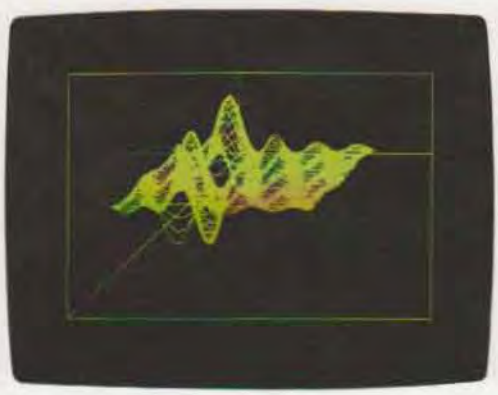
```
2040 HPLOT X0,Y0 TO X0,YN
2050 HPLOT XN,Y0 TO XM,Y0
2060 HPLOT X0,Y0 TO XN,YM
2065 HCOLOR=0
2070 FOR I=X0-3 TO 0 STEP -3
2080 HPLOT I,Y0
2090 NEXT
2100 HCOLOR=3
2110 RETURN
```

METHODE EMPLOYEE POUR LA REPRESENTATION 3D



Les segments affectés de la mention Erreur sont tracés parce que le programme ne dispose pas de toutes les fonctions nécessaires.

- Contour de référence (c'est le premier à être tracé)
- Contour déplacé
- Lignes de liaison des deux contours (épaisseur)
- Sommets de la base
- Sommets de la base déplacés
- Points d'intersection



Résultat final de la visualisation d'une fonction à 2 variables.

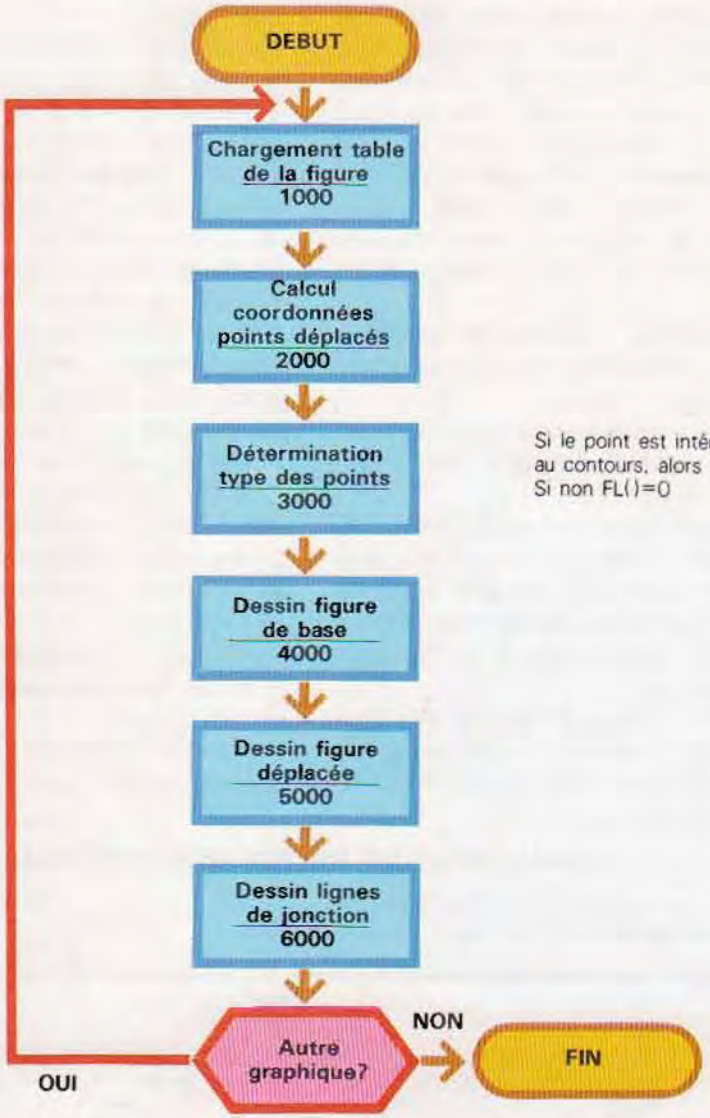
nous présentons (voir organigramme page ci-contre) est fondée sur la seule analyse de la figure de base, et elle peut, dans certains cas, aboutir à des résultats incorrects.

C'est pourquoi cette méthode ne doit être considérée que comme un exemple d'approche de ce problème et non pas comme une méthode universelle.

Les différentes fonctions employées sont :

- Acquisition des coordonnées des sommets de la figure de base (sous-programme 1000) dont la translation engendre la représentation 3D.
- Calcul des coordonnées des points déplacés (sous-programme 2000). La technique consiste tout simplement à incrémenter toutes les coordonnées de la quantité S.

ORGANIGRAMME DU PROGRAMME DE SIMULATION 3D



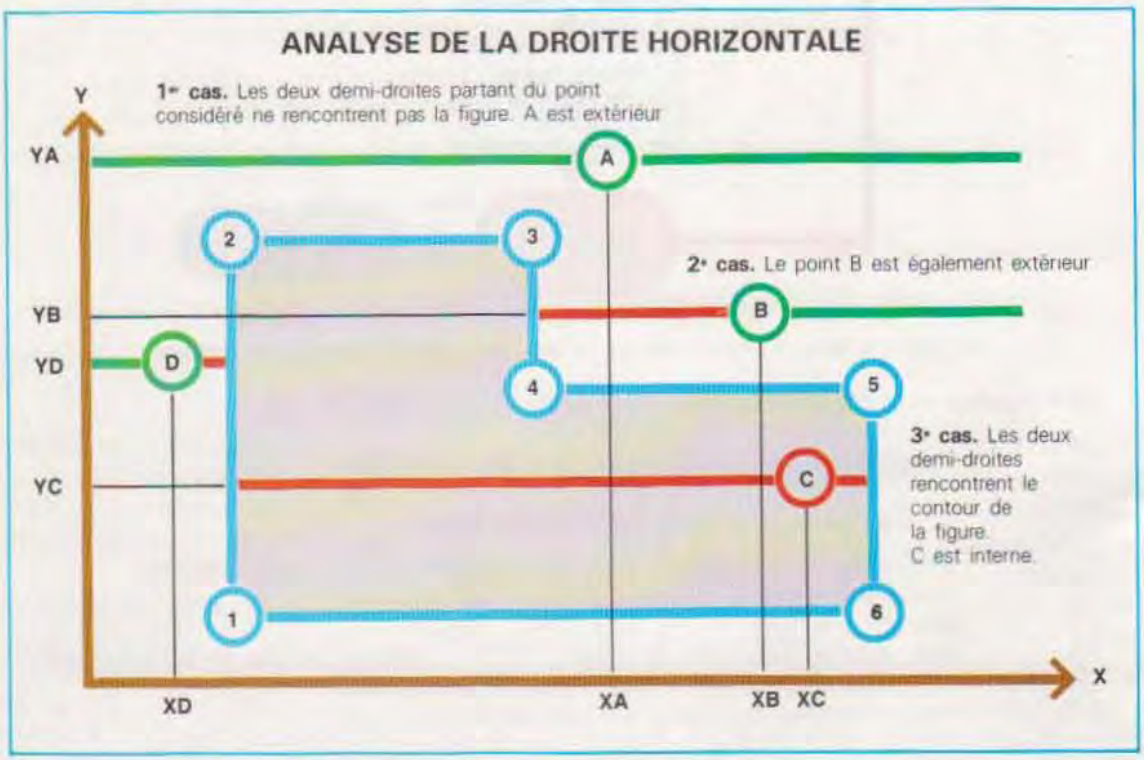
Par rapport au listing, les organigrammes de détail qui suivent ont, parfois, été simplifiés.

Principales variables utilisées

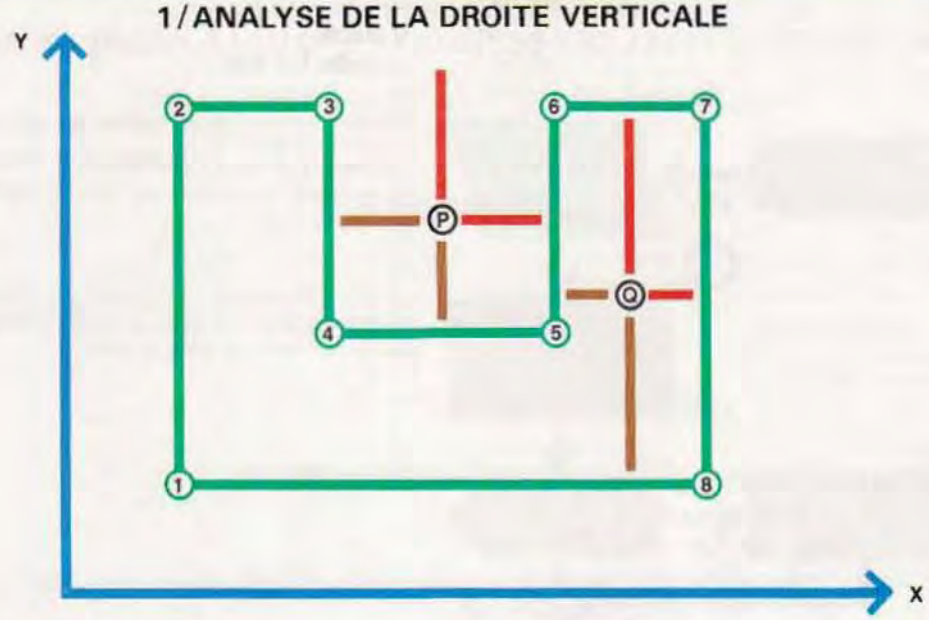
XP(21), YP(21)	Coordonnées de chaque point de la figure de base
XT(21), YT(21)	Coordonnées de la figure après translation
FL(20)	Indicateur de condition (1 si le point est interne, 0 s'il est externe)
N1	Nombre des points entrés
FS	Facteur d'échelle
OY	Indicateur d'orientation de l'axe Y (OY=-1 indique l'origine en haut à gauche)
S(2, 20)	Indicateur gauche/droite dans l'analyse horizontale
T(2, 20)	Indicateur haut/bas dans l'analyse verticale

■ Détermination du type de chaque point de la figure translatée. Une fois calculées les coordonnées des sommets, il faut vérifier si le segment unissant deux à deux les points correspondants est visible. Dans la figure de la page 1642, par exemple, le côté translaté 1 est uni au côté 2 par une ligne qui n'est pas visible ; à l'inverse, le segment qui unit les points translatés 2 et 3 doit l'être.
Le sous-programme 3000 analyse chacun des couples de sommets et détermine si le côté les unissant doit être affiché ou non.

La méthode consiste à déterminer quand un point du contour déplacé tombe à l'intérieur du contour de la figure de base.
Par exemple, le point translaté 1 doit être non visible puisqu'il se trouve à l'intérieur de la figure de base qui le recouvre.
Cette technique a ses propres limites. Si l'épaisseur (donc le déplacement S) est trop important, les coordonnées du point translaté se trouveront hors du contour de base, le point translaté sera lui aussi visible, et les résultats obtenus erronés.
Nous présentons ci-dessous l'algorithme utilisé pour déterminer si un point quelconque est interne ou non à un contour donné. Cette méthode, qui peut également être exploitée à d'autres fins, est fondée sur l'hypothèse selon laquelle les sommets ont été entrés de façon séquentielle.
Dans notre représentation, les numéros de 1 à 6 désignent les sommets du contour de base, et les lettres A à D les points dont on désire connaître la position par rapport à la figure. Un point est interne à la figure quand chacune des deux demi-droites horizontales, tracées à partir de ce point, coïncide avec le contour.
La figure 1 (page ci-contre) représente un type de contour particulier avec lequel l'analyse de la droite horizontale se révèle insuffisante. En effet, partant du point P, les deux demi-droites horizontales couperaient bien la figure de base, alors même que P est externe. L'analyse est également être effectuée le long de l'axe Y.
Une fois que le type de chaque point a été déterminé (interne/externe), le contour translaté peut être tracé (sous-programme 5000), et les sommets unis par les segments, créant ainsi une illusion d'épaisseur (sous-programme 6000). Le sous-programme 5000 contient un contrôle qui, dans certains cas, produit des résultats faux.
En effet, comme nous l'avons vu, il suffit, pour éliminer un côté, que l'une de ses deux extré-

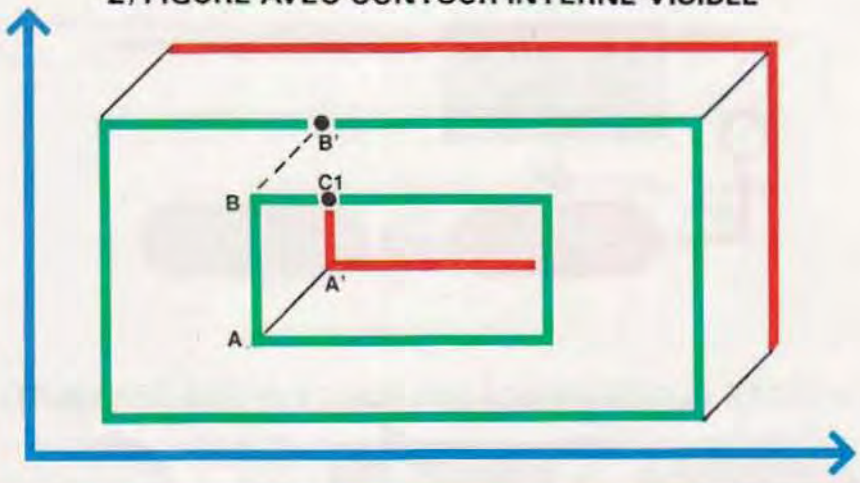


1/ANALYSE DE LA DROITE VERTICALE



Il s'agit d'un cas anormal qui rend nécessaire l'analyse selon l'axe Y. Avec uniquement l'analyse précédente, le point P serait vu intérieur à la figure (1er cas). Par contre, le second contrôle, le long de l'axe Y, indique qu'en réalité, il est extérieur. A l'inverse, le point Q relève du 1er cas d'après les deux analyses.

2/FIGURE AVEC CONTOUR INTERNE VISIBLE

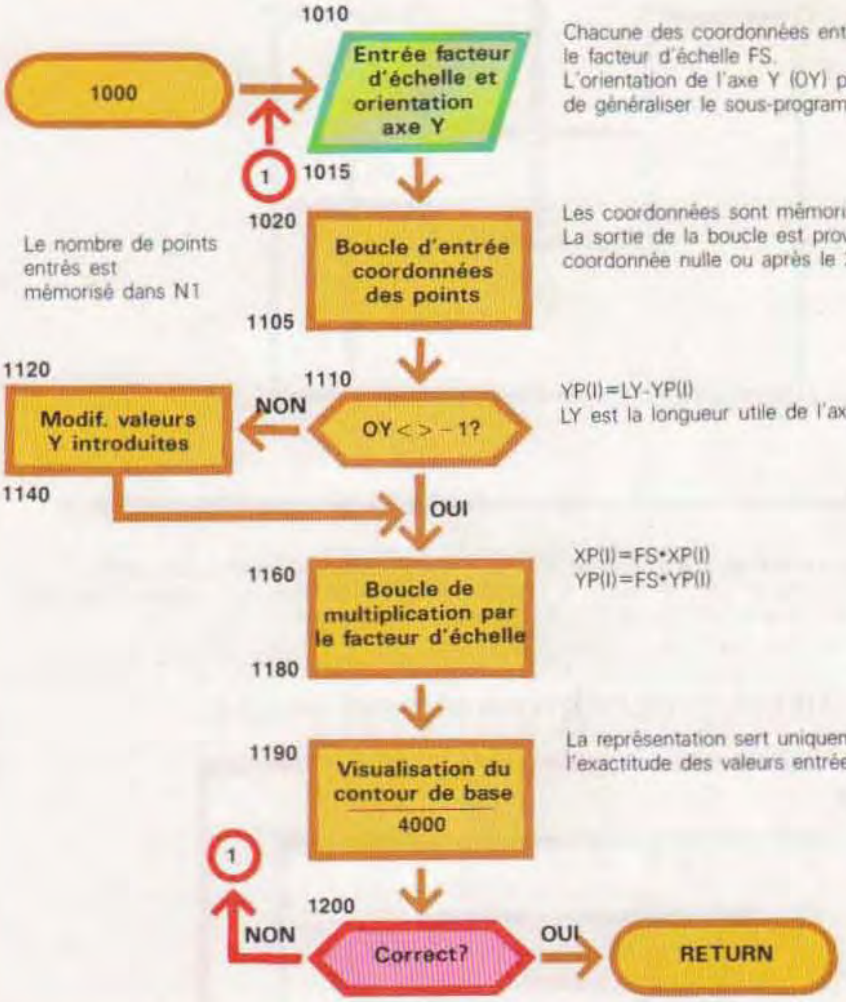


mités soit interne. D'après cette méthode, le côté unissant les sommets translatés 1 et 2 ne serait pas visible.

Il existe pourtant des cas où la logique décrite entraînerait l'omission de parties entières du dessin. Par exemple, dans celui de la figure 1, la partie interne serait totalement occultée. Il a donc fallu prévoir la possibilité de déterminer quand il existe des portions de côté visibles (sous-programme 5500). Là encore, certaines zones particulières du dessin peuvent toutefois être fausées (segments 2-a et 6-b de la figure, page 1642). L'erreur proviendrait de la sélection des points, dont la position n'est analysée que par rapport à la figure de base.

Le sous-programme 5500 illustre une méthode souvent employée pour identifier les points encore à l'intérieur de la fenêtre utilisée. Les pages 1646 à 1657 présentent les organigrammes et le listing de l'ensemble de la procédure.

ENTREE DE LA TABLE DES DEPLACEMENTS DE LA FIGURE DE BASE



Chacune des coordonnées entrées sera multipliée par le facteur d'échelle FS.
L'orientation de l'axe Y (OY) permet de généraliser le sous-programme

Le nombre de points entrés est mémorisé dans N1

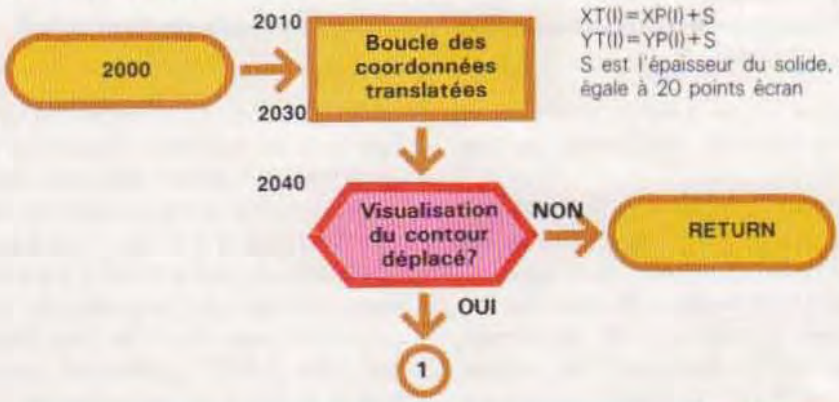
Les coordonnées sont mémorisées dans XP(I) et YP(I).
La sortie de la boucle est provoquée par l'entrée d'une coordonnée nulle ou après le 20e point

YP(I)=LY-YP(I)
LY est la longueur utile de l'axe Y

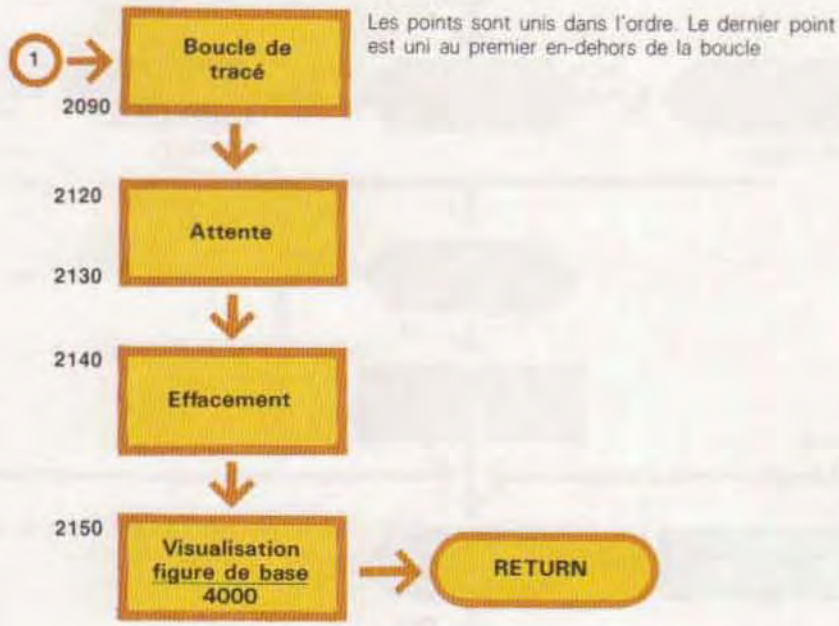
XP(I)=FS*XP(I)
YP(I)=FS*YP(I)

La représentation sert uniquement à contrôler l'exactitude des valeurs entrées

CALCUL DES COORDONNEES DES POINTS APRES TRANSLATION

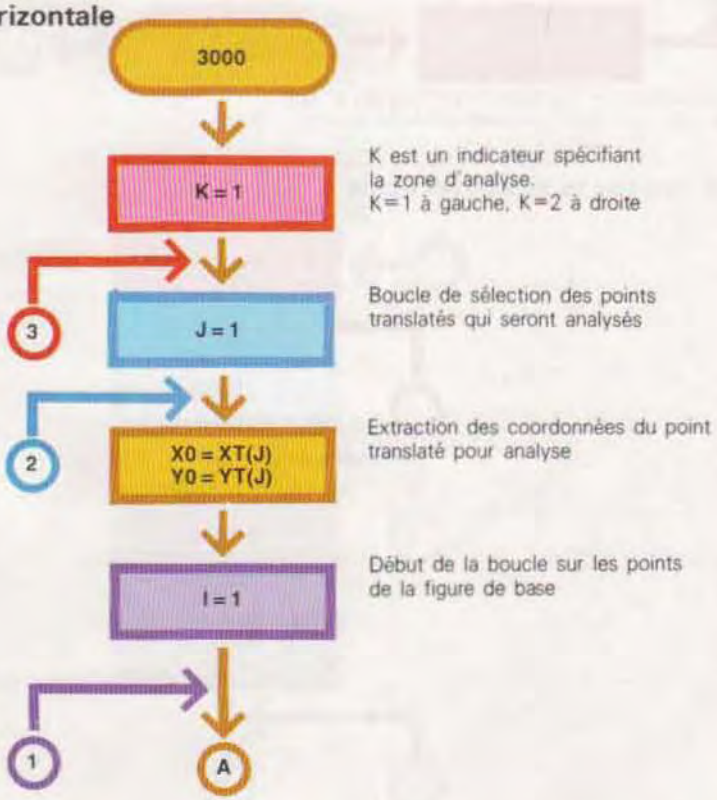


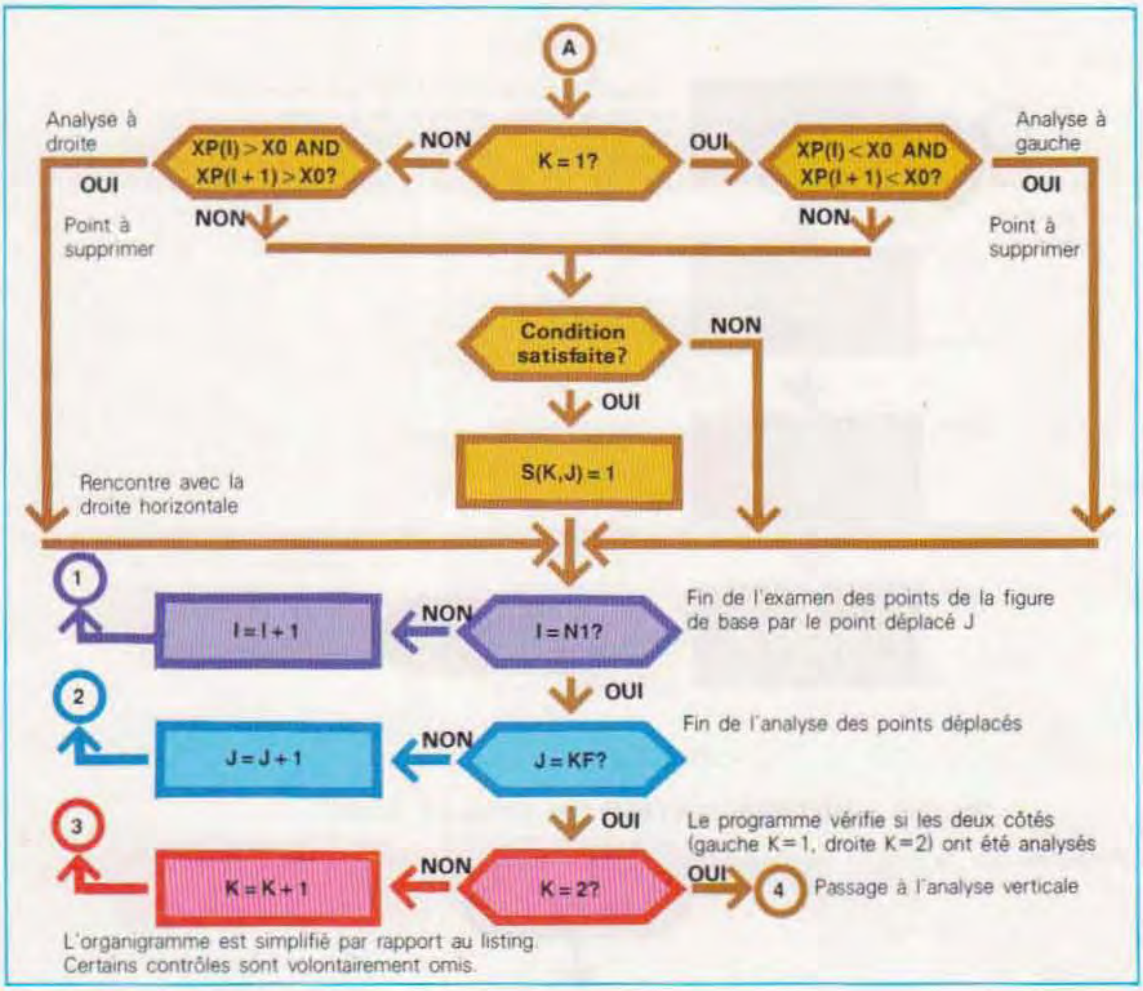
XT(I)=XP(I)+S
YT(I)=YP(I)+S
S est l'épaisseur du solide, supposée égale à 20 points écran



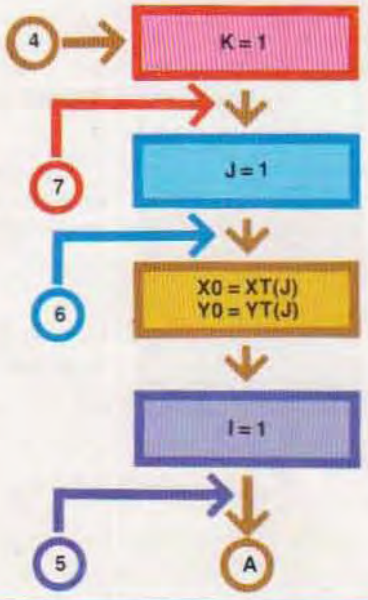
DETERMINATION DU TYPE DE POINT

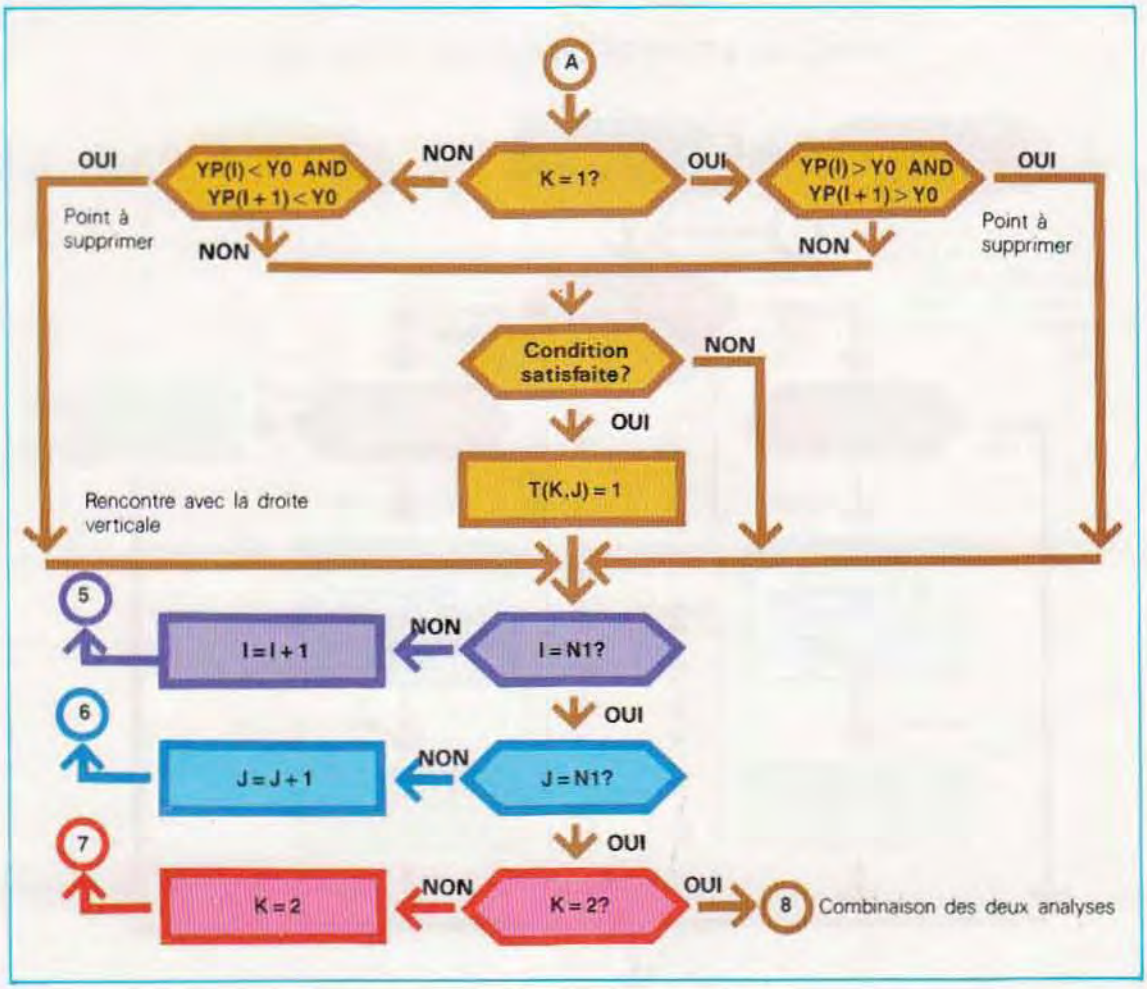
Analyse sur la droite horizontale

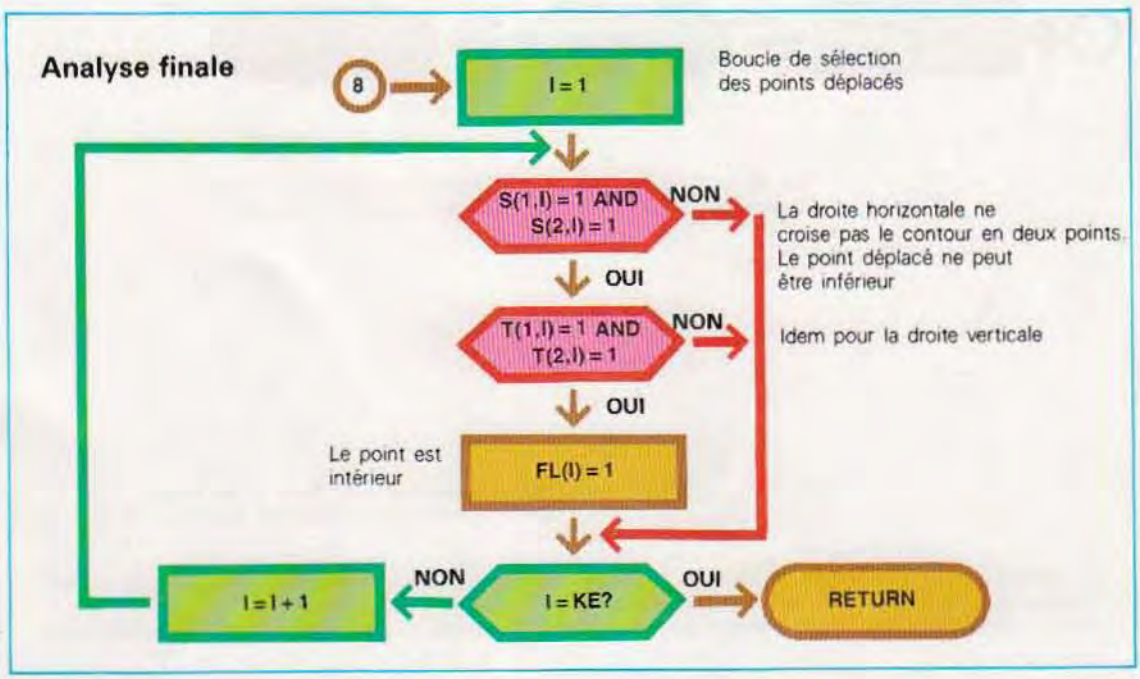




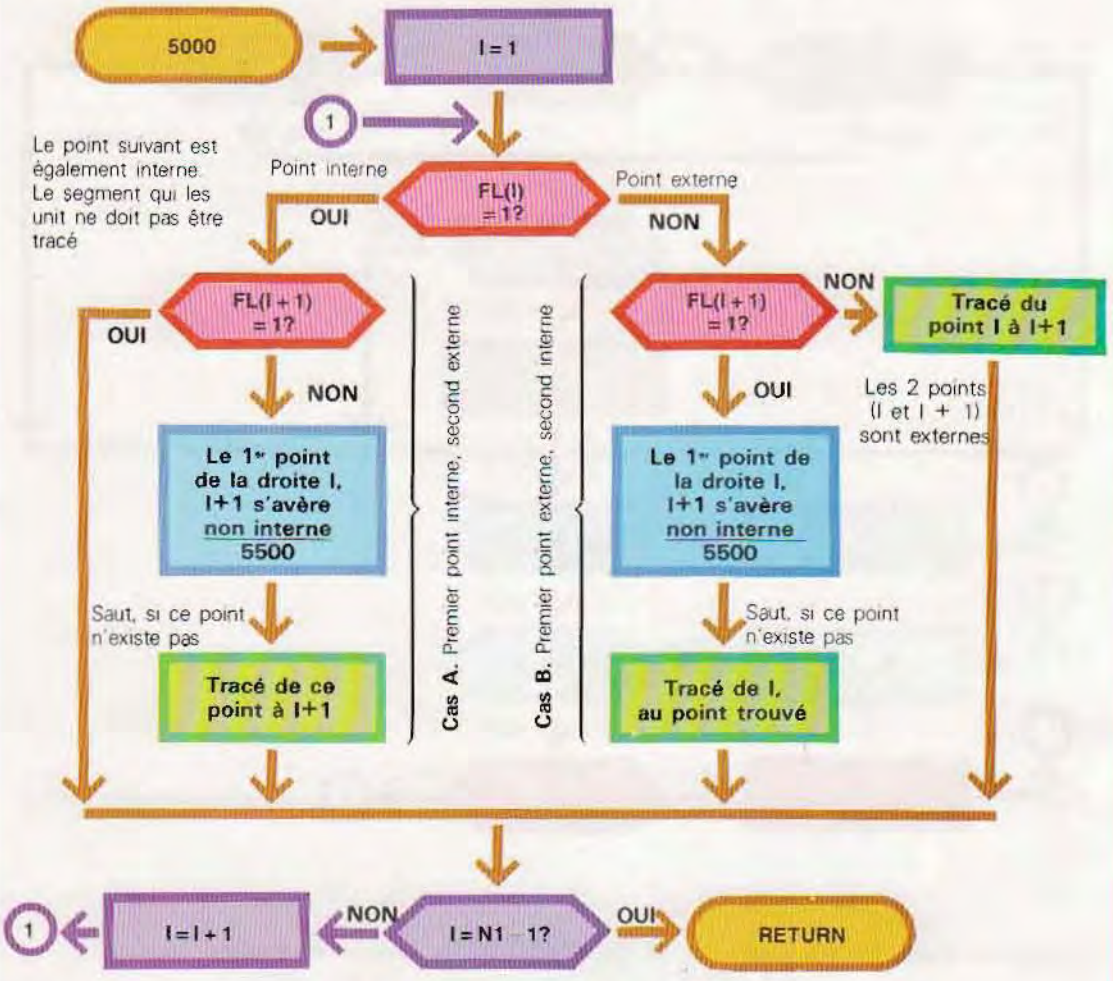
Analyse sur la droite verticale

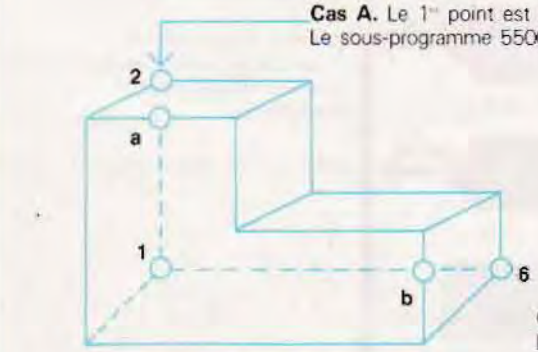



A
K = 1?
NON
OUI
YP(I) < Y0 AND YP(I + 1) < Y0
OUI
Point à supprimer
NON
YP(I) > Y0 AND YP(I + 1) > Y0
OUI
Point à supprimer
NON
Condition satisfaite?
NON
OUI
T(K,J) = 1
Rencontre avec la droite verticale
5
I = I + 1
NON
I = N1?
OUI
6
J = J + 1
NON
J = N1?
OUI
7
K = 2
NON
K = 2?
OUI
8
Combinaison des deux analyses


Analyse finale
8
I = 1
Boucle de sélection des points déplacés
S(1,I) = 1 AND S(2,I) = 1
NON
La droite horizontale ne croise pas le contour en deux points. Le point déplacé ne peut être inférieur
OUI
T(1,I) = 1 AND T(2,I) = 1
NON
Idem pour la droite verticale
OUI
Le point est intérieur
FL(I) = 1
I = I + 1
NON
I = KE?
OUI
RETURN

PRESENTATION DE LA FIGURE DEPLACEE



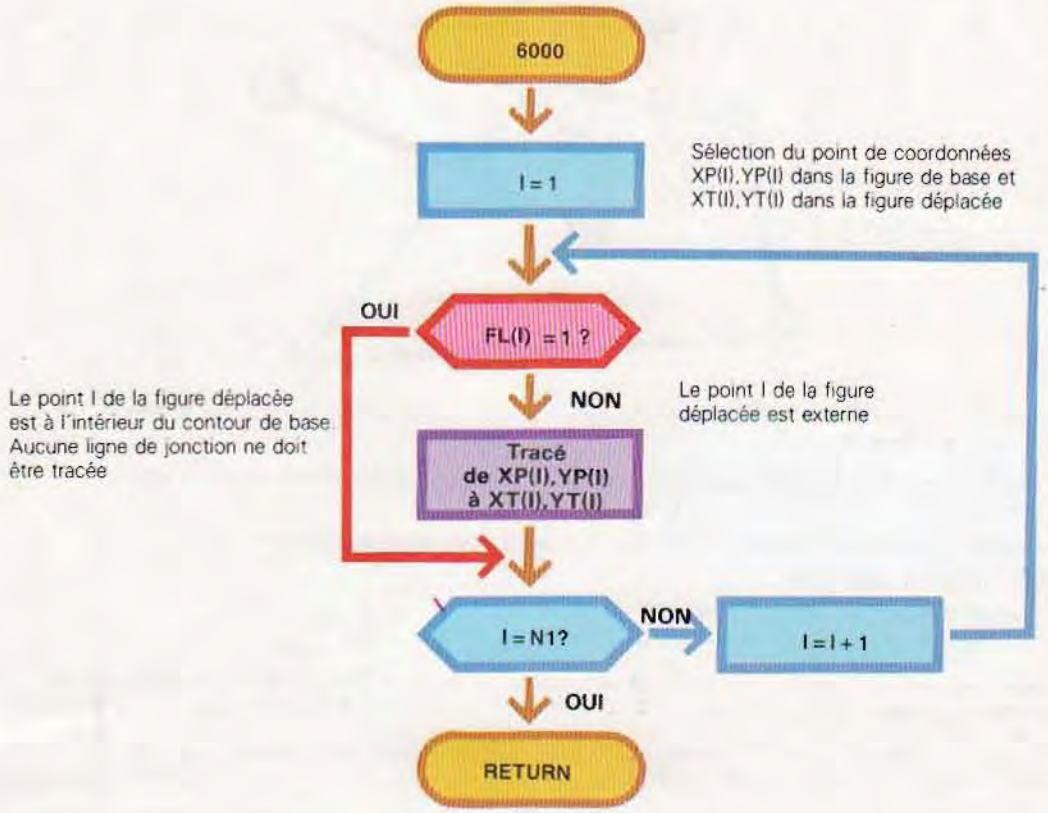


Cas A. Le 1er point est interne alors que le 2e point est externe. Le sous-programme 5500 détermine les coordonnées du point a

Cas B. Le point 6 est externe, le 1er est interne. Le sous-programme 5500 détermine la position du point b

La méthode présentée analyse la position des points déplacés uniquement par rapport à la figure de base et trace quelques segments erronés, comme les traits a,2 et 6,b. Pour avoir une vision complète, il faut aussi analyser chaque point déplacé par rapport aux surfaces générées par la translation, c'est-à-dire en fonction de l'épaisseur

LIGNES DE JONCTION



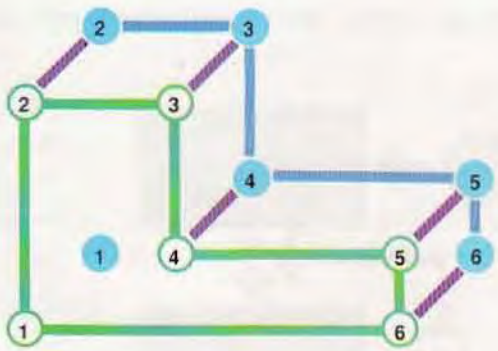


Figure de base

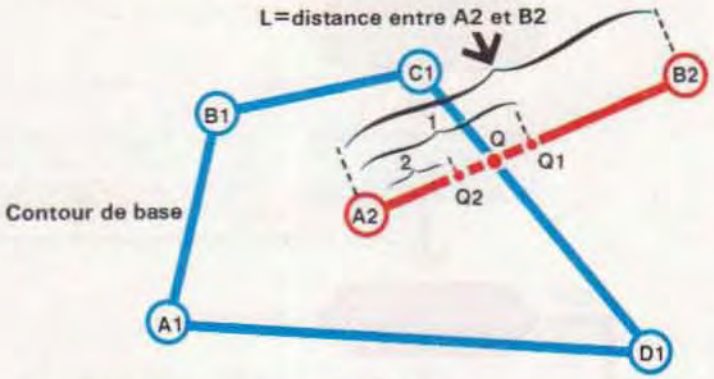
Points déplacés XT(I),YT(I)

Contour de la figure déplacée, présenté par le sous-programme 5000 (après suppression du sous-programme 5500)

Segments de jonction tracés par ce sous-programme

Les deux points 1 ne sont pas unis parce que celui de la figure déplacée se trouve à l'intérieur du contour de base

ALGORITHME UTILISE PAR LE SOUS-PROGRAMME 5500



A1, B1, C1 et D1 forment le contour de base. A2 et B2 sont deux points quelconques du contour déplacé, l'un interne et l'autre externe.

