ABC INFORMATIQUE

N° 83

COURS D'INFORMATIQUE PRATIQUE ET FAMILIALE

Trains et ordinateurs

Le traceur Penman

Procédures et fonctions Pascal

Des livres pour votre ordinateur

EDITIONS ATLAS

M 6062-83-12,50 F 90 FB - 3,80 FS - $ 1.95

Trains et ordinateurs

L'informatique a pris une place croissante à la S.N.C.F. Elle est particulièrement visible pour le client : depuis son domicile, il peut réserver sa place de train.

J.-Marc Fabbro/SNCF

Réservation à domicile
La réservation électronique des places à la S.N.C.F. existe depuis 1973 à partir des gares. Mais, dès cette année, les possesseurs de Minitel pourront le faire directement de chez eux, enfin... certains d'entre eux.

Avant tout service public, mais aussi entreprise commerciale, la Société nationale des chemins de fer français (S.N.C.F.) est présente sur l'ensemble du marché des transports, tant voyageurs que marchandises. Si sa principale raison d'être est l'exploitation des lignes de chemin de fer, son activité s'étend aussi, par l'intermédiaire de ses filiales, à des domaines connexes tels que la gestion d'hôtels, le tourisme, le camionnage, etc. Elle dispose également d'un patrimoine important comprenant voies, gares, triages, ateliers de réparation, installations de signalisation et de télécommunication, matériels roulants, équipements sociaux...

On conçoit qu'une telle entreprise, qui utilise pour son fonctionnement une main-d'œuvre et un matériel importants, et dont les établissements sont nombreux et dispersés, ait recours, pour gérer son activité, aux outils informatiques les plus modernes et les mieux adaptés du moment.

L'organisation informatique tient compte de la nature des applications mises en œuvre. Elle se répartit en trois domaines : l'informatique de gestion, à laquelle sont rattachés les calculs scientifiques et techniques et leurs traitements dérivés (conception, fabrication et enseignement assistés par ordinateur, infographie, etc.); l'informatique des processus industriels; et enfin la bureautique.

Ainsi, les équipements informatiques ont été largement utilisés pour les études de conception du TGV et la création de la ligne nouvelle Paris-Sud-Est. Ils le seront également pour les études de réalisation du TGV-Atlantique.

L'informatique de gestion

Parmi les applications de gestion dont est responsable la direction de l'informatique de la S.N.C.F., on peut distinguer deux catégories. Premièrement, les applications de gestion courante, comme celles de beaucoup d'entreprises, concernent notamment des tâches administratives à caractère répétitif (solde du personnel, comptabilité, etc.). Deuxièmement, les applications de gestion opérationnelles sont directement liées à l'activité de transport de la S.N.C.F.

De ces dernières applications, les deux plus importantes sont la gestion centralisée du trafic marchandises (G.C.T.M.) et la réservation électronique des places (RESA). Nous nous intéresserons plus particulièrement à cette dernière

Michel Henri/SNCF

Au guichet
Pour le candidat au voyage qui se déplace à la gare, la réservation peut être faite en quelques minutes grâce à l'interrogation en temps réel de l'ordinateur central. Cette liaison permet un gain de temps appréciable. A condition que la file d'attente ne soit pas trop longue...

application dans la seconde partie de cet article. Le transport des marchandises par wagons complets, du fait de l'importance du parc en service, des mouvements quotidiennement enregistrés (15 000 wagons répartis entre 4 000 gares) est apparu justiciable d'une gestion centralisée par ordinateur.

Mettant en œuvre le réseau de télé-informatique, la G.C.T.M. exploite, à partir de 1 200 terminaux, les informations recueillies lors de chaque mouvement de wagon. Les traitements informatiques couvrent le transport (suivi des wagons), composition des trains, préavis d'arrivée, etc.), le commercial (facturation et comptabilisation des frais, répartition des wagons vides, etc.) et le matériel (gestion des fichiers du matériel, inventaire permanent du parc et de l'utilisation des wagons, déclenchement des opérations d'entretien, etc.).

Concevoir une application informatique nécessite de répondre à un certain nombre de questions, certaines liées à des choix d'organisation du travail, d'autres à des choix techniques. Il en découle une « architecture » : ordinateurs de traitement, stockage des données, réseaux de transmission, postes de travail incluant les opérateurs, les terminaux, les documents...

La réservation électronique

Expérimentée dès le début des années soixante-dix et mise en service à la fin de 1973, la réservation électronique des places (RESA) couvre maintenant la presque totalité des prestations offertes aux voyageurs : places assises, couchettes, voitures-lits, repas à la place ou en voitures-restaurants, trains-autos et motos accompagnées.

Les informations nécessaires au fonctionnement de la réservation sont notamment : les jours de circulation des trains, les gares desservies, les heures d'arrivée et de départ, les tarifs, les schémas-voitures composant ces trains et les places qui y sont déjà réservées avec leur parcours de réservation...

La procédure de réservation consiste à attribuer à un client demandeur, au guichet d'une gare, dans une agence de voyage ou par téléphone, une place dans un train correspondant à sa demande.

Les souhaits particuliers de la clientèle, tels que fumeur ou non fumeur, position fenêtre ou couloir en places assises, inférieure ou supérieure en couchettes ou voitures-lits, sont également satisfaits dans la mesure des disponibilités.

La S.N.C.F. a choisi de faire de la RESA un système centralisé sur un seul site, Batignolles, où sont regroupés les données et les traitements. Par ailleurs, les terminaux sont largement distribués sur tout le territoire, dans les gares moyennes et grandes, ainsi que dans les agences de voyages assez importantes. Au départ, ces terminaux étaient soit entièrement passifs (Olivetti), soit disposant d'une aide par projection de diapositives (IBM). Ils étaient reliés à l'ordinateur central par le réseau de télé-informatique de la S.N.C.F. à basse vitesse (110 à 150 bauds).

A la fin des années soixante-dix, l'apparition de terminaux dits « intelligents », c'est-à-dire ayant des possibilités locales de stockage de données et d'exécution de traitements, a permis le développement de nouveaux terminaux.

En particulier, en 1981, la réservation dans le T.G.V. a nécessité la mise en œuvre de procédures spéciales et l'implantation de distributeurs de « réservation rapide », en libre service, connectés à l'ordinateur central. Ceux-ci assurent localement le dialogue avec les clients du T.G.V., par affichage de textes et de questions, la délivrance du titre de réservation et la gestion du paiement en espèces.

Une autre transformation qui a modifié l'architecture initiale de la RESA a été l'évolution des réseaux de transmission de données. Les premiers réseaux, avec leur architecture « en étoile », étaient assez vulnérables. Dans les réseaux récents, le schéma des câbles est maillé : il y a plusieurs chemins possibles d'un « nœud » à un autre. Les caractères à transmettre sont groupés en « paquets » d'une centaine de caractères, chaque paquet étant transmis par le chemin le meilleur à l'instant de la transmission. Le nouveau réseau S.N.C.F., RETIF, en cours d'installation, répond à ces critères, avec un objectif de 1 200 bauds pour les terminaux RESA, permettant de transmettre un titre de réservation en quatre secondes seulement. TRANSPAC, le réseau public de transmission de données, mis en place à l'initiative des P.T.T. sur tout le territoire français, répond aux exigences de la S.N.C.F.

Ainsi, les agences de voyages peuvent, au lieu de s'équiper de terminaux spécialisés, louer simplement un terminal banalisé, permettant d'accéder, via TRANSPAC, aux différents systèmes de réservation (non seulement S.N.C.F., mais aussi Air Inter, Air France, UTA, S.N.C.M., clubs de vacances, tours opérators, chaînes d'hôtels, etc.). Il s'agit du projet ESTEREL (Ensemble spécialisé de tourisme et de réservation électronique), dont la première phase est opérationnelle depuis le début de l'année 1984.

Agence de voyages à domicile

Un nouveau mode de transmission apparaît aujourd'hui avec l'utilisation, non plus des réseaux spécialisés, mais des réseaux téléphoniques publics (P.T.T.) ou privés (S.N.C.F.) par le biais des Minitels, ces petits terminaux peu onéreux, installés en masse chez les particuliers par les P.T.T.

Tout possesseur d'un Minitel pourra, dès le milieu de 1985, réserver des places de façon analogue à la réservation par téléphone, en formulant directement sa demande à l'écran grâce au nouveau service : RESA-Télématique.

De même, les professionnels (agents S.N.C.F., agents de voyages...), dans les petites gares et les petites agences de voyages qui ne disposaient pas encore de terminal, profiteront de cette expansion du réseau S.N.C.F. : en adjoignant une machine « Mirabel » au Minitel, il sera possible d'obtenir une édition des réservations. On éliminera ainsi presque totalement les recopies manuelles de références de place, sources d'erreurs et de doubles réservations.

En outre, des accords entre les différents réseaux de chemins de fer européens ont permis, au sein de l'Union internationale des chemins de fer (U.I.C.), de définir des échanges standardisés entre les différents ordinateurs de réservation européens. Les trains en réservation de onze pays européens sont ainsi accessibles par tout terminal de la RESA (Allemagne, Belgique, Luxembourg, Autriche, Pays-Bas, Danemark, Suisse, Italie, Espagne, Portugal et, bien sûr, la France), en attendant la Grande-Bretagne.

Quinze ans après le début de la conception de la RESA, on mesure le chemin parcouru par les procédures de réservation, depuis les bureaux de réservation manuels gérant les schémas-voitures dans de grands classeurs à tambour, jusqu'à la réservation depuis son domicile dans le « Munich-Athènes ».

Grâce à l'apparition des ordinateurs sur les guichets, le rôle des vendeurs est devenu encore plus important et plus efficace pour la S.N.C.F. Comme pour toute entreprise, l'informatique est là pour les aider dans leur tâche de vente et leur permettre de satisfaire les usagers, voyageurs et expéditeurs, au mieux et dans les meilleurs délais.

J.-Marc Fabbro/SNCF

Michel Henri/SNCF

Gérer des wagons
Il faut pas moins de 1 200 terminaux pour gérer l'ensemble des wagons et leurs mouvements, et établir ainsi un suivi commercial et technique des matériels.

Le marché

Le nombre des transactions journalières pour le trafic voyageurs S.N.C.F. oscille entre 30 000 et 210 000.
En période normale, 700 à 800 trains sont quotidiennement en réservation, ce qui constitue un fichier d'environ 40 millions de places gérées. En période de pointe, ce volume peut atteindre 60 millions de places pour un mouvement dépassant 1 000 trains.

Matériels et logiciels

Le parc des équipements informatiques de la S.N.C.F., qui représente une valeur d'acquisition de 900 millions de francs et une capacité de 120 millions de caractères en mémoire centrale et 340 milliards de caractères en mémoire de masse, est très diversifié.
Il se compose de matériels Bull, Burroughs, IBM, Sperry, Logabax, pour ne citer que les plus importants. A noter que, dans ce parc, la part du matériel français, actuellement 54 %, devrait atteindre 61 % à la mi-1985.
Les logiciels écrits pour ces machines sont, pour les plus utilisés à la S.N.C.F., dans les langages suivants : PROTÉE, COBOL, PL1, FORTRAN. En outre, des langages proches de l'utilisateur final (langages *« end user »*) ont été récemment mis au point. Ils permettent à l'utilisateur, à partir de bases de données, la sélection et la manipulation d'informations, ce qui réduit d'autant les tâches à effectuer par les informaticiens.

De bonnes relations

Les logiciels de gestion de bases de données font généralement usage de méthodes dites « d'organisation relationnelle ». Voyons les principes de base de ce système.

La plupart des « gestionnaires » de bases de données sont du type « relationnel ». Cela signifie que chaque enregistrement d'un fichier prend la forme d'un tableau, constitué de lignes et de colonnes, un peu à la façon d'un tableur. L'information peut prendre des formes variées qui suggèrent des structures plus complexes.

On emploie le terme « relationnel » parce que chaque « ligne » de la base de données est liée, de façon précise et sans ambiguïté, à chaque « colonne ». Ce n'est pas là une manière très souple d'organiser les données, mais elle se révèle suffisante pour les plus simples des programmes de ce type. Les micro-ordinateurs domestiques — même les tout nouveaux modèles à 16 bits, disposant de 128 K de mémoire ou plus — ont des capacités mémoire limitées; ils restent un peu lents pour ce qui est du stockage et du traitement de l'information. Les limites d'un système relationnel représentent une part du prix à payer, dès lors qu'on veut pouvoir manipuler des bases de données à des conditions acceptables.