Il faut déterminer les coordonnées du point Q, appartenant au segment A2,B2, à partir duquel on sort du contour de base.

Le segment L, qui unit les deux points A2 et B2, est divisé en deux, ce qui définit le point Q1.

Deux cas peuvent se présenter :

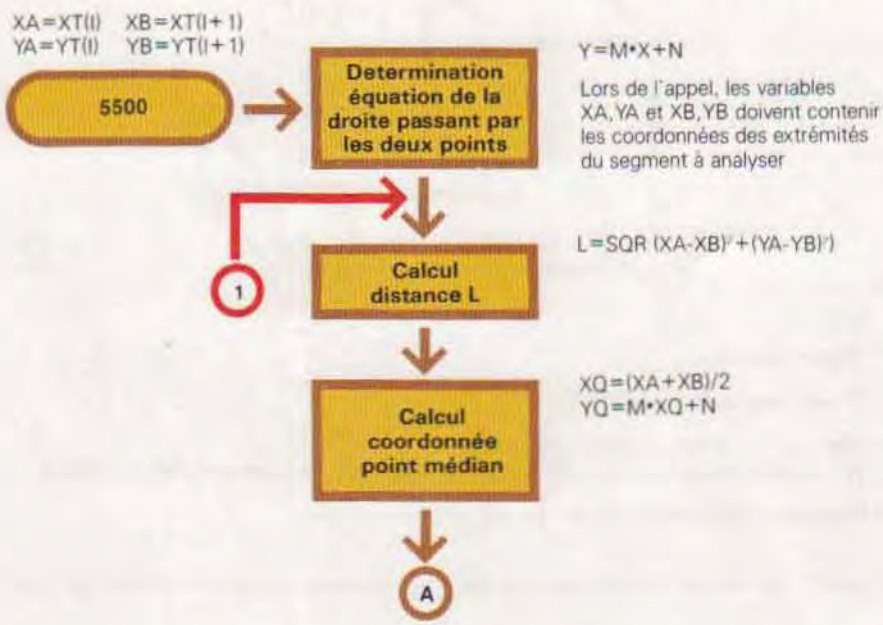
1 - Q1 est externe : l'analyse se poursuit sur la portion A2 Q1

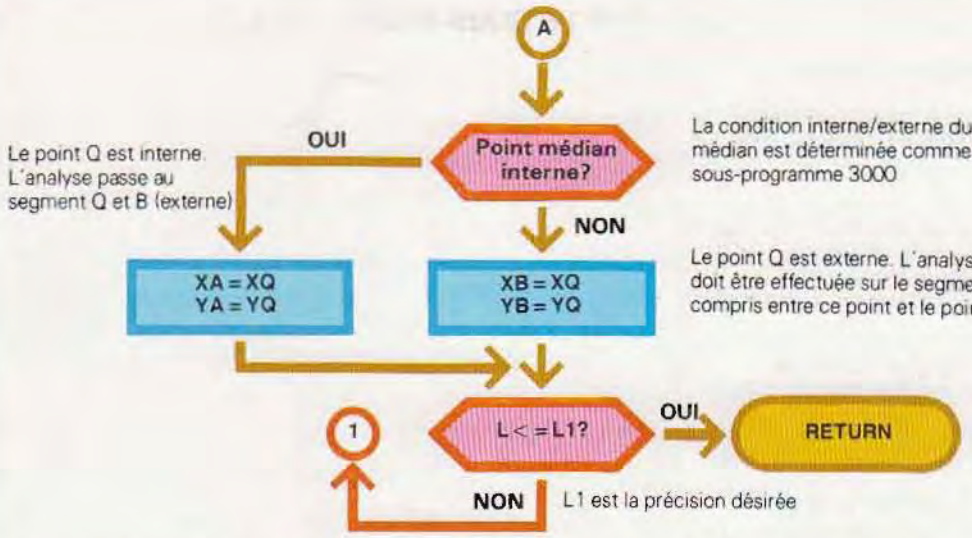
2 - Q1 est interne : l'analyse se poursuit sur la portion Q1 B2

Une fois sélectionné le segment sur lequel l'analyse va se poursuivre, le programme retourne au premier pas en remplaçant les deux extrémités du segment (A2,B2) par celles qui ont été trouvées au cours des pas précédents.

La boucle se répète quand la distance entre deux points Q successifs (Qn et Qn+1) est inférieure à une valeur prédéterminée (précision que l'on veut obtenir).

DETERMINATION DU POINT DE TRANSITION INTERNE/EXTERNE





La méthode illustrée ici donne des résultats approximatifs. En effet, la recherche est interrompue quand la précision requise (L1) est obtenue, alors qu'elle peut ne pas être suffisante du point de vue graphique. Un meilleur résultat peut être obtenu avec un autre système basé sur la lecture de la mémoire vidéo.
Il s'agit, en bref, de se déplacer le long du segment A2 B2 en déterminant l'état vidéo de chaque point. Les coordonnées de Q sont celles pour lesquelles le pixel est actif. Cette méthode donne de meilleurs résultats du point de vue graphique, mais elle demande plus de temps. En effet, le contenu de chacune des adresses correspondant aux points du segment A2 B2 doit être analysé. Il est possible de l'optimiser en la combinant avec la précédente. Ainsi, la première méthode détermine dans quel cadre se trouve Q, et la deuxième analyse ensuite ce cadre en vérifiant le contenu de la mémoire vidéo.

Le sous-programme 5500 peut être remplacé par : 5500 XQ=XB:YQ=YB:RETURN

De cette façon, le programme n'est que partiellement exécuté ; par contre, l'erreur est évitée.

GRAPHIQUE TRIDIMENSIONNEL AVEC EFFACEMENT DES LIGNES CACHEES

```
1 REM ====================
2 REM,* PROGRAMME  HAUTE *
3 REM * RESOLUTION  POUR *
4 REM * FIGURES EN 3D ET *
5 REM *  EFFACEMENT DES  *
6 REM *  LIGNES CACHEES  *
7 REM ====================
8 :
9 :
50 S=20:L1=5:LY=191
55 DIM YP(20),XP(20),YT(20),XT(20)
   ,FL(20),S(2,20),T(2,20)
60 ONERR GOTO 10000
65 CN=1
100 GOSUB 1000
110 GOSUB 2000
120 GOSUB 3000
130 GOSUB 4000
140 GOSUB 5000
150 GOSUB 6000
200 HOME:VTAB 21:INPUT "Autre
    graphique (O/N) ";RS$
210 IF LEFT$(RS$,1)="O" THEN RUN
```

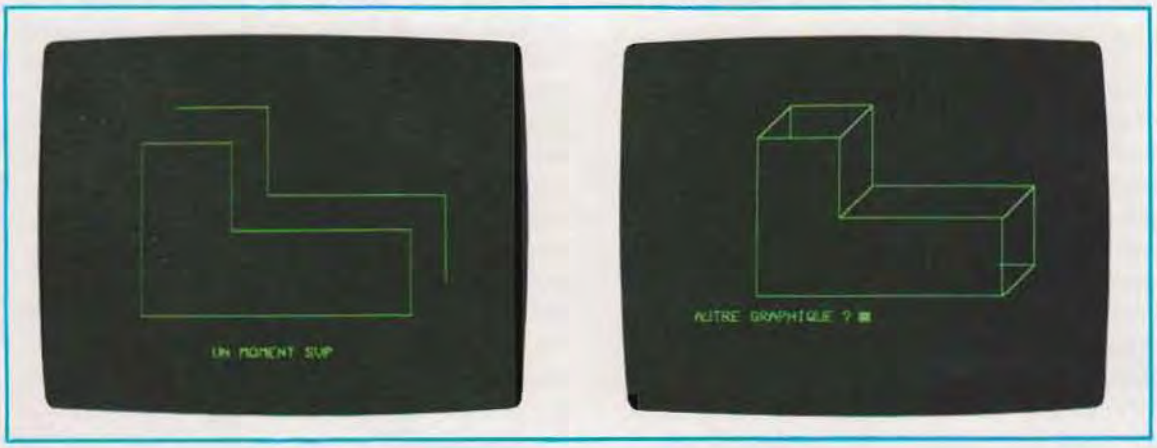
```
220 TEXT:HOME:VATB 22:END
999 :
1000 REM ======================
1001 REM ENTREE FIGURE DE BASE
1002 HGR
1003 :
1005 TEXT
1010 HOME:VTAB 10:INPUT "Entrez
      le facteur d'échelle ";F$
1015 VTAB 10:HTAB 10:INPUT
      "Orientation de l'ordonnée "
     ;OY
1020 I=1
1030 VTAB 10:HTAB 10:PRINT
      "Coordonnées du point "I"   "
1040 VTAB 10:HTAB 31:INPUT "X=
     ";XP(I)
1050 IF XP(I)*FS>278 THEN 1040

1060 VTAB 12:HTAB 31:INPUT "Y=
     ";YP(I)
1070 IF YP(I)*FS>191 THEN 1100

1080 IF YP(I)=0 OR XP(I)=0 THEN 1100

1090 IF I=20 THEN 1100
1095 I=I+1:GOTO 1030
1100 N1=I-1
1105 XP(N1+1)=XP(N1):YP(N1+
     1)=YP(N1)
1110 IF OY<>-1 THEN 1160
1120 FOR I=1 TO N1
1130 YP(I)=LY-YP(I)
1140 NEXT I
1160 FOR I=1 TO N1
1170 XP(I)=FS*XP(I):YP(I)=F
     S*YP(I)
1180 NEXT I
1190 GOSUB 4000
1200 VTAB 22:INPUT "Dessin correct
      (O/N) ";RS$
1210 IF LEFT$(RS$)="O" THEN
     RETURN
1220 GOTO 1000
2000 :
2001 REM ======================
2002 REM * CALCUL DES POINTS *
2003 REM *     DEPLACES      *
2004 REM ======================
2005 :
2010 FOR I=1 TO N1
2020 XT(I)=S+XP(I):YT(I)=YP
     (I)-S
2030 NEXT I
2035 XT(I)=XT(N1)+S:YT(I)=Y
     T(N1)-S
2040 HOME:VTAB 22:INPUT "Je présente
      la figure déplacée (O/N) ";RS$

2050 IF LEFT$(RS$,1)="O" THEN
     2070
2060 RETURN
2070 :
2090 FORI=1 TO N1-1
```

```
2100 HPLOT XT(I),YT(I) TO XT(I+1)
     ,YT(I+1)
2110 NEXT I
2120 HPLOT XT(N1-1),YT(N1-1) TO
      XT(N1),YT(N1)
2130 VTAB 22:PRINT "Enfoncez une
      touche pour effacer";:GET Q$

2140 HGR
2150 GOSUB 4000
2160 RETURN
2999 :
3000 REM --------------------
3001 REM * DETERMINATION DU *
3002 REM *  TYPE DE  POINT  *
3003 REM --------------------
3004 :
3005 KE=N1:HOME:VTAB 22:HTAB 10
     :PRINT "Un moment, svp"
3006 REM ----------------------
3007 REM :   ANALYSE  DE LA   :
3008 REM : DROITE HORIZONTALE :
3009 REM ----------------------
3010 FOR K=1 TO 2
3020 FOR J=1 TO KE
3025 IF KE=1 THEN 3040
3030 X0=XT(J):Y0=YT(J)
3040
      IF K=1 THEN 3070
3060 IF XP(I)>=X0 AND XP(I+1)<=X0
      THEN 3100
3065 GOTO 3080
3070 IF (YP(I)<=Y0 AND YP(I+1)>=Y0)
      OR (YP(I)>=Y0 AND YP(I+1)<=T0)
      THEN S(K,J)=1


3100 NEXT I,J,K
3105 :
3106 REM --------------------
3107 REM :  ANALYSE  DE LA  :
3108 REM : DROITE VERTICALE :
3109 REM --------------------
3110 FOR K=1 TO 2
3120 FOR J=1 TO KE
3125 IF KE=1 THEN 3140
3130 X0=XT(J):Y0=YT(J)
3140 FOR I=1 TO N1
3150 IF K=1 THEN 3170
3160 IF YP(I)>=Y0 AND YP(I+1)>=0
      THEN 3200
3165 GOTO 3180
3170 IF YP(I)<=Y0 AND YP(I+1)<=Y0
      THEN 3200
3180 IF (XP(I)<=X0 AND XP(I+1)>=X0)
      OR (XP(I)>=X0 AND XP(I+1)<=X0)
      THEN T(K,J)=1

3200 NEXT I,J,K
3205 :
3206 REM ------------------
3207 REM : ANALYSE FINALE :
3208 REM ------------------
3210 FOR I=1 TO KE
3220 IF S(1,I)=1 AND S(2,I)=1 THEN
      3240
```

```
3230 GOTO 3250
3240 IF T(1,I)=1 AND T(2,I)=1 THEN
      FL(I)=1
3250 NEXT I
3260 RETURN
3999 :
4000 REM ======================
4001 REM * REPRESENTATION DE *
4002 REM * LA FIGURE DE BASE *
4003 REM ======================
4004 :
4005 HGR:HCOLOR=3
4010 FOR I=1 TO N1-1
4020 HPLOT XP(I),YP(I) TO XP(I+1)
     ,YP(I+1)
4030 NEXT I
4035 GOTO 4060
4040 HPLOT XP(N1-1),YP(N1-1) TO
      XP(N1),YP(N1)
4060 RETURN
4999 :
5000 REM ======================
5001 REM * REPRESENTATION  DE *
5002 REM * LA FIGURE DEPLACEE *
5003 REM ======================
5004 :
5010 FOR P=1 TO N1-1
5020 IF FL(P)=CN THEN 5200
5030 IF FL(P+1)=CN THEN 5050

5040 HPLOT XT(P),YT(P) TO XT(P+1)
     ,YT(P+1)
5045 GOTO 5300
5050 XB=XT(P):XA=XT(P+1):YB=YT(P)
     :YA=YT(P+1)
5055 GOSUB 5500
5060 HPLOT XT(P),YT(P) TO X0,Y0
5070 GOTO 5300
5100 :
5200 IF FL(P+1)=CN THEN 5300

5210 XB=XT(P):XA=XT(P+1)
5211 YA=YT(P):YB=YT(P+1)
5215 GOSUB 5500
5220 HPLOT XQ,YQ TO XT(P+1),YT
     (P+1)
5300 NEXT P
5310 RETURN
5499 :
5500 REM ========================
5501 REM * POINTS DE TRANSITION *
5502 REM *    INTERNE/EXTERNE   *
5503 REM ========================
5504 :
5505 IF XA=XB THEN X=1:GOTO
     5515
5506 REM -------------
5507 REM : Y0=XQ*M+N :
5508 REM -------------
5510 X=XB-XA
5515 M=(YB-YA)/X:N=INT(
     YA-(XA*M))
5520 L=INT(SQR((XA-XB)^2+(YA-YB)^2))
```

```
5525 IF X=1 THEN X0=XA
     :Y0=INT((YA+YB)/2):GOTO 5540
5530 X0=INT((XA+XB)/2):Y0=INT(M*X0+N)
5540 X0=XQ:Y0=YQ
5545 S(1,1)=0:S(2,1)=0:T(1,1)=0
     :T(2,1)=0
5550 CF=FL(1):KE=1:FL(1)=0:GOSUB 3010
5580 IF FL(1)=CN THEN XA=XQ:YA=YQ
     :GOTO 5600
5590 XB=XQ:YB=Y0
5600 FL(1)=CF
5605 IF L<=L1 THEN SW=0:RETURN

5610 GOTO 5520

5999 :
6000 REM ==============
6001 REM * LIGNES DE *
6002 REM * FONCTION  *
6003 REM ==============

6004 :
6010 FOR I=1 TO N1
6020 IF FL(I)=CN THEN 6040
6030 HPLOT XP(I),YP(I) TO XT(I),YT(I)
6040 NEXT I
6050 RETURN

10000 :
10001 REM ==========
10002 REM * ERREUR *
10003 REM ==========

10004 :
10005 TEXT:HOME
10010 ER=PEEK(222)
10020 IF ER=53 THEN VTAB 10:PRINT
      "Coordonnées fausses "
     :GOTO 10050
10030 VTAB 10:PRINT "Données fausses "
10050 VTAB 22:HTAB 30:PRINT ">>";
10060 GET Q$:RUN
```

Exemple de fonctionnement du programme présenté.
A gauche, le programme a analysé les points de la figure de base et a présenté, après translation, ceux qui sont visibles. A droite, le dessin a été complété. Remarquer les deux segments erronés.



L'animation d'images par ordinateur

L'animation d'images graphiques constitue une activité bien particulière dans l'informatique graphique. Les techniques d'animation sont déjà largement utilisées dans les programmes de jeux vidéo. Il existe de très nombreux exemples dans lesquels l'animation d'une figure à l'écran peut simplifier considérablement la résolution d'un problème. C'est le cas quand on désire connaître l'effet d'une force sur une structure solide, ou observer le mouvement d'un organe mécanique.

Les objets animés

Nous avons vu que la visualisation d'une image graphique n'est pas une opération élémentaire. Elle passe par le tracé d'un grand nombre de segments qui serviront à la reconstitution de l'image. Le déplacement de l'objet correspondra à une succession d'effacements et de nouveaux tracés dans une autre position

Un **objet animé** (on dit « **sprite** », lutin, en anglais), est, à l'inverse, une image graphique qui est gérée comme telle par le programme, en choisissant une fois pour toutes sa forme et ses différentes positions à l'écran.

L'objet animé, qui est limité en fonction des dimensions de l'écran vidéo, possède trois caractéristiques principales :

— il est complètement programmable par l'utilisateur ;
— il se déplace dans les quatre directions du plan ;
— il mémorise d'éventuelles collisions avec d'autres objets animés ou d'autres figures présentes à l'écran.

Chacune de ces caractéristiques peut être mise en œuvre à l'aide de sous-programmes spéciaux, assurant la simulation d'objets animés même dans les systèmes ne les prévoyant pas. Les autres machines possèdent, quant à elles, des instructions spéciales de déplacement, de contrôle de collisions, etc. Elles sont donc dotées de certaines implémentations qui ne se trouvent pas dans le Basic Standard.

La gestion des objets animés est généralement commandée par des composants matériels ad hoc. Pour les machines plus banales il faut écrire des programmes qui vont simuler ces composants. Les fonctions exécutables sont toutefois plus limitées et la vitesse moins élevée, tout au moins en Basic interprété.

Pilotage des objets animés par logiciel

L'emploi des objets animés est étroitement lié à la production de dessins animés. Certaines des difficultés, rencontrées dans le pilotage par programme, sont justement dues à la nature de cette application.

Un objet animé peut être créé comme une table de figures et déplacé par variation de l'origine. Cette technique, bien connue, consiste à reproduire la figure à déplacer (l'objet animé) comme une série de déplacements (paramétrables) par rapport à une origine.

Si les coordonnées du point choisi comme référence sont modifiées, la figure se trouve déplacée (il faut naturellement prévoir son effacement).

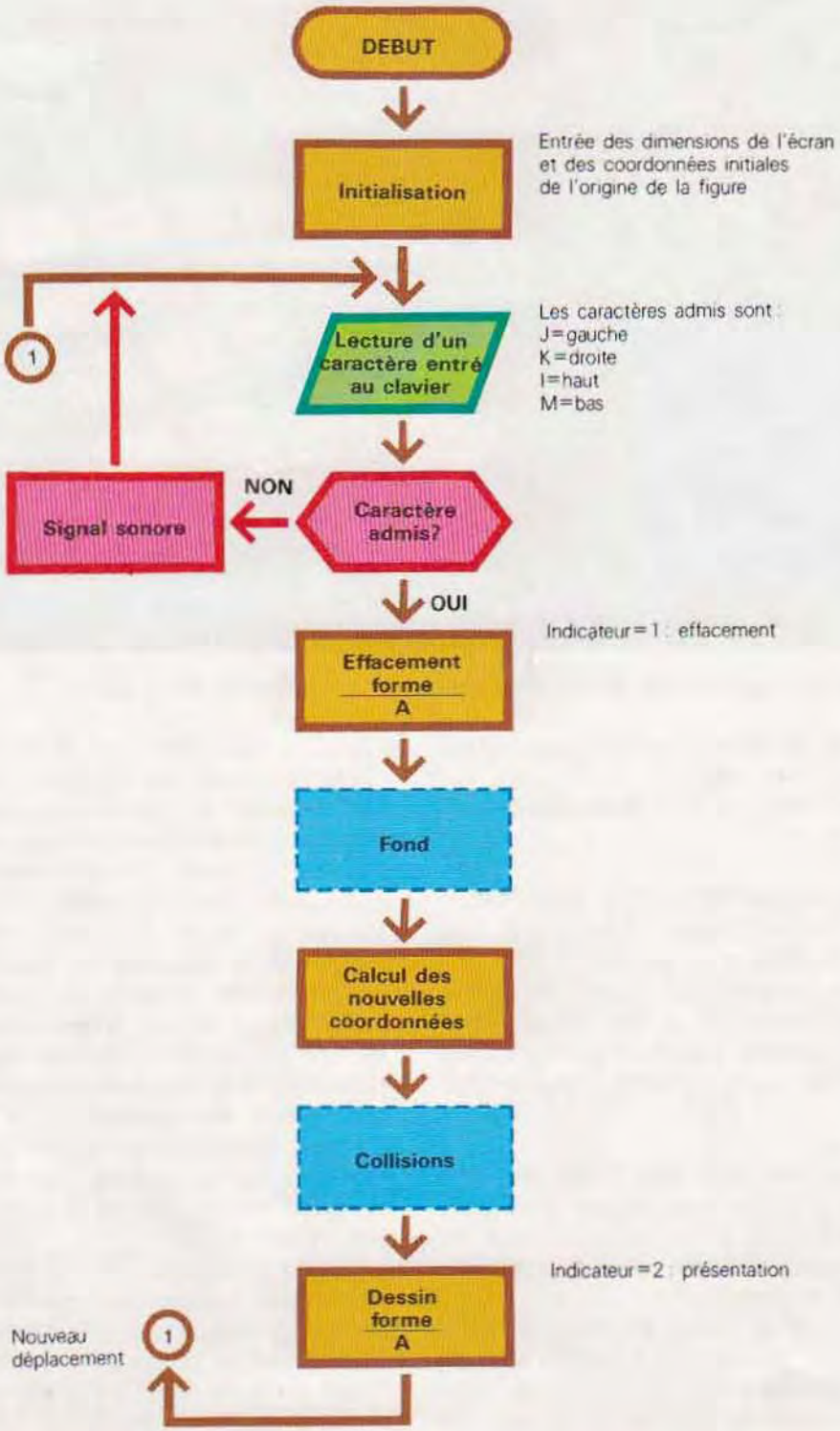
Page ci-contre, se trouve l'organigramme d'un programme qui déplace une figure à l'écran en suivant les commandes entrées au clavier. Les différentes phases de ce programme apparaissent immédiatement. Le sous-programme marqué A mérite, toutefois, une observation : il exécute aussi bien l'effacement que la visualisation. Il est déclenché par l'introduction d'un indicateur spécifiant laquelle de ces deux fonctions est désirée.

L'organigramme comporte également deux blocs (Fond et Collisions) dont l'emploi est spécifique à cette application.

L'animation d'images graphiques. L'animation par ordinateur est une technique très proche de celle employée au cinéma. La scène contient des images fixes (le fond) par rapport auxquelles se déplace l'objet de l'animation.

Au cinéma, la technique consiste à dessiner uniquement le fond, plus autant de tables, sur support transparent, que de mouvements désirés. Sur chacune de ces tables, la figure à déplacer est reproduite dans une position légèrement modifiée par rapport à la précédente. En posant ces différentes tables sur le fond et en photographiant toutes les scènes intermédiaires obtenues, on obtient un fond, qui, à la projection, donnera une réelle impression de mouvement.

DEPLACEMENT D'UNE FORME



P.A. Simon/Grazia Neri-Photostoke

Un moment de la conception assistée par ordinateur des image du film « Tron ».

En réalité le problème n'est pas aussi simple qu'on pourrait le croire.
Noter que la figure qui se déplace est gérée de deux façons :

1 / Lors du déplacement par rapport au fond, la figure reste inchangée ; il faut donc faire varier uniquement sa position.
2 / La figure se déplace par rapport au fond et subit des modifications. Dans ce cas, il faut ajouter au déplacement un nouveau tracé de la figure, partiel ou total, à chaque position.

Le deuxième cas, très facile à résoudre avec une feuille de papier et un crayon, pose quelques difficultés au niveau de l'ordinateur. La première grande difficulté, c'est de rétablir le fond après un mouvement.
Un point donné de l'écran appartient au fond, du moins tant que sa zone n'est pas touchée par la figure déplacée. De fait, il est actif (visible) ou éteint en fonction de ce que l'on doit montrer en fond.

Quand le même point est caché par la figure déplacée, il subit les impératifs de tracé de celle-ci et doit être rétabli à sa position initiale après un déplacement le dégageant.
Par exemple, si un petit bonhomme qui se promène passe devant un arbre, l'ordinateur devra, de pas en pas, remplacer une partie de l'arbre par la silhouette du bonhomme, puis rétablir l'image d'origine de l'arbre dès qu'il sera passé. Si, de plus, la figure est susceptible de subir des modifications, il ne s'agit alors plus seulement de déplacer un graphique, mais aussi de prévoir ses variations.
Dans les machines assurant l'animation, le fond est traité par le système : la figure mobile constitue l'objet animé et peut être déplacée par rapport au fond.
La sensation obtenue est celle de mouvement d'une figure statique. Pour obtenir une animation plus réaliste, il faut alors également modifier la forme de la figure, par exemple en dessinant quelques détails en différents points.
Avec la technique des objets animés, deux possibilités sont offertes : modifier la forme de

l'objet animé avant déplacement ou créer autant d'objets animés qu'il y a de positions à simuler et les présenter les uns à la suite des autres.
Dans ce cas, chacune des figures représente un objet, et l'animation consiste à les activer successivement. Ce système est alors tout à fait analogue à l'animation traditionnelle.
Les mêmes problèmes se posent dans la gestion des objets animés avec des programmes écrits par l'utilisateur. Il est nécessaire d'y prévoir un sous-programme de reconstruction du fond et un autre pour le contrôle des collisions. Avec les machines dédiées à ce type d'applications, cette dernière fonction est automatiquement prise en charge.

Gestion du fond. Il existe différentes méthodes de gestion du fond d'un dessin animé. Leur degré de complexité dépend principalement de deux paramètres :

- La vitesse d'exécution
- L'occupation de la mémoire

Pour donner une impression de déplacement continu, l'animation d'un dessin demande une vitesse d'exécution élevée.
Si le programme n'est pas suffisamment rapide, l'œil réussit à percevoir les différentes phases du déplacement (effacement, reconstruction du fond, nouvelle présentation), ce qui n'est pas agréable ; de plus, il faut effectuer des déplacements relativement importants pour ne pas obtenir un mouvement trop lent, ce qui serait inacceptable dans les jeux vidéo par exemple.
La méthode et les exemples exposés ci-après ne servent qu'à illustrer ces aspects de la gestion du fond. Pour rester suffisamment généraux, ils n'exploitent pas les particularités matérielles de certaines machines, qui pourraient permettre de réduire les problèmes de lenteur d'exécution et d'espace mémoire. Il est en tout cas pratique d'utiliser la version interprétée des programmes dans la phase d'écriture et de contrôle, puis de recourir à la version compilée pour l'application, afin d'obtenir des vitesses d'exécution acceptables.
Une autre solution consisterait à écrire certains sous-programmes en Assembleur.

Le fond peut être considéré comme un dessin faisant appel à l'écran tout entier. La gestion la plus simple, lors d'une animation, serait d'enregistrer le dessin dans une zone mémoire spéciale puis de le rappeler au moment de sa reconstruction.
Voici les fonctions impliquées dans le déplacement d'un objet animé :

1 / Transfert du fond de la mémoire de soutien à l'écran (visualisation)
2 / Présentation de l'objet dans sa position initiale
3 / Présentation du fond (entraînant l'effacement de l'objet)
4 / Présentation de l'objet dans la position suivante

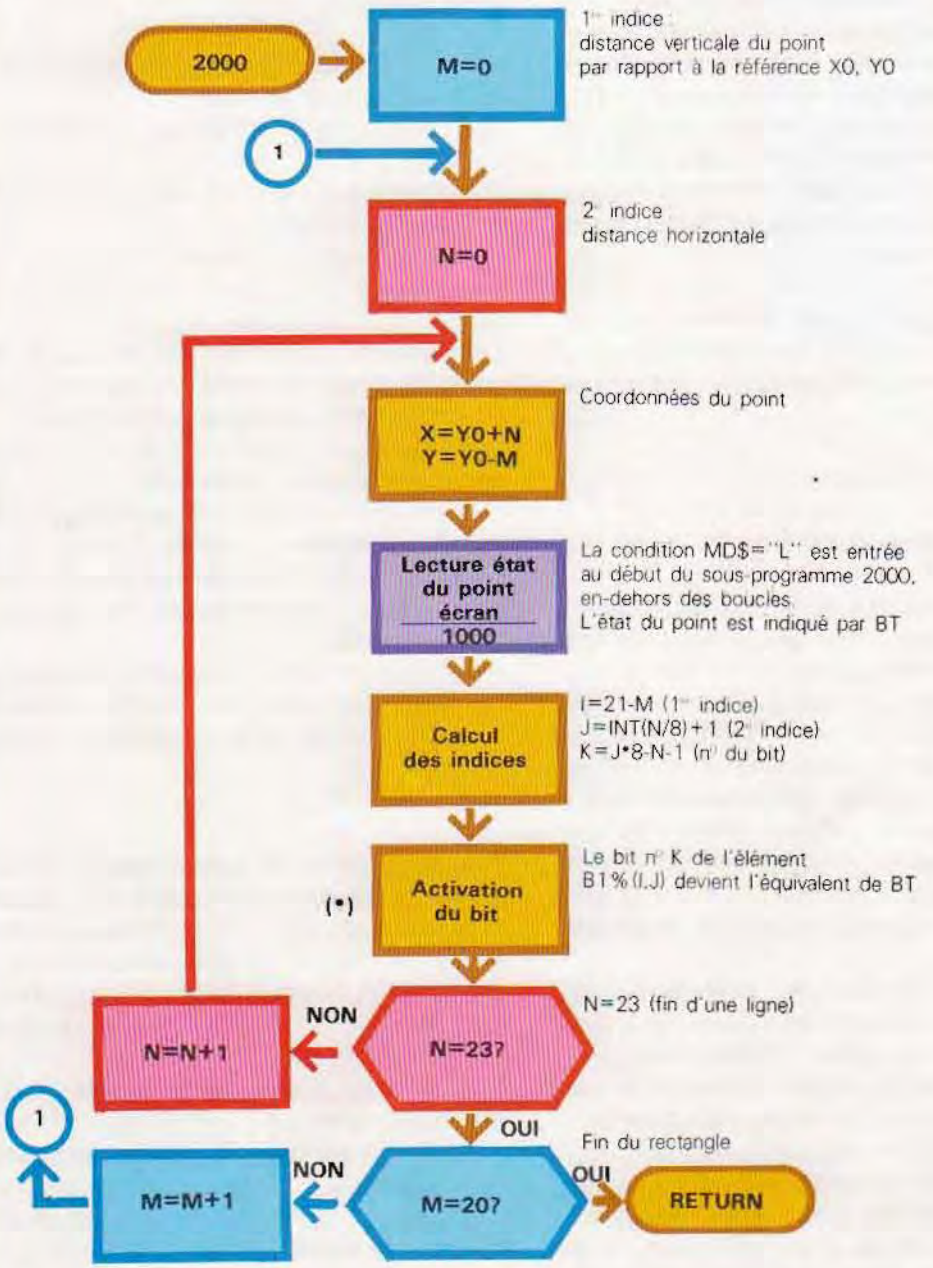
Le mouvement de l'objet est obtenu en activant, de façon itérative et rapide, ces quatre phases, jusqu'à reproduire le déplacement tout entier au moyen d'une série de petits mouvements donnant une impression de continuité.
L'effet visuel de cette méthode n'est toutefois pas des meilleurs. En effet, la faible vitesse d'exécution, liée à la quasi-totalité des interpréteurs Basic, rend perceptible l'enchaînement des opérations.
Il n'y a pas de solution logicielle généralisable, car une bonne animation nécessite obligatoirement l'emploi de certains éléments matériels particuliers.

Visualisation d'un objet animé. Voyons maintenant de près comment on peut résoudre le problème d'une machine. Nous voulons visualiser à l'écran un petit rectangle de 24 (base) sur 21 (hauteur) points écran *, position au point X0, Y0 sur une image graphique constituant le fond.
Chacun des déplacements de l'objet (rectangle) devra être accompagné de la reconstruction de la partie du fond précédemment masquée.
Ce processus comporte les trois phases successives suivantes :

*Ces valeurs ont été choisies pour conserver les dimensions de l'objet, utilisées par les systèmes prévoyant la gestion matérielle des objets animés.

Phase 1. Avant le positionnement de l'objet, la zone de l'écran concernée (rectangle ayant les mêmes dimensions que l'objet) est sauvegardée dans une zone mémoire (tableau bidimensionnel). Cette fonction consiste à calculer, à partir des coordonnées à l'écran de chaque point à sauvegarder, l'adresse correspondante dans la mémoire vidéo.

MEMORISATION DE LA FENETRE CONTENANT L'OBJET



• Ce bloc d'instructions est identique à celui utilisé dans le sous-programme 1000 pour activer un point écran. Cette méthode concerne les machines n'admettant pas d'employer les opérateurs logiques dans la création de masquages à l'intérieur de l'octet. Quand cette possibilité existe (par exemple sous CP/M), il suffit d'utiliser l'opérateur logique approprié.

Le contenu de cette mémoire est lu et transféré dans une mémoire tampon. Dans l'organigramme ci-contre, le tampon utilisé est B1%(21,3) (ce dimensionnement sera expliqué plus loin).

Phase 2. L'objet est introduit à partir du point X0, Y0. Le contenu de la mémoire vidéo correspondante est remplacé par le contenu du tableau S%(21,3) qui représente, sous une forme spéciale, l'objet.

Phase 3. Le déplacement de l'objet est obtenu en rétablissant le fond, c'est-à-dire en transférant B1%(21,3) dans la zone mémoire vidéo occupée par l'objet et en reprenant le cycle après avoir modifié les valeurs de X0,Y0.
Les phases 2 et 3 peuvent être décrites dans le même organigramme. Le sous-programme correspondant, qui est paramétré, présente, à partir des coordonnées X0,Y0, l'objet S%(21,3) ou le fond B1%(21,3).

Le transfert de données, à partir et à destination de la mémoire vidéo, nécessite le calcul de la position de mémoire correspondant aux coordonnées en points écran et du bit qu'elle contient. Sur ce point, nous avons repris le sous-programme 100 déjà présenté en l'adaptant.
Ce sous-programme demande, en entrée, les coordonnées X et Y du point écran et la valeur affectée à MD$ (MD$="L" pour la lecture de l'état allumé/éteint, MD$="S" pour l'écriture). Il fournit, en sortie, BT=1 si le point est allumé et BT=0 s'il est éteint.
Il est illustré, page suivante, par le tableau B1%(21,3). Chacun des éléments se composant de 8 bits, il peut contenir l'état d'autant de points écran. Les 24 points horizontaux sont donc mémorisés dans 3 cellules mémoire seulement. La même logique est utilisée dans certaines machines prévoyant la gestion des objets animés.
L'organigramme de reconstruction du fond ou de présentation de l'objet en fonction de l'indicateur F (F=1 fond, F=2 objet) est en page 1665. Les fonctions exécutées sont tout à fait analogues à celles du sous-programme 2000 avec, pour seule différence, l'utilisation de deux tableaux différents selon la valeur de F (F=1 pour B1%, F=2 pour S%).

Page 1666 enfin, c'est l'organigramme d'un programme dans lequel les sous-programmes 5000 et 6000 servent uniquement à préparer les données (voir listing en pages 1667 à 1669).

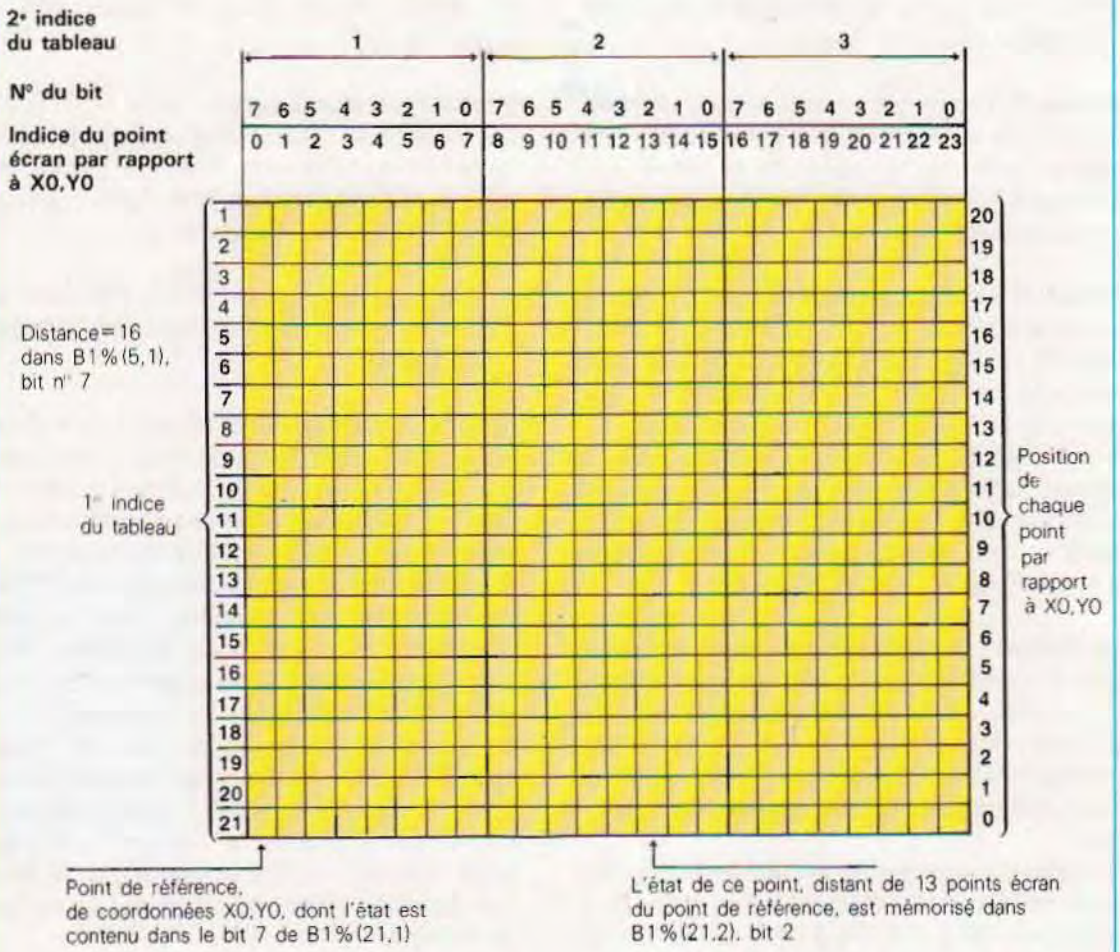
Problèmes d'animation. Dans le programme de la page 1666, l'objet est assimilé à un rectangle qui, converti en image binaire, est mémorisé dans le tableau S%. Cette méthode présente deux inconvénients :

1 / Elle ne permet pas de définir des contours.
2 / Elle ne prévoit pas d'animations à l'intérieur de l'objet.

Le premier oblige à remplir toute la zone disponible pour l'objet, faute de quoi il se créerait une zone toujours éteinte. Celle-ci correspondrait à la différence entre la zone du rectangle effectivement occupée par le dessin (objet) et la totalité de la zone à la disposition de l'objet.
Ce défaut peut être éliminé à l'aide d'un sous-programme de présentation plus évolué. Il ne s'agit plus simplement de présenter le rectangle formant l'objet, mais de construire l'intersection entre ses points et ceux de l'écran. Autrement dit, si un point de l'objet est actif, il doit être visualisé à l'écran (il couvre le fond) ; s'il est inactif (éteint), la mémoire vidéo doit rester intacte, comme couverte par le fond. Cela équivaut à dessiner l'objet sur un rectangle transparent.
Cette fonction peut être obtenue très simplement à l'aide d'un IF lors du transfert en mémoire vidéo de l'image de l'objet. Si le point concerné n'est pas actif, il n'y aura pas de transfert, de manière à garder le fond inchangé (voir schéma en page 1671).
Le deuxième défaut (pas d'animation dans l'objet) est nettement plus difficile à traiter. En fait, même avec des machines dédiées, l'animation reste une fonction difficile. Il s'agit, en effet, de modifier la forme de l'objet au fur et à mesure que le rectangle qui le contient se déplace.
On crée ce mouvement de deux façons :

1 / En mémorisant certaines positions intermédiaires.
2 / En schématisant les déplacements à l'aide de règles mathématiques.

ADRESSAGE DE LA MEMOIRE DE L'OBJET



Le balayage commençant par la dernière rangée (élément de 1e indice=21), les points écran de la rangée 21 peuvent être divisés en trois groupes :

— le premier, (2e indice=1) est contenu dans les bits 0 à 7 de l'élément B1%(21,1)
— le deuxième est contenu dans les bits 0 à 7 de B1%(21,2)
— le troisième est contenu dans B1%(21,3)

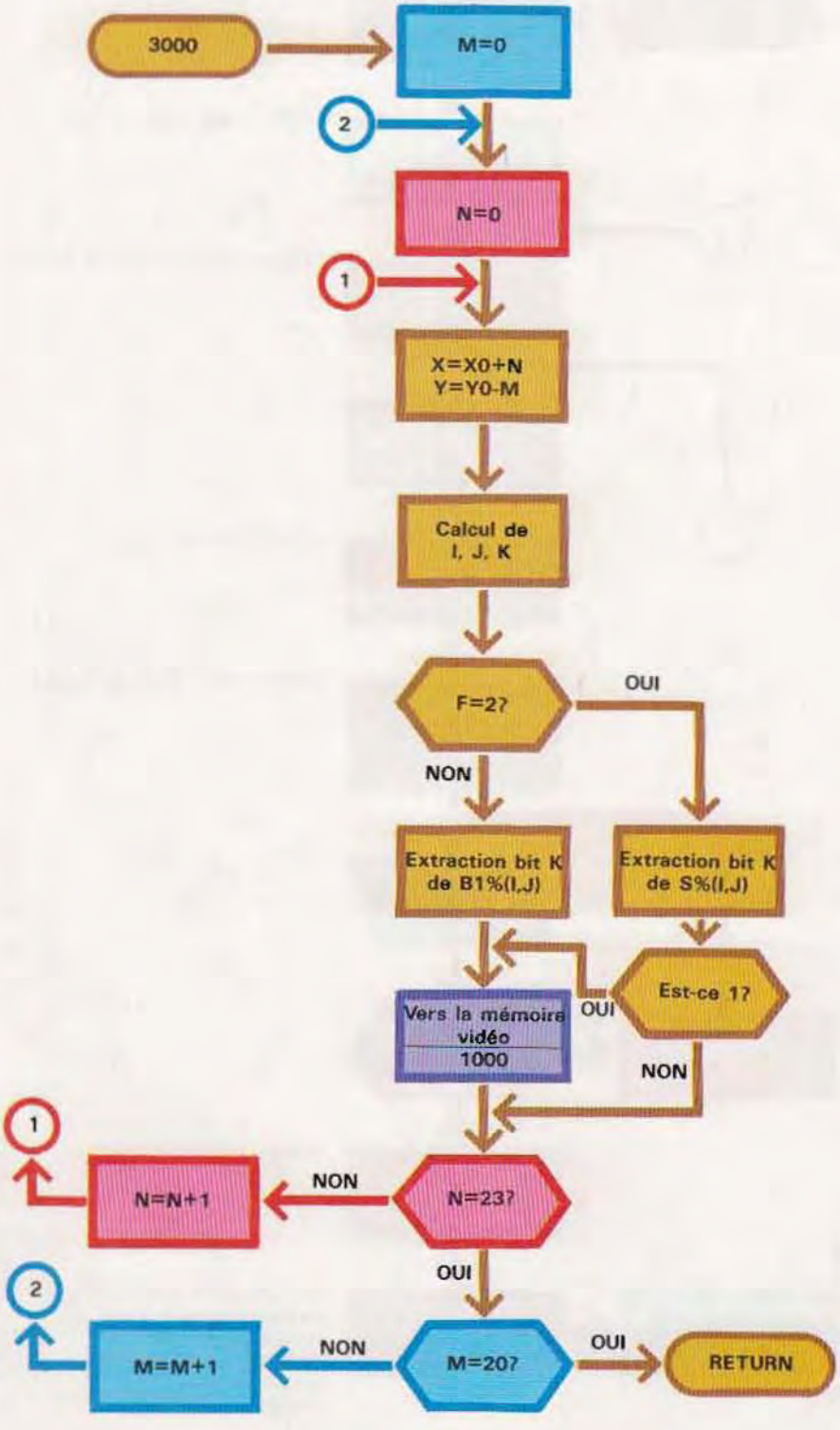
La même procédure permet d'obtenir l'adresse (indices et bits) de tout autre point. Par exemple, le point N=12, M=13 est contenu dans l'élément B1%(8,2) bit 3, selon la règle :

1e indice=21-M
2e indice=INT(N/8)+1
3e du bit=Second indice*8-N-1

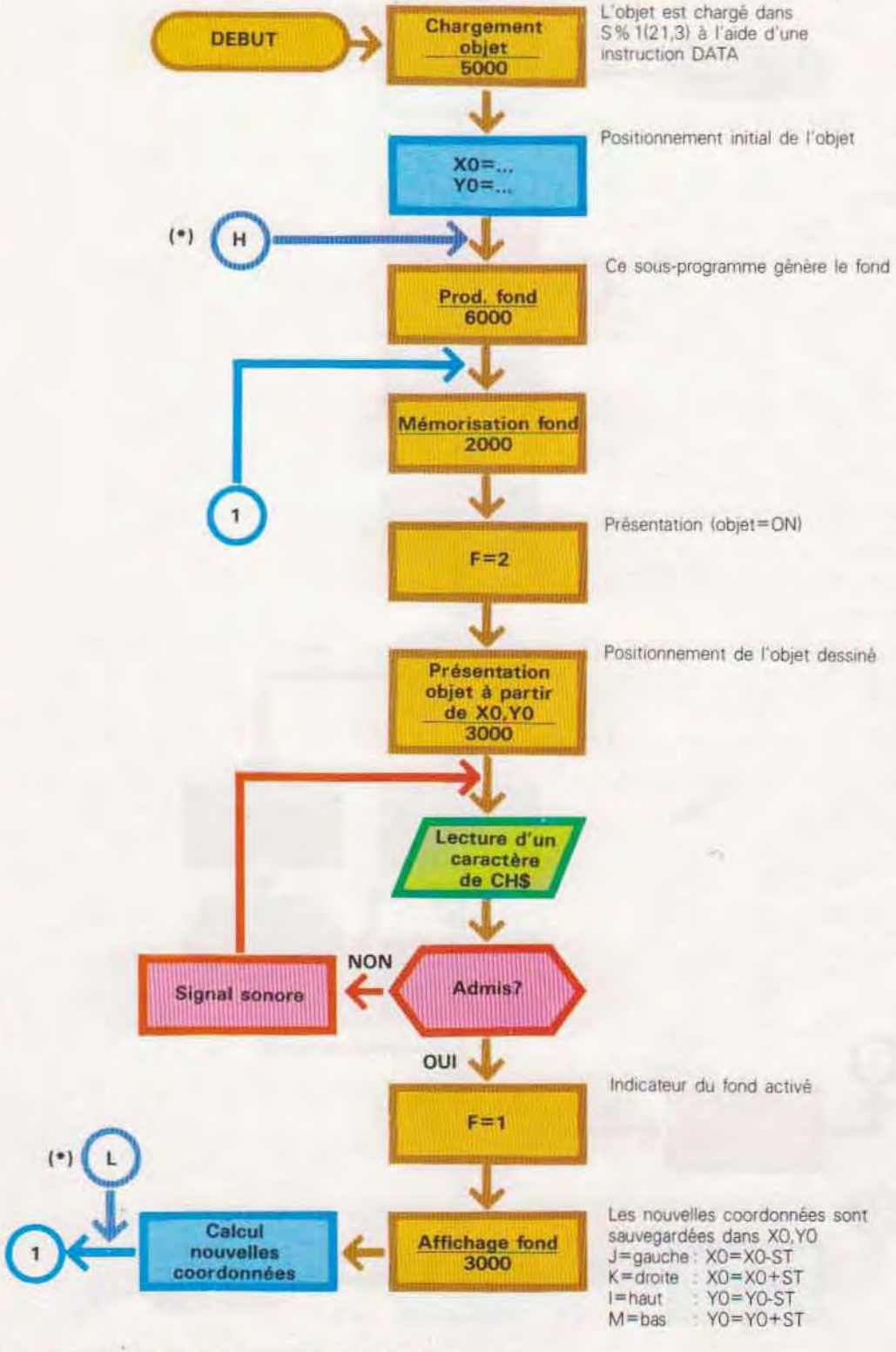
Noter que le point de référence (point X0,Y0) est distant de lui-même des quantités N=0, M=0. Par conséquent :

1e indice=21-0=21
2e indice=1
n° du bit=7

PRESENTATION DU FOND OU DE L'OBJET



EXEMPLE DE GESTION DES OBJETS ANIMES



(*) : Il sera fait référence aux connecteurs H et L plus loin.

EXEMPLE DE GESTION DES OBJETS ANIMES PAR LE LOGICIEL

```
10 REM ------------------------
20 REM * PROGRAMME PRINCIPAL *
30 REM ------------------------
40 DIM S%(21,3),BI%(21,3)
50 ST=5
60 GOSUB 5000
   :REM Chargement de l'objet animé
70 X0=10:Y0=180
80 GOSUB 6000:REM Dessin du fond
90 GOSUB 2000
   :REM Lecture de la fenêtre

100 F=2
110 GOSUB 3000
    :REM Ecriture de l'objet animé
120 GET CH$
130 IF CH$<>"I" AND CH$<>"J" AND
     CH$<>"M" AND CH$<>"K"
     AND CH$<>"CHR$(27) THEN 120
140 F=1
150 GOSUB 3000
    :REM Ecriture de la fenêtre
160 IF CH$="I" THEN Y0=Y0-ST
170 IF CH$="M" THEN Y0=Y0+ST
180 IF CH$="J" THEN X0=X0-ST
190 IF CH$="K" THEN X0=X0+ST
200 IF CH$=CHR$(27) THEN END

210 GOTO 90

1000 REM ------------------------
1010 REM * CALCUL DES ADRESSES *
1020 REM ------------------------
1030 GY=INT(Y/8)
1040 IF Y<64 THEN MR=8232+(GY-8)*128
     :GOTO 1070
1050 IF Y<128 THEN MR=8232+(GY-8)*128
     :GOTO 1070
1060 MR=8272+(GY-16)*128
1070 A=Y-GY*8
1080 MY=MR+A*1024
1090 GX=INT(X/7)

1100 MX=MY+GX
1110 BIT=X-7*GX
1120 NN=PEEK(MX)
1130 BY=NN
1140 FOR KK=0 TO BIT
1150 MS=NN/2
1160 NN=INT(MS)
1170 NEXT
1180 IF MS<>NN THEN BT=1:GOTO 1200
     :REM Le point écran est allumé
1190 BT=0
     :REM Le point écran est éteint

1200 RETURN

2000 REM ------------------------
2010 REM * LECTURE DE LA FENETRE *
2020 REM ------------------------
2030 MD$="L"
2040 FOR M=0 TO 20
2050 FOR N=0 TO 23
2060 X=X0+N:Y=Y0-M
```

```
2070 GOSUB 1000
2080 I=21-M:REM Premier indice
2090 J=INT(N/8)+1:REM Deuxième indice

2100 K=J*8-N-1
     :REM Numéro du point écran
2110 B1%(I,J)=B1%(I,J)+BT*2^K
2120 NEXT N
2130 NEXT M
2140 RETURN

3000 REM --------------------------
3010 REM * ECRITURE DE LA FENETRE *
3020 REM --------------------------
3030 MD$="S"
3040 FOR M=0 TO 20
3050 FOR N=0 TO 23
3060 X=X0+N:Y=Y0-M
3070 I=21-M:REM cf. 2080
3080 J=INT(N/8)+1:REM cf. 2090
3090 K=J*8-N-1:REM cf. 2100

3100 IF F=2 THEN N0=S%(I,J):GOTO 3120
3110 N0=B1%(I,J)
3120 FOR N1=0 TO K
3130 N2=N0/2
3140 N0=INT(N2)
3150 NEXT N1
3160 IF N2=N0 THEN BB=0:GOTO 3180
3170 BB=1
3180 GOSUB 1000
3190 IF BT=BB THEN 3220

3200 IF BT>BB THEN POKE MX,BY-2^BIT
     :GOTO 3220
3210 POKE MX,BY+2^BIT
3220 NEXT N
3230 NEXT M
3240 RETURN

5000 REM --------------------------
5010 REM * CHARGEMENT OBJET ANIME *
5020 REM --------------------------
5030 FOR I=1 TO 21
5040 FOR J=1 TO 3
5050 READ S%(I,J)
5060 NEXT J,I
5070 DATA 0,0,0
5080 DATA 0,0,0
5090 DATA 0,0,0

5100 DATA 0,0,0
5110 DATA 0,0,0
5120 DATA 0,0,0
5130 DATA 0,0,0
5140 DATA 28,0,248
5150 DATA 62,1,252
5160 DATA 103,3,254
5170 DATA 199,7,158
5180 DATA 143,15,28
5190 DATA 30,15,28

5200 DATA 60,30,60
5210 DATA 60,30,108
5220 DATA 31,252,248
5230 DATA 15,248,144
```

```
5240 DATA 7,240,48
5250 DATA 0,0,0
5260 DATA 0,0,0
5270 DATA 0,0,0
5280 RETURN

6000 REM ----------------------
6010 REM * CHARGEMENT DU FOND *
6020 REM ----------------------
6030 HOME
6040 INPUT "Nom du fond ";NS$

6050 HGR:POKE -16302,0:HCOLOR=3
6060 PRINT CHR$(4);"BLOAD";NS$
     ;"A$2000"
6070 RETURN
```

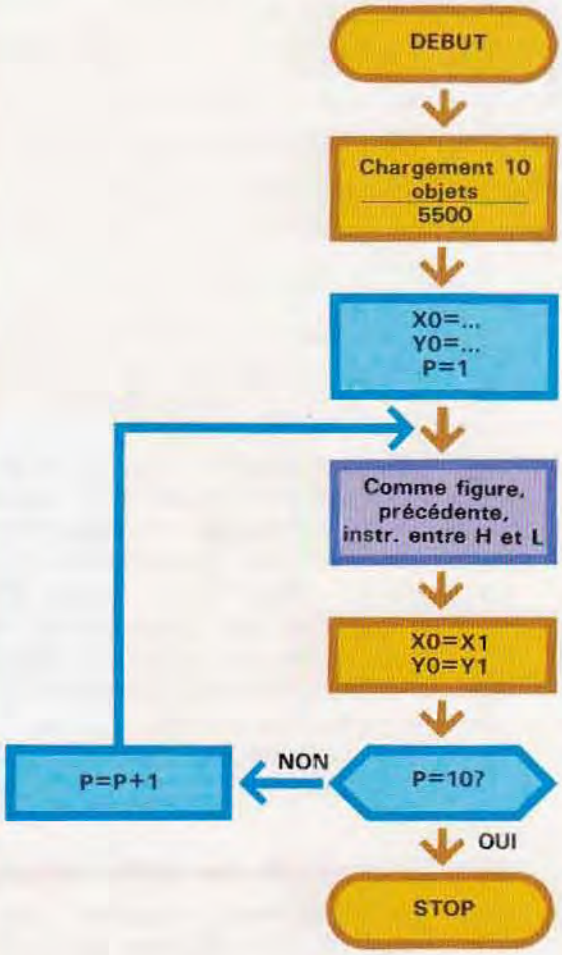
La seconde solution est la plus pratique, tant en raison de sa vitesse d'exécution que de la moindre occupation de l'espace mémoire. Elle n'est toutefois pas toujours applicable, les déplacements d'une figure étant difficiles à schématiser avec des formules mathématiques.
Quant à la première solution, elle ne nécessite qu'un compteur et une zone de mémoire contenant toutes les positions intermédiaires que l'on veut utiliser. Par exemple, si l'on range 100 positions intermédiaires dans T% (100,21,3), il suffit de les transférer (les unes après les autres, avec le compteur) dans S% (21,3) pour obtenir un mouvement également à l'intérieur de l'objet.
Une autre possibilité, enfin, consiste à ne dessiner que les positions initiales et finale, et éventuellement quelques intermédiaires si les deux précédentes sont très différentes. On laissera à l'ordinateur le soin de calculer tous les passages intermédiaires nécessaires à l'animation. Cette technique, qui présente des aspects extrêmement complexes, est encore à l'étude mais elle sera de plus en plus employée pour les productions cinématographiques.
On trouvera, à la page suivante, l'organigramme donné en page 1666, modifié de façon à permettre également l'animation à l'intérieur de l'objet. Cette animation s'obtient en préparant une dizaine de formes qui seront successivement présentées. Le déplacement de l'objet étant commandé par touches, l'animation est ralentie par la phase d'entrée. Pour obtenir un résultat plus réaliste, on définit préalablement un parcours qui sera exécuté automatiquement. La méthode la plus simple, pour décrire ce parcours consiste à définir un tableau (à 10 valeurs) dans lequel seront introduits (à l'aide d'une instruction DATA) les codes correspondant aux touches de déplacement. L'instruction de lecture sera alors activée par l'instruction CH$ = SP (P), SP étant un tableau contenant les codes de placement.
Une boucle dans laquelle P va de 1 à 10 simulera les pressions consécutives de 10 touches de déplacement et génèrera automatiquement le parcours défini dans l'instruction DATA.
La gestion des collisions est un élément indispensable pour ce type d'applications. Elle consiste à chercher si, lors de son déplacement, un objet animé en rencontre un autre ou une zone particulière du fond.
Ce contrôle s'effectue par lecture de la zone mémoire occupée par l'objet : il y a collision si elle est active.

Gestion matérielle des objets animés

Beaucoup d'ordinateurs personnels, certes plus proches des jeux vidéo que des applications professionnelles, sont équipés de composants matériels dédiés à la gestion des objets animés.
Ces composants peuvent mémoriser, déplacer et contrôler simultanément plusieurs objets. Il ne reste, à l'utilisateur, qu'à produire ces objets et le fond, puis à donner les instructions de mouvement, sans avoir à s'occuper ni d'effacement ni de reconstruction.
Dans ce type de machine, chaque objet animé se compose d'un ensemble de points divisés en rangées et colonnes ; on peut constituer la forme désirée en allumant certains d'entre eux.

Mémorisation de l'objet animé. Pour chaque objet il est prévu plusieurs centaines de points écran. Par exemple, la valeur utilisée par le Commodore 64 est de 504 points divisés en 21 rangées de 24 colonnes. L'occupation

GESTION DES OBJETS AVEC ANIMATION INTERNE



Les objets préparés (10) sont des formes différentes de la même figure. Présentées les unes après les autres, elles permettent d'animer l'intérieur du rectangle. Les 10 formes sont mémorisées dans T(10,21,3) à l'aide d'une instruction DATA

P est le compteur réglant la succession des 10 formes de l'objet

La seule différence se situe au niveau du sous-programme 3000, qui doit concerner le fond, alors qu'un autre (3700) gère l'objet

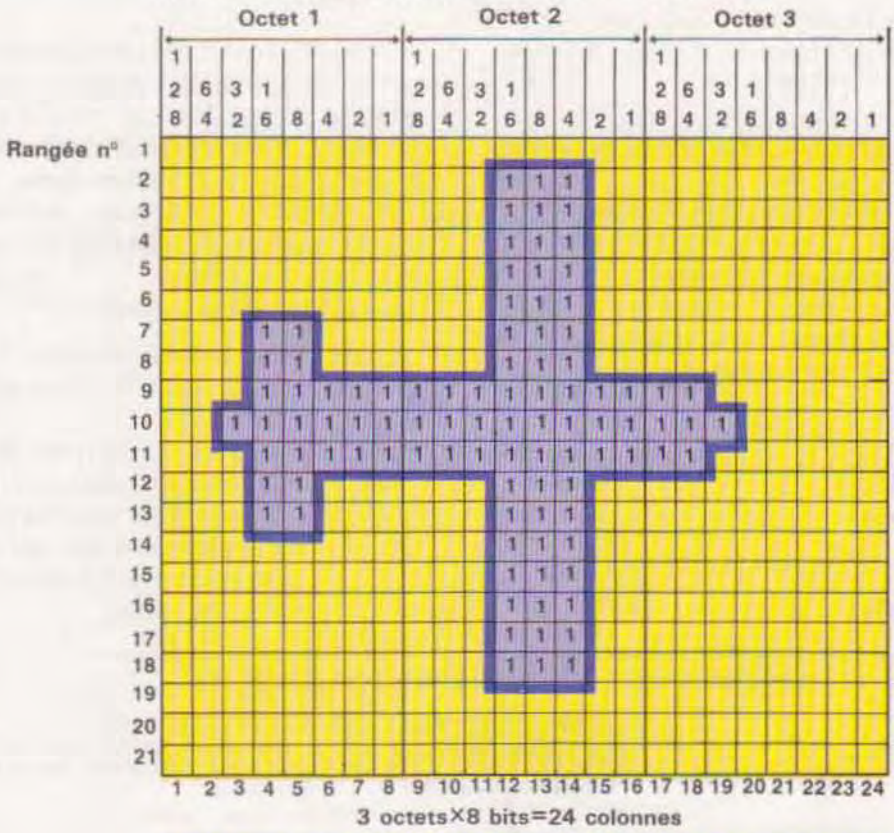
Le sous-programme 3700 est identique au 3000, à ceci près :
— une boucle initiale transfère T%(P,21,3) dans S%(21,3)
— le test de F et le bloc d'extraction du bit K de B1%(I,J) sont éliminés ; le sous-programme n'est plus paramétré et n'a donc pas à gérer le fond

en mémoire est donc de 504 bits soit 63 octets. La construction d'un objet consiste à mémoriser, dans 63 cellules de mémoire contiguës, l'état allumé/éteint de chacun des 504 bits. Pour allumer un point, il faut écrire, dans l'octet correspondant, une valeur décimale particulière qui active le bit sélectionné en le mettant à 1. Par exemple, pour activer le point se trouvant au croisement entre la première rangée et la première colonne, il faut écrire la valeur 128, tandis que pour la colonne 4, cette valeur sera 16 (ne pas oublier que les valeurs décimales correspondant à la fonction du bit sont 1, 2, 4, 8, 16, 32, 64, 128). Les valeurs pouvant être additionnées en écrivant le chiffre décimal 7, on active simultanément les bits se trouvant aux positions 1, 2 et 3.

Dans le schéma ci-contre, la 1re rangée ne contient aucun point actif : les points qui lui sont associés sont donc 0, 0, 0 (octets 1, 2 et 3 respectivement). A la 2e rangée, sont actifs les points 12, 13, 14 qui appartiennent à l'octet 2 et valent 16+8+4=28 ; elle est donc représentée par le groupe 0, 28, 0.

De même, pour le calcul des valeurs de la rangée 10, on a :

SCHEMA DE MEMORISATION D'UN OBJET



3 octets×8 bits=24 colonnes

Rangée n°	Valeurs		
	Octet 1	Octet 2	Octet 3
1	0	0	0
2	0	28	0
3	0	28	0
4	0	28	0
5	0	28	0
6	0	28	0
7	24	28	0
8	24	28	0
9	31	255	192
10	63	255	224
11	31	255	192
12	24	28	0
13	24	28	0
14	0	28	0
15	0	28	0
16	0	28	0
17	0	28	0
18	0	28	0
19	0	0	0
20	0	0	0
21	0	0	0

Points actifs de l'octet 1 : 3, 4, 5, 6, 7, 8 = =32+16+8+4+2+1=63
Points actifs de l'octet 2 : tous, soit =255
Points actifs de l'octet 3 : 17, 18, 19 = =128+64+32=224

La rangée 10 est donc représentée par le groupe de valeurs 63, 255, 224.
L'archivage de la table ainsi constituée est réalisé à l'aide de l'instruction POKE. En supposant que la mémorisation commence à la position 896 (valeur prise uniquement à titre d'exemple), il faudra 3 instructions pour mémoriser la première rangée (une instruction par octet) :

```
POKE 896,0        1er octet
POKE 897,0        2e octet
POKE 898,0        3e octet
```

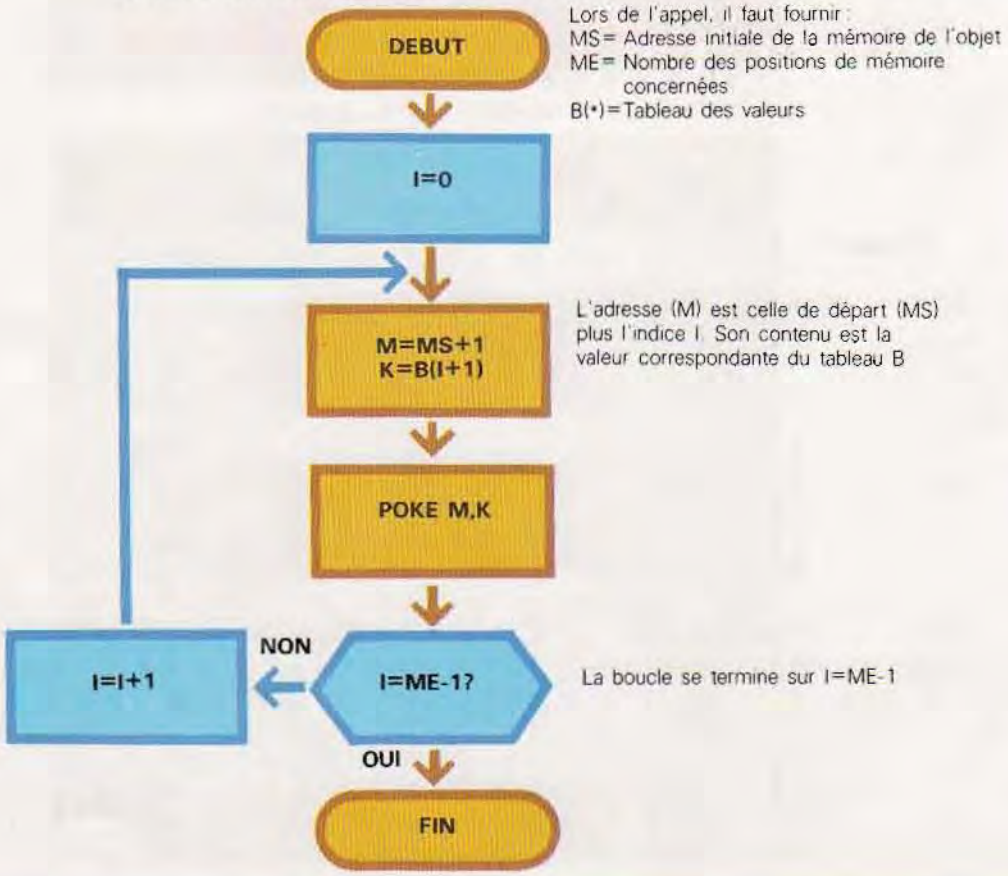
tandis que la 2e rangée sera entrée à l'aide de :

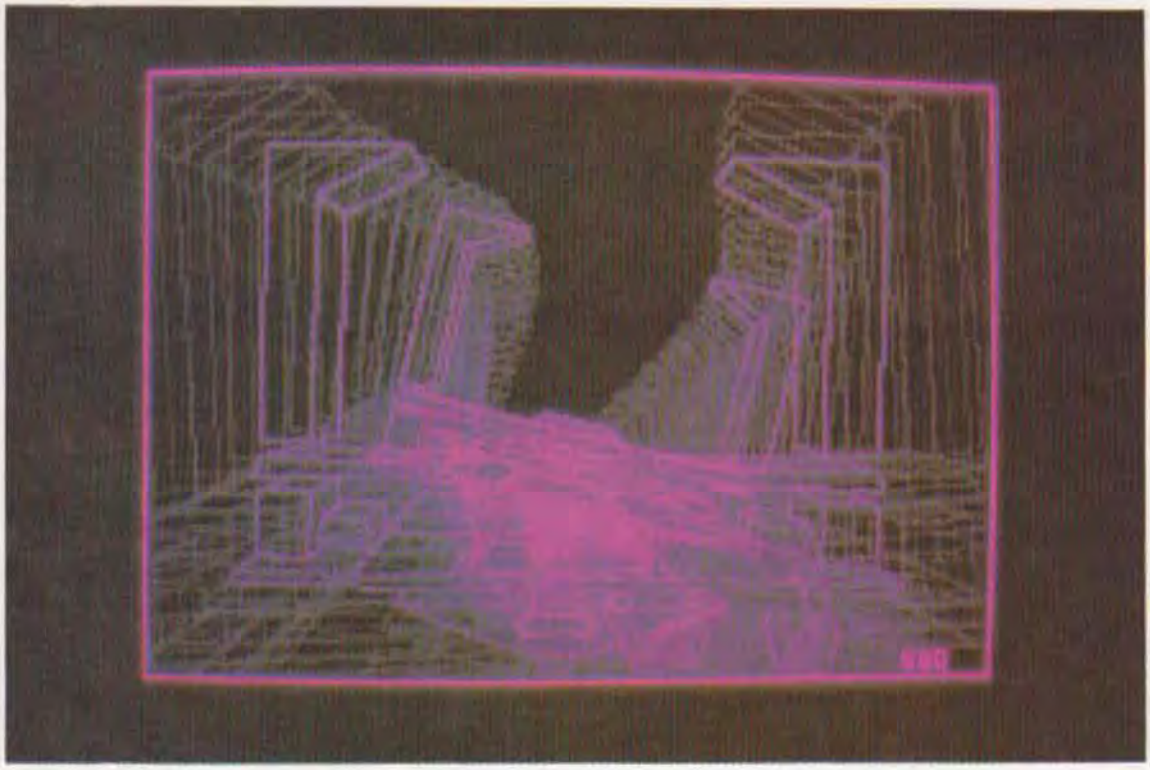
```
POKE 899,0
POKE 900,28
POKE 901,0
```

Il n'est évidemment pas nécessaire d'écrire autant d'instructions qu'il existe de positions de mémoire à activer. La meilleure solution consiste à utiliser une boucle depuis la position de départ jusqu'à la position finale.
L'organigramme ci-dessous représente un sous-programme de mémorisation de la table de la page précédente.
Les variables utilisées sont :

B (63) = Contient les 3 octets de chaque rangée (3 octets × 21 rangées = 63 octets)
MS = Position de mémoire de départ
ME = Nombre de positions de mémoire à activer. Pour l'ensemble de la table, il faut entrer ME=63 (64 si l'on tient compte de l'octet nul qui achève la description).

EXEMPLE DE MEMORISATION D'UN OBJET





P.A. Simon/Grazia Neri-Photolake

Une image animée réalisée pour le film « Tron ».

La table de difinition est rangée dans n'importe quelle zone de la mémoire vive (RAM) et pas nécessairement à proximité immédiate des définitions d'autres objets.
En effet, un registre pointeur permettra de retrouver chacune d'entre elles. Dans le Commodore 64, qui gère jusqu'à 8 objets différents, les pointeurs des définitions des objets occupent les positions 2040 à 2047.
Dans les applications, il est toutefois utile de réserver, à la mémorisation des tables de description des objets, une zone unique divisée en autant de blocs de 64 octets qu'il y a d'objets à gérer.
En supposant que la zone mémoire réservée aux objets commence à la position 832, l'objet défini précédemment (et chargé à la position 896) se trouvera dans le bloc 2. Après son activation, il faudra donc écrire la valeur 14 (12 + 2) à la position réservée à l'adressage (2040).
Le système peut ainsi retrouver sa position en effectuant la multiplication 14×64= 896.

Visualisation de l'objet animé. Une fois mémorisée la table, on doit activer la logique de visualisation de l'objet. La gestion des objets (présentation, déplacement, etc.) se fait généralement en activant ou en désactivant des positions de mémoire spéciales et préprogrammées.
Le système étant capable d'accepter plusieurs objets il faut lui indiquer sur lequel on désire travailler. Cette fonction est généralement activée en mettant à 1 un bit d'une position mémoire particulière. Par exemple, en écrivant 3 (décimal, c'est-à-dire 11 binaire) à cette adresse, on déclare actifs les objets n° 0 et n° 1.
Dans le Commodore 64, le registre de contrôle de la visualisation des objets se trouve à l'adresse 53269. Si le bit n° 0 de ce registre est mis à 1, la visualisation de l'objet n° 0 est activée. Il en sera de même pour tous les autres objets.
Toutefois il n'y aura visualisation que si le point de l'écran où se trouve l'objet est spécifié. Pour cela, chaque objet est contrôlé par deux registres dans lesquels il est nécessaire d'entrer les coordonnées (en points écran) du point désiré. Par exemple, la position de l'objet 0 est contrôlée par les registres 53248 et 53249 .
Dans le premier, c'est la valeur de la coordon-

née X qui devra être chargée (POKE), alors que dans le second c'est celle de la coordonnée Y. La visualisation de l'objet 0 (mémorisé dans le bloc 2) au point 100, 150 s'obtiendra par :

```
30 POKE 2040,2
40 POKE 53248,100
50 POKE 53249,150
60 POKE 53269,1
```

Déplacement de l'objet. Pour déplacer l'objet visualisé, il faut, à l'aide d'une boucle, faire varier les coordonnées de sa posotion contenues dans les deux registres de contrôle.
Dans le cas considéré, le déplacement horizontal de l'objet 0, s'obtiendra par :

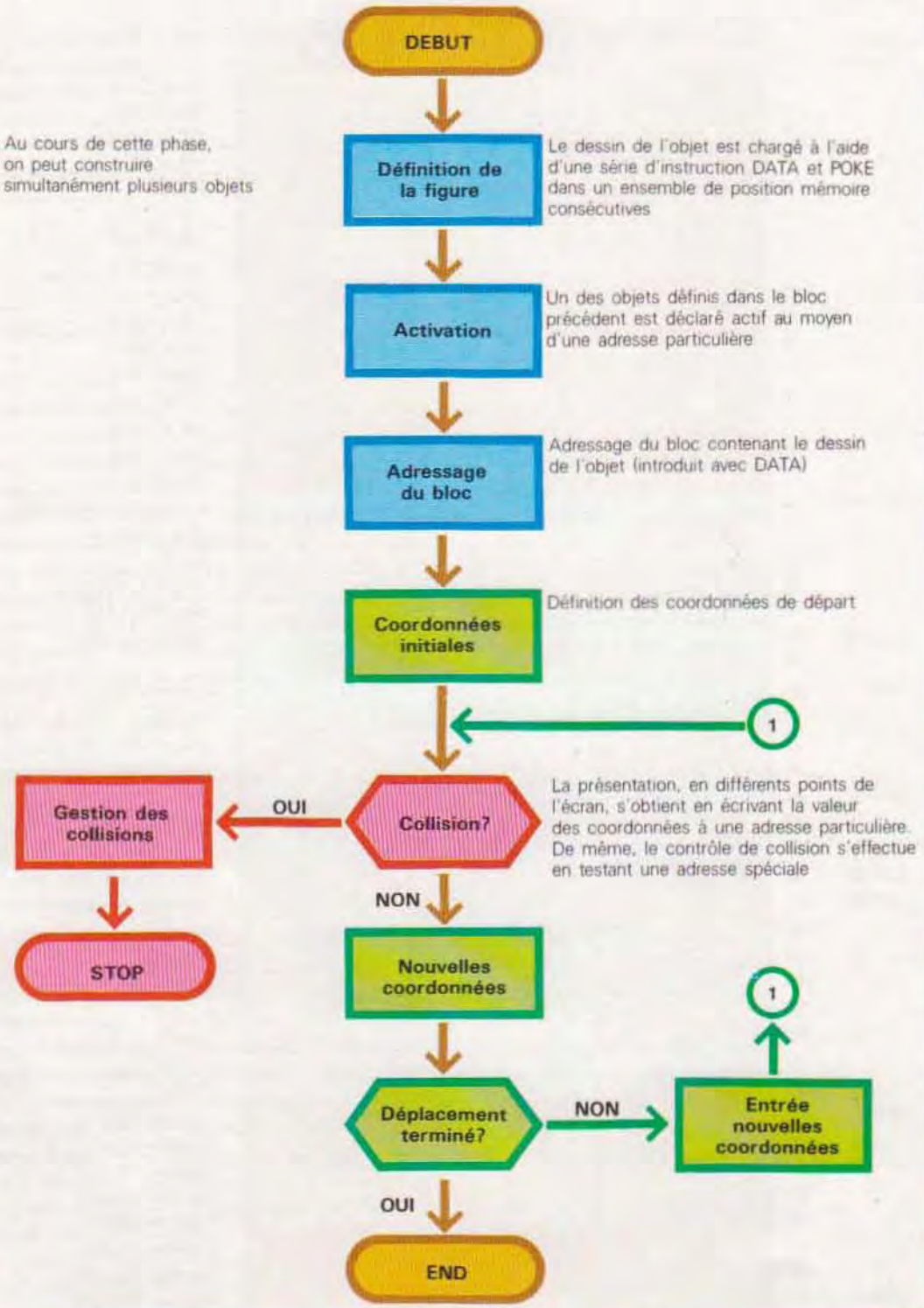
```
100 POKE 53249,100
110 FOR I = 50 TO 150
120 POKE 53248,I
130 NEXT I
```

La ligne 100 fixe la valeur de Y, tandis que la boucle attribue les valeurs 50 à 150 à la coordonnée X.
Il est souvent nécessaire d'associer un contrôle de collisions au déplacement d'un objet. Là encore, il existe des positions mémoire spéciales dont le contenu indique si une collision s'est produite.
Dans le Commodore 64, la collision entre deux objets est contrôlée par le registre 53278 et celle objet-fond par le registre 53279 (voir organigramme ci-contre).

Utilisation de la mémoire pour la gestion des objets animés. La zone mémoire du Commodore 64 chargée de la gestion des objets est schématisée en page 1676 ; elle commence à la position 53248.
Les 16 premières cellules sont les registres de positionnement des 8 objets que la machine peut gérer et dans lesquelles seront chargées les coordonnées de leur positionnement.
Vient ensuite un registre (53264) qui stockera les bits les plus significatifs des coordonnées X (1 bit par objet). Les 4 octets suivants ne concernent pas les objets animés. Le registre d'activation/désactivation est à l'adresse 53269 (adresse relative 21). En mettant à 1 un bit de ce registre, on active l'objet correspondant.
Aux adresses 53271 et 53277 se trouvent deux registres qui permettent de doubler les dimensions de l'objet, le premier le long de l'axe X et le second le long de l'axe Y, en mettant à 1 le bit correspondant (0 pour l'objet 0, 1 pour l'objet I, etc).
Le registre 53275 établit la priorité d'un objet par rapport au fond. Par exemple, si le bit n° 3 de ce registre a la valeur 0, l'objet 3 a la priorité sur le fond (c'est-à-dire qu'il couvrira le fond). S'il vaut 1, ce sera l'inverse.
Les positions 53278 et 53279 contiennent les indicateurs de collision objet-objet et objet-fond. Si l'objet 2 entre en collision avec l'objet 3, les bits 2 et 3 du registre 53278 seront mis à 1 ; il en sera de même en cas de contact avec le fond.
Pour finir, les registres 53287 à 53294 permettront de choisir la couleur des objets en chargeant les codes numériques correspondants (0 = blanc, 1 = noir, etc).
Les autres registres, assez nombreux, sont utilisés pour exécuter des fonctions particulières sur lesquelles nous ne nous arrêterons pas.
En revanche, voyons plus en détail la question de la mémorisation des objets.
Nous avons dit que la description d'un objet se compose d'une suite de 63 + 1 octets consécutifs (le dernier contenant toujours la valeur 0) mémorisés (instruction POKE) à partir d'une adresse de départ.
L'adresse de cette position doit ensuite être mémorisée dans le registre pointeur correspondant à l'objet en question.
En fait, ce qui y est réellement mémorisé ce n'est pas une adresse mémoire, mais un numéro d'ordre d'un bloc de mémoire de 64 octets. En multipliant cette valeur par 64, le système obtient l'adresse réelle à laquelle commence la description de l'objet.
Cette logique est détaillée page 1677.

Exemple d'application. Pages 1678 et 1680, on trouvera l'organigramme d'un jeu utilisant 4 objets animés multicolores.
Le jeu consiste à déplacer horizontalement un parachutiste afin d'éviter les collisions avec une série d'objets qui apparaissent de façon aléatoire. Le jeu est gagné si le parachutiste réussit à toucher le sol.
Pour compliquer sa descente et faire perdre son contrôle au joueur, un vent latéral déplacera le parachute de gauche à droite.