Une méthode tout à fait différente de cette organisation « tabulaire » consiste à réorganiser les données sous forme hiérarchique. Elles prennent la forme d'un arbre; les branches ont des « sous-branches », qui ont elles-mêmes des rameaux, des feuilles, etc.

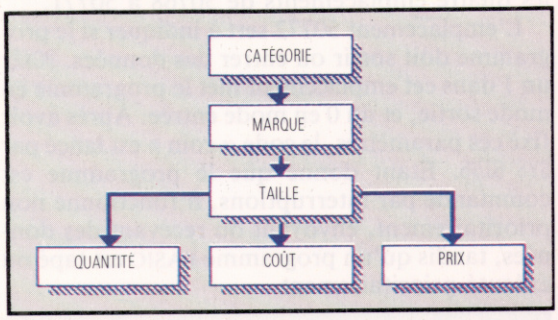
Pour illustrer ce point de vue, nous allons chercher comment créer une base de données permet-

Arbre analogique... En réorganisant les données sous forme hiérarchique, on obtient une structure en forme d'arbre, tout à fait comparable à celle d'un arbre généalogique. (Cl. SNARK/EDIMEDIA.)

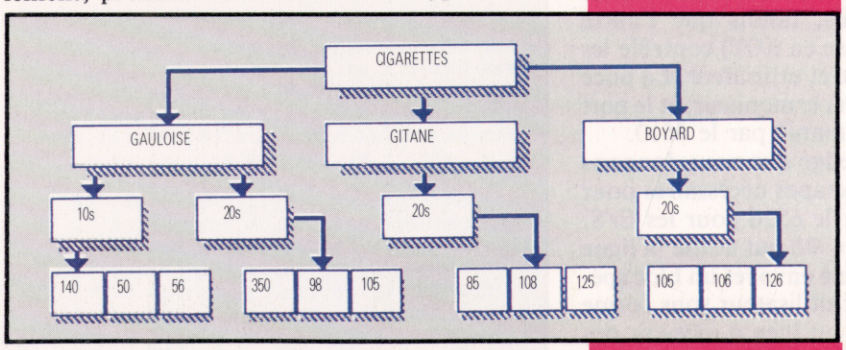
tant de traiter le stock d'un petit marchand de journaux-tabac.

CATÉGORIE	MARQUE	QUANTITÉ	STOCK	COÛT	PRIX DÉTAIL
CIGARETTES	GAULOISE	20	350	98	105
CIGARETTES	SENIOR SERVICE	10	140	50	56
CIGARETTES	BOYARD	20	85	108	125
CIGARETTES	BALTO	20	105	106	126
CONFISERIE	BOUNTY	1	106	14	17
CONFISERIE	MARS	1	95	12	16
CONFISERIE	BINGO	1	25	15	19
REVUES	ABC INFO	1	35	85	95
REVUES	VIE DU RAIL	1	12	60	80
REVUES	NEW LOOK	1	86	50	60
BOISSONS	LIMONADE	150	35	25	37
BOISSONS	VÉRIGOUD	150	40	18	27
BOISSONS	SPRITE	150	20	16	25

Représenté de façon hiérarchique, le tout aurait cette structure fondamentale :



Chaque enregistrement n'est pas organisé en « zones », mais en « fragments » de données, dont chacun peut correspondre à un ou plusieurs zones. Tout ce qui concerne les CIGARETTES, réorganisé hiérarchiquement, et non plus relationnellement, prendrait une forme de ce type :



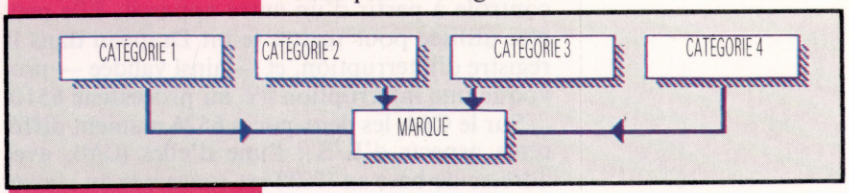
Liz Dixon

Cela a le gros avantage de présenter chaque élément d'information comme supérieur, ou subordonné à un autre. Il suffit de disposer d'un « pointeur » qui renvoie d'un « segment » d'information à un autre.

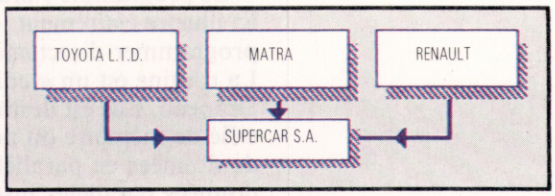
Supposons que notre détaillant n'ait en stock qu'une centaine de « catégories », mais qu'il propose en revanche près d'un millier de produits différents. Avec une base de données de type relationnel, nous aurions besoin de mille enregistrements, à raison de un par article. Un système de type hiérarchique permet de simplifier la présentation et de ne mettre en jeu que les cent catégories. Cela évite bien des répétitions fastidieuses, mais, il est vrai, donne à la base de données une organisation assez complexe.

Il existe encore un autre procédé, appelé CODASYL ou système en réseau. Ce type d'organisation des données a été mis au point par un groupe de recherche spécialisé du Conference on Data Systems Languages — d'où le nom. Le gros problème de la méthode hiérarchique est que l'information ne peut être organisée que dans un seul « sens »; pour reprendre notre analogie arboricole, une branche ne peut, par exemple, être rattachée à deux troncs différents. Or, c'est là un problème qui se pose quotidiennement. Pour en revenir à notre détaillant, CHIQUITO sera à la fois relié à SEITA et à CIGARES. Un même « élément » peut avoir plus d'un « créateur ». Par exemple, du point de vue du client, un pare-chocs est simplement une pièce automobile. Pour un garagiste, les choses sont moins évidentes; il doit aussi tenir compte de son origine.

Dans le système CODASYL, on fait usage d'un réseau d'ensembles (un peu au sens mathématique), dont chacun se compose d'un certain nombre d'enregistrements. Les dimensions de ceux-ci n'ont pas besoin d'être fixes (comme dans les bases de données relationnelles), et ils peuvent au besoin faire partie de plusieurs ensembles en même temps. Ceux-ci peuvent également se réduire à un seul enregistrement, tandis que ce dernier ne peut appartenir simultanément à plus d'une seule occurrence du même type d'ensemble. Il s'ensuit que des structures de ce genre ne sont pas envisageables :



C'est là une limitation assez gênante. Il est vrai que l'on a assez peu de chances de voir un article qui soit à la fois une REVUE, une BOISSON et une marque de CIGARETTES. Mais c'est parfaitement concevable pour une entreprise :



Liz Dixon

Les bases de données de type hiérarchique ou en réseau sont bien plus souples que les modèles relationnels, mais cette flexibilité a pour contrepartie une complexité qui ne les rend vraiment compétitives que sur mini-ordinateurs ou gros systèmes. Les ordinateurs personnels, quant à eux, mettent en œuvre, presque invariablement, des procédures de gestion relationnelles. On est donc amené, un jour ou l'autre, à faire face à la situation suivante : vous vendez certains articles (que ce soit des livres ou autre chose), et chacun d'eux a des fournisseurs différents. Il faut donc à la fois un relevé des articles, et une série d'enregistrements consacrés à tous ceux qui procurent le même élément.

Lignes d'échanges

Nous allons voir comment les deux puces CIA 6526 du C64 contrôlent les E/S selon le programme Parawedge destiné à envoyer ou recevoir un bloc mémoire spécifié par le port utilisateur.

Le Commodore 64 a deux puces CIA (adaptateur d'interface complexe) 6526, qui sont dédiées aux communications avec le monde extérieur. Ce ne sont pas les seules puces en rapport avec les entrées/sorties; les puces 6510 et vidéo manipulent, elles aussi, certaines E/S. Une puce 6526 a deux ports de données 8 bits, tous deux ayant des lignes programmables. La puce est adaptée à la communication 8 ou 16 bits, et elle a deux horloges 16 bits pouvant être couplées. De plus, elle a un registre de décalage 8 bits pour la communication série et, comme nous l'avons vu, une horloge programmable donnant l'heure du jour.

Elle a aussi deux lignes spécifiques d'entrée en communication — PC et Drapeau. PC décroîtra d'un cycle après avoir écrit une donnée au port B de la 6526, et peut servir à indiquer qu'elle est prête à recevoir des données d'un dispositif externe. La ligne Drapeau peut servir d'entrée de contrôle à partir d'un autre appareil. Elle peut être utilisée pour mettre le bit Drapeau dans le registre d'interruption, et — ainsi validée — provoque une interruption NMI au processeur 6510.

Sur le C64, les deux puces 6526 manient différents aspects d'E/S : l'une d'elles (CIA#1, avec adresse de base en $DC00) est consacrée au clavier et aux manches à balais, tandis que l'autre (CIA#2, avec adresse de base en $DD00) contrôle les données sur les ports série et utilisateur. La puce vidéo manipule les E/S vers le moniteur, et le port cassette est directement manié par le 6510.

Le programme Parawedge que nous donnons ici illustre clairement les étapes nécessaires pour programmer directement le 6526 pour les E/S. La routine est un « coin » NMI qui utilise la ligne Drapeau. Elle est destinée à envoyer un bloc spécifié de mémoire du port utilisateur sous forme de données en parallèle, ou bien à recevoir des données parallèles jusqu'à ce qu'un bloc spécifié de mémoire soit plein. Puisque c'est un code « coin », elle synchronisera les données en entrée ou en sortie sur les NMI, en laissant la machine libre de mener à bien d'autres tâches. Le seul hic est que, si le débit devient trop élevé, le C64 passera tout son temps en routines de service NMI, ce qui pourrait être extrêmement fâcheux.

Parawedge permet au C64 d'établir des communications bilatérales parallèles 8 bits avec un appareil extérieur — par exemple un autre ordinateur ou une imprimante parallèle — par le port utilisateur. Les broches du port utilisateur de PB0 à PB7 servent au transfert de données; la broche Drapeau 2 sert comme entrée en communication; PA2 signale une condition « prêt pour données » et PC2 une condition « données valides ». Pour utiliser le programme, il faut d'abord définir une zone de RAM à partir de laquelle vous voulez envoyer les données en sortie, ou vers laquelle vous voulez recevoir les données en entrée. Cela se fait en passant les adresses de début et de fin (sous forme octet lo/octet hi) vers le programme, en POKEant celles-ci dans les quatre emplacements de 50768 à 50771.

L'emplacement 50772 sert à indiquer si le programme doit sortir ou entrer des données. POKEr un 1 dans cet emplacement met le programme en mode sortie, et un 0 en mode entrée. Après avoir fixé ces paramètres, le code « coin » est lancé par SYS 50775. Étant donné que le programme est commandé par interruptions, il fonctionne non prioritairement, envoyant ou recevant des données, tandis qu'un programme BASIC est tapé ou exécuté prioritairement.

Programme Parawedge Commodore 64

Chargeur basic

```
1000 REM ** CHARGEUR BASIC PARAWEDGE **

1000 REM ** PARAWEDGE BASIC LOADER **
1010 DATA173,84,198,208,61,169,0,141,3
1020 DATA221,169,144,141,13,221,173,2
1030 DATA221,9,4,141,2,221,173,0,221,9
1040 DATA4,141,0,221,173,80,198,133,251
1050 DATA173,81,198,133,252,173,24,3
1060 DATA141,85,198,173,25,3,141,86,198
1070 DATA120,169,188,141,24,3,169,198
1080 DATA141,25,3,88,96,169,255,141,3
1090 DATA221,169,144,141,13,221,173,24
1100 DATA3,141,85,198,173,25,3,141,86
1110 DATA198,120,169,234,141,24,3,169
1120 DATA198,141,25,3,88,96,169,144,44
1130 DATA13,221,240,36,173,1,221,145
1140 DATA251,230,251,208,2,230,252,173
1150 DATA82,198,197,251,173,83,198,229
1160 DATA252,144,49,173,0,221,41,252
1170 DATA141,0,221,9,4,141,0,221,108,85
1180 DATA198,169,144,44,13,221,240,246
1190 DATA177,251,141,1,221,230,251,208
1200 DATA2,230,252,173,82,198,197,251
1210 DATA173,83,198,229,252,144,3,108
1220 DATA85,198,120,173,85,198,141,24,3
1230 DATA173,86,198,141,25,3,88,108,24
1240 DATA3
1250 DATA25596:REM*CHECKSUM*
1260 CC=0
1270 FORI=50775TO50971
1280 READX:CC=CC+X:POKEI,X
1290 NEXT
1300 READX:IFCC<>XTHENPRINT"CHECKSUM
  ERROR"
1310 END
```

Entrée de Parawedge
Nous donnons un listage en code d'assemblage pour Parawedge, qui peut être entré et assemblé avec un assembleur.
Le programme peut également être entré sous forme d'une série de DATA en tapant et en exécutant le code chargeur BASIC.