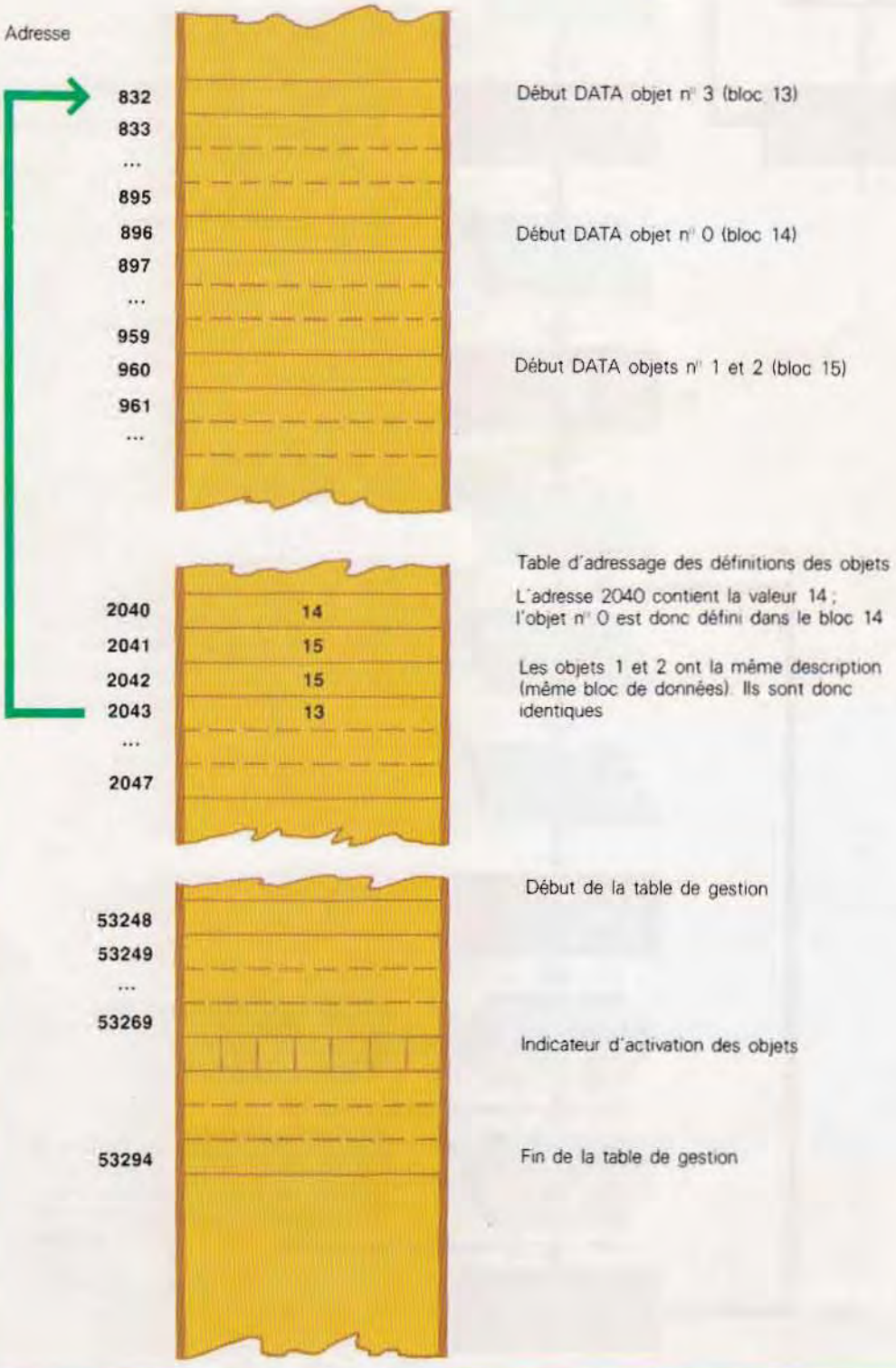
SCHEMA LOGIQUE DE GESTION DES OBJETS ANIMES



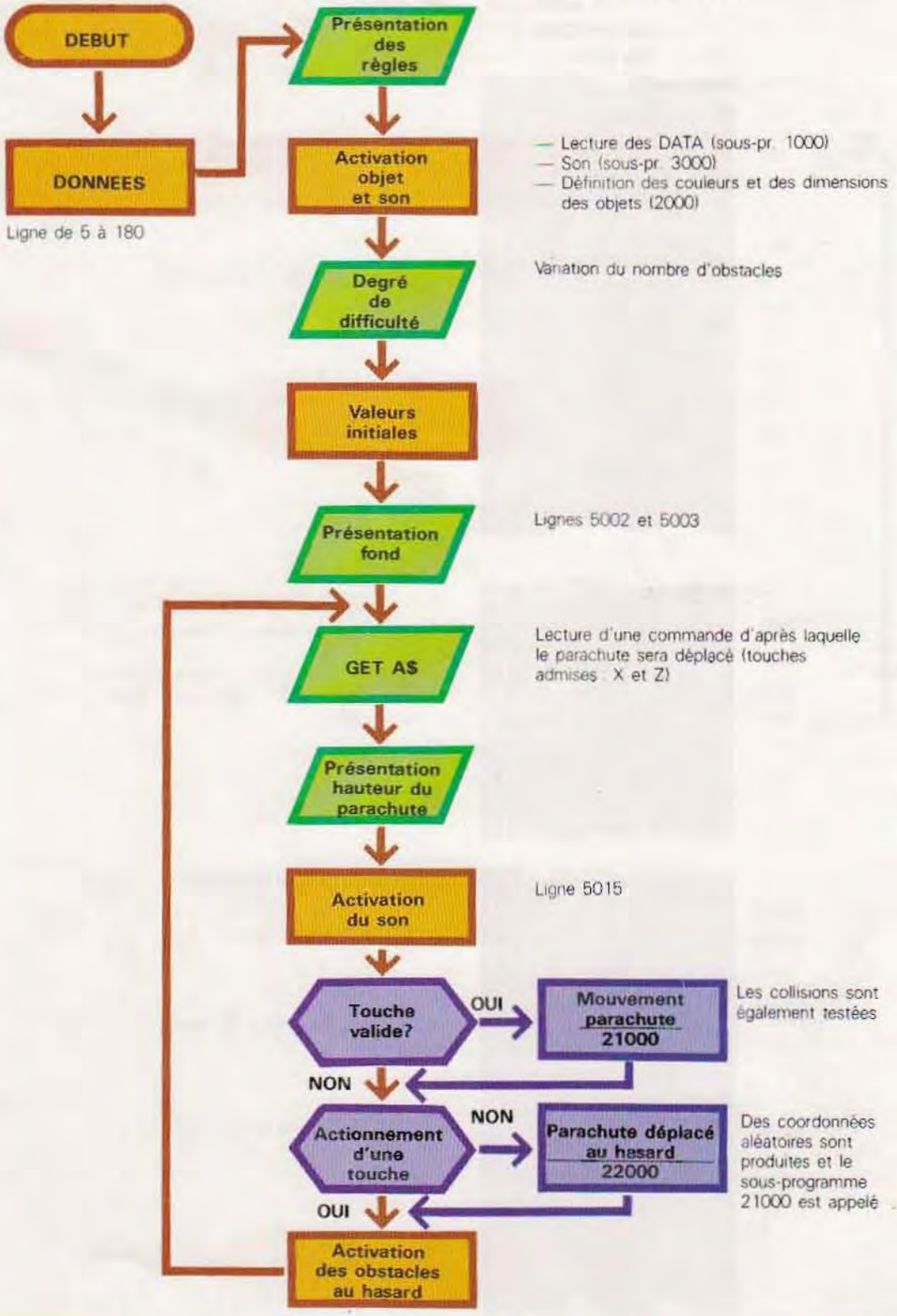
ORGANISATION DES REGISTRES DES OBJETS DANS LE COMMODORE 64

Bit n°		7	6	5	4	3	2	1	0	
Base = 53248		DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
	0									OBJET 0 X
	1									OBJET 0 Y
	2									OBJET 1 X
	3									OBJET 1 Y
	4									OBJET 2 X
	5									OBJET 2 Y
	6									OBJET 3 X
	7									OBJET 3 Y
	8									OBJET 4 X
	9									OBJET 4 Y
	10									OBJET 5 X
	11									OBJET 5 Y
	12									OBJET 6 X
	13									OBJET 6 Y
	14									OBJET 7 X
	15									OBJET 7 Y
53264	16									Bits de plus fort poids
	17									de la valeur X
	18									TABLE
	19									PHOTOSTYLE X
	20									PHOTOSTYLE Y
53269	21									Validation objet
	22									
53271	23									Augmentation Y de l'objet
	24									Mémoire vidéo de l'objet
	25									Demandes d'interruption
	26									MASQUES interruptions
53275	27									Priorité fond de l'objet
	28									Sélection objet multicolore
53277	29									Augmentation X de l'objet
53278	30									COLLISION Objet-Objet
53279	31									COLLISION Objet-Fond
	32			INFORMATIONS COULEUR						Marge
	33									Fond 0
	34									Fond 1
	35									Fond 2
	36									Fond 3
	37						Objets multicolores			OMC 0
	38									OMC 1
53287	39									Couleur objet 0
	40									Couleur objet 1
	41									Couleur objet 2
	42									Couleur objet 3
	43									Couleur objet 4
	44									Couleur objet 5
	45									Couleur objet 6
53294	46									Couleur objet 7

UTILISATION DE LA MEMOIRE DANS LA GESTION DES OBJETS ANIMES



ORGANIGRAMME DU PROGRAMME PARACHUTES



Gestion évoluée des objets animés

Les objets animés se retrouvent dans d'autres applications graphiques que les jeux vidéo, par exemple dans la préparation de symboles élémentaires servant à former des dessins complexes. Cet emploi particulier nécessite toutefois un logiciel de gestion évolué permettant d'appeler les fonctions nécessaires directement en Basic, sans s'occuper de questions aussi complexes que l'adressage de la mémoire ou que l'affectation des tables.

Les machines les plus modernes, notamment dans la catégorie des ordinateurs personnels, acceptent des versions de Basic avancé qui prévoient non seulement des instructions graphiques, mais également une gestion des objets animés. Les principales fonctions implémentées sont les suivantes :

— Définition des objets animés comme variables
— Présentation
— Contrôle des collisions

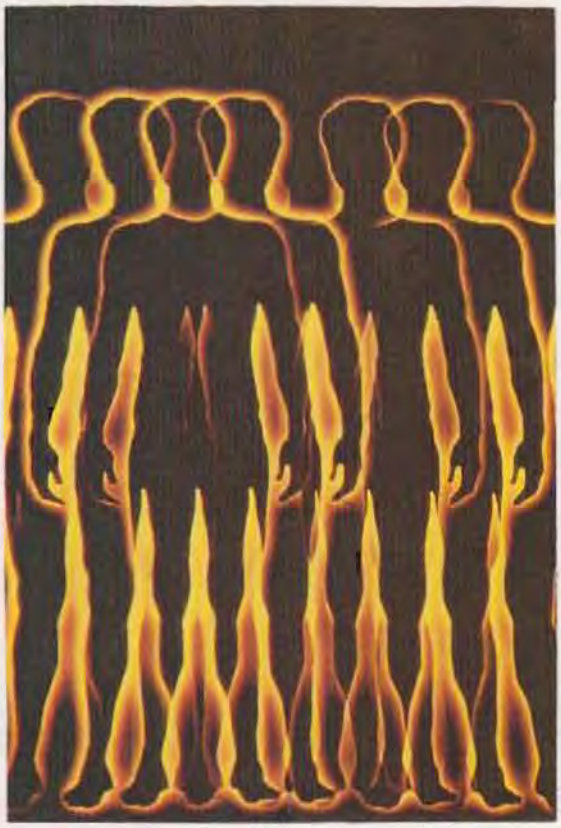
Définition des objets animés. La méthode reste la même que celle qui vient d'être décrite. Elle consiste à réserver une zone de mémoire dans laquelle l'image binaire de l'objet est transférée. Mais, contrairement aux formes précédentes, chaque objet est, dans ce cas, défini comme une variable de chaîne et sera donc chargé à l'aide d'une seule instruction DATA, sans passer par l'écriture de tables d'adressage.

Voici une forme très employée de cette instruction :

```
SPRITE$(N)=B$
```

Elle définit l'objet N comme égal au contenu de la variable de chaîne B$, qui est à son tour chargée au moyen d'une instruction DATA.

Cette instruction, comme les suivantes, est propre au système Philips VG8000, mais fonctionne également sur tous les ordinateurs utilisant le Basic Standard MSX. Cette forme de Basic est très proche du Basic 80 sous CP/M, tout en étant plus riche en instructions spécifiques. Le Basic MSX est d'une diffusion récente mais, du fait de sa remarquable facilité d'emploi (notamment dans les applications graphiques), il a été adopté dans la plupart des machines basées sur le microprocesseur Z80.



Marka

Effet de superposition d'images produit par l'ordinateur.

Visualisation. La forme la plus simple de cette instruction est

```
PUT SPRITE P, (X, Y), C, N
```

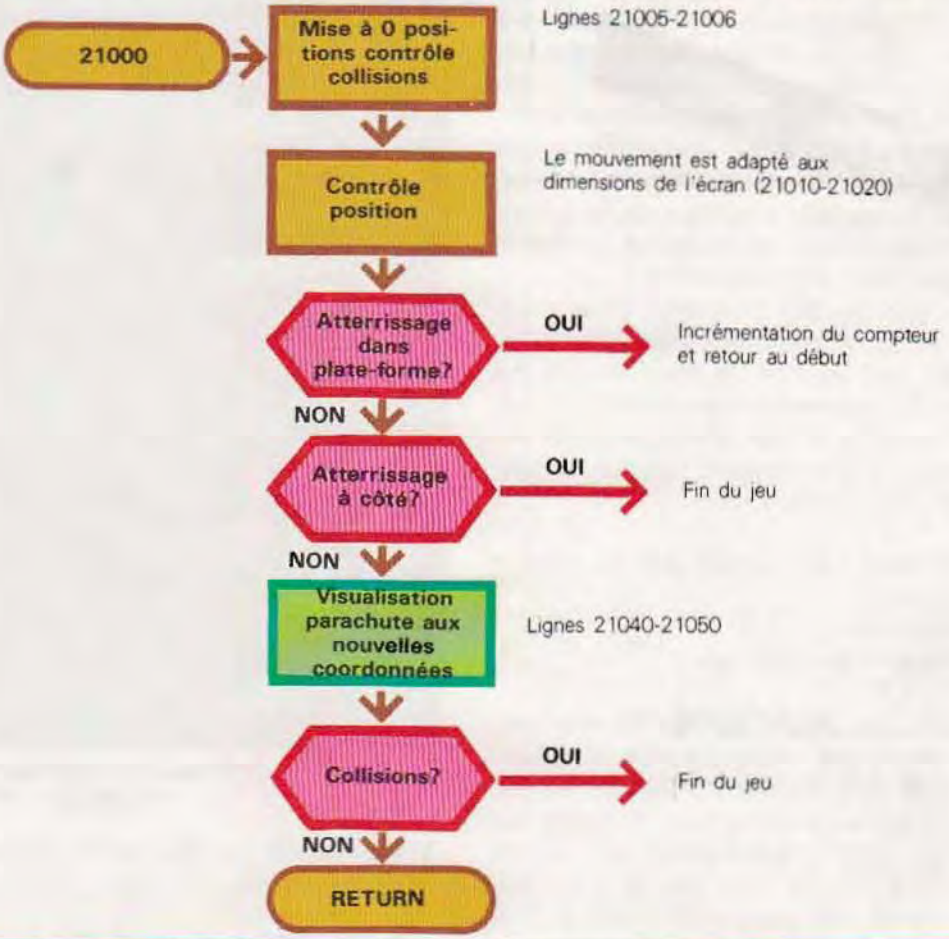
dans laquelle :

P : Indique la priorité de l'objet (pour éviter d'éventuels conflits en cas de présentation de deux objets au même point de l'écran)
X, Y : Sont les coordonnées
C : Est le nombre qui désigne la couleur choisie, de 0 à 15
N : Est le n° d'identification de l'objet, défini dans l'instruction SPRITE$(N)

La syntaxe présentée est, là aussi, propre au système Philips, également utilisable avec les ordinateurs acceptant le Basic MSX. Mais cette instruction n'établit pas les dimensions de l'objet, qui doivent être préalablement définies au moyen de l'instruction SCREEN,

```
SCREEN A, B, C, D
```

SOUS-PROGRAMME DE DEPLACEMENT ET DE CONTROLE



CARACTERES PARTICULIERS UTILISES DANS LE PROGRAMME PARACHUTE VERSION CMB 64

CHR $	Fonction	CHR $	Fonction
147	Vider l'écran	144	Ecrire en noir
18	ON inversé	31	Ecrire en bleu
146	OFF inversé	5	Ecrire en blanc
159	Ecrire en bleu clair	17	Déplacer le curseur vers le bas
28	Ecrire en rouge		

Positions de mémoires

650,128	Répétition automatique
53248+28,8	Définition objet multicolore
53248+42 53248+37 53248+38	Couleurs de l'objet
53248+29,1	Doubler la dimension de l'objet n° 0 (décimal 1) sur l'axe X
53278 53279	Collisions

Instruction d'activation de l'objet N.''n'':
POKE S+21, PEEK (S+21) AND ''n''

EXEMPLE D'UTILISATIONS DES OBJETS ANIMES CBM 64

```
5 REM *** Micro Resque CBM 64 ***
10 REM ** parachutes **
20 DATA 0,170,0,2,170,128,2,40,128,14,0,176
30 DATA 47,130,248,43,235,232,170,255,170,170,190,170
40 DATA 186,170,174,168,0,42,161,0,74,1,0,64
50 DATA 0,113,0,0,243,0,0,40,0,0,40,0
60 DATA 0,60,0,0,60,0,0,60,0,0,20,0,0,0,0
70 REM ** nuage **
80 DATA 0,60,0,0,255,0,3,255,128,15,191,192
90 DATA 63,255,240,255,255,184,239,254,126,243,253,255
100 DATA 253,243,255,127,247,255,62,111,255,28,63,239
110 DATA 0,31,222,0,15,204,2,7,248,12,15,224
120 DATA 16,31,128,8,127,192,1,239,224,0,126,112,0,56,0
130 REM ** faucon **

140 DATA 192,0,3,240,0,15,254,0,127,255,129,255
150 DATA 255,129,255,255,129,255,127,195,254,63,231,252
160 DATA 15,231,240,7,255,224,0,255,0,0,126,0
170 DATA 0,60,0,0,90,0,0,126,0,0,213,0
180 DATA 1,129,128,3,0,192,6,0,96,10,0,80
300 REM ** présentation **
301 GOSUB 3000
310 PRINT CHR$(147):POKE 53281,1:CLR
315 PRINT CHR$(18);CHR$(159);"                                    ";
320 PRINT CHR$(18);CHR$(28);"            Micro Resque             ";
325 PRINT CHR$(18);CHR$(159);"                                    ";
330 PRINT CHR$(17);CHR$(144)"Vous devez faire atterrir un groupe de"
331 PRINT "parachutistes sur le tremplin."
340 PRINT "Attention toutefois à ne pas heurter"
350 PRINT "les mines (;"CHR$(120);") pendant la descente."
360 PRINT "Et attention à éviter aussi les"
370 PRINT "buses géantes et les nuages toxiques."
380 PRINT "Le parachutiste descendra seul":PRINT "sous l'effet de la gravité"
390 PRINT "mais vous devrez le guider à droite et":PRINT "à gauche"
400 PRINT CHR$(17);CHR$(18);CHR$(31)"Touches de direction :";CHR$(146);CHR$(144)
410 PRINT TAB(20);"(";CHR$(18);CHR$(28);"X";CHR$(146);CHR$(144);")... droite"
430 PRINT TAB(20);"(";CHR$(18);CHR$(28);"Z";CHR$(146);CHR$(144);")... gauche"
435 PRINT TAB(12);CHR$(17);CHR$(17);CHR$(18);"F1 pour commencer"
440 GET A$:IF A$CHR$(133) THEN 440
443 PRINT TAB(12);CHR$(18);CHR$(28);"Un instant... "
445 POKE 650,128:REM ** répétition automatique pour toutes les touches **
450 REM ** adresse VIC-II **
460 S=53248
480 GOSUB 1000
490 GOSUB 3000
500 GOSUB 2000
545 PRINT CHR$(147);CHR$(144):INPUT "Difficulté :";DF
550 GOTO 5000:REM ** bloc principal **
1000 REM ** lecture et activation des objets animés **
1010 FOR D=0 TO 62:REM ** lecture parachutes **
1020 READ W
1030 POKE 832+D,W:REM ** objet n°3 bloc 13 **
1040 NEXT D
1050 FOR D=0 TO 62:READ W:REM ** lecture nuage **
1060 POKE 960+D,W:REM ** objet n°0 bloc 14 **
1070 NEXT D
1080 FOR D=0 TO 62:READ W:REM ** lecture faucon **
1090 POKE 960+D,W:REM ** objet n°1E2 bloc 15 **
1100 NEXT D
1120 REM ** activation des objets animés **
1140 POKE S+21,15:REM ** (décimales = 1+2+4+8) **
1150 POKE 2040,14:REM ** nuage au bloc 14 **
1160 POKE 2041,15
1170 POKE 2042,15:REM ** faucons au bloc 15 **
1180 POKE 2043,13:REM ** parachutistes au bloc 13 **
1200 RETURN
2000 REM ** définition des objets animés **
2010 POKE S+28,8:REM ** objet n°3 multicolore **
```

```
2020 POKE S+42,8:REM ** bit "10" orange **
2030 POKE S+37,0:REM ** bit "01" noir **
2040 POKE S+38,1:REM ** bit "11" blanc **
2050 REM ** objet n°0 noir et double en X **
2060 POKE S+39,0:POKE S+29,1
2070 REM ** objet n°1 couleur au hasard **
2080 POKE S+40,INT(RND(1)*2)+1
2090 REM ** objet n°2 couleur au hasard **
2100 POKE S+41,INT(RND(1)*7)+7
2140 RETURN
3000 REM ** activation registres du son **
3010 FOR RS=54272 TO 54296:POKE RS,0:NEXT RS
3020 POKE 54274,0:POKE 54275,8:POKE 54278,240:POKE 54277,0
3030 POKE 54296,15
3040 RETURN
4999 REM *** bloc principal ***
5000 SP=0:CS=0:X1=20:X2=20:X3=255
5001 REM
5002 PRINT CHR$(147):PRINT CHR$(142);:GOSUB 10000
5003 POKE 53278,0:POKE 53279,0
5005 X=INT(RND(1)*150)+100:Y=70:GOSUB 21000
5006 V1=0:V2=0:V3=0:NC=0
5010 GET A$
5011 IF Y<100 THEN B$=" ":GOTO 5013
5012 B$=" "
5013 PRINT CHR$(19);CHR$(18);CHR$(158);"Aterris: ";PAR;
5014 PRINT TAB(23);CHR$(18);CHR$(5)"Altitude: ";(200-Y);CHR$(157);B$;
5015 LB=8:HB=97:HZ=129:GOSUB 30000
5020 A=PEEK(197)
5030 IF A=23 THEN X=X+5:Y=Y+3:GOSUB 21000
5040 IF A=12 THEN X=X-5:Y=Y+3:GOSUB 21000
5045 IF A=64 THEN GOSUB 22000
5050 NC=INT(RND(1)*10)+1
5055 NC1=INT(RND(1)*50)+1
5070 IF NC=20RNC=60RNC=9 THEN GOSUB 23000
5080 IF NC=30RNC=70RNC=10 THEN GOSUB 23100
5085 IF NC1=17 AND Y>159 THEN 35000
5090 IF NC=40RNC=8 THEN GOSUB 23200
5210 GOTO 5010
10000 REM ** fond **
10003 PRINT CHR$(147);:POKE 53281,14:POKE 53280,6
10020 FOR MI=1 TO (DF*3)
10025 W=INT(RND(1)*500)+1
10030 POKE 1224+W,88:POKE 55496+W,INT(RND(1)*2)+1
10035 NEXT MI
10036 FOR I=1 TO 21:PRINT CHR$(17);:NEXT I
10037 PRINT CHR$(18);CHR$(30);"                    ";
10038 PRINT CHR$(18);CHR$(5);"     ";CHR$(30);"              ";
10040 FOR I=1 TO 2:PRINT CHR$(18);CHR$(30)"                    ";

10042 NEXT I
10043 PRINT CHR$(19)
10050 RETURN
21000 REM ** contrôle des coordonnées et visualisation de l'objet n°3
21005 POKE 53278,0
21006 POKE 53279,0
21010 IF X<25 THEN X=X+5:RETURN
21020 IF X>240 THEN X=X-5:RETURN
21030 IF X>170 AND X<176 AND Y>197 THEN 24000
21035 IF (Y>197 AND X<170) OR (Y>197 AND X<176) THEN 24100
21040 POKE S+6,X
21050 POKE S+7,Y
21060 REM ** collisions **
21065 IF (PEEK(53278) AND 9)=1 THEN RETURN
21070 IF (PEEK(53278) AND 9)=9 THEN 24500
21080 IF (PEEK(53278) AND 10)=10 THEN 24520
21090 IF (PEEK(53278) AND 12)=12 THEN 24520
21095 IF (PEEK(53279) AND 8)=8 THEN 26000
21097 IF PEEK(53279)<>8 THEN RETURN
21100 REM
```

```
21105 POKE 53278,0
21110 POKE 53279,0
21150 RETURN
22000 REM ** chute libre **
22010 Y=Y+3
22020 CS=INT(RND(1)*2)+1
22030 IF CS=1 THEN X=X+3
22040 IF CS=2 THEN X=X-3
22050 GOSUB 21000
22100 RETURN
23000 REM ** mouvement du nuage **
23005 Y1=INT(RND(1)*50)+100
23007 LB=1:HB=12:HZ=65:GOSUB 30000
23010 X1=X1+INT(RND(1)*10)+20
23012 IF X1>=24 AND V1>4 THEN SP=254:GOSUB 25000
23014 IF X1>=24 AND V1<4 THEN X1=20:GOTO 23005
23020 POKE S+0,X1:POKE S+1,Y1:GOSUB 21060
23035 RETURN
23100 REM ** mouvement buse 1 **
23102 IF V2<>0 THEN 23115
23107 LB=1:HB=12:HZ=65:GOSUB 30000
23110 Y2=INT(RND(1)*130)+50:V2=V2+1
23115 X2=X2+INT(RND(1)*10)+10
23117 IF X2>240 AND V2>4 THEN SP=253:GOSUB 25000
23119 IF X2>=240 AND V2<4 THEN X2=20:GOTO 23110
23120 POKE S+2,X2:POKE S+3,Y2:GOSUB 21060
23125 RETURN
23200 REM ** mouvement buse 2 **
23202 IF V3<>0 THEN 23215
23204 Y3=INT(RND(1)*150)+50:V3=V3+1
23207 LB=1:HB=12:HZ=65:GOSUB 30000
23215 X3=X3-10
23217 IF X3<=35 AND V3>4 THEN SP=251:GOSUB 25000
23219 IF X3<=35 AND V3<4 THEN X3=255:GOTO 23204
23220 POKE S+4,X3:POKE S+5,Y3:GOSUB 21060
23235 RETURN
24000 REM ** il a atterri **
24030 GOSUB 60000
24050 PRINT CHR$(19);CHR$(18);CHR$(144)"     Parachute atterri     ";
24055 PRINT TAB(15);CHR$(18);CHR$(158);"Félicitations"
24059 FOR T=1 TO 800: NEXT T
24060 PRINT CHR$(147):PAR=PAR+1:POKE 53279,247:GOTO 5001
24100 REM ** hors cible **
24105 PRINT CHR$(19);CHR$(18);CHR$(5);"     Hors cible     ";
24110 GOTO 35065
24500 REM ** collision avec d'autres objets **
24505 POKE 53278,0:SP=247:GOSUB 25000:SP=251:GOSUB 25000:SP=253:GOSUB 25000
      :SP=254:GOSUB 25000
24510 PRINT CHR$(147);CHR$(18);CHR$(5);"Le parachutiste est entré dans"
24515 PRINT CHR$(18);CHR$(5);"le nuage de ";CHR$(158);"trioxine";CHR$(5)
      ;"et est mort"
24516 PRINT CHR$(17);CHR$(5);"Total atterris: ";CHR$(18);CHR$(158);PAR
24517 LB=134:HB=24:HZ=65:FOR T=1 TO 3:GOSUB 30000:NEXT T
24519 GOTO 50000
24520 POKE 53278,0:SP=247:GOSUB 25000:SP=251:GOSUB 25000:SP=253:GOSUB 25000
      :SP=254:GOSUB 25000
24525 PRINT CHR$(147);CHR$(18);CHR$(5);"Le parachutiste a été"
24526 PRINT CHR$(18);CHR$(5);"tué par une";CHR$(158);" buse géante"
24528 LB=1:HB=12:HZ=17:FOR T=1 TO 3:GOSUB 30000:NEXT T
24529 PRINT CHR$(17);CHR$(15);"Total atterris: ";CHR$(18);CHR$(158);PAR
24530 GOTO 50000
25000 REM ** désactivateur **
25010 POKE S+21,PEEK (S+21) AND SP
25020 RETURN
26000 REM ** choc avec les mines **
26005 SP=251:GOSUB 25000:SP=247:GOSUB 25000:SP=253:GOSUB 25000:SP=254
      :GOSUB 25000
26010 PRINT CHR$(147);CHR$(18);CHR$(5);"Le parachutiste a heurté une "
26012 PRINT CHR$(158);CHR$(18);"mine anti-personnel"
26015 LB=1:HB=12:HZ=65:FOR T=1 TO 3:GOSUB 30000:NEXT T
```

```
26020 GOTO 50000
26025 PRINT CHR$(17);CHR$(5);"Total atterris: ";CHR$(18);CHR$(158);PAR
30000 REM ** sous-programme son **
30010 POKE 54276,HZ
30020 POKE 54272,LB
30030 POKE 54273,HB
30040 FOR T=1 TO 5:NEXT T
30050 POKE 54272,0:POKE 54273,0
30060 RETURN
35000 REM ** vent **
35010 SP=254:GOSUB 25000:SP=253:GOSUB 25000:SP=251:GOSUB 25000
35020 FOR K=X TO 255 STEP 9
35030 PRINT CHR$(19);CHR$(17);CHR$(18);CHR$(5);"Vent de 60 noeuds !!"
35035 LB=134:HB=24:HZ=65:GOSUB 30000
35040 POKE S+6,K:POKE S+7,Y
35045 PRINT CHR$(19);CHR$(17);"                "
35060 FOR T=1 TO 100:NEXT T:NEXT K
35065 FOR I=1 TO 7:PRINT CHR$(17):NEXT I:PRINT TAB(14);CHR$(18);CHR$(144)
      ;"Tu es mort !"
35070 SP=247:GOSUB 25000
35080 PRINT CHR$(19);CHR$(5)
35100 END
50000 REM
50010 PRINT CHR$(17);CHR$(5);"Une autre partie (O/N) ?";
50020 INPUT R$
50030 IF R$="O" THEN RUN
50040 PRINT TAB(15);CHR$(17);CHR$(18);CHR$(158);"Au revoir"
50050 FOR T=1 TO 1000:NEXT T:SYS770
60000 REM ** bruit atterrissage **
60005 POKE 54276,65
60010 POKE 54272,33:POKE 54273,134:POKE 54272,35:POKE 54273,132
60020 POKE 54272,37:POKE 54273,161:POKE 54272,42:POKE 54273,60
60030 POKE 54272,44:POKE 54273,191:POKE 54272,47:POKE 54273,104
60035 POKE 54272,47:POKE 54273,104:POKE 54272,44:POKE 54273,191
60040 POKE 54272,42:POKE 54273,60:POKE 54272,37:POKE 54273,60
60045 POKE 54272,35:POKE 54273,132:POKE 54272,33:POKE 54273,134
60060 POKE 54272,0:POKE 54273,0
60100 RETURN
```

Deux moments de l'exécution du programme « Parachutes ».

Les paramètres de cette instruction ont la signification suivante :

A : Mode d'utilisation de l'écran

A=1 texte, 40 caractères par ligne (cpl)
A=2 texte, 32 cpl
A=3 mode graphique 1 (256×192 pixels)
A=4 mode graphique 2 (même nombre de pixels, les figures étant constituées de rectangles de 4×4 pixels)

B : Dimension des objets animés ; ses valeurs vont de 0 (objets de 8×8 pixels) à 3 (32×32 pixels)

C : Quand il est à 1, il se produit un cliquetis chaque fois qu'une touche est actionnée

D : Vitesse de transmission lors de l'échange de données avec l'enregistreur. Les valeurs admises sont 1 (1 200 baud) et 2 (2 400 baud)

Contrôle des collisions. Les instructions de gestion de cette fonction sont :

SPRITE ON/OFF/STOP
pour activer, désactiver, exclure le contrôle
ON SPRITE GOSUB...
appelle le sous-programme spécifié quand une collision se produit.

Le programme ci-dessous et à la page suivante illustre l'emploi de ces instructions.
On se rend compte immédiatement de l'extrême facilité d'écriture des programmes qu'apporte ce type de Basic.

GESTION DES OBJETS ANIMES SUR MSX

```
 4 CLS:CLEAR
 5 COLOR 5,4,1
 7 PLAY "L406CDEFEDCREFGAGFEREDEDGAG"
 10 SCREEN 2,2
 11 GOSUB 9000
 13 C1=15:C2=1:C3=11:P=0
 15 REM ** objet araignée **
 20 DATA 1,3,7,63,79,159,47,79
 25 DATA 147,161,35,5,3,2,4,2
 30 DATA 128,192,224,252,242,249,244,242
 35 DATA 201,133,196,160,192,64,32,64
 40 REM ** objet phalène haut et bas **
 45 DATA 0,0,1,1,3,7,15,31
 50 DATA 63,119,235,221,255,127,61,24
 55 DATA 0,0,128,128,192,224,240,248
 60 DATA 252,238,215,187,255,254,188,24
 65 REM ** phalène gauche **
 70 DATA 0,0,0,1,3,7,15,63
 75 DATA 63,15,7,3,1,0,0,0
 80 DATA 56,124,238,233,191,222,235,254
 85 DATA 254,235,222,191,223,238,124,56
 90 REM ** phalène droite **
 95 DATA 28,62,119,251,253,123,55,127
 100 DATA 127,55,123,253,251,119,62,28
 105 DATA 0,0,0,128,192,224,240,252
 110 DATA 252,240,224,192,128,0,0,0
 140 REM ** objet oeufs **
 145 DATA 0,0,1,3,3,7,15,31
 150 DATA 31,31,31,15,7,0,0,0
 155 DATA 0,0,128,192,192,224,240,248
 160 DATA 248,248,248,240,224,0,0,0
 170 A$=" ":B$=" ":C$=" ":D$=" ":E$=" "
 200 FOR I=1 TO 32:READ A:A$=A$+CHR$(A):NEXT I
 205 SPRITE$(1)=A$
 210 FOR I=1 TO 32:READ B:B$=B$+CHR$(B):NEXT I
 215 SPRITE$(2)=B$
 220 FOR I=1 TO 32:READ C:C$=C$+CHR$(C):NEXT I
 225 SPRITE$(3)=C$
 230 FOR I=1 TO 32:READ D:D$=D$+CHR$(D):NEXT I
 235 SPRITE$(4)=D$
 240 FOR I=1 TO 32:READ E:E$=E$+CHR$(E):NEXT I
 245 SPRITE$(5)=E$
 263 X=100:Y=50:PUT SPRITE 2,(X,Y),C1,2
 264 X=1 INT(RND(1)*250)+1:Y1=INT(RND(1)*100)+50:PUT SPRITE 1,(X1,Y1),C2,1
 265 J=STICK(1)
```

```
267 IF J=1 THEN Y=Y-7:GOSUB 5000
268 IF J=2 THEN X=X+7:Y=Y-7:GOSUB 5000
269 IF J=5 THEN Y=Y+7:GOSUB 5000
270 IF J=4 THEN X=X+7:Y=Y+7:GOSUB 5010
271 IF J=6 THEN X=X-7:Y=Y+7:GOSUB 5020
272 IF J=3 THEN X=X+7:GOSUB 5010
275 IF J=7 THEN X=X-7:GOSUB 5020
276 IF J=8 THEN X=X-7:Y=Y-Y:GOSUB 5020
280 IC=INT(RND(1)*4)+1:IF IC=1 THEN X1=X1-20
282 IF IC=2 THEN X1=X1+20
283 IF IC=3 THEN Y1=Y1-20
284 IF IC=4 THEN Y1=Y1+20
290 GOSUB 1000
295 GOSUB 4000
297 IF X1=X OR Y1=Y THEN 3000
310 ON SPRITE GOSUB 2000
320 SPRITE ON
405 T=T+1
900 GOTO 265
1000 REM ** mouvement araignée **
1003 IF X1>250 THEN X1=X1-20:RETURN
1004 IF X1<10 THEN X1=X1+20:RETURN
1005 IF Y1>190 THEN Y1=Y1-20:RETURN
1006 IF Y1<10 THEN Y1=Y1+20:RETURN
1010 PUT SPRITE 1,(X1,Y1),C2,1
1050 RETURN
3000 REM
3005 SPRITE OFF
3010 PLAY "L20ZBDDC"
3017 COLOR 15,1,1
3018 LOCATE 15,10
3020 PRINT "Tu es mort"
3030 LOCATE 19,12
3030 PRINT "Total points: ";P
3050 LOCATE 1,1
3060 INPUT "Une autre partie (O/N) ";
3070 R$=INKEY$:IF R$="" THEN 3060
3080 IF R$="O" THEN RUN
3200 END
4000 REM ** apparition oeuf **
4010 X2=INT(RND(1)*250)+1
4015 Y2=INT(RND(1)*160)+1
4017 IF T>50 THEN 4070
4019 IF T<1 AND T<50 THEN RETURN
4020 PUT SPRITE 5,(X2,Y2),C3,5
4065 RETURN
4070 T=0:RETURN
5000 REM
5001 IF X>253 OR Y>187 OR X<0 OR Y<0 THEN RETURN
5002 PUT SPRITE 3,(X,209),C1,3:PUT SPRITE 4,(X,209),C1,4
5006 PUT SPRITE 2,(X,Y),C1,2:RETURN
5010 REM
5011 IF X>253 OR Y>187 OR X<0 OR Y<0 THEN RETURN
5013 PUT SPRITE 2,(XX,209),C1,2:PUT SPRITE 3,(X,209),C1,3
5016 PUT SPRITE 4,(X,Y),C1,4
5018 RETURN
5020 REM
5022 IF X>253 OR Y>187 OR X<0 OR Y<0 THEN RETURN
5023 PUT SPRITE 2,(X,209),C1,2:PUT SPRITE 4,(X,209),C1,4
5026 PUT SPRITE 3,(X,Y),C1,3
5029 RETURN
9000 REM ** dessin du fond **
9003 LINE (0,0)-(126-98),1:LINE (128,102)-(255,191),1
9005 LINE (128,89)-(255,0),1:LINE (126,102)-(0,191),1
9007 CIRCLE (126,100),2,1
9010 LINE (126,100)-(0,75),1:LINE (128,100)-(255,75),1
9013 LINE (126,101)-(0,125),1:LINE (128,101)-(255,125),1
9015 LINE (127,98)-(86,0),1:LINE (129,98)-(171,0),1
9017 LINE (125,102)-(86,191),1:LINE (129,102)-(171,191),1
9020 FOR I=S TO 140 STEP 6 CIRCLE (126,100),1,1:NEXT I
10000 RETURN
```

Marka

1687

GLOSSAIRE

A

Accès. Opération d'entrée/sortie sur l'unité de mémoire de masse (disque). Elle est réalisée en deux phases : la recherche et la lecture ou l'écriture. Dans le cas d'un accès disque, qui n'est normalement pas déclenché, une autre phase vient s'y ajouter : le temps nécessaire pour atteindre la vitesse de rotation correcte. C'est d'ailleurs cette phase qui fait la différence entre deux systèmes.

Accès aléatoire (Direct). Méthode de lecture ou d'écriture dans une zone quelconque de la mémoire. Cette méthode ne permet pas de prévoir quelle sera la prochaine position adressée. Elle peut s'appliquer à une zone mémoire RAM ou à un fichier.

Accès direct (ou aléatoire). Méthode permettant d'exécuter directement la fonction d'accès à une donnée. Un fichier est à accès direct (ou aléatoire) quand un enregistrement peut être lu ou écrit indépendamment des autres enregistrements.

Accès séquentiel. Méthode de lecture/écriture qui suit l'ordre normal des enregistrements sur le support (disque, bande magnétique...). Lors d'une opération d'écriture, chaque élément est physiquement positionné après l'élément précédent ; lors de la lecture, l'ordre de sortie des données est le même que l'ordre d'introduction. Il s'oppose donc à l'accès direct (ou aléatoire).

Accès sériel. Expression généralement synonyme d'accès séquentiel. L'accès à une donnée n'est réalisé qu'une fois traitées les données précédentes. Dans un fichier séquentiel, par exemple, un enregistrement n'est lu ou écrit qu'après que les enregistrements antérieurs ont été effectivement traités.

Accumulateur. Position de mémoire ou registre dans lequel sont temporairement rangés les résultats d'un calcul.

Activation. Opération qui lance l'exécution d'une procédure ou d'un sous-programme.

Adaptateur. Interface de nature mécanique qui établit une liaison entre des dispositifs ayant des connecteurs différents.

ADM. Accès Direct à la Mémoire (en anglais : DMA, Direct Memory Access). Transfert de données entre un périphérique et la mémoire sans passer par l'unité centrale.

Adressage calculé. Technique d'adressage consistant à convertir la clé de l'enregistrement en une valeur dont l'adresse est déduite.

Adressage indirect. Méthode d'adressage selon laquelle une position contenant une nouvelle adresse est indiquée. La donnée désirée se trouve à l'adresse ainsi spécifiée.

Adresse. Registre, position de mémoire ou dispositif générique. Une adresse se compose généralement de 8 bits, mais elle peut en avoir 16 ou 32 dans les plus grosses machines. Sa longueur indique l'extension de mémoire utilisable par le système.

Adresse d'une piste. Adresse à laquelle une donnée devrait logiquement se trouver. Ce terme ne s'utilise qu'avec les gros systèmes pour lesquels la gestion des données prévoit une zone dite de débordement. Cette zone est utilisée pour stocker temporairement les enregistrements quand les zones qui leur sont allouées sont occupées.
Il peut également s'agir d'une zone d'adressage (toujours avec une mémoire de masse).

Affectation. Transfert d'une valeur dans une variable. (La valeur précédente est généralement perdue).

Afnor. Association française de normalisation.

Aiguillage. Indicateur dont la valeur provoque une variation du flux de traitement.

Aléatoire. Qualifie une méthode d'adressage par référence à la position. Synonyme d'**Accès direct**.

Aléatoire. Qualifie des nombres produits au hasard à l'aide d'un algorithme particulier contenu dans presque tous les langages.

Algol. Algorithmic Oriented Language. Langage de programmation à usage scientifique, généralement utilisé dans les grands systèmes.

Algorithme. Série de règles permettant de résoudre un problème avec un nombre fini d'opérations de passages.

Alimentation (module d'alimentation). Circuit permettant de convertir une tension de manière à l'adapter aux niveaux requis par d'autres circuits. Il convertit, par exemple, un courant alternatif en courant continu.

Allocation. Désigne l'opération par laquelle une ressource est attribuée (par exemple un périphérique ou une zone de mémoire) à un utilisateur particulier ou à un programme (dans les systèmes autorisant la multiprogrammation).

ALU. Voir **UAL**.

Amorce. Version de base d'un programme capable de s'automodifier. Désigne également un programme capable de s'autocharger. L'amorce est généralement le premier programme chargé dans un ordinateur ; il permet à son tour de charger le système d'exploitation. Il arrive, très souvent, que l'amorce soit résidente et automatiquement activée lors de la mise en marche de la machine.

Amplitude. Valeur d'un signal à un instant déterminé.

Analogique. Qualifie un dispositif dont le fonctionnement est fondé sur l'analogie. Les calculateurs analogiques, à l'inverse des calculateurs numériques (les plus répandus), utilisent tous les niveaux

possibles d'un signal. Dans ce contexte, même les données sont représentées par ces niveaux.

APL. A Programming Language. Langage de programmation à usage scientifique.

Appel. Opération permettant de lancer un programme ou un sous-programme. C'est également, dans bon nombre de langages, un code reconnu (CALL).

Appel. Activation d'une procédure ou d'un sous-programme d'un de ses points de lancement.

APT. Automatically Programmed Tools. Langage de programmation spécial, utilisé pour préparer des données destinées à la commande numérique.

Area. Voir **Zone**.

Arbre. Structure logique ramifiée qui permet, en partant d'un point quelconque, de remonter à l'origine (le parcours suivi est univoque).

Argument. Valeur d'une variable affectée à une fonction fournie par le système ou par l'utilisateur. Dans certains cas, elle est fictive (dummy) et n'a qu'une utilité formelle. L'instruction BASIC FRE(n), par exemple, qui sert à connaître l'espace mémoire disponible, utilise « n » comme argument fictif. Le résultat obtenu sera valide quelle que soit la valeur de « n ».

Array. Voir **Tableau**.

Arrière-plan. Zone de l'écran entourant une figure ou un caractère (voir **Premier plan)**.

Arrondi. Méthode permettant d'éliminer quelques chiffres d'un nombre. Elle permet, en particulier, d'obtenir le nombre le plus proche d'une valeur réelle à gérer par l'ordinateur.

ASCII. American Standard Code for Information Interchange. C'est le code le plus usité pour l'échange d'informations entre un ordinateur et ses périphériques. Dans certains cas, il est également utilisé comme code de mémorisation des données ou de programmes sur disque.

Assemblage. Langage très proche de la logique de la machine, mais qui utilise néanmoins des codes mnémoniques. Il permet d'obtenir une version exécutable d'un programme.

Assembleur. Programme qui traduit, en langage machine, les instructions en langage d'assemblage.

Asynchrone. Désigne un processus dans lequel l'axe du temps n'est pas découpé d'une manière régulière. Qualifie le plus souvent une forme particulière de transmission de données ou un processus dont l'activation n'est pas liée à l'exécution des instructions d'un programme.

Attribut. Propriété caractéristique d'un ou de plusieurs éléments. Un point de l'écran peut avoir, comme attribut, sa couleur ; une zone de données d'un fichier peut avoir, comme attribut, son caractère numérique, le nombre de chiffres la composant ou le nombre de positions décimales de cette valeur.

Autohide. Voir **Elimination des faces cachées**.

Autoplacement. Fonction d'informatique graphique appliquée à la conception des circuits électroniques et permettant de déplacer un composant afin d'optimiser la conception du circuit.

Avant. En parlant d'un dérouleur de bande ou d'un fichier séquentiel, indique le sens de défilement vers l'arrière, de point actuel vers le début. Dans le cas des dérouleurs de bande, indique le rebobinage après lecture.

Avant-plan. Zone de l'écran occupée par un dessin ou par un caractère (voir aussi **Arrière-plan)**.

B

Background. Voir **Arrière plan**.

Backspace. Voir **Rappel arrière**.

Balayage. Opération par laquelle une zone déterminée est explorée séquentiellement. C'est notamment ce que réalise le faisceau électronique d'un terminal vidéo sur l'écran (voir **Balayage récurrent)**.

Balayage récurrent (TV). Méthode de formation de l'image basée sur l'excitation de certains points de l'écran au fur et à mesure que le faisceau électronique le parcourt. Cette méthode est semblable à celle qui est utilisée pour la formation des images de télévision.

Bande perforée. Ruban de papier sur lequel sont enregistrées des données, sous forme de perforations. La présence d'un trou à une position déterminée indique l'état du bit correspondant.

Banque de données. Ensemble de données relatives à un même sujet.

Barker (Code Barker). Forme particulière de codification dans la transmission des données.

Base de données. Ensemble de données en rapport les unes avec les autres, et pouvant servir à différentes applications. Dans les systèmes évolués, le type de mémorisation utilisé permet de les rendre indépendantes du programme qui les exploite. L'ensemble des programmes nécessaires à leur gestion prend le nom de SGBD (Système de Gestion de Base de Données). Ces programmes utilisent généralement des langages spécialisés pour leur description et leur manipulation.

Base de numération. Unité de base d'un système de numération. Les plus utilisées sont la base décimale (base 10), binaire (base 2), octale (base 8) et hexadécimale (base 16).

Batch. Voir **Traitement par lots**.

Baud. Unité d'expression du nombre d'états ou de conditions se produisant en une seconde lors d'une transmission des données. 1 200 bauds indiquent que la transmission s'effectue à une vitesse égale à 1 200 bauds par seconde (un état, ou une condition peut être un bit). Dans les systèmes utilisant 10 bits pour transmettre un caractère, la vitesse de transmission est égale à 1 200/10 = 120 caractères par seconde.

Baudot (Code Baudot). Code de transmission de données antérieur au code ASCII et qui utilise 5 bits par caractère.

Bibliothèque. Index des programmes (source, objet) généralement utilisés par le système d'exploitation.

Binaire. Se dit d'un dispositif ne pouvant avoir que deux états ou valeurs. Fait référence au système de numération en base 2.

Bistable. Circuit ayant deux états stables. Un signal spécial permet de le faire passer de l'un à l'autre et d'obtenir ainsi un compteur en base 2.

Bisynchrone. Méthode de communication permettant de repérer des erreurs multiples.

Bit. Contraction de binary digit (numérotation binaire). Information élémentaire dans un système de numération binaire. Il peut avoir les valeurs 0 ou 1.

Bit de parité. Bit de contrôle dont la présence ou l'absence indique si les bits précédents, qui constituent la donnée, doivent être en nombre pair ou impair.

Blanc. Espace non occupé par des caractères. Dans un ordinateur, un espace blanc est considérée comme un caractère ; il existe donc un caractère « blanc ». A ne pas confondre avec l'espace qui indique un véritable espacement. Ce dernier terme désigne généralement une position de la tête d'écriture d'une imprimante ou du curseur, mais il ne correspond à aucune donnée en mémoire. A l'inverse, « blanc » peut indiquer une quantité mise en mémoire.

Blinking. Voir **Clignotement**.

Blocking. Voir **Groupage**.

BOM. Bill Of Materials. Terme utilisé dans les applications de conception et de fabrication assistées sur ordinateur (CFAO) pour désigner la liste des composants nécessaires à la réalisation d'un appareil.

Bogue. Issu de l'anglais « bug » (« vermine ») et désignant une erreur contenue dans un programme. Termes dérivés : déboguer (mise au point) qui est l'opération de recherche d'erreurs dans un programme avant d'être opérationnel. Un programme spécial est déroulé pour ce faire. La mise au point permet normalement de localiser les failles en examinant, instant par instant, le contenu de la mémoire et l'état des variables.

Booléen. Se dit d'une valeur numérique interprétable comme « vraie » ou « fausse ». L'état vrai est généralement représenté par une valeur non nulle et par zéro. Il peut toutefois arriver qu'une logique inverse soit adoptée.

Bootstrap. Voir **Amorce**.

Boucle. Ensemble d'instructions exécutables récursivement jusqu'à ce qu'une condition déterminée soit vraie.

Boucle de courant. Procédé physique de transmission de données. Le signal est constitué d'un courant généralement compris entre 20 et 50 mA.

Boule roulante. Dispositif d'entrée constitué d'une sphère que l'on fait rouler pour déplacer le curseur. Son fonctionnement est identique à celui du manche à balai.

BPI. Bits Per Inch. Unité de mesure de densité d'enregistrement sur ruban magnétique (bit par pouce). Elle indique la quantité de données binaires contenues dans 27,5 mm de ruban.

BPS. Bits par seconde. Vitesse de transmission en bits. Pour obtenir la vitesse en caractères, il faut diviser la valeur en BPS par le nombre de bits contenues dans un caractère (par 8, si le caractère est un octet, par exemple).

Bruit. Signal, d'origine non déterminée, qui se superpose au signal utile et qui crée des perturbations.

B-Spline. Représentation mathématique d'une courbe plane.

Bubble Sort. Voir **Tri « à bulle »**.

Bucket. Voir **Compartiment**.

Buffer. Voir **Mémoire tampon**.

Bug. Voir **Bogue**.

By-pass. Elément de circuit évitant le passage du courant électrique dans d'autres circuits.

Byte. Voir **Octet**.

C

Cabestan. Mécanisme de réglage de la vitesse dans les enregistrements magnétiques des gros systèmes.

Câblé. Adjectif qualifiant un ensemble de liaisons électriques de transfert et de traitement des signaux.

Canal. Liaison par laquelle transitent les données.

CAO (anglais CAD, Computer Aided Design). Conception Assistée par Ordinateur. Procédé de conception informatisé. Cette expression désigne également un système, compris comme ensemble

matériel et logiciel, orienté vers la résolution de problèmes graphiques et de conception à l'aide de l'ordinateur.

Capacité. Appliqué à la mémoire, ce terme indique sa taille, généralement exprimée en Koctets. Plus généralement, désigne la capacité d'un circuit à emmagasiner une charge électrique, ou encore ce circuit lui-même. En cas d'utilisation de courant alternatif, il sert de pont. La capacité se mesure en micro-farads.

Caractère. Tout symbole utilisé comme partie de données ou pour leur commande et leur organisation. Un caractère n'est pas forcément alphabétique puisque sont également considérés comme caractères des codes spéciaux utilisés pour la gestion des périphériques ou pour la commande d'échanges avec d'autres systèmes (le code ASCII en présente tout un tableau).

Caractère de sollicitation. Symbole émis par l'ordinateur dans l'attente d'une donnée ou d'une instruction.

Caractère alphabétique. Lettre de l'alphabet. Les ordinateurs reconnaissent généralement l'alphabet latin (anglais).

Caractères alphanumériques. Ensemble de symboles constitué à la fois de lettres de l'alphabet et de chiffres, comme symboles et non pas comme valeurs.

Caractère de commande (ou de fonction). Tout caractère (au sens large, et non pas seulement alphabétique) utilisé pour exécuter des fonctions particulières comme l'activation, la modification ou l'exécution d'une opération de commande (transmission de données, enregistrement sur disque, etc.).

Carte de circuits imprimés. Support de réalisation de liaisons électriques. Elles sont fabriquées par photogravure d'une plaque d'un matériau isolant et par dépôt de cuivre.

Catalogue. Liste de tous les fichiers présents (données et programmes).

CCITT. Comité Consultatif International Télégraphique et Téléphonique. Organisme chargé de la normalisation internationale dans le domaine de la transmission.

Cellule. Groupe de positions de mémoire contiguës ayant généralement des contenus homogènes. Dans les bases de données, un ensemble de cellules est défini de telle sorte que son appellation ne recoupe pas celles des subdivisions mécaniques de la mémoire de masse, comme par exemple les secteurs.

CFAO. Conception et fabrication assistées par ordinateur.

Champ. Partie d'un enregistrement de données ayant un sens particulier. Dans un enregistrement d'un fichier d'identification, par exemple, il y a un champ nom de famille, un champ rue... (Voir **Zone**).

Chargeur (cartouche de disques). Ensemble de disques magnétiques utilisés uniquement dans les gros systèmes.

Checkpoint. Voir **Point de reprise**.

Chip. Voir **Puce**.

Circuit intégré. Circuit électronique miniaturisé dans un seul boîtier.

Clavier. Ensemble des touches, tout comme sur une machine à écrire, servant à entrer des données dans l'ordinateur.

Clé. Champ de données utilisé pour rechercher un enregistrement et pour y accéder. La clé peut être primaire, quand elle identifie un enregistrement de façon univoque, ou secondaire si elle est commune à plusieurs enregistrements.

Clé. Partie de l'enregistrement qui en permet l'identification. Dans les opérations de tri, la clé est le champ servant d'élément de comparaison entre les valeurs des enregistrements pour obtenir une sortie ordonnée.

CL File (Cutter Location File : fichier de positionnement de l'outil de coupe). Ensemble de données de sortie d'un système graphique constituant les coordonnées de commande d'une machine à commande numérique.

Clignotement. Variation de la luminosité d'un texte à l'écran. Cette forme de présentation sert à attirer l'attention de l'opération sur un message particulier. Le curseur à l'écran est souvent indiqué par un carré blanc clignotant.

Clipping. Voir **Ecrêtage**.

Clock. Voir **Horloge** et **Signal d'horloge**.

Cobol-Ansi. Langage de programmation développé par l'American National Standard Institute (ANSI) et orienté vers la gestion.

Cocktail Shaker Sort. Voir **Tri « à bulle »**.

Codasyl. (Conference of Data Systems Languages). C'est l'organisation chargée de mettre à jour les spécifications relatives au Cobol ou d'en émettre de nouvelles.

Code. Représentation de données ou de programmes intelligible pour l'ordinateur (par ex. le code ASCII). De ce terme, dérive « codage », qui désigne l'opération d'écriture d'un programme, c'est-à-dire la traduction des fonctions à exécuter sous forme binaire (voir **Indicateur**).

Code. Circuit servant à coder des données.

Code Huffman. Forme particulière de codage selon

laquelle les caractères les plus fréquemment utilisés sont représentés par un nombre de bits inférieur à la normale. Cette technique très complexe permet de diminuer l'espace occupé en mémoire de masse.

Collection de données. Ensemble de données groupées sous un nom collectif unique. Il en existe de deux types : les vecteurs et les groupes répétitifs. Un vecteur est un ensemble avec une seule dimension et des caractéristiques identiques. Le second type désigne un ensemble de données susceptible d'être présent dans le même enregistrement.

Commande. Directive demandant l'utilisation d'une fonction particulière du système d'exploitation ou du langage employé.

Commande numérique et **machines à commandes numériques.** Les commandes numériques sont des instructions mémorisées générées par les systèmes de FAO (fabrication assistée par ordinateur), par exemple pour piloter des machines à l'aide d'un ordinateur. Les machines à commandes numériques sont des machines de production automatisées ayant pour objet d'exécuter un travail particulier commandé par des instructions sous forme de données numériques, généralement produites pas les systèmes de FAO.

Commentaire. Ligne de programme ayant pour objet de fournir des explications. La possibilité d'inclure des lignes de commentaire existe avec tous les langages symboliques. Le compilateur est généralement capable de les éliminer de la version exécutable, de façon à ne pas occuper inutilement de l'espace mémoire (la version exécutable étant écrite en code numérique, les commentaires n'auraient donc aucun sens).

Communication. Opération de transmission ou de réception d'informations. La communication est directe dans le cas de distances faibles, par exemple entre un ordinateur et une imprimante ou un autre terminal local. A l'inverse, sur les grandes distances, l'emploi de dispositifs auxiliaires, comme les modems, est nécessaire.

Compactage. Méthode de réduction de la longueur du champ de la clé en utilisant des algorithmes de transformation appropriées. Désigne aussi la réduction de l'espace occupé par des données, par exemple en éliminant les expaces vides.

Compactage. Terme utilisé principalement en Basic pour désigner l'opération de réduction de la zone mémoire réservée aux chaînes. Elle permet à l'ordinateur d'éliminer celles qui ne sont plus exploitées, de regrouper toutes les zones occupées en une seule, de rendre disponible pour d'autres valeurs l'espace restant.

Compartiment. Zone de mémoire pouvant contenir plusieurs enregistrements et être exploitée globalement.

Compilateur. Programme capable de traduire des programmes utilisant des codes symboliques en une forme compréhensible pour l'ordinateur. Contrairement à l'interpréteur, le compilateur effectue la traduction de la totalité du programme et produit une forme intermédiaire, dite translatable, qui ne peut pas encore être exécutée.

Complément. Indique généralement un inverse. C'est le nombre obtenu par soustraction d'un nombre à un autre. Par exemple, le complément à 10 de 7 est 3 (10-7=3). En informatique, on utilise plus fréquemment le complément à 2, le système de numération étant binaire.

Complément à 2. Forme de représentation des nombres négatifs dans le système binaire.

Compteur. Circuit pouvant compter les impulsions reçues.

Concaténation. Union de deux chaînes pour en former une troisième constituant la somme des deux autres. Ce terme désigne également l'union de deux programmes ou le regroupement de données dans les mémoires de masse.

Constante. Toute valeur fixe, numérique ou alphabétique.

Conversationnel. Qualifie une méthode d'introduction de données selon laquelle l'ordinateur pose une série de questions à l'opérateur et attend les réponses, qu'il valide au fur et à mesure.

Coordonnées. Série de nombres permettant d'identifier un point par rapport à une référence. Les coordonnées peuvent se référer au plan et, dans ce cas, elles sont au nombre de deux (x, y), ou bien encore à l'espace, avec 3 valeurs. Elles servent généralement à identifier la position du curseur à l'écran ou celle de la pointe du traceur.

Coordonnées absolues. Coordonnées indiquant la position d'un point par rapport à l'origine du système de référence utilisé.

Coordonnées relatives. Coordonnées permettant d'identifier un point en indiquant son déplacement par rapport à un autre point.

Copie d'écran (hardcopy). Copie sur papier de ce qui apparaît sur un périphérique de sortie, généralement l'écran.

Corrélateur. Circuit dont la sortie vaut 1 si toutes les entrées sont 0. Sert à former des codes.

Couche. Dans les dessins complexes, l'ensemble des éléments représentés peut être divisé en sous-ensembles portant le nom de couches, afin de fournir une vue logique fractionnée permettant de composer le dessin par groupes.

CPU. Voir **UC**.

CR (Carriage Return : retour chariot). Lorsque ce code (ASCII) est envoyé à une imprimante ou à un écran, la

tête d'écriture ou le curseur va se placer à la première position utile de la même ligne. Pour obtenir un changement de ligne, il faut que le code LF (Line Feed), soit envoyé en même temps. A la fin d'un texte, les codes CR et LF sont donc presque toujours présents.

CRT (Cathode Ray Tube : tube à rayons cathodiques). Composant de l'écran permettant d'afficher les données de façon visible en exploitant les propriétés de certains luminophores excités par un faisceau d'électrons.

CRT. Désigne, par extension, le terminal de visualisation lui-même (écran).

Culot. Logement en matériau isolant protégeant des composants en évitant qu'ils ne soient soudés au circuit imprimé.

Curseur. Elément d'identification d'une position à l'écran. Il est généralement représenté par un tiret ou par un petit rectangle lumineux à l'écran.

Cycle. Désigne une phase de traitement de l'unité centrale, par exemple l'extraction d'une donnée de la mémoire. Une instruction se compose généralement de plusieurs cycles.

Cylindre. Zone pouvant être lue sans mouvement du mécanisme d'accès. Un cylindre désigne généralement l'ensemble de deux traces, chacune sur une face du disque, lisibles à partir d'une seule position des têtes de lecture.

D

DASD. (Direct Access Storage Device : unité de stockage à accès direct). Dispositif de mémoire à accès direct.

Data. Voir **Données**.

Débit. Vitesse d'exécution d'un traitement.

Debugger, debugging. De bug : vermine. Voir **Bogue**.

Décalage (retrait). Insertion d'un certain nombre d'espaces non significatifs au début d'une ligne de programme pour augmenter la lisibilité du listing. Ce décalage n'est possible qu'avec certains langages (Pascal et certaines formes du Basic).

Déclaration. Ensemble d'instructions définissant le type ou les dimensions des variables, ou encore les valeurs des constantes.

Déclaration. Opération d'indication du type ou des dimensions des variables. Elle peut être explicite, si elle se trouve dans les lignes du programme, ou implicite, si elle est automatiquement présumée par le langage utilisé. Par exemple, la déclaration d'un dimensionnement en Fortran est explicite, alors qu'en Basic elle peut, dans certaines limites, être implicite.

Décodeur. Circuit servant à décoder des données.

Décoder. Transposer une valeur de sa forme codée à sa forme normale.

Défaillance. Condition accidentelle d'erreur au niveau d'un périphérique. Elle entraine généralement sa mise hors service par le système d'exploitation.

Défaut (valeur par). Valeur optionnelle considérée par le système à défaut d'autre indication.

Défilement. Mouvement horizontal ou vertical d'une partie ou de la totalité de l'image à l'écran afin de faire apparaitre de nouvelles données. Dans les systèmes les plus complexes, dotés d'une mémoire vidéo spéciale, le défilement peut se faire dans les deux sens (on peut rappeler des rangées ou des colonnes déjà vues).

Délai d'attente. Temps nécessaire pour positionner la tête de l'unité de disque au-dessus de la surface en rotation.

Dépassement de capacité. Dans le cas d'un programme, cette expression signifie que la capacité de la mémoire a été dépassée.

Dépendance fonctionnelle. Il y a dépendance fonctionnelle quand un certain attribut d'une relation dépend d'un autre attribut de la même relation et que cette dépendance est univoque. On dit alors que le premier attribut identifie le second.

Déverminage, déverminer. Pour debugging et debugger. Voir **Bogue.**

Diagnostic. Opération par laquelle l'opérateur est informé des causes possibles d'une erreur. On reconnait deux types de diagnostic : celui du système et celui des programmes d'application. Le premier est géré par le système ou par les compilateurs et fait apparaitre les erreurs de forme ou de syntaxe. Le second doit être préparé par le programmeur en fonction de son application.

Dictionnaire. Liste de tous les types de données avec leurs noms et leurs structures.

Digital. Voir **Numérique**.

DIP. Dual In line Package. Boîtier dont les broches se trouvent sur deux côtés parallèles.

Directory. Voir **Répertoire**.

Disponibilité. Mesure l'exploitation d'un système sans défaillance.

Disque souple (floppy). Disque magnétique souple.

Disquette (mini disque). Ce terme désigne généralement les disquette de 5 pouces 1/4, d'une capacité de l'ordre de 160 Kockets et plus. Il peut également s'appliquer aux disques de 3 1/2 pouces adoptés par certains constructeurs de micro-ordinateurs professionnels.

Diviseur. Circuit logique à la sortie duquel est présentée

une impulsion pour toutes les « n » impulsions d'entrée. Il est généralement réalisé selon une configuration de compteur particulière.

DL/1. Langage particulier, implémenté sur machine IBM, pour la définition logique et physique des structures de données.

DMA (Direct Memory Access). Voir **ADM**.

Domaine. Ensemble des données du même type liées par une relation déterminée.
Espace sur disque réservé à un fichier.

Données. Informations à fournir avant le traitement. Toutes les données doivent être connues.

DOS (Disk Opérating System : système d'exploitation sur disque). C'est l'ensemble des programmes et sous-programmes, enregistrés sur disque, qui constituent un système d'exploitation. L'ensemble de ces programmes peut être divisé en deux groupes fondamentaux : les **transitoires** et les **résidents.** Les premiers ne sont chargés en mémoire qu'au moment de leur activation, tandis que les autres sont toujours résidents, par exemple, les modules de gestion des périphériques qui se retrouvent dans tous les programmes doivent être résidents.

Double précision. Méthode de mémorisation des données utilisant un espaces de mémoire deux fois plus grand que la normale afin d'utiliser plus de chiffres significatifs.

Drapeau. Voir **Indicateur**.

DRC (Design Rules Checking : contrôle des règles de conception). Procédé de contrôle des données servant à faire apparaître des types d'erreur prédéterminés, comme une incompatibilité des tolérances.

DTL (Diode Transistor Logic : logique à transistor et diode). Circuits obtenus en combinant des diodes et des transistors.

Duplex intégral. Mode de transmission simultané dans les deux sens. Est dit intégral par opposition au semi-duplex, système de transmission non simultanée. (Voir **Simplex**).

Dynamique. Qualifie un évènement se produisant en cours d'exécution d'un programme.

E

EAO. Enseignement Assisté par Ordinateur. Système d'enseignement autodidactique où l'ordinateur joue le rôle de professeur.

EBCDIC. Extended Binary Coded Decimal Interchange Code. Code spécial, parfois utilisé dans la transmission de données à la place du code ASCII, plus connu. Dans le code EBCDIC, chaque caractère est représenté par 8 bits. La première valeur est 0000 0000, qui représente l'espace en blanc (SP), tandis que la dernière est 1111 1001, qui correspond au caractère 0.

Echange. Processus au cours duquel deux variables échangent leurs contenus.

Echelle. Rapport entre les dimensions d'une image et celles de l'objet réel.

ECL. Emitter-Coupled Logic (Logique à couplage d'émetteurs). Famille particulière de circuits intégrés assez peu utilisée car difficilement interconnectables à d'autres.

Eclatement cellulaire. Technique particulière utilisée pour ranger les enregistrements ajoutés à un fichier. Les enregistrements sont placés dans une cellule qui est divisée en deux chaque fois qu'elle est remplie.

Echo. Retransmission de la donnée reçue. Par exemple, les caractères transmis par le clavier sont présentés à l'écran, en écho.

Ecrêtage. En électronique, il s'agit de l'élimination de la partie la plus élevée du signal. Dans les applications graphiques sur ordinateur, ce terme désigne parfois le procédé permettant d'éliminer une partie du dessin pour visualiser à l'écran des objets qu'il ne contiendrait pas autrement.

Edition. Modification de données. Ce terme est plus particulièrement utilisé pour désigner les opérations de correction (insertions ou modification) de programmes.

Edition. Opération de préparation des données pour l'impression ou pour l'opération de modification d'un programme.

Effet de bord. Il y a effet de bord quand une erreur générée dans une partie de programme par des causes déterminées réapparaît dans d'autres parties du programme.

Elément. Donnée particulière d'une liste ou d'un tableau.

Elimination des faces cachées (autohide). Option existant dans les systèmes graphiques les plus perfectionnés et permettant de visualiser une figure sous un angle particulier en effaçant les faces non visibles.

Emboîtement. Opération par laquelle une structure est englobée dans une autre. Ainsi, une boucle peut être emboîtée dans une autre ou un sous-programme imbriqué dans un autre.

En ligne (en direct). Qualifie un système dans lequel les données peuvent être entrées ou prélevées directement. C'est le contraire de hors ligne, ou différé, qui s'applique à des données accessibles uniquement après des actions particulières, comme la copie sur bande d'archives historiques.

Enregistrement. Ensemble d'informations liées entre elles, traitées comme une seule entité.

En-tête. Désigne généralement une partie initiale contenant la description ou la clé d'interprétation des parties suivantes. Ce terme est surtout employé pour désigner l'enregistrement initial d'un fichier contenant des informations sur ceux qui le suivent.

En tête (de gauche). Qualifie la partie initiale d'un champ de données. On peut, par exemple, parler des zéros en tête avant les valeurs numériques ou des espaces en tête avant les caractères d'une chaîne.

Entier. Nombre sans décimale.

Entrance. Terme désignant la charge produite par un circuit à son entrée. L'entrance donne un ordre de grandeur de la puissance nécessaire pour piloter ce circuit. (Voir **Sortance**).

EOF. End of File. Signal d'identification de la fin d'un fichier.

E/S. Entrée/Sortie. (anglais I/O). Désigne toutes les opérations d'entrée ou de sortie de données.

Escape. Code particulier utilisé dans de nombreux systèmes d'exploitation pour indiquer une action à exécuter. Correspond généralement à une touche spéciale.

Esclave. Dispositif faisant partie d'un réseau mais n'ayant pas une fonction dirigeante, par exemple un terminal vidéo.

Espace. Etat 0 (bas) lors de la transmission de données.

Etiquette (label). Terme désignant, dans de nombreux langages symboliques, le libellé d'adressage d'une instruction. Il peut également se référer au disque , dans ce cas, il s'agit de son nom.

Evénement. Action généralement extérieure pouvant provoquer une interruption de programme afin que des programmes particuliers soient lancés. Dans certains ordinateurs, des tests fonctionnels, gérés sous interruption, produisent de tels événements. Ces tests fonctionnels peuvent également être gérés par programme, auquel cas il n'y a pas d'interruption mais intégration continue de l'état des touches par le programme.

Exécution. Réalisation d'une instruction. Désigne le cycle d'exécution d'une instruction qui suit généralement l'extraction de son code en mémoire.

Extraction. Cycle de prélèvement en mémoire du code de l'instruction qui doit être ensuite exécutée.

F

Faces cachées. Plans à effacer parce que couverts par d'autres parties dans les graphiques d'objets en trois dimensions (3D).

Facteur de groupage. Nombre maximal d'enregistrements prévus dans un bloc. Est généralement utilisé en Cobol.

Facteur d'échelle. Valeur de multiplication des dimensions d'un dessin pour le ramener aux dimensions de l'écran ou du facteur.

FAO. Fabrication Assistée par Ordinateur. Système de conception et de préparation des données pour le fonctionnement des machines à commande numérique assistées par ordinateur. C'est un stade ultérieur de la CAO.

Fenêtre. Zone rectangulaire de l'écran sélectionnée par l'opérateur.

Fenêtre. En informatique graphique, et dans certains cas également en Mode texte, désigne une zone particulière de l'écran gérée indépendamment du reste, comme s'il s'agissait d'un écran séparé. En Pascal, désigne un ensemble d'identificateurs formant le « window diagram ».

FF. Form Feed. Caractère de commande (voir code ASCII) provoquant le saut à la page suivante.

Fichier circulaire. Fichier particulier dans lequel les nouveaux enregistrements remplacent les précédents.

Fichier hiérarchique. Fichier dans lequel certains enregistrements dépendent d'autres enregistrements selon une structure arborescente.

Fichier inversé. Structure particulière d'un fichier. Elle permet de rechercher des informations n'ayant pas été décrites précédemment. Un fichier inversé est obtenu à partir de listes d'indices inversées.

Fichier logique. Structure d'un fichier tel que le programme d'application le voit. Elle peut être totalement différente de sa structure physique.

Fictif. Qualifie une valeur utilisée uniquement pour suivre des règles formelles déterminées, sans être prises en compte. Par exemple, un argument d'une fonction peut être fictif s'il n'est pas réellement utilisé pour effectuer des calculs mais seulement pour respecter une règle formelle (en réalité, l'argument fictif est utilisé par le système, mais pas en tant que valeur).

File. Voir **Fichier**.

File d'attente. Liste d'éléments en instance de traitement. Il peut s'agir, par exemple, de la liste des demandes de service de plusieurs périphériques ou de procédures à exécuter.

Filtre. Composant permettant d'éliminer une grandeur en gardant les autres inchangées.

Flag (Drapeau). Voir **Indicateur**.

Floppy. Voir **Disque souple**.

Fonction. Procédure qui, à partir d'autres valeurs (paramètres) et d'un algorithme donné, rend une valeur dépendante. Les fonctions déjà prévues dans un langage (par exemple les fonctions mathématiques du Fortran) sont dites intrinsèques.

Foreground. Voir **Premier plan**.

Format. Forme de sortie ou d'entrée de données. Ce terme désigne également, parfois, la structure de mémorisation sur disque. A ne pas confondre avec le « formatage » du disque tout entier.

Formatage. Opération consistant à « préparer » le support (disque, par exemple), selon les spécifications de chaque système d'exploitation.

FORWARD. En Pascal, mot réservé, utilisé dans les déclarations de certaines procédures particulières.

Fréquence. Indique le nombre de fois par seconde qu'un signal prend la même valeur ; c'est l'inverse de la période.

FSK (Frequency Shift Keying). Méthode de modulation des signaux consistant à modifier la fréquence d'une onde porteuse en fonction des valeurs du signal modulant.

Fusion. Procédure selon laquelle deux listes en produisent une seule. Elle peut également s'appliquer à deux programmes.

G

Gamme. Limites entres lesquelles se trouve la valeur d'un élément donné. Par exemple, la gamme des nombres entiers d'une machine 8 bits va de - 32 000 à + 32 000.

Groupage. Combinaison de deux blocs (ou plus) de manière à faciliter leur lecture/écriture avec une seule instruction.

Guide-opérateur. Se dit d'un message ou d'un symbole utilisé par l'ordinateur pour guider l'utilisateur.

H

Haut niveau (évolué). Qualifie les langages symboliques orientés utilisateur.

Hertz. Unité de mesure de la fréquence. Ses multiples sont : kHz (kilo-Hertz = 1 000 Hz), MHz (méga-Hertz = 1 000 000 Hz), GHz (giga-Hertz = 1 000 MHz). Il n'a pas de sous-multiples.

Heuristique. Méthode de résolution d'un problème qui consiste à effectuer différents essais et à en analyser ensuite les erreurs.

Hiérarchie. Structure d'éléments subdivisée en différents niveaux ayant plusieurs degrés d'importance ou de priorité. La hiérarchie des opérations, par exemple, établit les règles de priorité de leur exécution arithmétique. Pour ce qui concerne les fichiers ou les structures de données, ce terme indique une subdivision et les lignes de dépendance de tous les éléments de niveau inférieur à un autre et liés à celui-ci.

Histogramme. Graphique à barres. La longueur de chaque barre représente la valeur de la variable.

Horloge. Composant capable de produire un signal périodique de synchronisation. Chaque signal est une impulsion d'horloge ; il est caractérisé par une amplitude et une période. L'amplitude est la valeur maximale atteinte par le signal à l'état actif, tandis que la période indique sa durée en secondes. On utilise aussi parfois la caractéristique de fréquence (mesurée en Hz ou MHz) dont la valeur numérique l'inverse de la période : elle représente le nombre d'impulsions émises en une seconde.

Hors ligne (en différé). Se dit d'un appareil sous le contrôle direct de l'ordinateur dirigeant le système. Quand elle s'applique à un traitement, cette expression signifie qu'il est exécuté dans le flot principal.

Hôte. Ordinateur central d'une installation à plusieurs machines (réseau).

Huffman. Voir **Code Huffman**.

I

Identificateur : Nom donné à un programme ou à une de ses parties.

Incrément. Valeur modifiant un compteur, comme le STEP (pas) dans les instructions FOR... du basic ou DO... du Fortran.

Indexé. Mode d'adressage dans lequel une position est obtenue en additionnant le contenu du registre d'index à l'adresse contenue dans l'instruction.

Indicateur (flag). S'utilise en un point donné d'un programme pour mémoriser un état ou un événement particulier. Certains langages, comme le RPG, utilisent beaucoup les indicateurs pour déterminer par exemple le type d'enregistrement ou indiquer le résultat d'une comparaison. Dans ces langages, on utilise généralement plutôt le terme de code, mais les fonctions ou la logique d'utilisation sont les mêmes. Les opérations activées en fonction de l'état de ces indicateurs sont dites « en rupture de codes ».

Indicateur. C'est un registre élémentaire qui permet d'afficher l'état d'un évènement. Il indiquera, par exemple, les débordements, le signe...

Indice. Valeur qui détermine la position d'un enregistrement ou un élément dans un tableau.

Inductance. Concerne un circuit constitué d'un certain nombre de bobines. Elle se mesure en (micro-Henry).

Infini machine. Le plus grand nombre qui peut être représenté selon le format interne d'un ordinateur.

Informatique répartie. Technique permettant d'obtenir un traitement simultané de données sur différents systèmes connectés en un réseau.

Initialisation. Dans un programme, affectation des valeurs initiales aux indicateurs, compteurs, chaînes...

Instable. Qualifie un circuit n'ayant pas d'état stable. L'exploitation de cette caractéristique permet d'utiliser ce type de circuit comme générateurs.

Instruction. Ensemble de mots réservés et de données formant une commande et indiquant une action particulière à exécuter par l'ordinateur. Un programme se compose d'une suite d'instructions (voir également **Mot réservé** et **Mot-clé**).

Instruction composée. Instruction complexe non prévue dans le langage et obtenue en réunissant des instructions élémentaires. Le bloc obtenu doit normalement commencer et finir par des mots-clés particuliers. Les instructions composées n'existent que dans certains langages évolués.

Intégré. Qualifie une procédure faisant partie intégrante du système d'exploitation.

Intégrité. Caractéristique indiquant l'absence d'erreurs dans un ensemble de données.

Interactif. Se dit d'un programme établissant un dialogue avec l'utilisateur (voir **Conversationnel**).

Interface. Dispositif qui adapte les signaux aux exigences de différents circuits. Dans les communications entre plusieurs ordinateurs (ou entre un ordinateur et des périphériques), les signaux doivent être adaptés tant à l'émission qu'à la réception.

Interpréteur. Programme capable d'interpréter et d'exécuter un programme source.

Interpréteur. Programme spécifique chargé de traduire un langage symbolique en codes exécutables. Contrairement au compilateur, l'interpréteur effectue la traduction et l'exécution instruction par instruction. La traduction d'une instruction n'étant pas mémorisée, une opération donnée doit être exécutée autant de fois que l'instruction est répétée.

Interruption. Suspension du programme en cours d'exécution, avec sauvegarde de tous les paramètres, de telle sorte que l'exécution puisse reprendre ultérieurement à partir du point d'interruption. L'interruption est dite « armée » quand le système, à l'état de validation, mémorise la demande. Elle est dite « traitée », ou « prise en charge » lorsque le système a activé un sous-programme spécial chargé d'exécuter les tâches demandées par l'interruption.

Intersection. Union de deux ou de plusieurs groupes de données, obtenue en prélevant seulement les éléments communs à ces groupes.

Intersection. Création d'un ensemble de données à partir de deux autres. Il regroupe tous les éléments communs aux deux premiers, et exclusivement ceux-là.

Invalidation. Etat interdisant certaines fonctions. Par exemple, quand les interruptions sont invalidées, le programme ne peut pas être interrompu (voir **Validé**).

Inversé. L'opposé d'un état quelconque. En parlant d'affichage vidéo, ce terme qualifie un libellé en noir sur fond blanc, généralement obtenu en éteignant les pixels composant les lettres. Normalement, c'est le contraire : l'écran est éteint et les lettres sont obtenues en activant certaines zones).

Inversé (fichier). Voir **Fichier inversé**.

Invitation à émettre. Phase de la transmission de données pendant laquelle le maître invite un esclave à envoyer ses données.

I/O. Voir **E/S**.

IPS. Inches Per Second. Unité de mesure de la quantité de ruban (exprimée en pouces) pouvant être traitée (lue ou écrite) en une seconde.

ISAM. Index Sequential Access Method. Méthode d'accès aux données d'un fichier à l'aide d'indices.

Itératif. Se dit d'une méthode de calcul consistant à chercher une solution par des passages successifs.

J

Jeu. Ensemble de symboles reconnus dans un langage donné ou par un ordinateur.

Jeu d'essai. Ensemble d'instructions utilisé pour contrôler la qualité d'un logiciel.

Joystick. Voir **Manche à balai**.

Jump. Voir **Saut**.

Justification. Opération de déplacement des valeurs à l'intérieur d'une zone ayant pour résultat de mettre tous les espaces du même côté. D'une façon générale, les valeurs numériques sont justifiées à droite, les éventuels espaces étant placés à gauche des valeurs, tandis que les données alphanumériques sont justifiées à gauche.

K

K. Abréviation de kilo, et plus généralement symbole du facteur de multiplication 1 000. Dans le cas des mémoires d'ordinateur, un K vaut 1 024.

L

Label. Voir **Etiquette**.

Langage machine. Ensemble des codes numériques que l'ordinateur interprète et exécute directement, c'est-à-dire sans passer par des traductions comme avec les langages dits évolués.

LF. Abréviation de Line Feed. Caractère de commande qui provoque le déplacement d'une ligne, mais sans changement de colonne.

LFU. Least Frequently Used (le moins fréquemment utilisé). Algorithme consistant à remplacer les données les moins utilisées par de nouvelles (voir également **LRU**).

Ligne. Donnée permettant de définir une position horizontale (par opposition à la colonne). On peut également parler de rangée.

Ligne. Quand il se rapporte à une sortie, ce terme désigne un ensemble de données contiguës et terminées par un CR. En entrée, il s'agit d'un groupe de données considérées comme un seul bloc (par ex. une chaîne). En transmission, il désigne tout moyen permettant d'envoyer des signaux : câbles électriques ou fibres optiques (pour la transmission des données sous forme de signaux lumineux).

Listing. Copie, sur papier ou à l'écran, des instructions constituant un programme.

Logiciel. Programmes, sous-programmes et, d'une façon générale, tout de qui n'est pas matériel. Traduction de Software.

Longueur fixe (fichier). Qualifie généralement un type particulier de fichier dans lequel tous les enregistrements ont la même longueur, par exemple les fichiers à accès direct en Basic.

Loop. Voir **Boucle**.

Lots. Voir **Traitements par lots**.

LRU. Least Recently Used (le moins récemment utilisé). Algorithme dit d'ancienneté consistant à remplacer les données utilisées le plus récemment par de nouvelles données (voir aussi **LFU**).

LSI. Large Scale Integration. Désigne des composants ayant un niveau élevé d'intégration. Il s'agit généralement de circuits composant un système tout entier.

M

M. Abréviation de Méga (10^6).

Macro. Groupe de commandes ou d'instructions considérées par l'ordinateur comme une seule instruction. Elles sont particulièrement utilisées en Assembleur ou dans certains progiciels comme les tableurs.

Macro-instruction. Instruction source qui génère une série d'actions équivalentes à un certain nombre d'autres instructions.

Manche à balai (joystick). Mécanisme constitué d'un levier et de transducteurs de position. Ses déplacements permettent de commander la position du curseur ou d'un dessin à l'écran.

Mantisse. Dans la représentation des nombres en virgule flottante, désigne la partie numérique qui doit être multipliée par l'exposant. Par exemple, le nombre 5 600 écrit sous la forme $5.6 \times E^3$ a pour mantisse 5.6 et pour exposant 3.

Marque. Etat 1 (haut) pendant la transmission de données.

Masque. Ensemble de caractères utilisé pour contrôler l'acquisition ou l'élimination de certains caractères appartenant à un autre ensemble. Le masque peut être représenté par un seul caractère ; dans ce cas, il sert à la sélection des bits individuels.

Matériel. Partie physique d'un système de traitement.

Matrice. Tableau à deux dimensions ou plus.

Mémoire associative. Mémoire dont les positions sont identifiées d'après leur contenu et non par leurs adresses. Elle permet d'augmenter sensiblement la vitesse de recherche (à l'aide des attributs du champ considéré).

Mémoire-tampon. Zone de mémoire utilisée pour stocker temporairement des données. On recourt généralement au tampon lors d'opérations d'E/S pour servir de « poumon » entre deux dispositifs échangeant des données à des vitesses différentes. En impression, par exemple, la présence d'un tampon permettra à l'ordinateur d'envoyer une certaine quantité de données et de se consacrer à d'autre tâches pendant que l'imprimante imprime ces données.