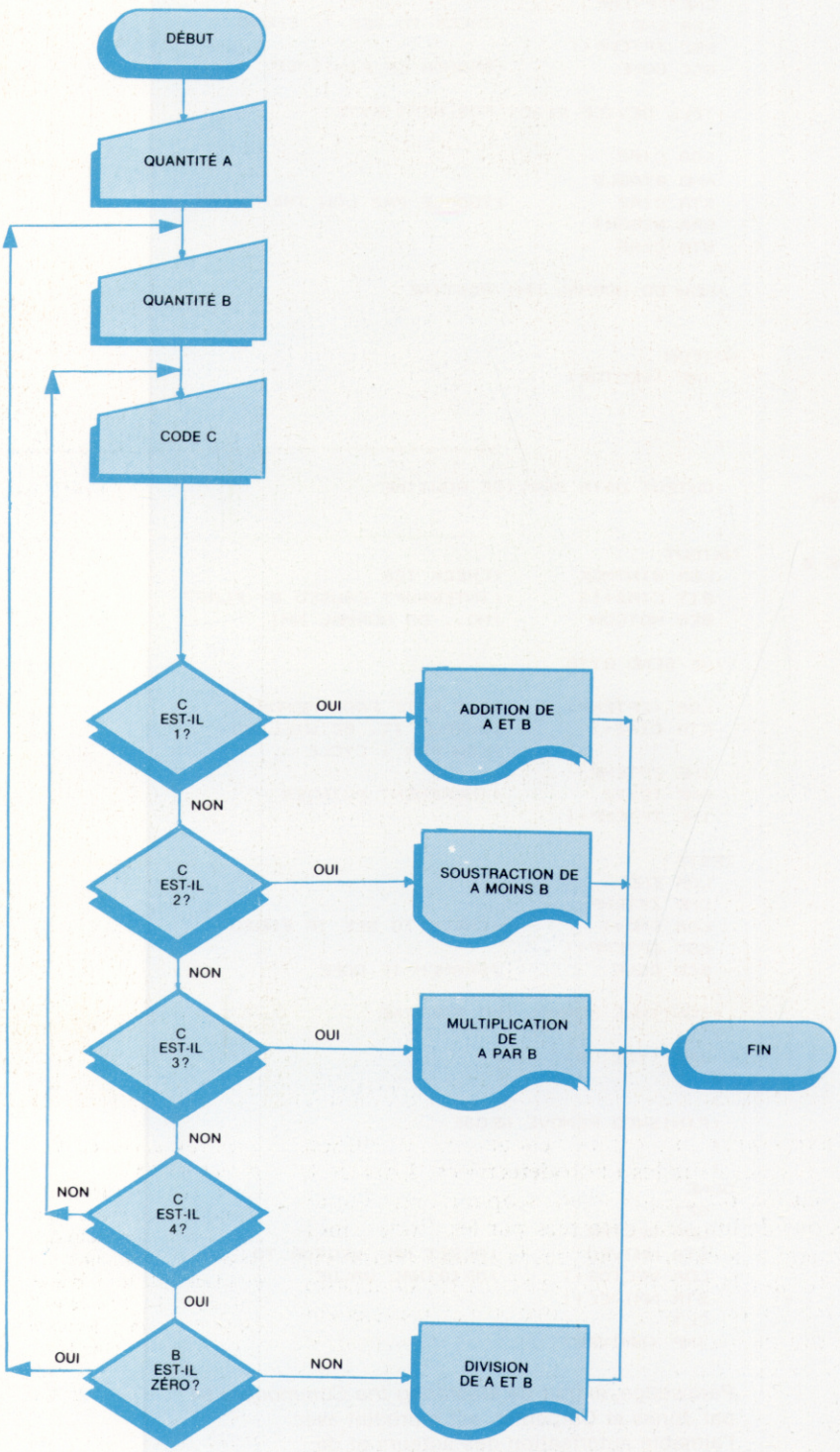
Listage d'assemblage

```
;*****************************************************
;*                                                   *
;*         PARAWEDGE - UN PROGRAMME COIN             *
;*                                                   *
;*  POUR L'ENVOI ET LA RECEPTION DE COMMUNICATIONS   *
;*                                                   *
;*         PARALLELES 8 BITS SUR CBM 64              *
;*                                                   *
;*****************************************************

   CIA2 = $DD00       ;6526 CHIP BASE ADDR
   OUTPUT = $FF
   INPUT = $00
   OUTSHK = $04
   INTMSK = $90
   TOGHI = $04
   TOGLO = $FC
   NMIVEC = $0318
   ZPTEMP = $FB
   * = $C650
   START *=*+2        ;START ADDRESS
   END *=*+2          ;END ADDRESS
   MODE *=*+1         ;INPUT/OUTPUT FLAG
   VECTOR *=*+2       ;STORAGE FOR NMI VECTOR

   LDA MODE           ;INPUT OR OUTPUT
   BNE OUTDAT         ;BRANCH IF OUTPUT
   LDA #INPUT
   STA CIA2+3         ;SET DDR FOR INPUT
   LDA #INTMSK
   STA CIA2+13        ;FLAG INTERRUPTS DISABLED
   LDA CIA2+2
   ORA #OUTSHK
   STA CIA2+2         ;SET PA2 FOR OUTPUT
   LDA CIA2
   ORA #TOGHI
   STA CIA2           ;SET HANDSHAKE LINE PA2 HIGH
   LDA START
   STA ZPTEMP
   LDA START+1        ;MOVE POINTERS TO ZERO PAGE 0
   STA ZPTEMP+1
  ;
  ;INITIALISE INPUT WEDGE
  ;
   LDA NMIVEC
   STA VECTOR         ;SAVE OLD NMI VECTOR
   LDA NMIVEC+1
   STA VECTOR+1
   SEI                ;
   LDA #<NXTIN        ;
   STA NMIVEC         ;INSERT DATA-INPUT WEDGE
   LDA #>NXTIN        ;
   STA NMIVEC+1       ;
   CLI                ;
   RTS
  ;
  ;INITIALISE OUTPUT WEDGE
  ;
OUTDAT
   LDA #OUTPUT
   STA CIA2+3         ;SET DDR FOR OUTPUT
   LDA #INTMSK
   STA CIA2+13        ;FLAG INTS DISABLED
  ;
  ;
   LDA NMIVEC
   STA VECTOR
   LDA NMIVEC+1       ;SAVE OLD NMI VECTOR
   STA VECTOR+1
   SEI                ;
   LDA #<NXTOUT       ;
   STA NMIVEC         ;INSERT DATA-OUTPUT WEDGE
   LDA #>NXTOUT       ;
   STA NMIVEC+1       ;
   CLI                ;
   RTS
  ;
  ;
  ;INPUT DATA SERVICE ROUTINE
  ;
  ;
NXTIN
   LDA #INTMSK        ;CHECK ICR
   BIT CIA2+13        ;INT CAUSED BY FLAG?
   BEQ NOTCOM         ;NO.. NORMAL NMI
  ;
  ;OK BYTE ON PORT
  ;
   LDA CIA2+1         ;READ BYTE
   STA (ZPTEMP),Y     ;STORE IN MEMORY
   INC ZPTEMP
   BNE TEST1          ;INCREMENT POINTER
   INC ZPTEMP+1
  ;
TEST1
   LDA END
   CMP ZPTEMP
   LDA END+1          ;CHECK TO SEE IF ENDED
   SBC ZPTEMP+1
   BCC DONE           ;BRANCH IF FINISHED
  ;
  ;TELL DEVICE READY FOR NEXT BYTE
  ;
   LDA CIA2
   AND #TOGLO
   STA CIA2           ;TOGGLE PA2 LOW THEN HIGH
   ORA #TOGHI
   STA CIA2
  ;
  ;NOW DO NORMAL NMI ROUTINE
  ;
NOTCOM
   JMP (VECTOR)


  ;
  ;
  ;OUTPUT DATA SERVICE ROUTINE
  ;
  ;
NXTOUT
   LDA #INTMSK        ;CHECK ICR
   BIT CIA2+13        ;INTERRUPT CAUSED BY FLAG?
   BEQ NOTCOM         ;NO.. DO NORMAL NMI
  ;
  ;OK SEND BYTE
  ;
   LDA (ZPTEMP),Y     ;GET BYTE FROM MEMORY
   STA CIA2+1         ;OUTPUT IT. PC WILL GO
                      ;LOW FOR 1 CYCLE
   INC ZPTEMP
   BNE TEST2          ;INCREMENT POINTER
   INC ZPTEMP+1

TEST2
   LDA END
   CMP ZPTEMP
   LDA END+1          ;CHECK TO SEE IF ENDED
   SBC ZPTEMP+1
   BCC DONE           ;BRANCH IF DONE
  ;
  ;CONTINUE NORMAL NMI ROUTINE
  ;
   JMP (VECTOR)
  ;
  ;
  ;FINISHED REMOVE WEDGE
  ;
  ;
DONE
   SEI
   LDA VECTOR
   STA NMIVEC         ;RESET NMI VECTOR TO
   LDA VECTOR+1       ;ORIGINAL VALUE
   STA NMIVEC+1
   CLI
   JMP (NMIVEC)
```

Parawedge, extrait de *Mastering the Commodore 64* par Jones et Carpenter, est reproduit avec l'aimable autorisation des auteurs et de Ellis Horwood Ltd.

Test en cascade (II)

Nous avons déjà comparé entre elles une variable et une constante. Nous poursuivons l'étude de ce test et introduisons un nouvel élément.



Dans le premier test en cascade, nous avions pris comme exemple les jours de la semaine pour voir comment éliminer une comparaison ; dans ce cas précis, elle découlait de la pure logique, puisque l'on savait qu'il n'y avait que sept jours dans la semaine. On peut donc réaliser la même opération avec les doigts de la main, ou les mois de l'année. Ces opérations sont possibles, car les exemples choisis appartiennent à quelques séries limitées dont le nombre d'éléments est déjà établi et fixe ; on voit immédiatement que, dans bien d'autres cas, il en ira tout différemment.

Ainsi, nous arrivons à la conclusion qu'il est nécessaire de contrôler quelques valeurs déterminées mais variables, pour pouvoir éliminer une question parmi l'ensemble des autres. On devra donc inclure une question générale au début de la série pour que le renseignement introduit soit filtré convenablement avant d'être comparé.

Nous développerons un exemple de deux façons différentes : la première est celle que nous voyons représentée sur l'organigramme. On n'a pas tenu compte ici du « filtre » initial. Nous verrons ultérieurement une représentation avec le filtre en question.

Sans filtre initial

Deux quantités étant introduites, on doit en inclure une troisième qui fait à la fois fonction de code et de marque de l'opération qui doit se réaliser avec les deux premières :

- 1 : somme
- 2 : soustraction
- 3 : multiplication
- 4 : division

Il convient de rappeler qu'il faut effectuer un contrôle pour que la seconde quantité, au cas où il s'agirait d'un code 4 (division), ne soit pas nulle, car cela nous donnerait une erreur. Dans cet exemple de test par élimination, le filtre qui détermine si le nombre est valide « se déplace » en passant par toutes les questions et après avoir vérifié que l'on ne peut opter pour aucune autre sortie. Une fois considéré comme valide, il n'est toutefois pas permis de trouver la démarche très recommandable, puisqu'elle implique une perte de temps et de travail, étant donné le questionnement à chaque contrôle. Dans le cas où il y aurait une quantité importante de décisions, on voit bien le problème posé : que de temps passé avant d'arriver à la conclusion que le code introduit est erroné !

Le traceur Penman

L'utilisation de dispositifs robotiques peu coûteux, comme les tortues, est devenue de plus en plus populaire. Découvrons dans ce domaine le traceur Penman, qui nous vient de Grande-Bretagne.

Le traceur Penman est composé d'une unité de commande, d'un traceur mobile, d'un bloc d'alimentation, d'un câble de connexion RS232 et du logiciel associé. Lorsqu'ils ne sont pas utilisés, l'unité de commande et le traceur ne forment qu'une seule unité, qui mesure 55 × 128 × 335 mm. Pour retirer le traceur de son boîtier dans l'unité de commande, vous n'avez qu'à appuyer sur le bouton situé sous l'unité, glisser le traceur à l'extérieur et dérouler son câble-ruban. Dès que le traceur se trouve hors de l'unité de commande, on aperçoit trois trous où peuvent être introduits des stylos, ainsi qu'un orifice central permettant de se servir d'un autre stylo pour effectuer des graphiques de type tortue. Les stylos feutres de 40 mm de longueur sont maintenus au-dessus du papier au moyen de pinces à ressort. Lorsqu'ils reçoivent des commandes de traçage, le ressort est attiré vers le bas par des électroaimants fixés à des leviers internes, et le poids du stylo provoque un contact sur le papier.

Trois roues sont présentes sous l'unité; deux de celles-ci sont des roues d'entraînement. La troisième est une petite roue de plastique placée sur un galet pivotant et a pour fonction d'assurer l'équilibre de l'ensemble. Les roues d'entraînement, métalliques, sont recouvertes d'un matériel qui accroît leur friction sur le papier. Devant chaque roue, un capteur optique détecte le bord du papier en enregistrant la différence de brillance entre le papier et le fond sur lequel il est placé.

Moteurs standards

Trois électroaimants sont connectés à des leviers à l'intérieur du traceur; chacun de ceux-ci commande les mouvements du stylo — la paire de moteurs électriques standards fixés aux roues d'entraînement est adjacente. Ce traceur utilise des moteurs électriques standards et non des moteurs pas à pas ou des servomoteurs; ceux-ci permettent à l'unité de tracer des courbes régulières et non des courbes « saccadées ».

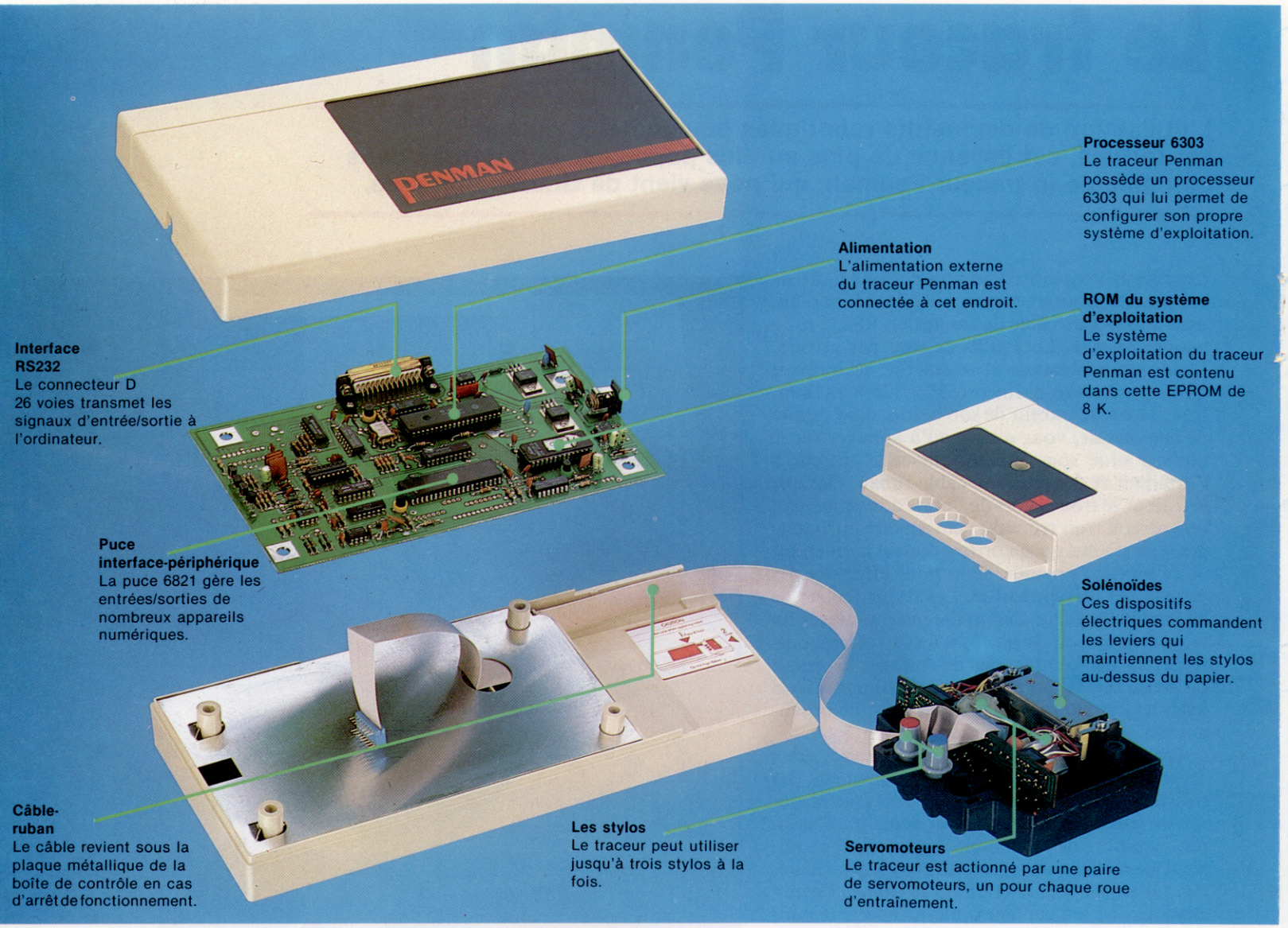
Cependant, ils exigent la présence d'un logiciel assez évolué, puisqu'il doit être en mesure de faire varier les tensions appliquées aux moteurs, et non simplement d'envoyer une série d'impulsions.

Finalement, une carte de circuits distribue l'alimentation et décode les signaux provenant de l'unité de commande, puis les envoie aux électroaimants ou aux moteurs. Sur chaque axe des moteurs, un disque stroboscopique accompagne la rotation du moteur; de chaque côté du disque, il y a deux diodes photodétectrices. Lors de la rotation du disque stroboscopique, des impulsions de lumière, détectées par les diodes, indiquent aux circuits logiques la vitesse du moteur afin que la machine puisse connaître sa position à tout moment. Cette logique est particulièrement utile lorsque le traceur Penman sert de « souris », puisque l'ordinateur doit savoir dans quelle direction et avec quelle rapidité les roues tournent, afin de déplacer le curseur sur l'écran.

La carte de circuits imprimés de l'unité de commande renferme trois puces principales. Il s'agit

Cl. Chris Stevens

Le traceur et son unité de commande
Le traceur Penman est composé d'un module de commande, connecté à un ordinateur via une interface RS232, et d'une tortue mobile. Le câble-ruban qui relie la tortue à la boîte de commande est bidirectionnel, ce qui signifie que les deux unités peuvent communiquer entre elles.



d'un microprocesseur 6303 (provenant du 6800), qui peut configurer une variété de systèmes d'exploitation. Le processeur se compose d'une RAM de zone de travail qui lui permet de stocker la position du traceur.

La seconde puce importante de cette carte est une EPROM qui contient les programmes de démonstration que peut effectuer le traceur Penman sans que le câble soit connecté.

Le traceur Penman peut être utilisé dans différents modes. Le mode « émulateur terminal » permet à l'unité d'être commandée du clavier. Pour passer à ce mode, vous devez soit charger directement le programme Pentlk, soit, en utilisant le traceur Penman comme une souris, charger le programme à partir du menu principal. Dès que le logiciel est chargé, les commandes sont envoyées à l'unité de commande via le port RS232 sous la forme de codes ASCII. Taper PRINT «I» initialise le traceur Penman. Il reçoit alors le signal et détermine lequel des trois rythmes de transmission (300, 1 200 ou 9 600 bauds) est utilisé. Cela étant fait, le traceur Penman est mis en position initiale en tapant la commande H. Le robot effectue alors une série de mouvements qui placent l'unité au coin supérieur gauche de la feuille de papier.

De cette position de départ, le traceur Penman essaie de trouver le bas de la page à l'aide de ses capteurs photosensibles. Dès qu'il trouve un bord, l'unité effectue une rotation à 90° et effectue la même intervention du côté gauche de la page. Afin de s'assurer que le contraste entre le papier et l'arrière-plan est suffisant, le constructeur a placé un morceau de papier noir sur lequel est posé le papier à dessin. Une fois revenu à sa position initiale, le traceur définit sa position d'origine à 50 mm de chaque bord du papier.

L'application Pentlk peut être exécutée dans l'un des deux modes directs (une commande à la fois), ou les commandes peuvent être fusionnées pour composer des programmes destinés à être chargés, sauvegardés et exécutés. Le mouvement en mode « terminal émulateur » est cartésien — ce qui signifie que le papier sur lequel est placé le robot est divisé en grille. Lorsque le traceur reçoit l'instruction d'aller à (500,500), il se déplace vers ces coordonnées au lieu de se dépla-

cer de 500 unités dans les deux directions. Une feuille de papier A4 a un maximum de 2 100 coordonnées sur l'axe des *x* et 2 970 sur l'axe des *y*. Les commandes peuvent être absolues ou relatives, selon le préfixe utilisé dans l'instruction MOVE, soit A, soit R. Les stylos sont sollicités par des commandes de type LOGO : U pour soulever le stylo, D pour le baisser et P pour le sélectionner.

Le traceur Penman peut aussi produire des graphiques tortue en mode direct, bien que les mouvements impliqués soient plus complexes que les mouvements cartésiens puisque la distance de déplacement doit être entrée en notation hexadécimale et avoir le préfixe $.

Cela s'explique par le fait que les graphiques tortue ignorent le logiciel normal qui interprète les mouvements cartésiens et interrogent directement les bits des adresses de commande de l'ordinateur. Cependant, c'est là la seule manière d'effectuer des graphiques tortue avec le traceur Penman. Les applications logicielles intégrées permettent au robot d'être commandé à partir de LOGO.

Commande en mode robotique

Un système similaire à celui qui contrôle le traceur Penman directement à partir du registre de direction de données du port utilisateur est utilisé pour commander le traceur en mode robotique. Chaque bit du registre commande un aspect différent des mouvements du robot — les bits 0 et 1 et les bits 2 et 3 commandent respectivement les moteurs de droite et de gauche. Les bits 4 et 5 exécutent les fonctions moteur générales comme la commutation des moteurs, tandis que le bit 6 commande le mouvement vertical du stylo. Le traceur Penman fournit également de l'information concernant son activité au registre de données, comme les erreurs de positionnement et la détection ou la non-détection du bord du papier par ses capteurs lumineux. Des commandes texte peuvent également être envoyées au traceur Penman. La gamme des commandes et de leurs applications est très similaire à celle des imprimantes/traceurs présents sur de nombreux ordinateurs domestiques. En fait, les formes des caractères tracés sont remarquablement semblables. La hauteur du texte varie de 1 à 127 mm, et le texte peut être imprimé dans quatre directions différentes.

Le traceur Penman peut aussi tracer un texte incliné, ce que ne peuvent faire les imprimantes/traceurs conventionnels. D'où un avantage appréciable.

Le manuel qui accompagne le traceur donne une liste complète des commandes disponibles, en expliquant brièvement comment elles sont prises en charge; de plus, le manuel présente les caractéristiques matérielles de l'unité. Il est peut-être un peu évolué pour le débutant; il semble qu'il ait été destiné aux amateurs qui en savent déjà assez long sur le fonctionnement de leur ordinateur.

Le traceur Penman représente un pas de plus vers la conception d'une tortue-traceur à usage général. Lorsqu'elle est adéquatement utilisée, l'unité offre une résolution qui commence à approcher celle des traceurs professionnels. Cependant, le logiciel existant ne permet pas encore à cette machine de servir d'outil professionnel. La série de programmes actuellement offerte vise principalement un usage éducatif. Les utilisateurs doivent écrire leurs propres programmes pour utiliser le traceur, ce qui leur permet d'apprendre les principes de base des logiciels robotiques.

Un utilisateur professionnel n'attache aucune importance au potentiel de programmation de l'unité, mais désire simplement se procurer une unité facile à utiliser.

En tant que périphérique éducatif, le traceur Penman apparaît comme un bon achat. Même s'il n'est pas le moins cher sur le marché, la variété des modes de fonctionnement offerts a de fortes chances de séduire de nombreuses écoles, où il pourrait servir à illustrer diverses applications.