Menu. Liste des fonctions prévues dans un programme donnant la possibilité d'en sélectionner une.

Méthode d'accès. Technique de transfert de données entre un ordinateur et un périphérique. Les principales sont les suivantes : accès série, direct (aléatoire), à distance, séquentiel virtuel (VSAM) et indexé hiérarchique séquentiel (HISAM). Ces deux dernières sont caractéristiques des grands systèmes.

Méthode des éléments finis. Méthode de calcul utilisée essentiellement en génie mécanique.

Micrologique (microprogrammes). Fonctions logicielles mémorisées de façon permanente, par ex. sur ROM.

Micro-ordinateur. Ordinateur basé sur des circuits

intégrés, voire parfois sur un circuit unique.

Migration. Opération de transfert des données dans une zone permettant d'augmenter la vitesse d'accès.

Mini-ordinateur. Ordinateur basé sur des circuits intégrés et, quelquefois, sur un seul circuit. Il est vrai que les micro-ordinateurs professionnels ont de plus en plus la puissance d'un mini.

Mise à jour. Opération de modification des données, généralement dans un fichier.

Mise au point. Opération de recherche et d'élimination des erreurs (voir déboguer).

Mode d'accès. Technique utilisée dans la lecture ou pour l'écriture d'un enregistrement logique d'un fichier. Les principaux sont les suivants : le mode séquentiel, dans lequel les enregistrements sont traités l'un après l'autre, et le mode direct, où l'on accède uniquement à l'enregistrement désiré.

Mode point (graphique). L'expression graphique en mode point (bit map graphics) désigne la représentation prenant en compte l'état de chaque point de l'écran.

Modèle. Les programmes qui simulent un certain phénomène ou un processus déterminé utilisent un modèle pour simuler le comportement du système réel.

Modèle conceptuel. Schéma d'ensemble d'une base de données.

Modem. Abréviation de Modulateur-Démodulateur. Appareil utilisé pour moduler et démoduler les signaux dans les transmissions à grande distance.

Module. Partie d'un programme logiquement séparée du reste et exécutant des fonctions définies et autonomes.

Moniteur. Synonyme d'écran.

Monitoring. Ce terme est parfois utilisé pour désigner l'opération par laquelle un programme présente l'état des variables tout en poursuivant le traitement.

Morgan. Voir **Théorème de Morgan**.

Mot. Ensemble constitué d'un nombre de bits égal à celui pouvant être inséré dans une adresse mémoire. Les longueurs les plus usitées sont : 8, 16 et 32 bits.

Mot-clé. Mot prédéfini et réservé, dans le langage de programmation utilisé, à un emploi particulier.

Mot réservé. Terme utilisé dans un langage (ou une commande) pour activer une fonction déterminée. Il ne peut généralement pas être utilisé à une autre fin, par exemple comme nom de variable.

MSI. Medium Scale Integration. Technologie particulière qui permet de contenir dans un seul boîtier un nombre de composants supérieur à celui de la TTL mais inférieur à celui de la LSI.

Multiplexeur. Dispositif permettant d'établir un dialogue entre plusieurs unités en utilisant un nombre réduit de liaisons. Deux types sont utilisés : temporel et fréquentiel.

Multipoint. Type de configuration de réseau dans lequel un maître dirige l'échange avec différents esclaves.

Multiprocesseur. Ordinateur utilisant simultanément plusieurs unités centrales.

N

Nœud. En parlant d'un réseau, il s'agit du point terminal d'une branche ou de la connexion de plusieurs dérivations.

Nœud. Point de branchement dans un organigramme. Ce terme est principalement employé en programmation structurée et désigne un point de branchement d'un diagramme en arborescence.

Nœud terminal. Terme utilisé, en programmation structurée, pour désigner un bloc n'ayant pas de branchement vers le bas.

Normalisation. Alignement de chiffres composant un nombre pour l'obtenir dans la forme requise par les règles arithmétiques (syntaxe).

Notation. Forme de représentation des données. En informatique, on utilise la notation binaire, octale, hexadécimale...

Nul. Qualifie un mot en blanc qui, d'une façon générale, ne génère aucune action. Il est intercalé entre des mots contenant les données des transmissions synchrones.

Numérique. Qualifie un dispositif ne pouvant reconnaître que des états définis et non pas les valeurs intermédiaires. Les deux états possibles sont généralement identifiés par 0 et 1. Noter cependant que certains dispositifs électroniques sont à trois états ; dans ce cas, tout de même, le troisième n'est pas un état logique mais seulement une condition électrique particulière dans laquelle le dispositif est comme branché.

Numériseur. Appareil permettant de convertir des coordonnées en une forme numérique compréhensible pour l'ordinateur.

O

Octal. Qualifie un système de numération utilisant la valeur 8 comme base.

Octet. Bloc de 8 bits. En binaire, un caractère est contenu dans un octet.

Off-line. Voir **Hors ligne**.

Offset. Désigne une valeur indiquant le décalage d'une donnée par rapport à une origine. Dans un fichier, il peut s'agir du nombre d'octets à partir du début d'un enregistrement, ce qui donne la position d'une donnée. Dans le cas de la mémoire, c'est une valeur constante à déduire d'une adresse pour connaître une position.

Offset. En électricité, désigne une tension constante à partir de laquelle le signal est appliqué.

On-line. Voir **En ligne**.

Opérande. Elément sur lequel doit être effectuée une opération.

Opérateur. Symbole particulier indiquant des opérations déterminées à exécuter sur les opérandes. Les principaux opérateurs sont de deux types : arithmétique et logique.

Opération. Toute action qui, appliquée à une combinaison d'éléments connus, génère un nouvel élément appartenant au même ensemble.

Ordinateur central. Elément central de traitement dans une configuration de grandes dimensions (voir **Hôte**).

Ordre. Indique la logique utilisée pour former une liste de données particulière. En parlant d'un bit, il s'agit de sa position à l'intérieur du mot.

Organigramme. Représentation graphique des fonctions à exécuter dans un programme.

Oscillateur. Circuit capable de produire un signal périodique (voir **Horloge**).

Overflow. Voir **Dépassement de capacité**.

Overlay. Voir **Zone de débordement**.

P

Page. Zone de mémoire généralement composée de 1 024 positions. Dans certains systèmes, il faut distinguer la page zéro et la page en cours. La page zéro contient des informations de chaînage utilisées par le système, alors que la page en cours est la zone de mémoire contenant l'instruction en cours d'exécution.

Page active. Dans certains systèmes, l'unité vidéo peut utiliser plusieurs pages de mémoire. La page active indique celle qui contient des informations. Elle est généralement différente de celle présentée à l'écran.

Pagination. Technique de segmentation utilisée par les systèmes d'exploitation les plus évolués et consistant à diviser les programmes ou les données en pages normalement résidentes sur disque et chargées en mémoire uniquement au moment opportun. C'est une façon d'augmenter l'utilisation de la mémoire par rapport à sa vraie capacité (la même zone est partagée, à des moments différents, par plusieurs groupes de données).

Paliers (par). Technique de développement d'un programme consistant à obtenir le résultat final par étapes de plus en plus approfondies et précises.

Paquet. Ensemble de valeurs constituant un bloc de données.

Paramètre. Nom symbolique utilisé pour transférer des valeurs, comme argument d'une fonction, lors de l'appel d'un sous-programme, ou encore entre procédures. En voici quelques types : paramètres effectifs, paramètres formels, paramètres à appel par nom, paramètres à appel par valeur.

Parité. Etat des données à contrôler avant leur transmission. Ce contrôle consiste à vérifier le nombre d'états ON en envoyant, en même temps que la donnée, un bit indiquant si ce nombre doit être pair ou impair. La parité paire est la logique selon laquelle le bit de parité est actif si le nombre de bits ON est pair, alors que dans le cas de la parité impaire (ou imparité), le bit de parité n'est actif que quand ce nombre est impair.

Pas à pas. Type de déroulement d'un programme. On recourt à la technique pas à pas (anglais step by step) pour effectuer des recherches d'erreurs. Toutes les machines n'offrent pas cette possibilité.

Passation. Opération de transfert des paramètres entre un programme (ou un sous-programme) appelant et un sous-programme ou une fonction appelé(e).

Période. Intervalle de temps entre deux points d'un signal de même amplitude.

Périphérique. Tout dispositif utilisé par l'ordinateur pour communiquer avec son environnement.

PEP (Parametric Element Processor). Langage évolué, utilisé pour résoudre des problèmes de graphique en trois dimensions (3D).

Phase. Représente l'origine d'un signal périodique.

Photostyle. Dispositif sensible à la lumière utilisé pour détecter un point de l'écran d'après sa luminosité.

Piping. Terme utilisé en informatique graphique (infographie) pour désigner la conception de tuyaux, lignes électriques ou éléments linéaires en général.

Piste. Division circulaire du disque, divisée en secteurs.

Pixel. Un point de l'écran (PICture Element) en mode graphique. Il peut également s'agir du bit précis de la zone mémoire contenant l'information sur l'état du point écran correspondant (voir **Point**).

Plaquette de circuits imprimés (PCB, Printed Circuit Board). Support isolant, par exemple la vétronite, sur lequel est gravé, à l'aide de procédés

photographiques, l'ensemble des liaisons à réaliser.

Plot. Petite zone d'un circuit imprimé sur laquelle sont soudés les composants.

Poids. Valeur multiplicative d'un chiffre dépendant de sa position dans un nombre écrit selon un système de numération positionnel. Par exemple, le 7 du nombre 275 a le poids 10.

Point. Désigne en général un point de l'écran (synonyme de pixel).

Point adressable. En informatique graphique, l'écran est divisé en un grand nombre de points formant une trame. Un dessin se construit en activant certains de ces points. D'une façon générale, l'écran ne peut pas être géré dans sa totalité et les zones activables sont appelées « points adressables ».

Point de reprise. Utilisé avec une méthode particulière consistant à écrire les paramètres d'un programme dans un fichier au fur et à mesure de son exécution. Si le programme s'interrompt pour des causes accidentelles, il peut être relancé à partir du point défini lors du dernier enregistrement (point de reprise). Cette technique est utilisée dans les procédures demandant un temps de traitement très long afin d'éviter de devoir repartir du début en cas d'interruption imprévue.

Pointeur. Variable dont la valeur indique un élément particulier d'un ensemble. Il s'agit soit d'un élément d'un tableau ou d'un enregistrement, soit d'un octet déterminé à l'intérieur de l'enregistrement.

Pointeur intégré. S'utilise dans un type de gestion particulière qui prévoit que les pointeurs se trouvent dans l'enregistrement des données auxquelles ils donnent accès et non pas dans le répertoire.

Port. Voir **Porte**.

Porte (d'accès). Point d'accès, c'est-à-dire d'entrée ou de sortie, des données. Les portes sont dites numériques quand elles permettent le transfert de signaux numériques (ON/OFF), ou analogiques. Dans ce dernier cas, elles doivent être équipées d'un convertisseur analogique/numérique.

Position. Désigne une position en mémoire ou un espace sur disque. Plus généralement désigne toute zone servant à la mémorisation d'une donnée.

Position. Position repérable à l'aide d'une valeur numérique. Par exemple, un enregistrement au début d'un fichier ou un caractère dans une chaîne.

Postprocesseur. Programme capable d'interpréter des données en sortie d'un autre logiciel afin de les convertir en un format adapté à des usages ultérieurs, par exemple pour produire des dessins avec un traceur de courbes.

Précision. Désigne, d'une façon générale, la capacité de relever des différences.

Précision. Indique la « qualité » de la machine. Elle est donnée par la différence entre la valeur réelle d'un nombre et celle que la machine est capable de gérer. Par exemple, dans une machine utilisant 9 chiffres significatifs (pour les nombres réels), la précision est d'environ $1/1\ 000\ 000\ 000^e$.

Premier plan. Zone de l'écran occupée par un dessin ou par un caractère.

Préparation. Voir **Pré-traitement**.

Pré-traitement. Méthode de traitement des données permettant de les rendre conformes aux exigences d'un logiciel particulier (synonyme de **Préparation**).

Procédure. Ensemble des programmes exécutant une tâche particulière. Le sens de ce terme est différent en Pascal : il désigne une partie de programme qui peut être exécutée ; il est alors synonyme de sous-programme.

Progiciel : Logiciel complexe.

Programmation structurée. Méthode logique de développement d'un programme qui consiste à diviser le problème en plusieurs problèmes de moindre importance et à résoudre chacun d'eux à l'aide d'un module conceptuellement séparé des autres.

Programme d'application. Programme écrit en vue de résoudre un problème particulier.

Protégé. Se dit d'une zone mémoire à accès contrôlé. Peut également qualifier une disquette. Dans ce cas, il indique la présence ou l'absence du dispositif matériel permettant l'écriture (la protection est généralement réalisée en couvrant, ou dans certains cas en découvrant, une encoche de la disquette).

Protocole. Ensemble de règles concernant l'échange de messages entre deux unités.

Puce. Synonyme de circuit intégré.

Puck. Dispositif manuel servant à entrer des coordonnées.

R

Racine Valeur mathématique particulière. En programmation structurée, c'est le nœud de départ d'un organigramme en arbre.

Rafraîchissement (régénération). Méthode de maintien de l'image vidéo consistant à la retracer à intervalles réguliers avant qu'elle ne disparaisse.

RAM (Random Access Memory). Mémoire vive : mémoire à accès direct, ou sélectif, également dite mémoire vive.

Randomisation (algorithme de). Algorithme de calcul des nombres aléatoires.

Rappel arrière. Déplacement en arrière d'un caractère. Une touche d'espacement arrière est prévue dans de nombreux claviers. Quand elle n'existe pas, cette fonction est exécutée par une combinaison particulière d'autres touches.

Raster display. Voir **Balayage récurrent**.

Read only. Expression qualifiant un dispositif, le plus souvent des mémoires, qui ne peut être utilisé qu'en lecteur, donc non modifiable.

Recherche. Examen, par prélèvement d'un élément d'une liste, si satisfaction de conditions déterminées.

Recherche dichotomique. Algorithme particulier utilisé pour accélérer la procédure de recherche. L'ensemble à examiner est divisé en deux à chaque phase de la recherche.

Récursif. Qualifie une procédure telle que chaque pas utilise les résultats du précédent. Exemple, la technique permettant la solution d'une équation par des approximations successives, ou des types de programmes particuliers (également dits « réentrants »), qui peuvent s'appeler eux-mêmes.

Recouvrement. Utilisation de la même zone de mémoire par plusieurs programmes à tour de rôle. Dans les petits systèmes, le recouvrement est géré par le programme d'application ; dans les systèmes plus évolués, il est pris en charge par le système d'exploitation en fonction de l'activité en cours.

Réel. Adjectif qualifiant une variable ou une constante appartenant à l'ensemble des nombres réels.

Relationnelle : Se dit d'une base de données construite en fonction des relations de ses éléments. Sa gestion demande l'utilisation d'un logiciel capable de traiter ses éléments de façon à produire de nouvelles relations, ce qui donne une grande souplesse d'emploi au système.

Remark. Voir **Commentaire**.

Remplissage. Bourrage d'un bloc avec des données factices comme des zéros ou des espaces.

Répertoire (Directory). Table d'identification ou de correspondance d'un ensemble de données. Le répertoire d'une disquette est la zone de ce disque réservée au système. Elle contient le nom de chacun des fichiers et sa position (et d'autres informations spécifiques à chaque système).

Réseau. Ordinateurs connectés entre eux.

Résistance. Composant d'un circuit offrant une résistance au passage du courant. Celle-ci se mesure en Ohms et ses multiples (K=x1 000, M=x1 000 000).

Résolution. Quantité minimale mesurable entre deux points de l'écran. En informatique graphique, elle se mesure d'après le nombre de lignes comprises dans l'unité de longueur. En parlant des transducteurs, il s'agit de la plus petite quantité de la grandeur considérée pouvant engendrer un signal exploitable.

Retracer. Reconstruire un dessin pour y ajouter les dernières modifications.

Return. Fonction de retour dans le programme au terme de l'exécution d'un sous-programme. Dans de nombreux langages, c'est aussi le mot-clé de cette fonction.

RGB. Abréviation désignant un écran couleur fonctionnant selon la technique additive (Red, Green, Blue : Rouge, Vert, Bleu).

ROM. (Read Only Memory). Mémoire morte : mémoire à lecture seule (voir également **Read Only**).

Router. Logiciel capable de déterminer automatiquement les interconnexions des composants d'un circuit imprimé.

Routine. Programme.

RS 232. Sigle désignant une interface de communication utilisée dans les liaisons de transmission de données entre ordinateurs ou entre un ordinateur et des périphériques.

RTL (Resistance Transistor Logic). Logique obtenue en combinant des résistances et des transistors.

S

Saut. Rupture dans le traitement séquentiel normal.

Sauvegarde. Copie des données destinée à être utilisée pour une éventuelle reconstitution en cas d'erreur ou de panne.

Scalaire. Qualifie une grandeur non orientée (à l'inverse d'une grandeur vectorielle) ou ne figurant pas dans un tableau.

Schéma canonique. Modélisation d'un ensemble de données représentant la structure de ces données, indépendamment de l'usage qui en sera fait et du matériel ou du logiciel qui les exploitera.

Schéma d'implantation. Dessin représentant une entité physique grandeur nature.

Scissor (programme de découpage). Logiciel de division de graphiques en parties plus petites dans un écran.

Secteur. Quantité minimale adressable par la mémoire de masse.

Segment. Désigne une zone mémoire définie comme une extension, par exemple de 64 Koctets.

Sélection. Phase de transmission des données au cours de laquelle le maître invite un esclave à recevoir (on dit aussi « invitation à recevoir »).

Sémantique. Ensemble de règles de description d'un langage de programmation.

Semi-duplex. Méthode de transmission dans les deux sens en alternance. C'est la méthode la plus utilisée pour relier un écran ou un clavier, et l'unité centrale.

Séparateur. Caractère de séparation entre des valeurs. Dans certains langages, la virgule, par exemple, servira de séparateur lors de la saisie des données.

Séquentiel. Se dit d'un type de fichier dans lequel les enregistrements sont mémorisés par ordre croissant d'après le contenu d'un champ particulier de la clé.

Série. Expression contenant un ensemble d'éléments produits à l'aide de règles mathématiques précises.

Signal. Impulsion électrique indentifiable par l'ordinateur.

Signal d'horloge. Signal, de forme généralement carrée, par rapport auquel les composants d'un système sont synchronisés.

Simplex (unidimensionnel). Se dit d'une méthode de transmission de données. Un « canal » ou une voie simplex ne permet qu'un sens de communication, par exemple de l'ordinateur au périphérique, ou inversement.

Software. Voir **Logiciel**.

Sortance. Terme désignant la capacité produite par un circuit, c'est-à-dire l'ordre de grandeur de la puissance que ce circuit peut délivrer. Elle s'exprime généralement par le nombre de circuits analogiques qu'il peut piloter. (Voir **Entrance**)

Sous-chaîne. Série de caractères extraite d'une chaîne.

Sous-programme. Partie d'un programme contenant des instructions utilisables par différentes autres parties y accédant à l'aide d'un saut avec retour (Return).

Structure. Terme désignant une série d'éléments ayant des liens logiques.

Structure de recherche. En Pascal, il s'agit d'une chaîne qui devra être comparée à d'autres pour déterminer si elle est contenue dans celles-ci.

Syntaxe. Ensemble de règles indiquant comment doivent être utilisées les instructions d'un langage déterminé. Elles peuvent être représentées synthétiquement à l'aide de diagrammes syntaxiques.

Système d'exploitation. Logiciel contrôlant l'exécution des programmes d'application, y compris des éventuels interpréteurs ou compilateurs, et les programmes de gestion des périphériques (voir **DOS**).

T

Table de correspondances. Désigne la description des liens de dépendance entre les différents enregistrements de fichiers.

Tableau. Ensemble d'éléments, à une ou à plusieurs dimensions, désigné par un nom symbolique et par un ou plusieurs indices. Un tableau à deux dimensions, par exemple, est une matrice dont chaque élément est défini par sa ligne et sa colonne.

Tableau bidimensionnel. Tableau de données à deux dimensions.

Tambour. Dans le cas des traceurs, le tambour est le dispositif sur lequel défile le papier. Le tambour peut aussi être un support de mémoire de masse.

Tampon. Voir **Mémoire-Tampon**.

Taux de succès. Exprime le nombre d'enregistrements que l'on pense devoir être traités en un passage. C'est généralement un pourcentage de la dimension du fichier (en nombre d'enregistrements).

Temps d'exécution. Période pendant laquelle un programme est activé.

Temps partagé. Méthode d'utilisation d'une même mémoire par plusieurs programmes.

Temps réel. Une application en temps réel est une tâche dont les actions, après la phase d'entrée, sont immédiatement exécutées.

Terminal. Dispositif d'entrée/sortie dans un réseau de télécommunications.

Théorème de Morgan. Théorème utilisé dans l'analyse des circuits numériques pour convertir des schémas à logique négative et vice versa.

Time sharing. Voir **Temps partagé**.

Topographie. Schéma ou description du contenu de la mémoire décrivant l'utilisation de ses différentes zones.

Total de contrôle. Totalisation de certaines zones des enregistrements d'un fichier afin d'obtenir une valeur numérique (sans signification en tant que donnée) permettant de contrôler qu'aucune donnée n'a été perdue ou modifiée. Une technique analogue est utilisée en transmission ; dans ce dernier cas, on parle parfois de procédure d'« apurement ».

Tracé élastique. Technique de génération d'une ligne consistant à en fixer une extrémité et à tracer l'autre en fonction des déplacements d'un périphérique d'entrée, par exemple un manche à balai.

Traitement par lots. Mode particulier d'exécution

d'un programme selon lequel le traitement est différé par rapport au moment de l'introduction des données.

Transaction. Opération provoquant la création ou la suppression d'un enregistrement.

Transfert anticipé. Technique visant à accélérer l'accès aux mémoires de masse ou, en général, à des périphériques lents. Elle consiste à déplacer des blocs entiers de données vers un périphérique plus rapide, avant même que le programme ne les demande.

Transfert sur demande. Procédé consistant à transférer un bloc de données d'un périphérique lent dans un périphérique rapide quand le programme veut l'utiliser. C'est le contraire du transfert anticipé.

Tri « à bulle » (par permutation de paires). Technique consistant à exécuter un tri, dans un tableau, par sélection d'une partie des éléments du tableau et par échange de positions. On obtient ainsi le classement désiré. Ce processus est répété jusqu'à ce que tout le contenu du tableau soit trié. Le nom « à bulle » (de l'anglais Bubble Sort) provient de la technique utilisée, qui consiste à prélever au fur et à mesure les valeurs les plus petites, c'est-à-dire les plus « légères », qui « montent » vers le sommet de la liste. Une version particulière du Bubble Sort, plus rapide si un tri partiel a déjà été effectué, consiste à considérer simultanément une valeur située vers le sommet de la liste et une valeur vers la fin de la liste. Elle est appelée « Cocktail Shaker Sort ».

Trois états. Se dit d'un dispositif électronique numérique qui peut se permettre à l'état 0, 1, ou à l'état de haute impédance.

TTY. Abréviation de Teletype (télescripteur).

U

UAL. Unité arithmétique et logique (anglais ALU) utilisée dans les systèmes à micro-ordinateur et chargée d'exécuter des calculs.

UART (Universal Asynchronous Receiver Transmitter). Dispositif programmable de gestion des fonctions d'E/S.

UC. Unité centrale de traitement (CPU, Central Processing Unit). Composant matériel commandant le fonctionnement des autres unités. Il effectue les calculs et gère les périphériques.

V

Validation. Opération généralement effectuée dans les programmes d'application et consistant à vérifier les données.

Validé. Etat d'un processus répondant à certaines conditions et permettant de ce fait des interruptions. (Contraire d'**Invalidation**).

Valorisé à l'exécution. Dans certains systèmes d'exploitation ou dans certains langages, se dit d'un module particulier contenant les informations nécessaires à l'utilisation d'un programme (run-time module en anglais).

Vidage. Opération d'entrée sur imprimante ou écran, sans traitement, du contenu d'une zone de mémoire ou d'une partie de disque.

Vide. Qualifie un ensemble, par exemple une chaîne, ne contenant que des blancs.

Virgule fixe. Définit une représentation de valeurs numériques réelles dans laquelle la virgule a une position définie.

Virgule flottante. Définit une représentation de valeurs numériques réelles sous forme d'une mantisse et d'un exposant.

Viseur. Dispositif permettant d'afficher une grandeur.

Vitesse de transfert. Vitesse à laquelle les données sont transférées entre un dispositif et l'unité centrale. Elle s'exprime normalement en milliers de caractères par seconde.

W

Walking-one. Technique de contrôle des mémoires.

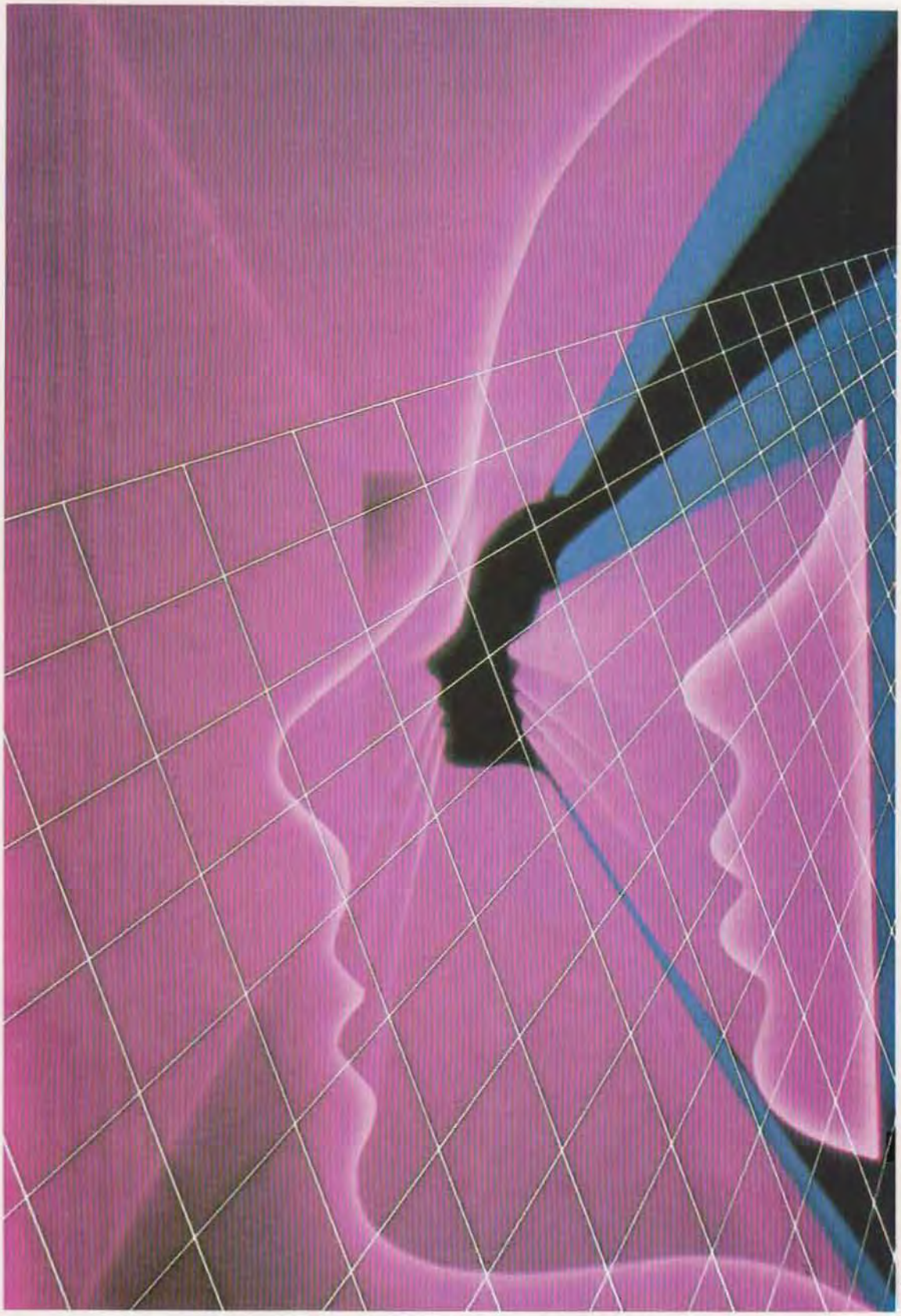
Window. Voir **Fenêtre**.

Z

Zone. Désigne généralement une partie de la mémoire, ou encore une position de mémoire de masse adressable dans une base de données. En multiprogrammation, une zone est allouée à chaque utilisateur pour ses traitements.

Zone de débordement. Désigne, dans le cas des fichiers, une zone particulière dans laquelle sont mémorisés les enregistrements qui ne peuvent pas être placés dans la zone logique à laquelle ils appartiennent.

Zoom (grossissement). Fonction de représentation grossie de parties d'un dessin.



Marika

1705

Table des matières

Volume 1

Volume 6

Index des mots réservés

FONCTIONS DU SYSTEME D'EXPLOITATION

BASIC

COMPILATION

INSTRUCTIONS PARTICULIERES AU BASIC 80

COMMANDES DOS

ASSEMBLEUR

COBOL

FORTRAN

PASCAL

INSTRUCTIONS DU BASIC POUR LA COMMUNICATION DES DONNEES

INSTRUCTIONS BASIC POUR LE GRAPHISME

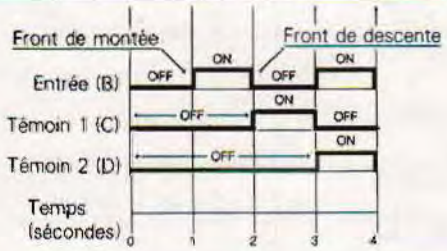
Index des listings et des programmes d'exemples

Index des tests

Index des articles

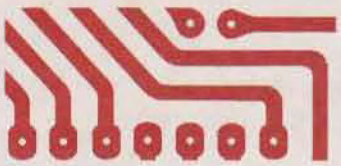
Errata

Dans la colonne de gauche "Références", sont successivement indiqués la page, la colonne (A pour gauche et B pour droite) ou bien s'il s'agit d'un tableau ou d'un encadré, enfin, en 3e position, le numéro de la ligne. Dans la colonne du milieu "Errata", on trouve les mauvais libellés ; dans la colonne de droite "Corrections" sont indiqués les bons libellés.

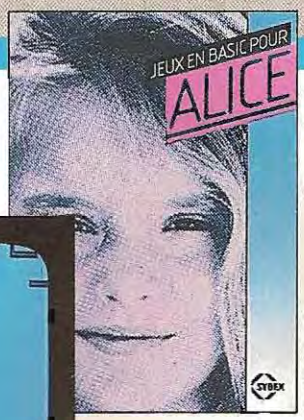
REFERENCES	ERRATA	CORRECTIONS
47/A/7	2101	2001
47/A/tableau		
57 7e question/b	En notation hexadécimale, on ne peut indiquer de nombre supérieur à 9.	En notation hexadécimale, on peut indiquer des nombres supérieurs à 9.
62/A/7	226-174 : 152 décimal =	226-174 : 52 décimal =
77/encadré/22	7/ a-vrai b-vrai c-faux d-vrai	7/ a-vrai b-vrai c-vrai d-vrai
85/B/8	Nom = Michel OU Frédéric	Prénom = Michel OU Frédéric
97/A/28	Ce boîtier comporte quatorze branches, trois	Ce boîtier contient quatorze broches, trois
108/tableau	Sortie = représentation négative inversée	Sortie = représentation négative inversée (H=Ø)
111/A/15	code ASCII (American Standard Computer	code ASCII (American Standard code for
121/B/27	masque = 1111111 1	masque = 1111111 0
185/tableau	code = A code = A code = A code = A	code = A code = A code = A code = B
208//B/tableau	BxT déperdition par rayonnent	BxT déperdition par conduction
264/tableau	Sortie du module (en direction du bloc 450 du schéma précédent).	Sortie du module (en direction du bloc 350 du schéma précédent).
287/encadré/5	250 Ko et d'1 Mo (1 Ko= 1000 octets ; 1 Mo = 1 méga-octet=1000 Ko).	250 Ko et d'1 Mo (1 Ko=1000 octets environ — en réalité 1024 ; 1 Mo : 1 méga-octet=1000 Ko).
347/tableau	Les valeurs sont : N(0)=30 N(1)=11 N(2)=15	Les valeurs sont : N(0)=5 N(1)=11 N(2)=30
350/encadré/44	1400/120=11 fois 1 minute, soit 110 minutes. C'est-à-dire qu'il faudrait presque 2 heu-	1400/120=11 fois 1 minute, soit 11 minutes. C'est-à-dire qu'il faudrait presque 2 heu-
385/encadré/24	Les solutions du test se trouvent pages 398 et 399	Les solutions du test se trouvent pages 398 et 400
391/listing		`160 FOR J=3 TO 7 STEP 2 'DEUXIÈME BOUCLE (FERMEE LIGNE 220)`
395/listing		`80 LPRINT"VILLE=";VILLE$;`
396/tableau	WHILE F Si la valeur (VILLE) lue sur le disque n'est pas Paris, l'indicateur F est mis à	WHILE C Si la valeur (VILLE) lue sur le disque n'est pas Paris, l'indicateur C est mis à
428/tableau	MOIS=MF AND JOUR<JF? MOIS=MD AND JOUR>JD?	MOIS=MF AND JOUR<=JF? MOIS=MD AND JOUR>=JD?

429/listing		50 IF FLAG=1 GOTO 103:REM ON ABANDONNE LA LECTURE
429/listing		100 NEXT I
429/listing		103 END
429/listing		1070 REM
430/listing		5020 IF(MOIS>MF) OR (MOIS=MF AND JOUR<=JF) THEN FLAG=1:PRINT MD$:GOTO 5090
430/listing		5030 IF(MOIS<MD) OR (MOIS=MD AND JOUR>=JD)THEN PRINT MD$:GOTO 5090
432/B/22	Un manuel, une page vide ou un menu mal écrit peuvent nuire à l'esprit, davantage qu'une	Un manuel, une page vidéo ou un menu mal écrits peuvent nuire à l'esprit davantage qu'une
452/listing		205 IF 47<N AND N<58 THEN B$=" ":GOTO 240
454/A/1	CV$(A$). Convertit une chaîne de quatre	CVS(A$). Convertit une chaîne de quatre
455/A/9	Arrivé à cette instruction, le système indique, par un curseur clignotant, qu'il attend une entrée et suspend l'exécution du programme jusqu'à la frappe d'une touche.	Contrairement à l'instruction suivante INPUT$(N), cette commande n'indique pas par un curseur clignotant qu'elle attend une entrée et ne suspend pas l'exécution du programme en cours.
555/listing		220 B$=INPUT$(1) 'LECTURE D'UN SEUL CARACTERE A LA FOIS
556/listing		70 LPRINT"ESSAI N.2",A1;A2;A3
556/listing		80 LPRINT"ESSAI N.3";A1;A2;A3
559/listing		*Noter que la lettre "P" est mise pour "Paris" ou "Parisiens"* 370 END P 123.8
817/A/10	COUNT(NS..NE). Cette fonction restitue la	COUNT(NS...NE). Cette fonction restitue la
817/B/4	SUM(NS..NE). Dans la page suivante, on a	SUM(NS...NE). Dans la page suivante, on a
817/B/8	est immédiate : SUM(C2..C5) équivaut à	est immédiate : SUM(C2...C5) équivaut à
818/A/1	MAX(NS..NE), MIN(NS..NE). Elles extraient	MAX(NS...NE), MIN(NS...NE). Elles extraient
818/A/4	NPV(S, NS..NE). NPV signifie Net Present	NPV(S,NS...NE). NPV signifie Net Present
818/A/9	térêt, les valeurs NS..NE sont les chiffres que	térêt, les valeurs NS...NE sont les chiffres que
818/B/9	LOOKUP (M,NS..NE). Cette fonction recherche la valeur M dans le champ NS..NE. La	LOOKUP (M,NS...NE). Cette fonction recherche la valeur M dans le champ NS...NE. La
818/tableau	COUNT(A2..A5) SUM(C2..C5) SUM(D2..D5) SUM(E2..E5)	COUNT(A2...A5) SUM(C2...C5) SUM(D2...D5) SUM(E2...E5)
819/ 2e col/8 et 9	B14 apparait la moyenne des valeurs de la colonne B et en D14	B12 apparait la moyenne des valeurs de la colonne B et en D12
823/tableau	SUM(B2..B6) SUM(C2..C6	SUM(B2...B6) SUM(C2...C6)
824/1re col/10	fonction SUM(B2..B6), qui dé-	fonction SUM(B2...B6), qui dé-
826/tableau	SUM(E4..G4) SUM(E5..G5) SUM(E6..G6) SUM(E7..G7) SUM(E8..G8) SUM(E9..G9)	SUM(E4...G4) SUM(E5...G5) SUM(E6...G6) SUM(E7...G7) SUM(E8...G8) SUM(E9...G9)
838/ 2e col/5 et 6	lonne H) par le coût unitaire (identification, colonne D). La	lonne E) par le coût unitaire (identification, colonne C). La
878/2e schéma	S=[(NON A)ET(B)]OUX[(A)ET(NON B)] OU exclusif = OUX	S=[(NON A)ET(B)OU[(A)ET(NON B)] OU
965/A/16,43-B/2,3	ENVIRONNEMENT DIVISION	ENVIRONMENT DIVISION

968/B/11	BLOCK CONTAINS 3 RECORDS	BLOCK CONTAINS 3 RECORDS.
968/B/13 à 16	BLOCK CONTAINS 10 RECORDS BLOCK CONTAINS 1 TO 10 RECORDS BLOCK CONTAINS 100 CHARACTERS BLOCK CONTAINS 30 TO 900 CHARACTERS	BLOCK CONTAINS 10 RECORDS. BLOCK CONTAINS 1 TO 10 RECORDS. BLOCK CONTAINS 100 CHARACTERS. BLOCK CONTAINS 30 TO 900 CHARACTERS.
968/B/21	BLOCK CONTAINS 1 RECORDS	BLOCK CONTAINS 1 RECORDS.
973/B/26	01 CHAMP-1	01 CHAMP-1.
973/B/28	05 SOUS-CHAMP-2	05 SOUS-CHAMP-2.
973/B/38	77 DEPARTEMENT PIC X(2)	77 DEPARTEMENT PIC X(2).
974/A/8	IF DEPARTEMENT IS EQUAL TO''69'	IF DEPARTEMENT IS EQUAL TO '69'
974/A/13	88 MARSEILLE VALUE '13'	88 MARSEILLE VALUE '13'. .
974/A/14	88 PARIS VALUE '75'	88 PARIS VALUE '75'.
974/B/8	01 EMPLOYE	01 EMPLOYE.
974/B/10	05 PRENOM ET NOM	05 PRENOM ET NOM.
974/B/13	05 DATE DE NAISSANCE	05 DATE DE NAISSANCE.
975/tableau	Code-1 : 33 caractères	Code-1 : 23 caractères
975/A/4	longueur totale de 33 caractères. Elle peut	longueur totale de 23 caractères. Elle peut
976/A/1	01 DATA-TR	01 DATA-TR.
976/A/2	05 JOUR PIC 9(2)	05 JOUR PIC 9(2).
976/A/3	05 MOIS PIC 9(2)	05 MOIS PIC 9(2).
976/A/4	05 ANNEE PIC 9(2)	05 ANNEE PIC 9(2).
976/B/25	05 SIEGE PIC X(2) VALUE'LY'	05 SIEGE PIC X(2) VALUE'LY'.
978/A/1	A X 9 P X * $ B 0 + - , / S V CR DB	A 9 P X * $ B 0 + - , / S V CR DB
979/B/27	01 SERIE-ARTICLE PIC AA9(2)	01 SERIE-ARTICLE PIC AA9(2).
980/tableau/19	1234 9(4) $1234	1234 $9(4) $1234
993/A/17	OPEN OUTPUT FICHIER-SORTIE	OPEN OUTPUT FICHIER-SORTIE.
995/A/14	SELECT FICHIER-A ASSIGN TO DISC FICHIER A	SELECT FICHIER-A ASSIGN TO DISC.
996/A/7	OPEN INPUT-FICHIER-A	OPEN INPUT-FICHIER-A.
997/B/33	OPEN E-S nom-du-fichier	OPEN E-S nom-du-fichier.
1000/A/4	OPEN EXTEND nom-du-fichier	OPEN EXTEND nom-du-fichier.
1000/B/6	FICHIER-5	FICHIER-5.
1000/B/11	FICHIER-5	FICHIER-5.
1001/B/21	UPON PRINTER	UPON PRINTER.
1001/B/22	CLOSE FICHIER-A	CLOSE FICHIER-A.
1004/B/34	[LINE AT BOTTON marge inférieure]	[LINE AT BOTTOM marge inférieure]
1011/A/17-18	01 CHAMP PIC S9 (5)	01 CHAMP PIC 59 (5).
1025/A/11,14	DISPLAY «NOMBRE PAIR»	DISPLAY 'NOMBRE PAIR'
1027/A/6	01 FICHE PIC X(3).	01 FICHE
1027/A/33	MOVE SPACES TO TO BAC-A-FICHE	MOVE SPACES TO BAC-A-FICHE.
1027/A/41,44	MOVE SPACES TO BAC-A-FICHE	MOVE SPACES TO BAC-A-FICHE.
1027/B/1	MOVE BAC-A-NOM TO NOM	MOVE BAC-A-NOM TO NOM.
1027/B/4	MOVE 'ABC' TO TYPE-FICHE	MOVE 'ABC' TO TYPE-FICHE.
1027/B/7	MOVE 83 TO ANNEE	MOVE 83 TO ANNEE.
1028/A/1	01 DEPART PIC X (3) VALUE'ABC	01 DEPART PIC X (3) VALUE'ABC'.
1041/A/14	01 COULEUR PICX(20) VALUE	01 COULEUR PIC X(20) VALUE.
1042/B/15	SPACES	SPACES.
1046/A/23	01 CHAINE	01 CHAINE.
1047/B/10	INTO DATE-CONVERTIE	INTO DATE-CONVERTIE.
1054/B/18	REMAINDER RESTE	REMAINDER RESTE.
1117/tableau	MOVE 2 TO DIMENSIONE	MOVE 2 TO DIMENSION
1145/A/13	01 B	01 B.
1261/tableau	ELEVE RECORD	ELEVE=RECORD



D.C.A.