TRACEUR PENMAN

PRIX

...

DIMENSIONS

335 × 128 × 55 mm.

INTERFACE

RS232, ce qui lui permet d'être connecté à tout ordinateur compatible RS232.

RÉSOLUTION

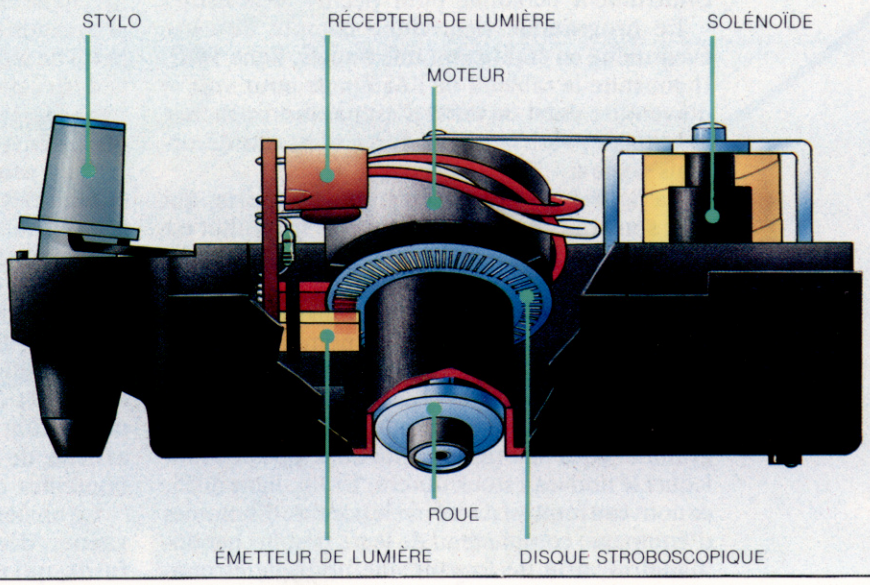
Dimension texte : 1 à 127 mm; résolution graphique : 0,1 mm.

DOCUMENTATION

Le manuel donne une profusion de détails techniques qui permettent aux utilisateurs d'exploiter le potentiel du traceur Penman; mais il n'y a pas assez d'explications destinées au réels débutants.

Positionnement

Pour permettre au système de commande de connaître la position du traceur, il était nécessaire de prévoir un mécanisme quelconque de réaction à l'intérieur du robot qui compterait le nombre de « pas » effectués. Pour cela, les fabricants ont fixé un disque métallique près de l'axe de la roue d'entraînement. Le disque est doté d'un certain nombre de fentes très fines sur sa bordure extérieure. Sur la carte des circuits, un dispositif émet et reçoit de la lumière. En tournant, le disque produit un effet stroboscopique entre l'émetteur et le récepteur lumineux, qui est traduit en une série d'impulsions. Le nombre d'impulsions détermine le nombre de pas effectués.



Chris Stevens

Cl. Kevin Jones

La mer est immense

Nous concluons la programmation des petits événements avec un module où nous pêcherons du poisson, recueillerons de l'eau fraîche, tandis que le temps...

Nous allons mettre au point un ensemble de petits programmes dont l'un sera choisi de façon aléatoire par le sous-programme ligne 5500. Nous avions déjà incorporé cinq événements différents. Si nous incorporons les nouveaux, nous arrivons à un total de 13, et c'est pourquoi la valeur de RM, ligne 46, doit être fixée à 13. Il faut aussi modifier la ligne 5525, pour ce qui est de l'instruction ON X GOTO, parce qu'elle renverra désormais à davantage de numéros de ligne. Le programme principal fait choix, chaque semaine, d'un événement mineur qu'il prend au hasard dans la liste.

```
5525 ON X GOTO 5540, 5570, 5570, 5570, 5570
5600, 5700, 5800, 5850, 5900, 5950, 6000,
6050.
```

Nous ajoutons tout d'abord à l'instruction ON...GOTO la ligne 5600 : c'est là que commence le sous-programme consacré à la prise du poisson. Si X reçoit une valeur de 6 (le nombre étant tiré au hasard), alors le programme principal se dirigera là.

Lorsqu'un membre de l'équipage meurt, sa disparition n'est pas prise en compte (dans la variable CN) avant la fin de la semaine; mais si par extraordinaire tous les autres matelots étaient morts durant la semaine en cours, il n'y aurait évidemment personne pour pêcher le poisson!

Le programme tient donc compte de cette éventualité en établissant une boucle, ligne 5610. Il consulte le tableau de l'équipage pour voir si d'aventure untel ou untel n'est pas mort pendant la semaine, vérifiant si sa force n'est pas désormais fixée à -999.

La ligne 5630 fait le décompte des morts, qui sont soustraits de l'effectif total. Si le résultat est inférieur à 1, il ne se passera rien, et l'on retournera en programme principal.

Si, en revanche, certains marins sont encore debout, la ligne 5650 génère un nombre aléatoire compris entre 11 et 20, qui représente, en kilogrammes, la quantité de poisson prise, le tout est ajouté ligne 5680 à la viande qui reste; le programme vous informe ensuite du temps pendant lequel le nouveau stock durera. Enfin, ligne 5685, ce nouveau total est divisé par le nombre d'hommes d'équipage, compte tenu de leurs besoins hebdomadaires, afin de fournir une nouvelle estimation relative au nombre de semaines pendant lesquelles on pourra y recourir.

L'événement suivant de ON X GOTO est une tempête. Elle se produit ligne 5700. Vos hommes sortiront des barriques afin de recueillir un peu d'eau de pluie, ce qui vous permettra de renouveler vos provisions d'eau. Là encore, ligne 5735, le programme génère un nombre aléatoire compris entre 11 et 20, qui représente le nombre de tonneaux ainsi récupérés; il est ajouté, ligne 5750, à ce qui restait encore. La ligne 5755 calcule et affiche ensuite une estimation permettant de savoir combien de temps le nouveau stock pourra être utilisé.

Les vents favorables constituent le huitième événement. Ils augmentent l'allure du vaisseau, et permettent de raccourcir d'une demi-semaine la longueur du trajet. C'est ce que fait la ligne 5835. L'événement numéro neuf correspond à du beau temps. Le sous-programme correspondant commence ligne 5850 (en neuvième position ligne 5525). Les membres de l'équipage ont ainsi l'occasion de se reposer un peu, ce qui est très profitable à leur santé : leurs forces, contenues en TS(,), sont donc augmentées. La ligne 5882 crée une boucle qui parcourt ce tableau. En effet, si la force d'un mort était augmentée, il reviendrait à la vie! La ligne 5884 s'assure donc de l'existence éventuelle de valeurs égales à 0 ou -999 (correspondant à des décès), et ignore les marins qui sont dans ce cas. Un nombre aléatoire compris entre 5 et 15 est ensuite ajouté, ligne 5886, aux forces de tous les survivants.

Les médicaments

La ligne 5900 marque le début d'un sous-programme consacré au dixième événement : la perte de médicaments. Le mauvais temps a causé la destruction des flacons, qui se sont brisés; les remèdes sont perdus, et cela aura des conséquences gênantes si la plupart des marins tombent malades. Le programme vérifie d'abord qu'il y a bel et bien des médicaments à bord, en examinant le tableau OA(), consacré aux marchandises, et plus précisément son premier élément; s'il a une valeur de 0 ou de -999, il ne se passera rien. Dans le cas contraire, la quantité existante est divisée par deux, ligne 5925. Il est vrai, cependant, que les bouteilles ne se brisent pas véritablement par moitié, et c'est pourquoi seule la partie entière du résultat sera retenue. Un message est alors affiché, de façon que le joueur sache combien de bouteilles il lui reste.

Le onzième événement est lié à la tempête : les vagues déchaînées provoquent la rouille des fusils, qui ne sont donc plus en état de fonction-

ner. Or, ils ont beaucoup d'importance (défense, commerce, poursuite de la nourriture). Le sous-programme est celui qui est consacré aux bouteilles brisées : y a-t-il des fusils à bord (ligne 5955)? Si oui, diviser la quantité par deux (ligne 5975), et afficher la partie entière du résultat.

Le douzième événement est introduit ligne 6000. Cette fois, les souris ont dévoré la moitié des ballots de tissu. Ceux-ci constituent le quatrième élément du tableau OA(), et le programme étudie donc OA(4) pour voir s'il y a du tissu à bord (ligne 6005). Si oui, les souris, dissimulées ligne 6025, rongent la moitié du tout, et le joueur est ensuite informé de ce qui reste.

A ce stade, il y a moyen de rendre le jeu plus facile, ou plus difficile, en modifiant les facteurs qui permettent l'apparition de tel ou tel événement.

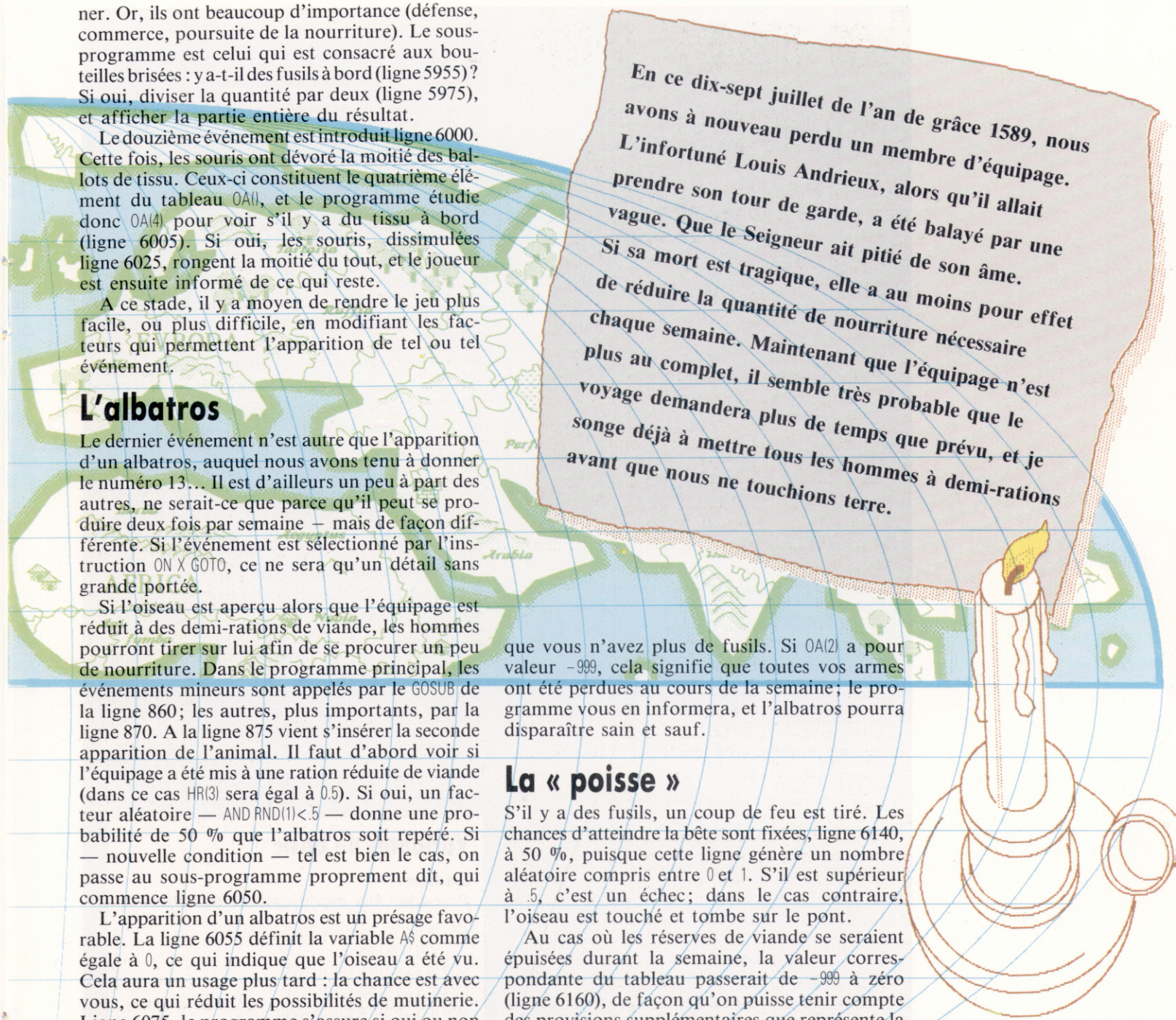
L'albatros

Le dernier événement n'est autre que l'apparition d'un albatros, auquel nous avons tenu à donner le numéro 13... Il est d'ailleurs un peu à part des autres, ne serait-ce que parce qu'il peut se produire deux fois par semaine – mais de façon différente. Si l'événement est sélectionné par l'instruction ON X GOTO, ce ne sera qu'un détail sans grande portée.

Si l'oiseau est aperçu alors que l'équipage est réduit à des demi-rations de viande, les hommes pourront tirer sur lui afin de se procurer un peu de nourriture. Dans le programme principal, les événements mineurs sont appelés par le GOSUB de la ligne 860; les autres, plus importants, par la ligne 870. A la ligne 875 vient s'insérer la seconde apparition de l'animal. Il faut d'abord voir si l'équipage a été mis à une ration réduite de viande (dans ce cas HR(3) sera égal à 0.5). Si oui, un facteur aléatoire — AND RND(1)<.5 — donne une probabilité de 50 % que l'albatros soit repéré. Si — nouvelle condition — tel est bien le cas, on passe au sous-programme proprement dit, qui commence ligne 6050.

L'apparition d'un albatros est un présage favorable. La ligne 6055 définit la variable A$ comme égale à 0, ce qui indique que l'oiseau a été vu. Cela aura un usage plus tard : la chance est avec vous, ce qui réduit les possibilités de mutinerie. Ligne 6075, le programme s'assure si oui ou non l'équipage manque de viande. Pour cela, il cherche à savoir si la quantité restante est égale ou plus grande que les exigences hebdomadaires des matelots, multipliées par le temps qui reste à parcourir avant que le voyage n'ait pris fin. Si les réserves sont suffisantes, l'albatros disparaît au loin et la ligne 6130 renvoie au programme principal.

Si, en revanche, il ne reste guère de viande, le joueur se voit informé d'un détail intéressant : l'oiseau pèse 10 kg; veut-il tirer sur lui? La ligne 6115 détermine la première lettre de la réponse. Il est bien sûr possible de ne pas tirer. Si toutefois vous avez absolument besoin de la chair de l'animal, la ligne 6133 vérifie d'abord que OA(2) n'est pas égal à zéro, ce qui indiquerait que vous n'avez plus de fusils. Si OA(2) a pour valeur -999, cela signifie que toutes vos armes ont été perdues au cours de la semaine; le programme vous en informera, et l'albatros pourra disparaître sain et sauf.



La « poisse »

S'il y a des fusils, un coup de feu est tiré. Les chances d'atteindre la bête sont fixées, ligne 6140, à 50 %, puisque cette ligne génère un nombre aléatoire compris entre 0 et 1. S'il est supérieur à .5, c'est un échec; dans le cas contraire, l'oiseau est touché et tombe sur le pont.

Au cas où les réserves de viande se seraient épuisées durant la semaine, la valeur correspondante du tableau passerait de -999 à zéro (ligne 6160), de façon qu'on puisse tenir compte des provisions supplémentaires que représente la carcasse de l'animal. La quantité de viande, PA(3) (dans le tableau des provisions), est augmentée de 10 kg, et le joueur se voit préciser quel est le stock actuel.

Il n'est pas toujours de bon augure de tuer un albatros, comme le savent les lecteurs du poète anglais Coleridge. Notre programme entreprend en tout cas de donner à B$, ligne 6162, la valeur 0. C'est le facteur de « poisse » qui vient s'opposer au facteur de chance de la ligne 6055. Nous vous préciserons plus loin ce qu'il faut entendre par là précisément.

Le joueur ne connaîtra pas exactement les raisons de ce qui se produit, mais certains facteurs seront modifiés de façon que la marche du navire se trouve ralentie.

Module 7 : encore de l'imprévu

Addition à la boucle principale

```
875 IF HR(3)=.5ANDRND(1) .5THENPRINTCHR$
```

S./P. Encore de l'imprévu

```
5600 REM EVENT 6 - ATTRAPER DES POISSONS
5605 X=0
5610 FORT=1TO16
5615 IFTS(T,2)=-999THENX=X+1
5620 REM COMPTE DES MORTS DE LA SEMAINE
5625 NEXT
5630 IFCN-X<1THENRETURN
5635 REM PAS D'ACTION SI TOUS MORTS
5640 PRINT
5645 S$=" PENDANT LA SEMAINE+":GOSUB9100
5646 PRINT:GOSUB9200
5650 S$="L'EQUIPAGE A PRIS+":GOSUB9100
5655 X=INT(RND(1)*10)+11
5660 REM DE 10 A 20 KG
5662 PRINTX;"KILOS DE POISSON"
5665 PRINT:GOSUB9200
5670 S$="VOTRE RESERVE DE VIANDE PEUT+":GOSUB9100
5675 S$="DESORMAIS SUFFIRE POUR ENVIRON+":GOSUB9100
5678 IFPA(3)=-999THENPA(3)=0
5680 PA(3)=PA(3)+X
5685 PRINTINT(PA(3)/(CN+PN(3)));SEMAINES"
5690 GOTO5530
5700 REM EV.7 EAU DE PLUIE
5705 PRINT
5710 S$=" PENDANT LA SEMAINE+":GOSUB9100
5715 PRINT:GOSUB9200
5720 S$="LES PLUIES ONT PERMIS+":GOSUB9100
5725 S$="DE REMPLIR D'EAU VOS TONNEAUX+":GOSUB9100
5730 PRINT:GOSUB9200
5735 X=INT(RND(1)*10)+11
5736 REM 10 TO 20 BARRELS
5740 S$="VOTRE RESERVE D'EAU PEUT+":GOSUB9100
5745 S$="DESORMAIS SUFFIRE POUR ENVIRON+":GOSUB9100
5748 IFPA(4)=-999THENPA(4)=0
5750 PA(4)=PA(4)+X
5755 PRINTINT(PA(4)/(CN+PN(4)));:"SEMAINES"
5760 GOTO5530
5800 REM EV 8 - VENTS FAVORABLES
5805 PRINT
5810 S$="VENTS FAVORABLES TOUTE LA SEMAINE!+":GOSUB91
00
5815 PRINT:GOSUB9200
5820 S$="VOUS AVEZ BIEN AVANCE+":GOSUB9100
5825 S$="ET LA DUREE DU VOYAGE EST+":GOSUB9100
5830 S$="REDUITE D'UNE DEMI-SEMAINE+":GOSUB9100
5835 EW=EW-.5
5839 GOTO5530
5850 REM EV - 9 BEAU TEMPS
5855 PRINT
5860 S$="BEAU TEMPS TOUTE LA SEMAINE!+":GOSUB9100
5865 PRINT:GOSUB9200
5870 S$="L'EQUIPAGE EST DE BONNE HUMEUR+":GOSUB9100
5875 GOSUB9200
5880 S$="            ET EN MEILLEURE SANTE!+":GOSUB9100
5882 FORT=1TO16
5884 IFTS(T,2)=0 OR TS(T,2)=-999THEN5888
5886 TS(T,2)=TS(T,2)+INT(RND(1)*11)+5
5888 NEXT
5889 GOTO5530
5900 REM EV 10 - PERTE FLACONS
5905 IF DA(1)=0 OR DA(1)=-999THENRETURN
5910 PRINT
5915 S$="VOUS VOUS APERCEVEZ QUE LA MOITIE+":GOSUB91
00
5920 S$="DES FLACONS SE SONT BRISES+":GOSUB9100
5925 DA(1)=INT DA(1)/2)
5930 PRINT:GOSUB9200
5935 S$="IL NE RESTE PLUS QUE+":GOSUB9100
5940 PRINTDA(1);"DES BOUTEILLES VIDES"
5945 GOTO5530
5950 REM EV 11 - LA ROUILLE !
5955 IF DA(2)=0 OR DA(2)=-999THENRETURN
5960 PRINT
5965 S$"VOUS VOUS APERCEVEZ QUE LA MOITIE+":GOSUB9100
5970 S$="DE VOS FUSILS ONT ROUILLE+":GOSUB9100
5972 S$="ET NE PEUVENT PLUS SERVIR+":GOSUB9100
5975 DA(2)=INT(DA(2)/2)
5980 PRINT:GOSUB9200
5985 S$"IL NE VOUS RESTE PLUS QUE+":GOSUB9100
5990 PRINTDA(2);"FUSILS INUTILES"
5995 GOTO5530
6000 REM EV 12 - TISSU PERDU
6005 IF DA(4)=0 OR DA(4)=-999THENRETURN
6010 PRINT
6015 S$="VOUS VOUS APERCEVEZ QUE LA MOITIE+":GOSUB9100
6020 S$="DE VOS BALLOTS DE TISSU+":GOSUB9100
6022 S$="SONT RONGES PAR LES SOURIS+":GOSUB9100
6024 S$="ET N'ONT PLUS DE VALEUR+":GOSUB9100
6025 DA(4)=INT(DA(4)/2)
6030 PRINT:GOSUB9200
6035 S$="IL NE VOUS RESTE PLUS QUE+":GOSUB9100
6040 PRINTDA(4);"BALLOTS VIDES"
6045 GOTO5530
6050 REM EV 13 - ALBATROS
6055 PRINT:A$="Y"
6060 S$="UN ALBATROS SURVOLE LE BATEAU+":GOSUB9
100
6062 GOSUB9200
6065 S$="C'EST UN HEUREUX PRESAGE+":GOSUB9100
6068 S$="ET L'EQUIPAGE SE REJOUIT+":GOSUB9100
6070 PRINT:GOSUB9200
6075 IFPA(3)<(CN+PN(3)+(JL-WK+1))THEN6090
6080 REM PAS DE DISETTE DE VIANDE
6085 GOTO6122
6090 S$="VOUS ETES A COURT DE VIANDE+":GOSUB9100
6095 S$="ET CET OISEAU PESE 10 MG !+":GOSUB9100
6100 PRINT:GOSUB9200
6105 S$="AIMERIEZ-VOUS LE CAPTURER ?+":GOSUB9100
6110 INPUTI$
6112 PRINT:GOSUB9200
6115 IFLET$(I$,1)="o" THEN6133
6120 S$="C'EST AUSSI BIEN!+":GOSUB9100
6122 PRINT:GOSUB9200
6125 S$="L'ALBATROS DISPARAIT...+":GOSUB9100
6130 GOTO5530
6133 IFDA(2)=0ORDA(2)=-999THEN6180
6135 S$="ON TIRE SUR L'OISEAU.....+":GOSUB9100
6138 GOSUB9200:GOSUB9200
6140 IFRND(1)<.5THEN6150
6145 S$="...MAIS EN VAIN!+":GOSUB9100
6148 GOTO6122
6150 S$="ET IL TOMBE SUR LE PONT!+":GOSUB91
00
6155 PRINT:GOSUB9200
6160 IFPA(3)=-999THENPA(3)=0
6162 PA(3)=PA(3)+10:B$="o"
6165 S$="VOUS AVEZ MAINTENANT+":GOSUB9100
6167 S$="10 Kg DE VIANDE EN PLUS...+":GOSUB9100
6170 S$="MAIS PEUT-ETRE LA CHANCE+":GOSUB9100
6172 S$="VA-T-ELLE TOURNER...+":GOSUB9100
6174 GOTO5530
6180 S$="ET COMMENT FAIRE SANS FUSILS?+":GOSUB9100
6190 GOTO6122
```

Variantes de basic

Spectrum :

Procédez aux modifications suivantes :

```
875  IF HR(3)=.5ANDRND(1)<5.THENCLS
     :GO SUB 6050
5527 IF X=6 THEN GO TO 5600
5528 IF X=7 THEN GO TO 5700
5529 IF X=8 THEN GO TO 5800
5530 IF X=9 THEN GO TO 5850
5531 IF X=10 THEN GO TO 5900
5532 IF X=11 THEN GO TO 5950
5533 IF X=12 THEN GO TO 6000
5534 IF X=13 THEN GO TO 6050
5535 PRINT:S$=K$:GO SUB 9100
5536 LET I$+INKEY$:IF INKEY$=" " THEN GO TO 5536
6115 IF I$ (1 TO 1)="Y" THEN GO TO 6133
```

BBC Micro :

Procédez aux modifications suivantes :

```
875  IF HR(3)=.5AND RND(1)<.5
     THEN CLS: GOSUB6050
```

Connaître votre ordinateur

Voici une première sélection de quelques livres concernant les ordinateurs les plus répandus sur le marché français pour vous permettre de mieux exploiter toutes leurs possibilités.