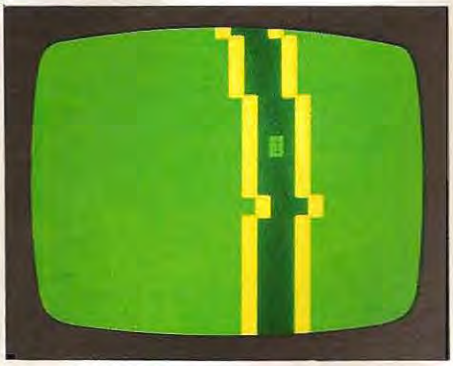
Un jeu pour Alice

Tiré de « Jeux en BASIC pour ALICE » de Pierre Monsaut, Éditions SYBEX, Réf. 320, Ft 16 x 22, 96 p., 49 F.

Les rôles sont maintenant inversés. Vous manœuvrez la D.C.A. et devez essayer d'abattre les avions qui passent au-dessus de vous. Pour tirer, utilisez n'importe quelle touche. Vous disposez au départ de dix obus. Si vous abattez huit avions, vous obtenez un bonus de huit points et huit obus supplémentaires.

```
10 REM D.C.A
20 CLEAR 250
30 GOTO 10000
40 A$=RIGHT$(A$,1)+LEFT$(A$,31)
50 B$=RIGHT$(B$,31)+LEFT$(B$,1)
60 PRINT@ 0,A$;:PRINT@ 64,B$;
80 R$=INKEY$
90 IF R$<>"" AND M=495 THEN M=46
3:NM=NM-1
100 IF M<>495 THEN M=M-64:PRINT@
 M,M$;:PRINT@ M+64,O$;
110 IF M<>143 THEN 200
120 IF POINT(32,5)<>0 THEN 200
130 PRINT@ 79,CHR$(191);
140 PRINT@ M,O$;
150 SOUND 1,1
160 S=S+1
170 B$=LEFT$(B$,14)+O1$+RIGHT$(B
$,13)
180 GOTO 300
200 IF M<>79 THEN 320
210 IF POINT(30,1)<>0 THEN 320
220 PRINT@ 15,CHR$(191);:S=S+1
240 PRINT@ M,O$;
250 SOUND 1,1
260 A$=LEFT$(A$,13)+O1$+RIGHT$(A
$,14)
300 IF S>1 AND INT(S/8)=S/8 THEN
 GOSUB 1000
310 M=495
320 PRINT@ 480,"S:";S;"M:";NM;
330 IF NM<1 AND M=495 THEN 400
340 IF M<32 THEN M=495
350 GOTO 40
400 IF S>R THEN R=S
410 PRINT@ 195,"SCORE :";S,"RECO
RD :";R;
420 PRINT@ 238,"UNE AUTRE ?";
430 R$=INKEY$
440 IF R$="" THEN 430
450 IF R$<>"N" THEN 30
460 CLS:END
1000 PRINT@ 173,"BONUS";
1010 A$=A1$:B$=B1$:NM=NM+8
1040 FOR I=1 TO 300:NEXT I
1060 PRINT@ 173,O1$+O1$;
1070 S=S+8
1080 RETURN
10000 S=0:A$="":B$=""
10030 FOR I=1 TO 32
10040 READ A,B
10050 A$=A$+CHR$(A)
10060 B$=B$+CHR$(B)
10070 IF I/8=INT(I/8) THEN RESTO
RE
10080 NEXT I
10090 A1$=A$:B1$=B$
10110 J$=CHR$(174)+CHR$(168)+CHR
$(172)
10120 M=495:M$=CHR$(171)
10140 NM=10:O$=CHR$(175):CLS 3
10170 PRINT@ 494,J$;:PRINT@ 0,A$
;
10190 PRINT@ 64,B$;:O1$=""
10210 FOR I=1 TO 5
10220 O1$=O1$+CHR$(175)
10230 NEXT I
10240 GOTO 40
15000 DATA 175,175,164,172,172,1
72,172,168,175,175,175,175,175,1
75,175,175
```


Dans le prochain numéro : **Apollo II,** un jeu pour Vic 20.



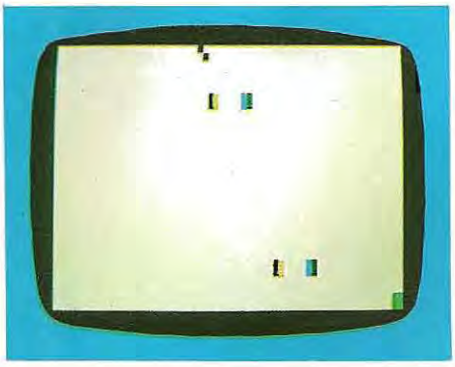
GRAND-PRIX

Un jeu pour ALICE

Au volant de votre formule 1, essayez de parcourir la plus grande distance possible. Votre voiture dispose de deux vitesses que vous pouvez sélectionner à l'aide des touche <1> et <2>. La direction est commandée par les touche <,> et <.>. En seconde vitesse, votre voiture roule deux fois plus vite. Mais gare à l'accident.

```
5 REM  ***************
10 REM * GRAND-PRIX *
15 REM ***************
20 CLS
30 N$=CHR$(128)
35 REM R$=ROUTE
40 R$=CHR$(159)+CHR$(128)+CHR$(1
28)+CHR$(128)+CHR$(159)
45 REM R=POSITION ROUTE
50 R=13
55 REM T=KM PARCOURUS/TEMPS
60 T=1
65 REM V$=CARACTERE VOITURE
70 V$="I"
75 REM V=POSITION VOITURE
80 V=207
90 V1=V
95 REM DEBUT ROUTE DROITE
100 FOR I=0 TO 15
110 PRINT@ R,R$;
120 R=R+32
130 NEXT I
140 R=R-32
145 REM AFFICHAGE VOITURE
150 PRINT@ V,V$;
154 REM
155 REM BOUCLE PRINCIPALE
156 REM
160 D$=INKEY$
165 REM CHANGEMENT DE VITESSE
170 IF D$="1" THEN T=1:GOTO 160
180 IF D$="2" THEN T=2:GOTO 160
185 REM DIRECTION
200 V=V+(D$=",")-(D$=".")
210 IF D$<>"" THEN 160
215 REM NOUVELLE POS. VOITURE
220 P=V+32
225 REM CALCUL COORDONNEES X,Y
230 Y=INT(P/32)*2
240 X=(P-16*Y)*2
245 REM ACCIDENT?
250 IF POINT(X,Y)<>0 THEN 400
255 REM AFFICHAGE ROUTE
260 R=R+(RND(2)=1)-(RND(2)=1)
270 IF R<480 THEN R=480
280 IF R>506 THEN R=506
290 PRINT@ 511," ";
295 REM AFFICHAGE VOITURE
300 PRINT@V1-32,N$;
310 PRINT@ R,R$;
320 PRINT@ V,V$;
345 REM COMPTE KM
350 K=K+T
355 REM DELAI
360 DL=(2-T)*50
370 FOR I=1 TO DL:NEXT I
380 V1=V
390 GOTO 160
394 REM
395 REM ACCIDENT
396 REM
400 PRINT@ P,CHR$(191);
410 PRINT@ V1,N$;
420 FOR I=1 TO 4
430 FOR J=1 TO 20
440 SOUND J*10,1
450 NEXT J
460 NEXT I
465 REM AFFICHAGE SCORE
470 PRINT@ 166,"KMS PARCOURUS :"
;K;
480 D$=INKEY$
490 PRINT@ 230,"UNE AUTRE ?";
500 D$=INKEY$
510 IF D$="" THEN 500
520 IF D$<>"N" THEN RUN
530 CLS
540 END
```

Tiré de «Jeux en BASIC sur ALICE»
de Pierre Monsaut, Editions SYBEX
Réf. 320 Ft 16 x 22, 96 p., 49 F.



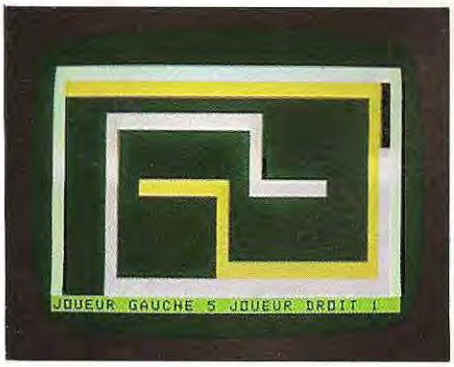
SLALOM

Un jeu pour ALICE

Partez aux sports d'hiver sans risque de vous casser une jambe ! Lancez-vous du haut de la piste et essayez de passer le plus grand nombre possible de portes sans heurter les piquets. Frappez n'importe quelle touche pour changer de direction.

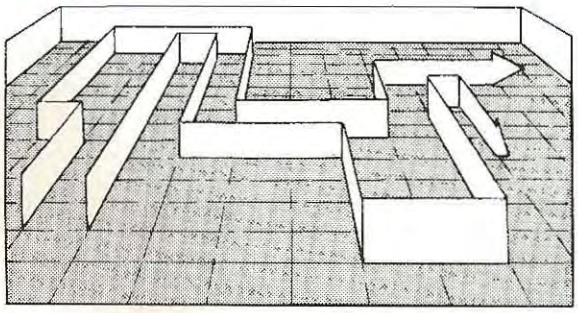
```
10 REM SLALOM
15 REM TABLEAU DES POSITIONS
16 REM        DU SKIEUR
20 DIM S$(1)
30 FOR I=1 TO 32
40 E$=E$+CHR$(207)
50 NEXT I
55 REM SKIEUR ALLANT A GAUCHE:
60 S$(0)=CHR$(201)
65 REM SKIEUR ALLANT A DROITE:
70 S$(1)=CHR$(198)
75 REM ECRAN BLANC
80 CLS 5
85 REM DIRECTION INITIALE:
86 REM              GAUCHE
90 D=-1
95 REM POSITION INITIALE
96 REM           DU SKIEUR
100 J=16
105 REM DESSIN DES PORTES
110 P$=CHR$(181)+CHR$(207)+CHR$(
207)+CHR$(170)
115 REM
116 REM BOUCLE PRINCIPALE
117 REM
120 FOR K=1 TO 300
125 REM CALCUL DES COORDONNEES
126 REM    X ET Y DU SKIEUR
130 Y=INT(J/32)*2
140 X=(J-16*Y)*2
145 REM SKIEUR AU NIVEAU D'UNE
146 REM              PORTE?
150 IF K>=16 AND (K-5)/10=INT((K
-5)/10) THEN GOSUB 350
155 REM AFFICHAGE D'UNE PORTE?
160 IF K<284 AND K/10=INT(K/10)
THEN GOSUB 400
165 REM MOUVEMENT DU SKIEUR
170 IF INKEY$<>"" THEN D=-D
180 J=J+D
190 IF J<2 THEN J=2
200 IF J>29 THEN J=29
210 PRINT@ 511,E$;
220 PRINT@ J,S$(D/2+0.5);
230 NEXT K
235 REM FIN DE LA DESCENTE
240 PRINT@ 164,"PORTE(S) RATEE(S
) :";T;
250 PRINT@ 230,"UNE AUTRE DESCEN
TE ?";
260 D$=INKEY$
270 IF D$="" THEN 260
280 IF D$<>"N" THEN RUN
290 CLS
300 END
344 REM
345 REM PORTE RATEE?
346 REM
350 IF POINT(X-2,Y)<>0 OR POINT(
X+4,Y)<>3 THEN IF POINT(X-4,Y)<>
0 OR POINT(X+2,Y)<>3 THEN T=T+1:
SOUND 1,1
360 RETURN
394 REM
395 REM AFFICHAGE D'UNE PORTE
396 REM
400 P1=RND(3)-2
410 P=P-6*P1
420 IF P<482 THEN P=488
430 IF P>506 THEN P=500
440 PRINT@ P,P$;
450 RETURN
```

Tiré de «Jeux en BASIC sur ALICE» de Pierre Monsaut, Editions SYBEX, Réf. 320, Ft 16 x 22, 96 p., 49 F.



TRACE

Un jeu pour ALICE



Deux joueurs s'affrontent pour se partager l'espace vital. Chacun doit s'efforcer, tout en se déplaçant, de ne jamais recouper sa trace ou celle de son adversaire, et de ne pas sortir du rectangle dessiné sur l'écran. Les commandes à utiliser sont :

Joueur de droite : <P>, <L>, <M> et <.>
Joueur de gauche : <Z>, <Q>, <S> et <W>

```
10 REM TRACE
20 CLS 0
30 C$=CHR$(207)
40 J1$=CHR$(239)
50 J2$=CHR$(255)
60 P1=248
70 P2=232
80 D1=-1
90 D2=1
120 FOR I=0 TO 31
130 PRINT@ I,C$;
140 PRINT@ I+448,C$;
150 NEXT I
160 FOR I=1 TO 13
170 PRINT@ I*32,C$;
180 PRINT@ I*32+31,C$;
190 NEXT I
200 PRINT "JOUEUR GAUCHE";J2,"JO
UEUR DROIT";J1
210 PRINT@ P1,J1$;
220 PRINT@ P2,J2$;
230 D$=INKEY$
240 C1=(D$="L")-(D$="M")+32*((D$
="P")-(D$="."))
250 C2=(D$="Q")-(D$="S")+32*((D$
="Z")-(D$="W"))
260 IF C1<>0 THEN D1=C1
270 IF C2<>0 THEN D2=C2
280 P1=P1+D1
290 Y=INT(P1/32)*2
300 X=(P1-16*Y)*2
310 IF POINT(X,Y)<>0 THEN 1000
320 PRINT@ P1,J1$;
330 P2=P2+D2
340 Y=INT(P2/32)*2
350 X=(P2-16*Y)*2
360 IF POINT(X,Y)<>0 THEN 2000
370 PRINT@ P2,J2$;
380 SOUND 1,1
390 GOTO 230
1000 J2=J2+1
1010 GOSUB 5000
1020 IF J2=10 THEN 3000
1030 D$=INKEY$
1040 GOTO 10
2000 J1=J1+1
2010 GOSUB 5000
2020 IF J1=10 THEN 4000
2030 D$=INKEY$
2040 GOTO 10
3000 CLS
3010 PRINT@ 164,J2$;"LE JOUEUR G
AUCHE GAGNE";J2$
3020 PRINT@ 202,J2;"A";J1
3030 GOTO 4500
4000 CLS
4010 PRINT@ 164,J1$;"LE JOUEUR D
ROIT GAGNE";J1$
4020 PRINT@ 202,J1;"A";J2
4500 R$=INKEY$
4510 PRINT@ 266,"UNE AUTRE ?"
4520 R$=INKEY$
4530 IF R$="" THEN 4510
4540 IF R$<>"N" THEN RUN
4550 END
5000 FOR I=5 TO 255 STEP 5
5010 SOUND I,1
5020 NEXT I
5030 RETURN
10000 SOUND 1,1
10010 RETURN
```

Tiré de «Jeux en BASIC pour ALICE» de Pierre Monsaut, Editions SYBEX, Réf. 320, Ft 16 × 22, 96 p., 49 F

SYBEX 6-8, impasse du Curé
75881 PARIS CEDEX 18
Tél. : 203.95.95

CHIFFRES

Un jeu pour ATARI

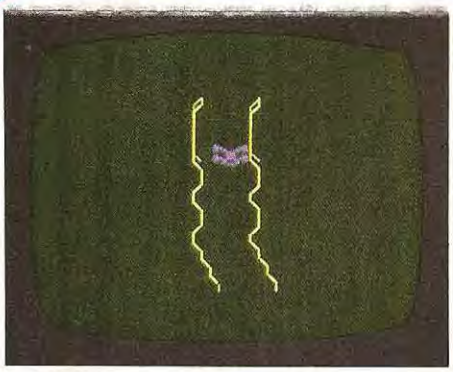
Utilisez les touches <Z> et <X> pour modifier votre croqueur de chiffres. A droite de l'écran, des nombres apparaissent les uns après les autres et si vous ne parvenez pas à les dévorer, vous aurez perdu. Pour dévorer un chiffre, utilisez la touche <.>, vous ne pouvez dévorer qu'un chiffre identique à celui de votre croqueur de chiffres. Si vous dévorez un signe «*», vous obtenez un bonus.

```
0 GOTO 135:REM ** CHIFFRES DE P.BUNN **
1 FOR I=50 TO 75 STEP 2:SOUND 0,I,10,15:
NEXT I:I=0:Y=M+48:IF M=10 THEN Y=10
2 Q$=CHR$(Y):IF A$(1,1)=Q$ THEN I=1:S=S+
10
3 IF A$(2,2)=Q$ AND I=0 THEN A$(2,2)=A$(
1,1):I=1:S=S+20
4 IF A$(3,3)=Q$ AND I=0 THEN A$(3,3)=A$(
2,2):A$(2,2)=A$(1,1):I=1:S=S+30
5 IF A$(4,4)=Q$ AND I=0 THEN A$(4,4)=A$(
3,3):A$(3,3)=A$(2,2):A$(2,2)=A$(1,1):I=1
:S=S+40
6 IF A$(5,5)=Q$ AND I=0 THEN A$(5,5)=A$(
4,4):A$(4,4)=A$(3,3):A$(3,3)=A$(2,2):I=1
:S=S+50
7 IF A$(6,6)=Q$ AND I=0 THEN A$(6,6)=A$(
5,5):A$(5,5)=A$(4,4):A$(4,4)=A$(3,3):A$(
3,3)=A$(2,2):I=1:S=S+60
8 IF I=1 THEN A$(1,1)=" ":N=N+1
10 IF I=1 AND Q$=CHR$(10) THEN FOR I=255
 TO 20 STEP -18:SOUND 0,I,10,15:NEXT I:S
=S+(INT(RND(0)*7+3)*100)
11 POSITION 1,0:? #6;SC$;S:? #6;" RECORD
:";REC
12 SOUND 0,0,0,0:RETURN
20 FOR I=14 TO 0 STEP -2:SOUND 0,M+15%30
,10,I:NEXT I:RETURN
135 DIM A$(6),Q$(1),SC$(6):GOSUB 2000
140 GRAPHICS 2:SETCOLOR 4,2,2:SETCOLOR 2
,2,4
150 A$="      ":A$(6,6)=CHR$(RND(0)*9+48)
160 S=0:N=5:M=0:U=18:GOSUB 11
165 Y=M+48:IF M=10 THEN Y=10
170 POSITION 5,5:? #6;CHR$(Y);":";A$
180 I=PEEK(764):I2=PEEK(53775)
190 IF I=22 AND I2=251 THEN M=M+1:GOSUB
20:IF M>10 THEN M=0
200 IF I=23 AND I2=251 THEN M=M-1:GOSUB
20:IF M<0 THEN M=10
210 IF I=34 AND I2=251 THEN GOSUB 1
215 I=0
220 TRAP 1000:O=O+1:IF O/U=INT(O/U) THEN
 A$(N,N)=CHR$(RND(0)*10+48):I=1
222 IF I=1 THEN IF A$(N,N)=CHR$(58) THEN
 A$(N,N)=CHR$(10)
224 IF I=1 THEN N=N-1:I=0
225 IF INT(RND(0)*20)=3 THEN U=U-1:IF U<
8 THEN U=8
230 TRAP 40000:GOTO 165
1000 ? #6:? #6;"..perdu..":FOR I=90 TO 2
20 STEP 11:FOR H=80 TO 90 STEP 2:SOUND 0
,H,10,15:SOUND 1,I,10,15
1010 SETCOLOR 4,H,4
1020 POKE 709,INT(RND(0)*15)*16+8
1030 NEXT H:NEXT I:SETCOLOR 4,2,4:POKE 7
64,255
1040 SOUND 0,0,0,0:SOUND 1,0,0,0:IF S>RE
C THEN REC=S:POSITION 0,2:? #6;" record!
!!"
1050 ? "UNE AUTRE ";:TRAP 1050:INPUT A$
1055 IF A$(1,1)="N" THEN GRAPHICS 0:? "A
U REVOIR.":? :? :END
1060 IF A$(1,1)="O" THEN 140
1070 GOTO 1050
2000 FOR P=1 TO 6
2010 READ N
2020 SC$(P,P)=CHR$(N)
2030 NEXT P
2040 DATA 243,227,239,242,229,186
2050 RETURN
```

Tiré de «Jeux en BASIC sur ATARI» de Paul Bunn, Éditions SYBEX, Réf. 282, Ft 16 × 22, 96 p., 49 F.

SYBEX 6-8, impasse du Curé 75881 PARIS CEDEX 18 Tél. 203.95.95

Dans le prochain numéro : **Cavernes**, un programme-jeu pour ZX 81



GRAND PRIX 2

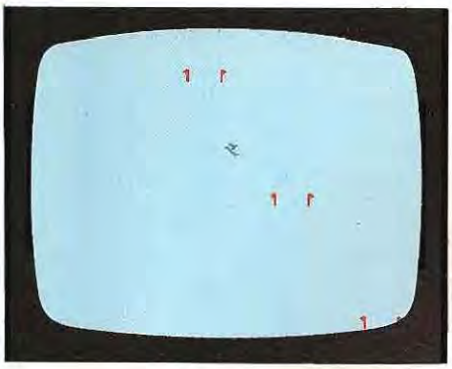
Un jeu pour ATARI

Pour conduire votre bolide, utilisez le manche à balai. Si vous réussissez à atteindre l'arrivée, vous aurez un bonus, et la valeur de la distance parcourue sera doublée.

```
10 REM ** GRAND PRIX 2 - P.BUNN **
15 PI=5:GOSUB 3000:ACCIDENT=1000
20 X2=14:X=116:S=PEEK(106)-8:Q=S*256:FOR
 N=Q+512 TO Q+640:POKE N,0:NEXT N:POKE 5
4279,S
30 POKE 559,46:POKE 53248,X:POKE 704,216
:POKE 704,70:POKE 53256,1
40 FOR N=Q+552 TO Q+581:READ A:POKE N,A:
NEXT N
50 DATA 129,195,165,24,24,153,219,165,36
,24
60 GRAPHICS 0:SETCOLOR 2,0,0:POKE 53277,
3:POKE 559,46
65 POKE 752,1:POKE 53278,A
66 FOR P=0 TO 23:POKE 201,X2:? ,S2$:NEXT
 P
70 S=STICK(0):X=X+(S=7)*2:X=X-(S=11)*2:P
OKE 53248,X
72 SOUND 0,40,10,15:SC=SC+PI:SOUND 0,0,0
,0
75 IF Q THEN Q=0:Z=2:GOTO 110
80 A=PEEK(53770)
90 IF A>85 AND A<170 THEN Z=2
100 IF A>170 AND X2<15 THEN Z=3:Q=1
105 IF A<85 AND X2>2 THEN X2=X2-1:Z=1:Q=
1
110 POKE 201,X2
115 IF A=192 AND  NOT I THEN ? ,F$:I=1:G
OTO 145
120 IF Z=1 THEN ? ,S1$
130 IF Z=2 THEN ? ,S2$
140 IF Z=3 THEN ? ,S3$:X2=X2+1
145 Y=PEEK(53252):IF Y<>0 AND I THEN GOT
O 2000
150 IF Y<>0 THEN GOTO ACCIDENT
160 GOTO 70
1000 REM ** ACCIDENT!!!! **
1010 N=INT(RND(0)*10):POKE Q+552+N,PEEK(
53770):SOUND 0,RND(0)*20+20,80,15:POKE 7
04,PEEK(53770)
1020 IF PEEK(53770)<240 THEN 1010
1030 POKE 53248,0:? :? "VOTRE SCORE:";SC
:? :? :? "TAPEZ UNE TOUCHE.":POKE 764,25
5
1035 SOUND 0,0,0,0
1040 IF PEEK(764)=255 THEN 1040
1050 RUN
2000 FOR Y=0 TO 255:POKE 710,Y:NEXT Y
2005 ? CHR$(125);"      SCORE :";SC:POKE
710,0
2010 ? "     BONUS  :1000":B=1000
2020 FOR G=1 TO 1000 STEP 10:B=B-10:SC=S
C+10:POSITION 14,0:? SC:POSITION 14,1:?
B;"  ":SOUND 0,G/4,10,10:NEXT G
2030 SOUND 0,0,0,0
2040 ? :PI=PI*2:? "1 KM VAUT ";PI;" POIN
TS"
2045 RESTORE :I=0
2050 GOTO 20
3000 DIM F$(8),S1$(8),S2$(8),S3$(8)
3010 RESTORE 3000:FOR P=1 TO 8
3020 READ A,B,C,D
3030 F$(P,P)=CHR$(A)
3040 S1$(P,P)=CHR$(B)
3050 S2$(P,P)=CHR$(C)
3060 S3$(P,P)=CHR$(D)
3070 NEXT P
3080 RESTORE :RETURN
3090 DATA 160,6,22,7,198,32,32,32
3100 DATA 201,32,32,32,206,32,32,32
3110 DATA 201,32,32,32,211,32,32,32
3120 DATA 200,32,2,32,160,6,32,7
```

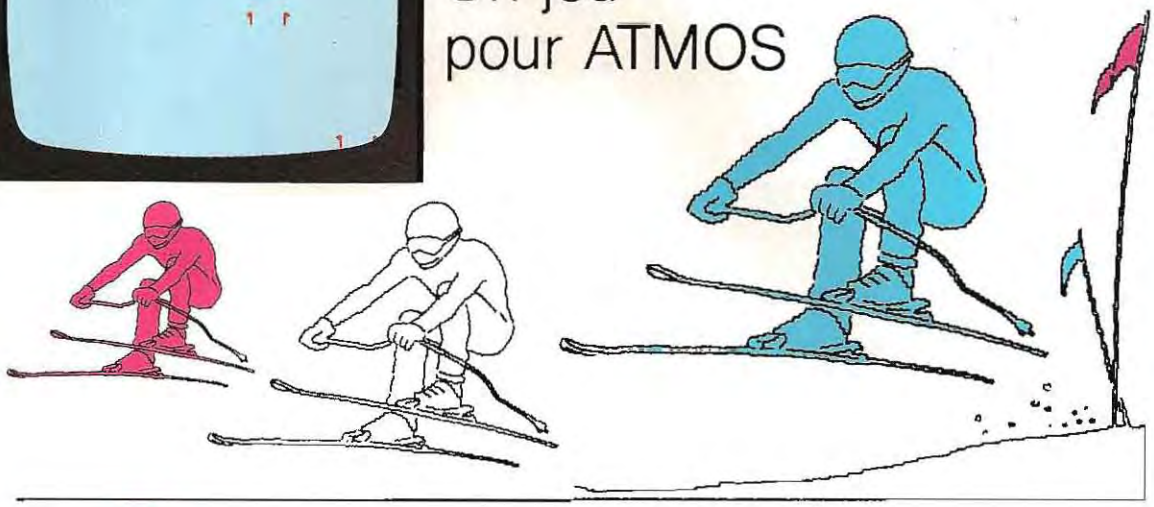
Tiré de «Jeux en BASIC sur ATARI» de Paul Bunn, Editions SYBEX, Réf. 282, Ft 16 × 22, 96 p., 49 F.

SYBEX 6-8, impasse du Curé 75881 PARIS CEDEX 18 Tél. : 203.95.95



SLALOM

Un jeu pour ATMOS

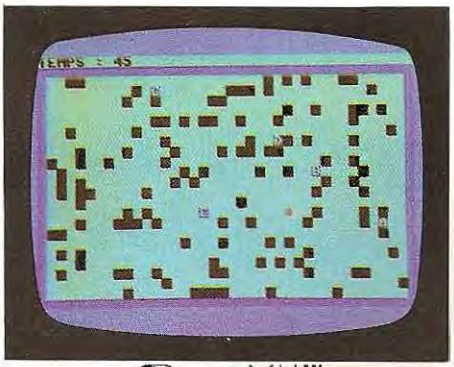


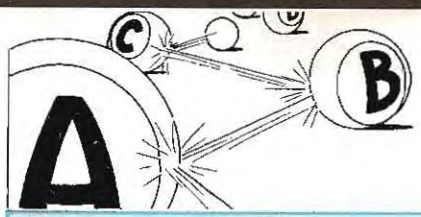
Partez aux sports d'hiver sans risque de vous casser une jambe ! Lancez-vous du haut de la piste et essayez de passer le plus grand nombre possible de portes sans heurter les piquets. Frappez n'importe quelle touche pour changer de direction.

```
 10 REM **********
 20 REM * SLALOM *
 30 REM **********
 50 GOSUB 460
 60 FOR K=1 TO 300
 90 IF K>=16 AND (K-5)/10=INT((K-5)/10)T
HEN GOSUB 360
 100 IF K<284 AND K/10=INT(K/10) THEN GO
SUB 380
 110 IF KEY$<>"" THEN DX=-DX
 120 S1=SX
 130 SX=SX+DX
 140 IF SX<2 THEN SX=2:DX=-DX
 150 IF SX>37 THEN SX=37:DX=-DX
 160 PLOT S1,SY,N$
 170 PRINT@ 38,26;N$
 180 PLOT SX,SY,S$(DX/2+0.5)
 190 NEXT K
 200 PLOT 8,15,"PORTE(S) RATEE(S) :"+STR
$(T)
 210 PLOT 8,18,"UNE AUTRE DESCENTE ?"
 220 REPEAT
 230 D$=KEY$
 240 UNTIL D$=""
 250 REPEAT
 260 D$=KEY$
 270 UNTIL D$<>""
 280 IF D$<>"N" THEN PRINT CHR$(17):RUN
 290 INK 0
 300 PRINT CHR$(17)
 310 CLS
 320 END
 360 IF SCRN(SX-1,SY)=93 AND SCRN(SX+2,S
Y)=94 THEN 375
 365 IF SCRN(SX-2,SY)=93 AND SCRN(SX+1,S
Y)=94 THEN 375
 370 T=T+1
 372 ZAP
 375 RETURN
 380 P=P+(INT(RND(1)*3)-1)*8
 390 IF P<2 THEN P=10
 400 IF P>34 THEN P=26
 410 PLOT P,26,P$
 420 PLOT 1,26,CHR$(1)
 430 RETURN
 460 CLS
 470 PAPER 7
 480 INK 4
 490 PRINT CHR$(17)
 500 FOR I=0 TO 31
 510 READ A
 520 POKE 46808+I,A
 530 NEXT I
 540 S$(0)=CHR$(91)
 550 S$(1)=CHR$(92)
 560 N$=CHR$(32)
 570 P$=CHR$(93)+N$+N$+CHR$(94)
 580 DX=-1
 590 SY=11
 600 SX=19
 610 P=18
 620 RETURN
1000 DATA 4,56,17,26,4,8,16,32
1010 DATA 8,7,34,22,8,4,2,1
1020 DATA 12,28,12,4,4,4,4,4
1030 DATA 6,7,6,4,4,4,4,4
```

Tiré de «Jeux en BASIC sur ATMOS» de Pierre Monsaut, Editions SYBEX, Réf. 346, Ft 16 × 22, 96 p., 49 F.

SYBEX 6-8, impasse du Curé 75881 PARIS CEDEX 18 Tél. : 203.95.95





ALPHABET

Un jeu pour COMMODORE 64

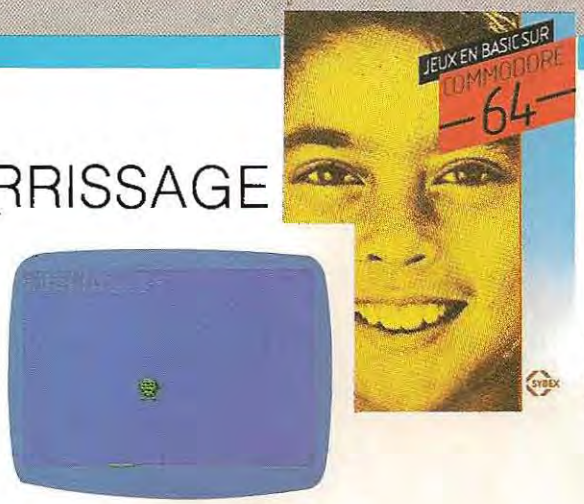
Voici un jeu d'adresse assez difficile. Vous tenterez de marquer le plus grand nombre de points possible en effaçant les lettres affichées par l'ordinateur. Les touches à utiliser sont <W> (haut), <Z> (bas) <A> (gauche) et <S> (droite). La barre d'espacement vous permet de vous arrêter. *Attention :* vous devez effacer les lettres dans l'ordre alphabétique en évitant les obstacles placés au hasard sur l'écran, et ceci en un temps limité. (Le temps qui vous reste est affiché sur la première ligne.) Lorsque toutes les lettres sont effacées, le jeu reprend avec une lettre supplémentaire.

```
5 REM  ************
10 REM * ALPHABET *
15 REM ************
20 GOSUB 1000
100 FOR I=1 TO X
110 GET X$
120 D=(X$="A")-(X$="S")+40*((X$="W")-(X$
="Z"))
130 IF D<>0 THEN D0=D
140 IF X$=" " THEN D0=0
150 T=T-0.1
160 T$=STR$(INT(T+1))
170 PRINT H$;
180 PRINT "TEMPS :";T$;" ";
190 IF T<0 THEN 3000
200 J=J+D0
210 C=PEEK(J)
220 IF C=I+128 THEN S=S+I:GOTO 290
230 IF C<>32 THEN J=J1
240 POKE J1,CR
250 POKE J,CJ
260 POKE J+M,JC
270 J1=J
280 GOTO 110
290 POKE J1,CR
300 POKE J,CJ
310 POKE J+M,JC
320 J1=J
330 NEXT I
340 GOTO 20
1000 X=X+1:CJ=81
1020 JC=10:X$=""
1040 D=0:D0=D
1060 T=50:T$=""
1080 H$=CHR$(19)
1090 J=0:J1=J
1110 C=0:CR=32
1130 M=54272
1140 CB=160
2000 PRINT CHR$(147);
2010 POKE 53280,14
2020 POKE 53281,1
2030 FOR I=0 TO 39
2040 POKE 1064+I,CB
2050 POKE 1064+I+M,6
2060 POKE 1984+I,CB
2070 POKE 1984+I+M,6
2080 NEXT I
2090 FOR I=1 TO 22
2100 POKE 1064+I*40,CB
2110 POKE 1064+I*40+M,6
2120 POKE 1103+I*40,CB
2130 POKE 1103+I*40+M,6
2140 NEXT I
2150 FOR I=1 TO 120
2160 GOSUB 5000
2170 POKE P,CB
2180 POKE P+M,0
2190 NEXT I
2200 FOR I=1 TO X
2210 GOSUB 5000
2220 POKE P,I+128
2230 POKE P+M,6
2240 NEXT I
2250 GOSUB 5000
2260 J=P
2270 POKE J,CJ
2280 POKE J+M,JC
2290 J1=J
2300 PRINT CHR$(144);
2310 RETURN
3000 IF S>RE THEN RE=S
3010 PRINT CHR$(147):PRINT:PRINT
3040 PRINT TAB(13)"TEMPS ECOULE"
3050 PRINT:PRINT:PRINT
3080 PRINT TAB(13)"SCORE :";S
3090 S=0:PRINT:PRINT:PRINT
3130 PRINT TAB(13)"RECORD :";RE
3140 PRINT:PRINT:PRINT:PRINT:GET X$
3180 PRINT TAB(13)"UNE AUTRE ?"
3190 GET X$
3200 IF X$="" THEN 3190
3210 IF X$<>"N" THEN 20
3220 END
5000 P=INT(RND(TI)*960)+1064
5010 IF PEEK(P)<>32 THEN 5000
5020 RETURN
```

Tiré de «Jeux en BASIC sur COMMODORE 64» de Pierre Monsaut, Editions SYBEX Réf. 317, Ft 16 × 22, 96 p., 49 F.

SYBEX 6-8, impasse du Curé 75881 PARIS CEDEX 18 Tél. : 203.95.95

ATTERRISSAGE



Un jeu pour Commodore 64

Tiré de « Jeux en BASIC sur Commodore 64 » de Pierre Monsaut, Éditions SYBEX, Réf. 317, Ft 16 x 22, 96 p., 49 F.

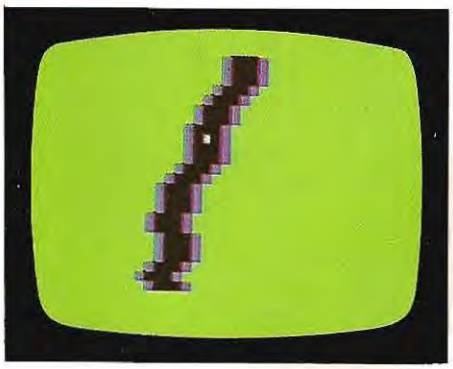
Après un long voyage en apesanteur, poser une navette spatiale en douceur n'est pas chose aisée ; mais grâce à votre ordinateur, vous allez être en mesure de vous entraîner sans danger. Vous devez poser votre navette sur l'aire prévue à cette effet. Vous pouvez vous diriger vers la droite et vers la gauche à l'aide des touches de contrôle du curseur ou freiner votre descente avec la barre d'espacement. Pour réussir un atterrissage, les vitesses verticales et horizontales doivent être inférieures ou égales à 1.

```
5 REM  ******************
10 REM * ATTERRISSAGE *
15 REM ******************
20 GOSUB 1000
100 GET X$:IF FU=0 THEN 150
110 IF X$<>"" THEN FU=FU-1
120 IF X$=" " THEN UU=UU-1
130 IF X$=G$ THEN UH=UH-1
140 IF X$=D$ THEN UH=UH+1
150 UU=UU+0.1:U=U+UU:H=H+UH
190 IF H>255 THEN H=0:MS=M1
200 IF H>90 AND MS=M1 THEN 4540
210 IF H<0 AND MS=M1 THEN H=255:MS=M0
220 IF H<0 AND MS=M0 THEN 4540
240 PRINT HO$;
250 PRINT "CARBURANT :";STR$(FU);"     "
255 IF V<0 THEN 300
260 POKE D+16,MS:POKE D,H
270 POKE D+1,INT(V)
275 IF V>V1-3 THEN 3000
300 PRINT "VITESSE VERT. :";STR$(INT(10*
UU)/10);"     "
310 PRINT "VITESSE HORIZ.:";STR$(UH);"
320 GOTO 100
1000 D=53248:RESTORE
1010 FOR I=0 TO 190
1020 READ A:POKE 832+I,A
1035 NEXT I
1040 M1=5:M0=0:POKE D+39,7:POKE D+40,7
1090 POKE D+41,0:POKE 2040,13
1110 POKE 2041,14:POKE 2042,15
1130 V1=230:V0=0:FU=90:G$=CHR$(29)
1170 D$=CHR$(17):HO$=CHR$(19):M=54272
2000 PRINT CHR$(147);:POKE D+21,0
2020 H=INT(RND(TI)*191)+65
2030 AH=INT(RND(TI)*191)+65
2050 FU=FU+10:MS=M0:UH=0:UU=0:U=0
2100 POKE D+2,AH:POKE D+3,V1
2120 POKE D,H:POKE D+1,V:POKE D+21,3
2150 RETURN
3000 IF ABS(H-AH)>4 THEN 4000
3005 IF UU>1 THEN 4000
3010 IF ABS(UH)>1 THEN 4000
3020 FOR I=1 TO 4000:NEXT I
3030 SC=SC+1:GOSUB 2000
3050 GOTO 100
4000 POKE D+5,V+5:POKE D+4,H
4020 POKE D+21,6
4500 PRINT:PRINT:PRINT:PRINT:PRINT
4530 PRINT TAB(6)"VOTRE NAVETTE S'EST EC
RASEE"
4540 IF SC>RE THEN RE=SC
4545 FOR I=1 TO 2000:NEXT I
4560 PRINT:PRINT:PRINT
4580 PRINT TAB(13)"SCORE :";SC
4590 SC=0
5000 PRINT:PRINT:PRINT
5030 PRINT TAB(13)"RECORD :";RE
5040 IF RE<SC THEN RE=SC
5050 SC=0:PRINT:PRINT:PRINT:GET X$
5100 PRINT TAB(13)"UNE AUTRE ?"
5110 GET X$
5120 IF X$="" THEN 5110
5130 IF X$<>"N" THEN POKE D+21,0:GOTO 20
5140 END
10000 DATA 0,255,0,1,255,128,3,255,192
10010 DATA 7,255,224,12,195,48,12,195,48
10020 DATA 15,255,240,12,102,48,12,102,4
8
10030 DATA 15,255,240,7,255,224,3,129,19
2
10040 DATA 1,129,128,0,255,0,1,255,128
10050 DATA 1,24,128,2,60,64,2,36,64
10060 DATA 4,0,32,4,0,32,14,0,112,0
10100 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
10110 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
10120 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
10130 DATA 0,0,0,0,0,0,0,0,0,255,255,255
10140 DATA 255,255,255,255,255,255,0
10200 DATA 0,0,0,0,0,0,45,0,27,3,0,0,8,0,1
27,0
10210 DATA 0,5,0,0,236,0,0,67,0,2,64,240
10220 DATA 7,32,112,7,32,56,15,145,56
10230 DATA 31,23,120,31,25,124,95,97,254
10240 DATA 127,255,254,255,255,255,255
10250 DATA 255,255,0,0,0,0,0,0,0,0,0
```

Dans la même collection : ATARI, ALICE, ATMOS, DRAGON, ORIC, Spectrum, TO 7, TRS-80, TRS-80 couleur, TRS-80 MC-10, VIC 20 et ZX 81.

SYBEX
6-8, impasse du Curé
75018 PARIS
Tél. 203-95-95

Dans le prochain numéro : un programme-jeu pour Alice.