Vous avez acquis un ordinateur avant d'aborder cette encyclopédie, ou bien vous avez fait cet achat au cours de sa lecture. Peut-être avez-vous déploré que l'on parle peu des particularités de votre machine, et vous êtes-vous senti un peu seul à parler votre « dialecte » BASIC.

Mais rassurez-vous. Pour tous les micro-ordinateurs présents sur le marché, il existe une bibliographie plus ou moins abondante. Aussi avons-nous sélectionné pour vous quelques titres d'ouvrages consacrés aux matériels les plus répandus en France. Ils vous aideront à maîtriser parfaitement votre ordinateur et, dès lors, vous serez à même d'adapter la plupart des programmes présentés dans « ABC Informatique » à votre machine. Cette première sélection est plus particulièrement tournée vers les matériels Thomson. Nous verrons ultérieurement des livres consacrés, entre autres, aux ordinateurs Alice de chez Matra et à ceux de Philips.

Thomson-S.I.M.I.V.

Aller plus loin en Basic TO7

Les onze programmes originaux présentés ici mettent en valeur l'utilisation pratique de l'ensemble des instructions du BASIC TO7. L'auteur explique dans le détail la démarche qu'il a suivie en insistant sur les diverses étapes : conception d'ensemble, rédaction des blocs fonctionnels, vérification et mise au point des blocs, etc.

Par J.C. Wanner.
312 pages. Format 17 × 22 cm.
Eyrolles. Collection Microplus.

La conduite du MO5

Ce livre vous permettra d'exploiter toutes les capacités de votre micro. Vous apprendrez à faire tenir 16 K de mémoire vive avec seulement 8 K d'adresses, à changer de couleur sans utiliser les commandes BASIC. Vous trouverez un programme vous permettant de définir automatiquement des formes, puis de les sauvegarder sur cassette. Une large place est accordée à l'explication des commandes graphiques.

Par J.Y. Astier et A. Kauf.
152 pages. Format 14,5 × 21,5 cm.
Eyrolles. Collection Micro-ordinateurs.

La conduite du TO7-70

Obtenez le maximum de votre ordinateur TO7-70 grâce à ce livre, qui vous apprend comment en utiliser toute la mémoire et en maîtriser les possibilités graphiques. Vous apprendrez à utiliser le crayon optique, à déjouer les pièges de la programmation en langage machine, etc. Un détail d'une instruction vous fait défaut ? Vous y retrouverez l'instruction voulue, expliquée et accompagnée d'un exemple clair. Préférez-vous vous adonner à la programmation en assembleur ? Ce livre vous dévoile tout sur les points d'entrée du moniteur et ses variables système...

Par G. Guillon.
204 pages. Format 14,5 × 21,5 cm.
Eyrolles. Collection Micro-ordinateurs.

Pratique du MO5

Deux volumes contenant respectivement vingt-quatre programmes *(Niveau 1)* et vingt-huit programmes *(Niveau 2)*. Le premier apprend l'art et la manière d'utiliser le MO5. Vous saurez comment fonctionne ce micro-ordinateur, vous apprendrez à le programmer en BASIC, à exploiter les logiciels du commerce en cartouche ou cassette, en un mot, tout ce qu'il faut savoir lorsqu'on veut acquérir un MO5.

Thomson C.S.F.

Le second volume constitue le complément indispensable pour pousser plus à fond votre approche de l'informatique, et notamment de la programmation en BASIC. Vous apprendrez entre autres à « déboguer » un programme, à manipuler les chaînes de caractères, les sous-programmes, les fichiers, les tableaux, les graphismes et leur animation.

Par H. Lilen.
Format 21 × 29,7 cm.
Vol. 1 : 192 pages.
Vol. 2 : 176 pages.
Éditions Radio.

Pratique du TO7-70

Voici encore deux volumes, de vingt-quatre programmes chacun. Le *Niveau 1* s'adresse aux tout débutants, et leur explique tout ce qu'il faut savoir pour exploiter ce micro-ordinateur : en quoi consiste une telle machine, comment on la connecte et on la met en marche, comment on exploite des programmes tout faits (jeux, enseignement, petite gestion familiale), en cartouche ou en cassette, et comment on la programme en BASIC. Vérifiez et exploitez vos nouvelles connaissances en exécutant les deux douzaines de programmes, exercices ou jeux proposés. Suite du précédent, le *Niveau 2* développe les notions de BASIC acquises, les généralise et vous explique les règles importantes, et en particulier l'art de « déboguer » un programme. Il vous enseigne également le traitement des tableaux, des chaînes, des sous-programmes, des fichiers. Vous apprendrez à créer des « modules » graphiques, à les animer, etc. Grâce aux programmes, exercices ou jeux que vous découvrirez tout au long de ce livre, vous acquerrez la maîtrise du BASIC.

Par H. Lilen.
Format 21 × 29,7 cm.
Vol. 1 : 192 pages.
Vol. 2 : 176 pages.
Éditions Radio.

Apprivoiser TO7, MO5, TO7-70

Cette initiation s'adresse surtout, par son contenu et sa présentation aérée et illustrée de dessins, aux jeunes à partir de 10 ans. Guidé et conseillé par le prince Silicone, cousin du personnage imaginé par Saint-Exupéry, le lecteur apprendra à conserver en BASIC et découvrira la programmation au travers d'exemples aboutissant à un jeu complet : le jeu des drapeaux du monde. L'ouvrage est complété par une cassette contenant des jeux, des utilitaires et un logiciel éducatif.

Par B. Dupuy et B. Violet.
175 pages. Format 18 × 25 cm.
Foucher éditeur.

MO5 - TO7 vos programmes

Les vingt programmes proposés ici exploitent toutes les capacités des trois micro-ordinateurs de Thomson (MO5, TO7 et TO7-70) : graphisme,

couleur, musique, crayon optique. Les sujets abordés vont des jeux de réflexes aux spectacles musicaux ou graphiques, en passant par les jeux de réflexion, de hasard, de stratégie ou de mémoire, tous originaux, pour un ou plusieurs joueurs. Les listages sont commentés et structurés pour permettre au lecteur de les comprendre et de les adapter.

Par S. Pouts-Lajus et P. Champeaux.
130 pages. Format 15 × 23 cm.
Cedic Nathan.

Passeport pour Basic TO7 et TO7-70

Très facile d'usage et très pratique, ce livre s'adresse aussi bien au débutant qu'au programmeur averti. Tous les mots clés sont répertoriés dans l'ordre alphabétique, accompagnés d'un programme et d'une explication détaillée.

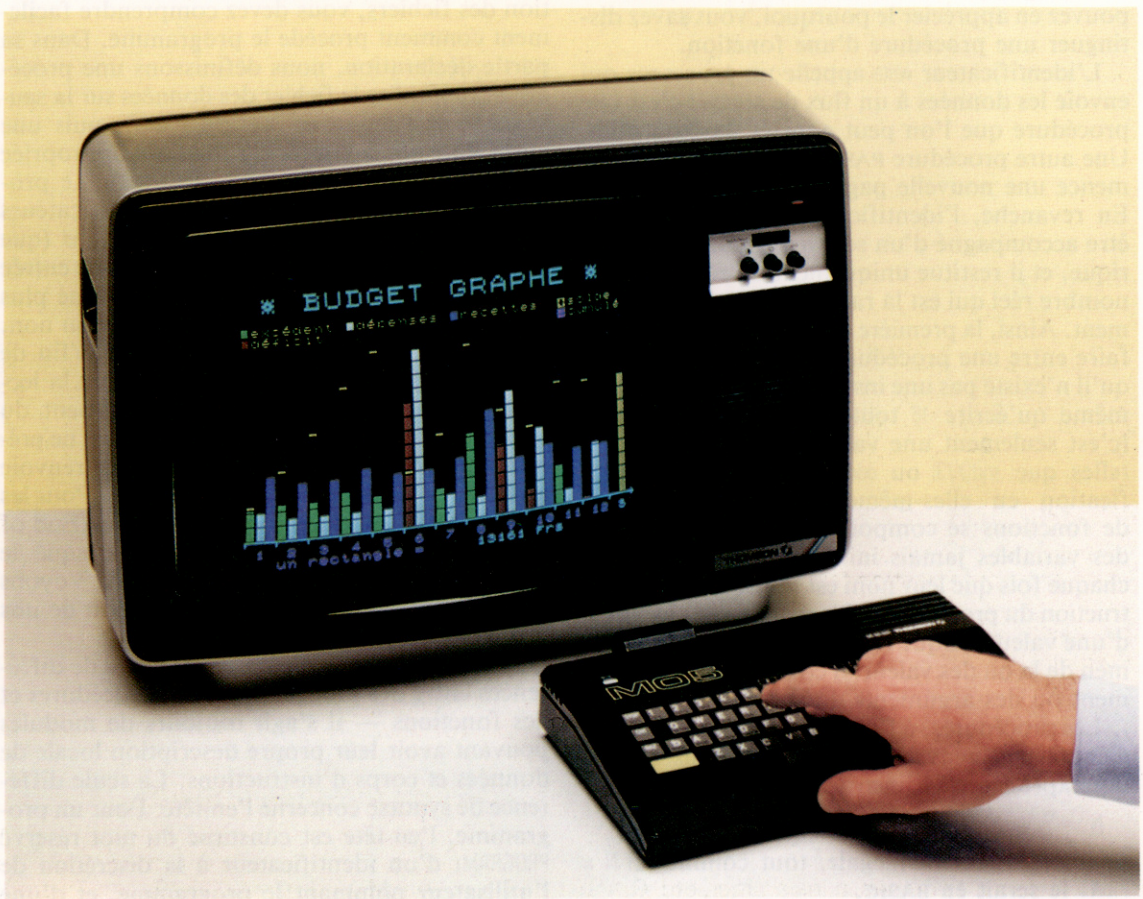
Par C. Galais.
160 pages. Format 11,5 × 16,5 cm.
E.T.S.F. Collection Poche-informatique.

MO5 premiers programmes

Écrivez votre premier programme BASIC sur MO5 en moins d'une heure !
D'une présentation claire, comportant de nombreux diagrammes et illustrations en couleurs, ce livre vous enseigne les bases de la programmation en BASIC sur MO5. Avec lui, vous apprendrez à programmer en quelques heures, quels que soient votre âge et votre formation.

Par R. Zaks.
245 pages. Format 16 × 22 cm.
Sybex.

Thomson-S.I.M.I.V.



Thomson

Procédures et fonctions

On peut définir en PASCAL ses propres procédures et fonctions. Nous allons nous y attarder ainsi que sur les concepts très importants de « portée » et de « passage de paramètres ».

Nous allons définir nos propres procédures et fonctions, pour compléter celles du PASCAL, telles que write et sqrt. Rappelons la différence entre une procédure comme write et une fonction comme sqrt. Lorsque l'on écrit par exemple :

```
write ('Bonjour !')
```

nous envoyons sur le fichier standard de sortie la chaîne de caractères indiquée comme paramètre unique. En d'autres termes, write désigne un sous-programme qui sait comment envoyer au périphérique de sortie (la visu) des données sous la forme d'un flux de caractères. Mais que ferait l'instruction suivante ?

```
sqrt (256)
```

La réponse est, bien sûr, que cela générerait un message d'erreur à la compilation, l'instruction étant illégale. Par ailleurs :

```
write (sqrt (256))
```

ne serait pas seulement légal, mais afficherait également quelque chose comme 1.60000E+01 ; si vous pouvez en apprécier le pourquoi, vous savez distinguer une procédure d'une fonction.

L'identificateur write appelle un processus qui envoie les données à un flux de sortie ; c'est une procédure que l'on peut appeler par son nom. Une autre procédure PASCAL est page (F) qui commence une nouvelle page sur le fichier texte F. En revanche, l'identificateur sqrt doit toujours être accompagné d'un seul « argument » numérique, et il restitue uniquement un résultat — le nombre réel qui est la racine carrée de son argument. Ainsi, la première distinction que l'on peut faire entre une procédure et une fonction, c'est qu'il n'existe pas une instruction de fonction. De même qu'écrire 16 tout court ne signifie rien (c'est seulement une valeur), et des expression telles que sqrt(X/3) ou odd(N) n'ont pas de signification en elles-mêmes. Les identificateurs de fonctions se comportent davantage comme des variables jamais initialisées, mais évaluées chaque fois que leur nom est utilisé dans une instruction du programme. Le résultat sous la forme d'une valeur unique est obtenu à partir de l'examen de la ou des valeur(s) courante(s) de l'argument ou des arguments de la fonction.