GRAND PRIX

Un jeu pour COMMODORE 64

Au volant de votre formule 1, essayez de parcourir la plus grande distance possible. Votre voiture dispose de deux vitesses que vous pouvez sélectionner à l'aide des touches <1> et <2>. La direction est commandée par les touches de contrôle du curseur. En seconde vitesse, votre voiture roule deux fois plus vite. Mais gare à l'accident !

```
5 REM  xxxxxxxxxxxxxx
10 REM x GRAND-PRIX x
15 REM xxxxxxxxxxxxxx
20 GOSUB 1000
100 GET X$
110 IF X$=V1$ THEN T=1
120 IF X$=V2$ THEN T=2
130 V=V+(X$=G$)-(X$=D$)
140 P=PEEK(V+M+40)
150 IF INT(P/16)<>P/16 THEN 3000
160 R=R+(RND(TI)<0.5)-(RND(TI)<0.5)
170 IF R<VN THEN R=VN
180 IF R>VM THEN R=VM
190 PRINT TAB(R);R$
200 POKE V1-40,CR
210 POKE V1-40+M,0
220 POKE V,CV
230 POKE V+M,VC
240 K=K+T
250 DL=(2-T)*50
260 FOR I=1 TO DL
270 NEXT I
280 V1=V
290 GOTO 100
1000 PRINT CHR$(147)
1010 POKE 53280,5
1020 POKE 53281,5
1030 R=18
1040 V=1364
1050 V1=V
1060 M=54272
1070 T=1
1080 S=0
1090 V1$="1"
1100 V2$="2"
1110 D$=CHR$(29)
1120 G$=CHR$(17)
1130 CR=160
1140 CV=90
1150 VC=1
1160 R$=CHR$(18)+CHR$(31)+" "+CHR$(144)
1170 R$=R$+"    "+CHR$(31)+" "
1180 VN=1
1190 VM=34
2000 FOR I=1 TO 25
2010 PRINT TAB(R);R$
2020 NEXT I
2030 POKE V,CV
2040 POKE V+M,VC
2050 RETURN
3000 POKE V1,CR
3010 POKE V1+M,0
3020 POKE V+40,160
3030 POKE V+M+40,VC
3040 FOR I=1 TO 500
3050 GET X$
3060 NEXT I
3070 PRINT CHR$(147)
3080 PRINT
3090 PRINT
3100 PRINT
3110 PRINT TAB(15)"ACCIDENT !!"
3120 PRINT
3130 PRINT
3140 PRINT
3150 PRINT TAB(12)"KMS PARCOURUS :";K
3160 PRINT
3170 PRINT
3180 PRINT
3190 PRINT TAB(15)"UNE AUTRE ?"
3200 GET X$
3210 IF X$="" THEN 3200
3220 IF X$<>"N" THEN RUN
3230 END
```

Tiré de «Jeux en BASIC sur Commodore 64» de Pierre Monsaut, Editions SYBEX, Réf. 317, Ft 16 × 22, 96 p., 49 F.

SYBEX 6-8, impasse du Curé
75881 PARIS-CEDEX 18
Tél. : 203.95.95

SERPENT

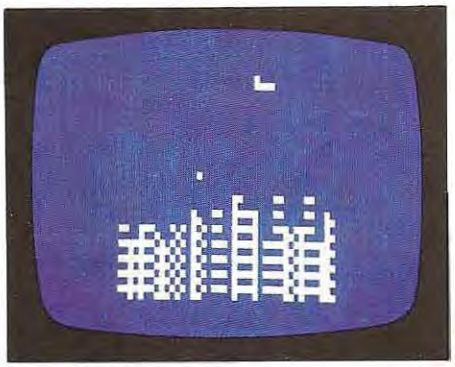
Un jeu pour COMMODORE 64

Dans ce jeu, vous êtes un serpent qui se déplace en ondulant sur l'écran. Le changement de direction s'effectue en tapant n'importe quelle touche. Pour pouvoir vous déplacer, vous devez vous nourrir. Heureusement, vous êtes entouré par un grand nombre de champignons. Mais attention ! Si les bleus sont excellents, vous devez absolument éviter les noirs qui, eux, sont vénéneux. Chaque champignon bleu vous apporte suffisamment de calories pour avancer de dix lignes. Essayez de ne pas mourir de faim sans pour autant finir empoisonné !

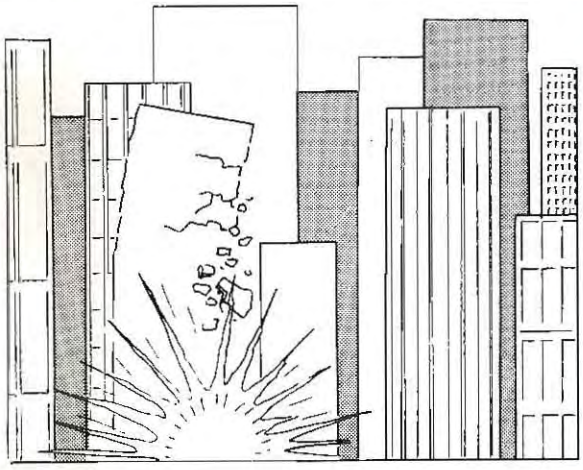
```
5 REM  ***********
10 REM * SERPENT *
15 REM ***********
20 GOSUB 1000
100 GET X$
110 IF X$<>"" THEN D=-D
120 T=T+D
130 IF T<L0 THEN T=L0
140 IF T>L1 THEN T=L1
150 T1=T+40
160 IF PEEK(T1)=65 THEN 500
170 IF PEEK(T1)=88 THEN S=S+10:H=H+10
180 PRINT B$;
190 P=INT(RND(TI)*40)
200 POKE 1984+P,88
210 POKE 1984+P+M,BC
220 P=INT(RND(TI)*40)
230 IF RND(TI)<0.5 THEN 260
240 POKE 1984+P,65
250 POKE 1984+P+M,RC
260 POKE T,CT
270 POKE T+M,TC
280 S=S-1
290 IF S=0 THEN 500
300 H=H+1
310 GOTO 100
500 PRINT B$;
510 POKE T,CT
520 POKE T+M,TC
530 FOR I=1 TO 500
540 NEXT I
550 IF H>RE THEN RE=H
560 PRINT CHR$(19);
580 FOR I=1 TO 10
590 PRINT B$;
600 NEXT I
610 PRINT TAB(13)"SCORE :";H;CHR$(19);
620 FOR I=1 TO 15
630 PRINT B$;
640 NEXT I
650 PRINT TAB(13)"RECORD :";RE;CHR$(19);
660 FOR I=1 TO 20
670 PRINT B$;
690 GET X$
700 NEXT I
710 PRINT TAB(13)"UNE AUTRE ?";
720 GET X$
730 IF X$="" THEN 720
740 IF X$<>"N" THEN 20
750 END
1000 PRINT CHR$(147)
1010 POKE 53280,5
1020 POKE 53281,13
1030 CT=19
1040 TC=9
1050 D=-1
1060 T=1404
1070 T1=T
1080 BC=6
1090 RC=0
1100 S=100
1110 M=54272
1120 B$=CHR$(17)
1130 L1=1423
1140 L0=1384
1150 H=0
1200 FOR I=0 TO 24
1210 PRINT B$;
1220 NEXT I
1230 PRINT CHR$(144);
1240 RETURN
```

Tiré de «Jeux en BASIC sur Commodore 64» de Pierre Monsaut, Editions SYBEX, Réf. 317, Ft 16 × 22, 96 p., 49 F.

SYBEX 6-8, impasse du Curé
75881 PARIS-CEDEX 18
Tél. : 203.95.95



BLITZ



Un jeu pour DRAGON

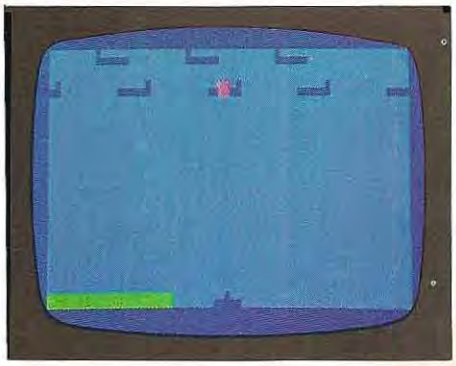
Votre mission est de détruire la ville que vous survolez afin de pouvoir atterrir. A chaque passage votre avion vole un peu plus bas. Vous ne pouvez larguer une bombe (en appuyant sur une touche quelconque) que lorsque la bombe précédente a atteint son objectif ou le sol. Lorsque votre avion a atterri (ou lorsqu'il s'est écrasé contre un immeuble), le score est affiché ainsi que le record du jour. Si ce jeu vous paraît trop difficile, vous pouvez changer les limites de la ville (6 et 26, ligne 100).

```
10 REM *********
20 REM * BLITZ *
30 REM *********
34 REM
35 REM INITIALISATION
36 REM
39 REM A$=AVION
40 A$=CHR$(128)+CHR$(155)+CHR$(1
47)
49 REM B$=BOMBE
50 B$=CHR$(145)
58 REM H=POSITION DE L'AVION ET
59 REM SCORE
60 H=0
69 REM B=POSITION DE LA BOMBE
70 B=0
80 B1=B
90 CLS 0
94 REM
95 REM AFFICHAGE DE LA VILLE
96 REM
99 REM 6 ET 26 : LIMITES VILLE
100 FOR I=6 TO 26
110 C=RND(7)+151
114 REM LIGNE 120: REMPLACER 8
115 REM PAR VALEUR SUPERIEURE
116 REM POUR DIMINUER LA HAUTEUR
117 REM DES IMMEUBLES
120 FOR J=15 TO RND(4)+8 STEP-1
130 PRINT@ J*32+I,CHR$(C);
140 NEXT J
150 NEXT I
154 REM
155 REM BOUCLE PRINCIPALE
156 REM
159 REM AFFICHAGE AVION
160 PRINT@ H,A$;
169 REM AVION ECRASE?
170 IF PEEK(1027+H)<>128 THEN 24
0
179 REM TIR
180 IF INKEY$<>"" AND B=0 THEN B
=H+33
189 REM BOMBE ATTEINT LE SOL
190 IF B<>0 THEN GOSUB 360
199 REM PAS DE BOMBE LACHEE?
200 IF B=0 THEN GOSUB 400
209 REM AVANCE AVION
210 H=H+1
219 REM AVION POSE?
220 IF H=507 THEN 240
230 GOTO 160
234 REM
235 REM AVION POSE OU ECRASE
236 REM
239 REM RECORD BATTU?
240 IF H>R THEN R=H
250 PRINT @ 3,"SCORE :";H,
260 PRINT "RECORD :";R;
270 FOR I=1 TO 100
280 NEXT I
290 R$=INKEY$
300 PRINT@ 73,"UNE AUTRE ?";
310 R$=INKEY$
320 IF R$="" THEN 310
330 IF R$<>"N" THEN 40
340 CLS
350 END
354 REM
355 REM BOMBE LACHEE
356 REM
359 REM SOL ATTEINT?
360 IF B>=510 THEN B=0
365 REM AFFICHAGE BOMBE
370 PRINT@ B1,CHR$(128);
380 IF B<>0 THEN PRINT@ B,B$;:B1
=B:B=B+32
390 RETURN
394 REM
395 REM DELAI POUR RALENTIR
396 REM L'AVION SI AUCUNE BOMBE
397 REM N'EST LACHEE
398 REM
400 FOR I=1 TO 20
410 NEXT I
420 RETURN
```

Tiré de «Jeux en BASIC sur ORAGON» de Pierre Monsaut, Éditions SYBEX, Réf. 324, Ft 16 × 22, 96 p., 49 F.

SYBEX 6-8, impasse du Curé
75881 PARIS CEDEX 18
Tél. 203.95.95

Dans le prochain numéro : **Chiffres**, un programme-jeu pour Atari.



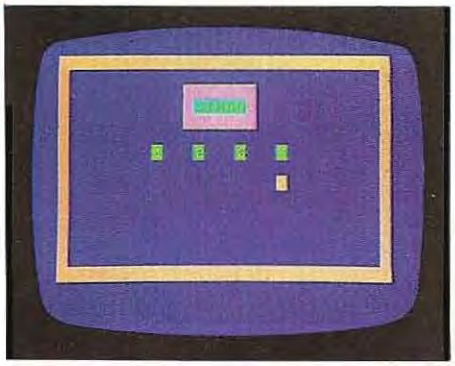
D.C.A.

Un jeu pour DRAGON

Les rôles sont maintenant inversés. Vous manœuvrez la D.C.A. et devez essayer d'abattre les avions qui passent au-dessus de vous. Pous tirer, utilisez n'importe quelle touche. Vous disposez au départ de dix missiles. Si vous abattez huit avions, vous obtenez un bonus de huit points et huit missiles supplémentaires.

```
10 REM **********
20 REM * D.C.A. *
30 REM **********
40 CLEAR 250
50 GOTO 490
60 A$=RIGHT$(A$,1)+LEFT$(A$,31)
70 B$=RIGHT$(B$,31)+LEFT$(B$,1)
80 PRINT@ 0,A$;
90 PRINT@ 64,B$;
100 R$=INKEY$
110 IF R$<>"" AND M=495 THEN M=4
63:NM=NM-1
120 IF M<>495 THEN M=M-64:PRINT@
 M,M$;:PRINT@ M+64,O$;
130 IF M<>143 THEN 210
140 IF POINT (32,5)<>0 THEN 210
150 PRINT@ 79,CHR$(191);
160 PRINT@ M,O$;
170 SOUND 1,1
180 S=S+1
190 B$=LEFT$(B$,14)+O1$+RIGHT$(B
$,13)
200 GOTO 280
210 IF M<>79 THEN 300
220 IF POINT(30,1)<>0 THEN 300
230 PRINT@ 15,CHR$(191);
240 S=S+1
250 PRINT@ M,O$;
260 SOUND 1,1
270 A$=LEFT$(A$,13)+O1$+RIGHT$(A
$,14)
280 IF S>1 AND INT(S/8)=S/8 THEN
 GOSUB 420
290 M=495
300 PRINT@ 480,"S:";S;"M:";NM;
310 IF NM<1 AND M=495 THEN 340
320 IF M<32 THEN M=495
330 GOTO 60
340 IF S>R THEN R=S
350 PRINT@ 195,"SCORE :";S,"RECO
RD :";R;
360 PRINT@ 233,"UNE AUTRE ?";
370 R$=INKEY$
380 IF R$="" THEN 370
390 IF R$<>"N" THEN 50
400 CLS
410 END
420 A$=A1$
430 B$=B1$
440 NM=NM+8
450 FOR I=1 TO 300
460 NEXT I
470 S=S+8
480 RETURN
490 S=0
500 A$=""
510 B$=""
520 FOR I=1 TO 32
530 READ A,B
540 A$=A$+CHR$(A)
550 B$=B$+CHR$(B)
560 IF I/8=INT(I/8) THEN RESTORE
570 NEXT I
580 A1$=A$
590 B1$=B$
600 J$=CHR$(174)+CHR$(168)+CHR$(
172)
610 M=495
620 M$=CHR$(171)
630 NM=10
640 O$=CHR$(175)
650 CLS 3
660 PRINT@ 494,J$;
670 PRINT@ 0,A$;
680 PRINT@ 64,B$;
690 O1$=""
700 FOR I=1 TO 5
710 O1$=O1$+CHR$(175)
720 NEXT I
730 GOTO 60
740 DATA 175,175,164,172,172,172
,172,168.175,175,175,175,175,175
,175,175
```

Tiré de «Jeux en BASIC sur DRAGON» de Pierre Monsaut, Editions SYBEX, Réf. 324, Ft 16 × 22, 96 p., 49 F.



SIMON

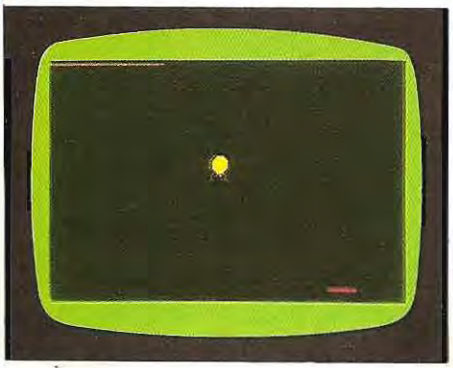
Un jeu pour DRAGON

Dans cette version du jeu bien connu, vous utiliserez les touches <1>, <2>, <3>, et <4> pour répéter la séquence produite de façon aléatoire par l'ordinateur. A chaque touche correspondent une couleur et un son :
(du plus grave au plus aigu)
1 = blanc ; 2 = bleu ; 3 = violet ; 4 = orange.
La limite de la longueur de la séquence à répéter est 100. Lorsque vous faites une erreur, le jeu s'arrête et votre score, correspondant à la longueur de la plus longue séquence que vous avez réussi à reproduire, est affiché ainsi que le record du jour.

```
10 REM *********
20 REM * SIMON *
30 REM *********
40 DIM T(100)
50 X=0
60 CLS 0
70 GOSUB 370
79 REM X=LONGUEUR DE LA SEQUENCE
80 X=X+1
90 T(X)=RND(4)
100 FOR I=1 TO X
110 GOSUB 330
120 NEXT I
130 FOR I=1 TO X
139 REM ATTENTE
140 FOR J=1 TO 1000
150 D$=INKEY$
160 IF D$<>"" THEN 190
170 NEXT J
179 REM TROP LONG, PERDU
180 GOTO 250
189 REM ERREUR?
190 IF ASC(D$)-48<>T(I) THEN 250
200 GOSUB 330
210 NEXT I
220 FOR I=1 TO 500
230 NEXT I
240 GOTO 80
244 REM
245 REM PERDU
246 REM
250 X=X-1
260 IF X>R THEN R=X
270 PRINT@ 324,"SCORE :";X,"RECO
RD :";R;
280 PRINT@ 394,"UNE AUTRE ?";
290 D$=INKEY$
300 IF D$="" THEN 290
310 IF D$<>"N" THEN 50
320 END
324 REM
325 REM AFFICHAGE DE LA SEQUENCE
326 REM
330 POKE 1285+4*T(I),191+T(I)*16
340 SOUND 50+T(I)*45,2
350 POKE 1285+4*T(I),128
360 RETURN
364 REM
365 REM INITIALISATION
366 REM
369 REM TRACE DU CADRE
370 FOR I=1024 TO 1055
380 POKE I,255
390 POKE I+448,255
400 NEXT I
410 FOR I=1 TO 13
420 POKE 1024+I*32,255
430 POKE 1055+I*32,255
440 NEXT I
449 REM AFFICHAGE DES CHIFFRES
450 FOR I=1 TO 4
460 POKE 1221+4*I,I+48
470 NEXT I
480 FOR I=1 TO 7
490 POKE 1099+I,239
500 POKE 1163+I,239
510 NEXT I
520 POKE 1132,239
530 POKE 1138,239
540 PRINT@ 109,"SIMON";
550 RETURN
```

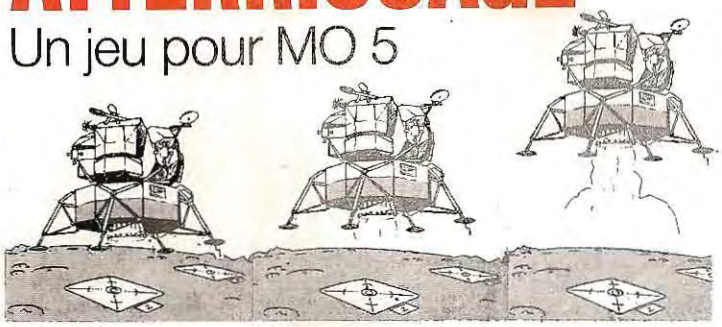
Tiré de «Jeux en BASIC sur DRAGON»
de Pierre Monsaut, Editions SYBEX
Réf. 324, Ft 16 × 22, 96 p., 49 F.

SYBEX 6-8, impasse du Curé
75881 PARIS CEDEX 18
Tél. : 203.95.95



ATTERRISSAGE

Un jeu pour MO 5



Après un long voyage en apesanteur, poser une navette spatiale en douceur n'est pas chose aisée; mais grâce à votre ordinateur, vous allez être en mesure de vous entraîner sans danger. Vous devez poser votre navette sur l'aire prévue à cet effet. Vous pouvez vous diriger vers la droite et vers la gauche à l'aide des touches < , > et <. > ou freiner votre descente avec la barre d'espacement. (Vous pouvez également utiliser le joystick.) La quantité de carburant disponible est indiquée par la longueur du trait horizontal, en haut de l'écran.

Si vous n'avez pas de joystick, supprimez les lignes 140 à 160.

```
10 REM ****************
20 REM * ATTERRISSAGE *
30 REM ****************
40 DEFINT A-Z
50 DIM N(15,20)
60 DIM R(15,20)
70 DIM C(15,10)
80 FU=100
90 GOTO 620
100 GOSUB 750
110 DV=DV+1
120 IF FU<1 THEN D$="":GOTO 170
130 D$=INKEY$
140 Q=STICK(0)
150 DV=DV+3*(Q=5)
160 DH=DH+(Q=3)-(Q=7)
170 DV=DV+3*(D$=" ")
180 DH=DH+(D$=".")-(D$=",")
190 IF D$<>"" OR Q<>0 THEN FU=FU-1:FF=1
200 DRAW "COL"+STR$(FF)
210 FF=0
220 IF MH<0 OR NH>319 THEN 250
230 IF MV<0 THEN 250
240 PUT (MH,MV)-(NH,NV),R
250 NH=NH+DH
260 NV=NV+DV
270 IF NV>187 THEN 340
280 MV=NV-20
290 MH=NH-15
300 IF MH<0 OR NH>319 THEN 110
310 IF MV<0 THEN 110
320 PUT (MH,MV)-(NH,NV),N
330 GOTO 110
340 IF DV>5 THEN 420
350 IF ABS(NH-A)>4 THEN 420
360 IF ABS(DH)>1 THEN 420
370 PUT (MH,167)-(NH,187),N
380 FOR I=0 TO 1000
390 NEXT I
400 S=S+10
410 GOTO 100
420 FOR I=1 TO 10
470 NEXT I
480 CLS
490 SCREEN 4,6,6
500 LOCATE 5,10
510 PRINT "VOTRE NAVETTE S'EST ECRASEE"
520 LOCATE 10,13
530 PRINT "SCORE :";S;
540 LOCATE 10,16
550 PRINT "UNE AUTRE ?";
560 IF INKEY$<>"" THEN 560
570 D$=INKEY$
580 IF D$="" THEN 570
590 IF D$<>"N" THEN RUN
600 CLS
610 END
620 CLS
630 SCREEN 3,0,2
640 LOCATE 0,0,0
650 FOR I=1 TO 15
660 FOR J=0 TO 10
670 C(I,J)=RND*1000
680 NEXT J
690 NEXT I
700 DRAW "C3BM50,50R2L1U3E3H4U6E4R7F4D6G
4F3D3L1R2L1U3H3L3D2R1D1U1L3D1U1R1U2L3"
710 PAINT (55,40),3
720 GET (50,30)-(65,50),N
730 GET (0,0)-(15,20),R
740 GOTO 100
750 CLS
760 A=INT(RND*260)+20
770 DRAW "C1BM"+STR$(A)+",188R4L23D1R23D
1L23"
780 NH=160
790 NV=20
800 DH=0
810 DV=0
820 MV=0
830 MH=NH-15
840 FU=FU+10
850 DRAW "C1BM0,3R"+STR$(FU)
860 RETURN
```

Tiré de « MO 5 JEUX D'ACTION»
de Pierre Monsaut, Editions SYBEX,
Réf. 367, Ft 16 × 22, 96 p., 49 F.

SYBEX 6-8, impasse du Curé
75881 PARIS CEDEX 18
Tél. : 203.95.95

PARACHUTE

Un jeu pour MO5

Essayez, en sautant d'un hélicoptère en vol, d'atteindre la cible qui se trouve au sol. Une première pression sur une touche vous permet de descendre verticalement en chute libre. Une deuxième pression entraîne l'ouverture du parachute. La descente continue alors plus lentement et avec un angle de 45 degrés car le vent vous pousse. Plus vous attendrez pour ouvrir votre parachute et moins vous serez déporté. Mais n'attendez pas trop longtemps car, au-dessous de cent mètres, le parachute ne s'ouvre plus.

```
10 REM *************
20 REM * PARACHUTE *
30 REM *************
40 GOSUB 470
50 HH=HH-4
60 IF HH=0 THEN PUT (0,1)-(27,9),R
70 IF HH=0 THEN HH=288
80 PUT (HH,1)-(HH+27,9),H
90 D$=INKEY$
100 IF D$="" THEN 140
110 IF PV>100 THEN 140
120 IF SP=1 THEN OP=1 ELSE SP=1
130 IF OP=0 THEN PV=10:PH=HH
140 IF SP=0 THEN 230
150 IF OP=0 THEN PV=PV+8
160 IF OP=1 THEN PV=PV+1:PH=PH-1
170 IF PV>167 OR PH<1 THEN 260
180 IF OP=1 THEN 210
190 PUT (PH,PV)-(PH+14,PV+23),PF
200 GOTO 50
210 PUT (PH,PV)-(PH+14,PV+23),PO
220 GOTO 50
230 FOR I=1 TO 50
240 NEXT I
250 GOTO 50
260 IF ABS(PH-A)>4 THEN 320
270 FOR I=1 TO 1000
280 NEXT I
290 S=S+10
300 GOSUB 670
310 GOTO 50
320 CLS
330 SCREEN 1,2,2
340 LOCATE 10,10
350 ATTRB 1,1
360 PRINT "SCORE :";S;
370 LOCATE 10,16
380 PRINT "UNE AUTRE ?";
390 ATTRB 0,0
400 IF INKEY$<>"" THEN 400
410 D$=INKEY$
420 IF D$="" THEN 410
430 IF D$<>"N" THEN RUN
440 CLS
450 SCREEN 4,6,6
460 END
470 LOCATE 0,0,0
480 SCREEN 4,6,6
490 DEFINT A-Z
500 DIM H(27,8)
510 DIM PF(14,23)
520 DIM PO(14,23)
530 DIM R(27,8)
540 CLS
550 DRAW "C4BM51,50R14L6G4L2G1L1G1D1F1R2
1E1U5L2D1G2L5H1L1H1L1H1L1"
560 PAINT (58,56),4
570 GET (50,50)-(77,58),H
580 GET (100,50)-(127,58),R
590 CLS
600 DRAW "C1BM54,50G1L1G1D1G1D1R12U1H1U1
H1L1H1L4"
610 PAINT (56,54),1
620 DRAW "C1BM51,56D1F1D1F1D3R2D6L1R1U3R
2D3R1L1U4L1U4L1U1R2D1L1D4R1U2R2U3E1U1E1U
1"
630 GET (50,46)-(64,69),PO
640 CLS
650 DRAW "C1BM54,61R1D6L2D2U2R3D2L1D1R2U
1L1U2R3D2U2L3U3D3R1U6R1"
660 GET (50,46)-(64,69),PF
670 CLS
680 HH=288
690 HV=1
700 A=INT(RND*191)+10
710 DRAW "C0BM"+STR$(A)+",191R12"
720 SP=0
730 OP=0
740 PV=0
750 PH=0
760 RETURN
```

Tiré de «MO5 JEUX D'ACTION»
de Pierre Monsaut, Editions SYBEX
Réf. 367, Ft 16 x 22, 96 p., 49 F.

SERPENT

Un jeu pour MO 5

Dans ce jeu, vous êtes un serpent qui se déplace en ondulant sur l'écran. Le changement de direction s'effectue en tapant n'importe quelle touche. Pour pouvoir vous déplacer, vous devez vous nourrir. Heureusement, vous êtes entouré par un grand nombre de champignons. Mais attention ! Si les bleus sont excellents, vous devez absolument éviter les rouges qui, eux, sont vénéneux. Chaque champignon bleu vous apporte suffisamment de calories pour avancer de dix lignes. Essayez de ne pas mourir de faim sans pour autant finir empoisonné !

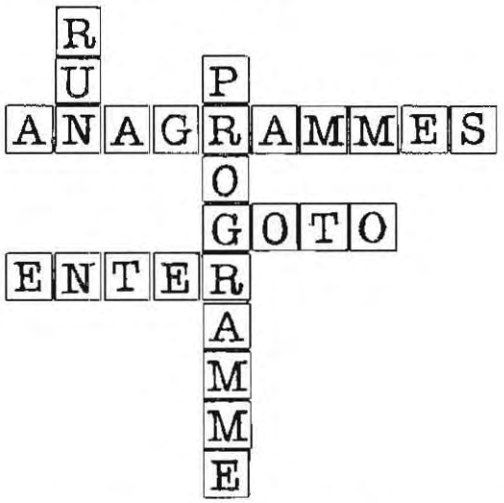
```
10 REM ***********
20 REM * SERPENT *
30 REM ***********
40 CLEAR ,,2
50 GOSUB 540
60 D$=INKEY$
70 IF D$<>"" THEN D=-D
80 X=X+D
90 IF X<0 THEN X=1
100 IF X>39 THEN X=38
110 IF POINT(X*8+4,Y*8+4)=1 THEN 260
120 IF POINT(X*8+4,Y*8+4)=4 THEN S=S+10:
H=H+10:BEEP
130 LOCATE X,Y
140 COLOR 0
150 PRINT S$;
160 LOCATE 0,24,0
170 PRINT
180 LOCATE INT(RND*38)+1,24
190 COLOR 4
200 PRINT C$;
210 IF RND>0.5 THEN LOCATE INT(RND*38)+1
,24:COLOR 1:PRINT C$;
220 S=S-1
230 IF S=0 THEN 260
240 H=H+1
250 GOTO 60
260 BEEP
270 LOCATE 0,24
280 PRINT
290 LOCATE X,Y-1
300 COLOR 5
310 PRINT S$;
320 FOR I=1 TO 5
330 BEEP
340 FOR J=1 TO 50
350 NEXT J
360 NEXT I
370 IF INKEY$<>"" THEN 370
380 IF H>R THEN R=H
390 LOCATE 4,20
400 COLOR 0
410 PRINT "SCORE :";H;
420 LOCATE 23,20
430 PRINT "RECORD :";R;
440 LOCATE 15,23
450 PRINT "UNE AUTRE ?"
460 D$=INKEY$
470 IF D$="" THEN 460
480 IF D$<>"N" THEN 520
490 SCREEN 4,6,6
500 CLS
510 END
520 GOSUB 610
530 GOTO 60
540 CLS
550 SCREEN 2,2,2
560 DEFINT A-Z
570 DEFGR$(0)=0,0,102,255,219,126,60,24
580 DEFGR$(1)=60,126,126,255,255,24,24,6
0
590 S$=GR$(0)
600 C$=GR$(1)
610 CLS
620 S=100
630 H=0
640 D=1
650 X=19
660 Y=10
670 RETURN
```

Tiré de «MO 5 JEUX D'ACTION» de Pierre Monsaut, Editions SYBEX, Réf. 367, Ft 16 × 22, 96 p., 49 F.

SYBEX 6-8, impasse du Curé 75881 PARIS-CEDEX 18 Tél. : 203.95.95

ANAGRAMME

Un jeu pour ORIC

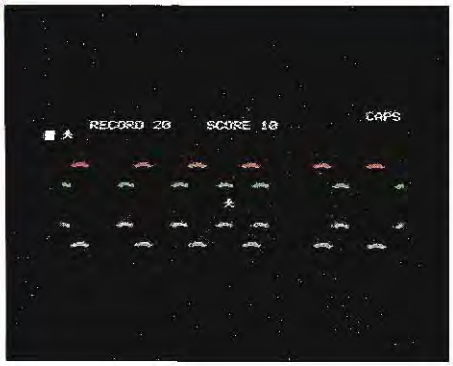


Dans ce jeu, vous disposez de dix essais pour trouver l'anagramme. Ce programme utilise un dictionnaire de quarante mots. Il peut être étendu en modifiant le nombre de mots (42) dans la ligne 9000.

```
10 REM ANAGRAMME
20 REM PETER SHAW 1983
30 GOSUB 9000'CHOIX MOT
35 ZAP
40 GOSUB 8000
45 EXPLODE
50 FOR A=1 TO 10
60 GOSUB 7000
70 GOSUB 6000
75 PING:GOSUB 8000
80 NEXT A
90 REM ********************************

100 REM PERDU
110 CLS:PLOT 7,2,"LE MOT EST "+W$
120 A=1
130 REPEAT
140 PLOT 12,10,CHR$(A)+"PERDU!!"
150 A=A+1:IF A=8 THEN A=1
155 WAIT 4:PLAY 1,0,1,100:SOUND 1,100,A
160 UNTIL KEY$<>""
170 PLAY 0,0,0,0:RUN
5000 REM GAGNE
5010 CLS
5020 A=1
5030 REPEAT
5040 PLOT 12,10,CHR$(A)+"BRAVO!!"
5050 A=A+1:IF A=9  THEN A=1
5055 WAIT 4:PLAY 1,0,1,100:SOUND 1,100,A

5060 UNTIL KEY$<>""
5070 PLAY 0,0,0,0:RUN
6000 IF G$=W$ THEN 5000
6010 RETURN
7000 CLS
7010 PRINT"ESSAI ";A
7020 PRINT
7030 PRINT"VOTRE ANAGRAMME EST ";S$
7040 PRINT
7050 INPUT"TAPEZ LE MOT ";G$
7060 RETURN
8000 L=LEN(W$)
8010 S$=LEFT$("                    ",L)
8020 FOR I=1 TO L
8030 R=INT(RND(1)*L)+1
8040 IF MID$(S$,R,1)>" " THEN GOTO 8030
8050 S$=LEFT$(S$,R-1)+MID$(W$,I,1)+MID$(
S$,R+1)
8060 NEXT I
8070 RETURN
9000 FOR A=1 TO INT(RND(1)*42)+1
9010 READ W$
9020 NEXT A
9030 RETURN
9040 DATA PROGRAMME,SOFTWARE,BASIC,HARDW
ARE,AFFICHAGE
9050 DATA ORDINATEUR,GRAPHIQUE,CLAVIER,I
MPRIMANTE
9060 DATA JEUX,SILICONE,ARCADE,MEMOIRE,P
UCE
9070 DATA ASSEMBLEUR,INTERFACE,VIDEO
9080 DATA DIGITAL,PARALLELE,DISQUE,MONIT
EUR
9090 DATA PROCESSEUR,SERIE,TRANSFERT,MAC
HINE
9100 DATA LANGUAGE,CASSETTE,PRISME,ACCES
,ALEATOIRE
9110 DATA CODEUR,COMPILER,ADAPTER,CIRCUI
T
9120 DATA PASCAL,COBOL , ECRAN,FORTRAN
9130 DATA LISTING,ANAGRAMME,EDITEUR,CONN
ECTER
```

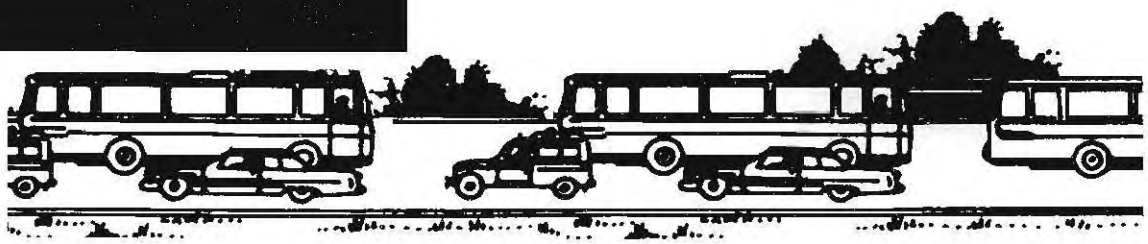
Tiré de «Jeux en BASIC sur ORIC»
de Peter Shaw, Editions SYBEX
Réf. 278, Ft 16 × 22, 96 p., 49 F

SYBEX 6-8, impasse du Curé
75881 PARIS CEDEX 18
Tél. : 203.95.95



AUTOROUTE

Un jeu pour ORIC



Traverser l'autoroute à l'heure de pointe n'est une partie de plaisir pour personne, mais c'est pourtant le but de ce jeu. Vous devez éviter deux files de voitures roulant en sens inverse ; si vous êtes écrasé par un véhicule, vous perdez l'une de vos six vies. Pour chaque piéton arrivé sain et sauf, vous gagnez dix points. Utilisez la barre d'espacement pour faire traverser votre piéton. Il y a un record à battre !

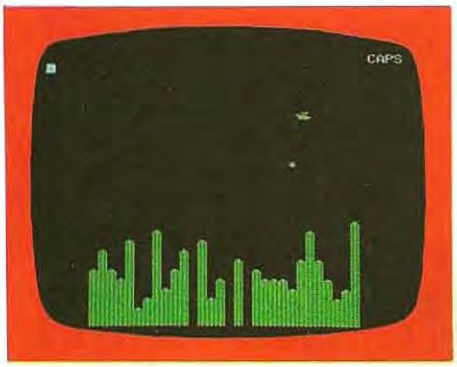
```
10 REM AUTOROUTE
20 REM PETER SHAW 1983
30 RE=0:PRINT CHR$(6)
40 GOSUB 9000
50 GOSUB 8000
60 V=14
70 H=19
80 GOSUB 7000
90 PLOT H,V," "
100 IF KEY$=" " THEN V=V-1
110 PLAY 0,1,1,10
140 PLOT 1,4,A$:PLOT 1,6,B$
150 PLOT 1,10,B$:PLOT 1,12,A$
160 IF SCRN(H,V)<>32 THEN 1000
170 PLOT H,V,"#"
180 L1$=LEFT$(A$,1):R1$=RIGHT$(A$,37)
190 L2$=LEFT$(B$,37):R2$=RIGHT$(B$,1)
200 A$=R1$+L1$:B$=R2$+L2$
210 IF V=2 THEN 2000
220 GOTO 90
1000 EXPLODE:WAIT 50
1010 IFML>0 THEN ML=ML-1:GOTO 60
1020 IF SC>RE THEN RE=SC
1030 PLOT 6,20,"TAPEZ UNE TOUCHE POUR JO
UER"
1040 GET A$
1050 GOTO 50
2000 PING:WAIT 50
2010 SC=SC+10
2020 GOTO 60
```

```
7000 CLS:PAPER 0:INK 7
7010 PRINT,"RECORD ";RE,"SCORE ";SC
7020 PLOT 0,4,CHR$(1)+A$
7030 PLOT 0,6,CHR$(2)+B$
7040 PLOT 0.10,CHR$(5)+B$
7050 PLOT 0,12,CHR$(6)+A$
7070 PLOT H,V,"#"
7080 FOR A=1 TO ML
7090 PLOT A,1,"#"
7100 NEXT A
7110 RETURN
7120 REM
8000 PAPER 0:INK 7
8010 SC=0
8020 B$="$%      $%      $%      $%      $%  $%
     "
8030 A$="'&      '&      '&      '&      '&
'&      "
8040 ML=5
8050 RETURN
8060 REM
9000 FOR A=(46080+(ASC("#")*8)) TO (4608
0+(ASC("'")*8)+7)
9010 READ US:POKE A,US:NEXT:RETURN
9020 DATA 12,12,30,45,12,18,33,0
9030 DATA 0,0,0,7,15,63,12,0
9040 DATA 0,0,0,48,62,62,6,0
9050 DATA 0,0,0,56,60,63,6,0
9060 DATA 0,0,0,3,31,63,24,0
```

Tiré de «Jeux en BASIC sur ORIC» de Peter Shaw, Éditions SYBEX, Réf. 278, Ft 16 × 22, 96 p., 49 F.

SYBEX 6-8, impasse du Curé 75881 PARIS CEDEX 18 Tél. 203.95.95

Dans le prochain numéro : **Blitz**, un programme-jeu pour Dragon.



BLITZ

Un jeu pour ORIC

Larguez vos bombes sur les immeubles pendant que votre avion traverse l'écran. Utilisez la barre d'espacement pour larguer vos bombes. Si vous désirez augmenter la difficulté, changez le nombre 10 ligne 8020 pour un nombre plus petit et le nombre 16 pour un plus grand.

```
10 REM BLITZ
20 REM PETER SHAW 1983
25 GOSUB 9000
26 GOSUB 8000
60 FOR A=0 TO 26
70 FOR B=1 TO 34
80 PLOT B,A," #$"
90 IF SCRN(B+3,A)=39 OR SCRN(B+4,A)=40 T
HEN GOTO 1000
95 PLOT B,A," #%"
100 X$=KEY$
110 IF X$=" " THEN GOSUB 250
120 PLAY 0,1,1,15
130 NEXT B
140 PLOT B,A,"    "
150 NEXT A
160 PLOT 1,26,"#$"
170 PRINT"                BRAVO!!"
180 PRINT
190 PRINT"ATTERRISSAGE REUSSI!!"
200 PRINT"VOUS AVEZ TUE ";A*B;" PERSONNE
S POUR"
210 PRINT"          SAUVER VOTRE VIE"
220 PRINT
230 PRINT" J'ESPERE QUE CELA NE PERTURBE
RA PAS  VOTRE SOMMEIL"
240 END
250 REM LANCEMENT BOMBE
260 B1=B
270 FOR DR=A TO 25
280 PLOT B,A," #$"
290 IF SCRN(B+3,A)=39 OR SCRN(B+4,A)=40
THEN GOTO 1000
300 PLOT B,A," #%"
305 B=B+1:IF B>34 THEN B=1:A=A+1:PLOT 34
,A-1,"      "
307 PLAY 0,1,1,10
320 PLOT B1,DR," ":PLOT B1,DR+1,"&"
340 NEXT DR
360 PLOT B,A,"    ":PLOT B1,26," "
370 RETURN
999 STOP
1000 EXPLODE
1010 FOR L=A TO 26
1020 PLOT B,L," #$"
1030 EXPLODE:WAIT L
1040 PLOT B,L,"    "
1050 NEXT L
1060 PRINT" SCORE ";A*10+B*10
1070 FOR A=0 TO 6
1080 INK A:WAIT 10
1090 NEXT A
1100 IF KEY$<>"" THEN 1100
1110 GET A$:RUN
8000 CLS:PAPER 0:INK INT(RND(1)*7)+1
8010 FOR S=33 TO 4 STEP -1
8020 HG=INT(RND(1)*10)+16
8030 PLOT S,HG,"'"
8040 FOR K=HG+1 TO 26
8050 PLOT S,K,"("
8060 NEXT K
8070 NEXT S
8080 RETURN
9000 FOR U=46359 TO 46407
9010 READ US:POKE U,US
9020 NEXT U:RETURN
9030 DATA 0,15,34,53,63,31,31,1
9040 DATA 0,48,32,4,52,60,52,4
9050 DATA 0,48,32,0,48,60,48,0
9060 DATA 0,18,12,30,30,12,0,0
9070 DATA 0,12,30,63,45,63,45,63
9080 DATA 45,63,45,63,45,63,45,63,45
```

Tiré de «Jeux en BASIC sur ORIC»
de Peter Shaw, Editions SYBEX
Réf. 278, Ft 16 x 22, 96 p., 49 F.

SYBEX 6-8, impasse du Curé
75881 PARIS CEDEX 18
Tél. : 203.95.95