Les procédures, à l'inverse, ne donnent pas de valeurs et ne peuvent donc pas être utilisées dans des expressions. L'instruction :

```
N: = WriteLn
```

est manifestement illégale, tout comme LET N = PRINT le serait en BASIC.

L'écriture de programmes importants en PASCAL est grandement facilitée par la possibilité que l'on a de donner un nom à ses propres procédures et fonctions, d'en contrôler l'accès et leur utilisation conjointe. De même que pour l'attribution de noms aux variables, le manque relatif de restrictions sur les identificateurs signifie que l'on peut librement choisir des noms appropriés et commodes pour nos sous-programmes. Soit le programme suivant :

```
PROGRAMME Incomplet (entrées, sorties, fichierdesdonnées) ;
{déclarations..}
BEGIN
   Ouvrir (fichierdesdonnées):
   WHILE NOT EoF (fichierdesdonnées) DO
      BEGIN
         read (fichierdesdonnées, élément) ;
         Traitement (élément)
      END
END
```

Bien que nous n'ayons pas encore traité la gestion des fichiers, vous devez comprendre facilement comment procède le programme. Dans sa partie déclaration, nous définissons une procédure qui localise le fichier des données sur la sauvegarde et l'ouvre en lecture (Ouvrir), puis une autre destinée à traiter de manière appropriée chaque élément de données (Traitement). Le programme utilise également deux identificateurs prédéfinis du PASCAL. La procédure read (que nous avons utilisée jusqu'ici pour lire une entrée de texte) peut être utilisée comme indiqué plus haut pour lire toute donnée, structurée ou non. La fonction booléenne EoF (End of File/fin de fichier) donne la valeur « vrai » lorsque la lecture a porté sur le dernier enregistrement du fichier donné en argument. A nouveau, en ne précisant pas l'identificateur du fichier, on renvoie au fichier d'entrée standard par défaut. Pour les fichiers texte seulement, la fonction EoLn (End of Line) donne le résultat « vrai », lorsque le dernier caractère de la ligne a été lu. Voyons comment cela s'applique à la définition de nos propres sous-programmes.

De manière fondamentale, il y a peu de différences entre des programmes, des procédures et des fonctions — il s'agit toujours de modules pouvant avoir leur propre description locale de données et corps d'instructions. La seule différence de syntaxe concerne l'en-tête. Pour un programme, l'en-tête est constitué du mot réservé PROGRAM, d'un identificateur à la discrétion de l'utilisateur nommant le programme, et d'une

liste de paramètres identifiant les fichiers avec lesquels le programme communique. L'en-tête d'une procédure comprend le mot réservé PROCÉDURE suivi de la liste des paramètres formels, identificateurs du type de chaque paramètre de données. Ainsi, par exemple :

```
PROCÉDURE Lignes (NombLignes : entier);
   VAR
      N : entier;
   FOR N ; = 1 TO NombLignes DO
   WriteLn
END;
```

A la fin d'une procédure, END est suivi d'un point-virgule et non d'un point comme à la fin d'un programme. NombLignes, identificateur de paramètre formel, reçoit sa valeur par l'instruction d'appel :

```
Lignes (5)
```

Cette procédure simpliste est néanmoins très utile lorsque nous voulons, pour des raisons de clarté de présentation, laisser plusieurs lignes blanches (cinq, ici). Elle nous épargne (ainsi qu'au compilateur) ces séquences fastidieuses d'instructions WriteLn séparées par des points-virgules. Cette procédure vous permet de laisser en blanc autant de lignes que vous le voulez, Lignes (10) laissera 10 lignes, Lignes (20), 20, et ainsi de suite.

Portée

Cet exemple illustre également la sûreté du PASCAL. La variable de contrôle de boucle FOR doit toujours être déclarée comme vraie variable locale. Nous n'aurions pas pu dire par exemple :

```
For NombLignes: = 1 TO NombLignes DO...
```

Même si c'était acceptable dans un programme principal, la sûreté de la boucle ne serait pas garantie si son contrôleur n'était pas local ou « relativement global ». Avez-vous déjà eu à déboguer un programme BASIC comme celui-ci :

```
300 FOR N = 1 TO T
400 GOSUB 2000
500 NEXT N
```

pour finir par découvrir qu'un sous-programme lointain, qui peut être appelé indirectement et conditionnellement, utilise également N. Le PASCAL interdit formellement une telle conduite et vous permet de gagner du temps de développement. Tout identificateur déclaré dans un pavé de programme a une région définie qui porte sur toute sa longueur. Cependant, sa « portée », c'est-à-dire la portion de programme depuis laquelle il est accessible, ne s'étend que depuis l'endroit de sa déclaration jusqu'à sa fin. Cette limitation peut être encore accrue par redéfinition. Dans l'exemple précédent, le N auquel il est fait référence dans la procédure Lignes est naturellement l'entier déclaré localement. Cette déclaration a priorité sur toute autre relativement globale lors de chaque activation de la procédure. Par rapport à l'étendue du programme principal (portée) dans l'exemple donné, les régions propres aux variables globales N et X sont constituées par l'ensemble du programme. La portée de X s'étend depuis sa déclaration jusqu'à la fin du programme, à l'exclusion du champ de la procédure A. Cela, parce qu'un autre X (SET OF char) est défini en tant qu'identificateur de type; sa région étant le pavé A. De manière similaire, dans B, N se réfère au paramètre formel de B, et non pas à la variable globale.

Bien que les aires de A et B soient la totalité du programme, la portée de B ne commence qu'à son point de définition. C'est-à-dire que lorsque vous utilisez A à l'intérieur de B, B est invisible à A. Cela renforce la logique implacable du PASCAL : aucun programme ne peut être exécuté avant d'avoir été écrit ! Cela explique également pourquoi les définitions et déclarations *doivent* figurer dans l'ordre suivant : CONST, TYPE , VAR, suivis des définitions de procédures et de fonctions ordonnées selon les besoins structurels du programme. De nombreux compilateurs PASCAL sont à un passage, c'est-à-dire qu'ils lisent le code source une seule fois. Cela serait rendu impossible sans cet ordre logique.

L'effort de devoir déclarer les données locales à chaque procédure est largement récompensé. La modularité est assurée en transmettant aux procédures toutes les valeurs pour les données sous la forme de paramètres. En effet, même s'il se peut que vous trouviez des programmes PASCAL qui utilisent des procédures sans listes de paramètres, c'est-à-dire en accédant de manière globale à toutes les données, cette pratique doit être absolument évitée. On peut du reste considérer qu'il s'agit là d'une des faiblesses (rares) du PASCAL, de permettre ce genre d'abus. Vous remarquerez que l'utilisation de N dans B renvoie à son paramètre formel local, l'entier N global étant temporairement inaccessible. Outre la sûreté inhérente à ce type d'organisation, cela

Portée

```
PROGRAM Portée
VAR
      N :entier;
      X :réel;
PROCÉDURE A (Y : réel);
      TYPE
         X = SET OF char;
      {etc.}
PROCÉDURE B (N:entier);
      {etc.}
BEGIN {Programme principal -
      Portée}
      ...
      A (succ (N)/3);
      B (N);
      A (X)
         {etc.}
END.
```

Dans le programme Portée esquissé ici, les régions et les portées des diverses procédures et variables sont indiquées dans la table suivante :

Proc/Var	Région	Portée
A	Principale	A, B, principale
B	Principale	B, principale
N (global)	Principale	A, principale
X (réel)	Principale	B, principale
X (dans proc A)	A	A
N (dans proc B)	B	B

permet à toute une équipe de programmeurs de travailler sur un grand projet sans avoir à se préoccuper des éventuels conflits d'identificateurs. L'appel de B transmet la valeur de l'identificateur global N en tant que paramètre réel. Ce dernier, également appelé B, est en fait une variable totalement distincte. Cela implique les points suivants :

• Une copie locale de la valeur transmise à un pavé est faite lors de l'accès à ce dernier.

• Aucune modification effectuée sur ce « paramètre valeur » à l'intérieur du pavé n'a d'effet sur le paramètre réel transmis depuis le point d'appel.

• La valeur transmise peut être une expression de type correction.

Naturellement, toute déclaration de procédure doit lister les paramètres réels de l'appel et en outre correspondre exactement en nombre, ordre et type à la liste des paramètres formels de l'en-tête.

Le mécanisme de transmission d'une valeur de paramètre peut être considéré comme l'initialisation de l'identificateur du paramètre formel à la valeur spécifiée dans la déclaration d'appel. Ainsi :

```
Lignes (succ(N + interligne)DIV 2)
```

suppose l'instruction suivante d'affectation dès l'accès à Lignes :

```
NombLignes: = succ(N + interligne)DIV2
```

De base à base

Le programme ValeurBase illustre l'utilisation des procédures avec paramètres de valeur. On peut prendre n'importe quelle base, de 2 (binaire), à 16 (hexadécimale). Les nombres décimaux saisis au clavier sont affichés selon la notation appropriée. Par exemple 32767 — MaxInt sur plusieurs petits compilateurs PASCAL — devient 7FFF en hexadécimal, 111111111111111 en binaire, ou 77777 en notation octale. Essayez les modifications suivantes :

• Pour assurer la représentation négative, nous devons convertir le nombre dans son complément à 2. Niveau machine, cela signifie lui affecter un signe négatif et ajouter 1. Pouvez-vous penser à un moyen simple de le faire en PASCAL ?

• Peut-être voulez-vous effectuer les conversions d'une autre manière ? Vous pouvez utiliser le programme comme modèle de conversion d'un nombre donné dans une base quelconque (de 2 à 16), en décimal. Vous pourriez incorporer cette procédure dans le programme ci-dessous. On vous donne une clé : pensez aux données et à l'édition des liens.

```
PROGRAM  ValeurBase ( entrées, sorties);
          {convertit  les  nombres  décimaux
           selon   n'importe   quelle   base
           depuis le binaire jusqu'à l'hex.}
CONST
          Colonnes = 79; {par écran/imprimante}
TYPE
          octet    = 0...255;
          cardinal = 0...MaxInt;
VAR
          nombre,
          base   : entier;
          Plein,
          légal  : booléen;
          {11111111111111111111111111111111}
PROCEDURE EcrireChiffre  (  chiffre  :  octet );
        {valeur  de  liste  [compte]   donné
         au chiffre}
   BEGIN




     IF chiffre IN [0 .. 9]
        THEN
           write ( chiffre : 1 )
        ELSE  {écrit de A .. F}
           CASE chiffre OF
              10 : write ( 'A' );
              11 : write ( 'B' );
              12 : write ( 'C' );
              13 : write ( 'D' );
              14 : write ( 'E' );
              15 : write ( 'F' );
           END   {CASE}
   END;  EcrireChiffre
         11111111111111111111111111111111
PROCEDURE Print ( n  : cardinal;
                  base : octet );
   CONST {ces données sont locales à Print}
         ChiffresMax= 32;
   TYPE
          limites  = 1 ..  ChiffresMax;
   VAR
          list   :  TABLEAU   [ limites ]   OF  octet;
          index  :  limites;
          compte :  octet;
   BEGIN
      compte := 0;
      REPEAT
           compte := succ ( compte );
           liste [compte] := N MOD base;
           N := NDIV base
      UNTIL N = 0;
           {affichez en premier MSB :}
      FOR compte := compte DOWNTO 1 DO
         EcrireChiffre ( liste [compte] )
   END; {Print}
         11111111111111111111111111111111
BEGIN  ValeurBase - Programme principal
   REPEAT
      WriteLn  (  'choisissez  un  nombre pour
      la base : ' );
      WriteLn ( '  2 .. 16 ' : Colonnes );
      WriteLn  (  '(tout autre avorte l'exé-
      cution)' : Colonnes );
      write ( 'Base ? ' );
      read ( base );
      legal := base IN [2 .. 16] ;
      IF legal THEN
         BEGIN
            write  (  'Nombre  (0  change  de
            base) ? ' );
            read ( nombre );
            Plein := nombre<= 0;
            WHILE NOT Plein DO
               BEGIN
                  write  (  nombre  : Colonnes
                  DIV 2, '  vers  la  base  ', base
                   : 1, ' est  ' );
                  PRINT ( nombre, base );
                  WriteLn;
                  write ( 'Nombre ? ' );
                  read ( nombre );
                  Plein := nombre  <= 0;
                  END

         END
   UNTIL NOT legal
END.
```