ABC INFORMATIQUE

N° 10

COURS D'INFORMATIQUE PRATIQUE ET FAMILIALE

Alice le merveilleux

L'animation informatisée

L'ordinateur parle

Les pirates de l'informatique

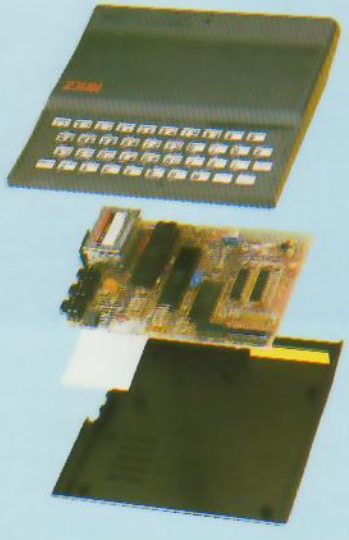
EDITIONS ATLAS

M6062-10-12F 85FB-3,80FS-$1.95

Sommaire

Prochain fascicule

- Le Sinclair ZX-81, premier micro produit en série, effectue de véritables miracles avec seulement quatre puces et est toujours considéré comme un exemple de conception.
- Les simulateurs de vol destinés à l'entraînement des pilotes utilisent des ordinateurs et sont très coûteux. Certaines de leurs fonctions ont été reprises sous la forme de jeux vidéo sophistiqués sur micro-ordinateurs.
- Il est possible d'envoyer des données d'un ordinateur à un autre au moyen d'un coupleur acoustique. Nous expliquons le fonctionnement de ce dispositif.



ABC INFORMATIQUE
cours d'informatique pratique et familiale
EDITIONS ATLAS

Édité par ÉDITIONS ATLAS s.a., tour Maine-Montparnasse, 33, avenue du Maine, 75755 Paris *Cedex 15*. Tél. : 538-52-70.

Belgique : ÉDITIONS ATLEN s.a., Bruxelles.

Canada : ÉDITIONS ATLAS CANADA Ltée, Montréal Nord.

Suisse : FINABUCH s.a., ÉDITIONS TRANSALPINES, Mezzovico.

Réalisé par EDENA s.a., 29, boulevard Edgar-Quinet, 75014 Paris. Tél. : 320-15-01.

Direction éditoriale : J.-Fr. Gautier. *Service technique et artistique* : F. Givone et J.-Cl. Bernar. *Iconographie* : J. Pierre. *Correction* : B. Noël.

VENTE

Les numéros parus peuvent être obtenus chez les marchands de journaux ou, à défaut, chez les éditeurs, au prix en vigueur au moment de la commande. Ils resteront en principe disponibles pendant six mois après la parution du dernier fascicule de la série. (Pour toute commande par lettre, joindre à votre courrier le règlement, majoré de 10 % de frais de port.)

Pour la France, s'adresser à ÉDITIONS ATLAS, tour Maine-Montparnasse, 33, avenue du Maine, 75755 Paris *Cedex 15*. Tél. : 538-52-70.

Pour les autres pays, s'adresser aux éditeurs indiqués ci-dessous.

SOUSCRIPTION

Les lecteurs désirant souscrire à l'ensemble de cet ouvrage peuvent s'adresser à :

France : DIFFUSION ATLAS, 3, rue de la Taye, 28110 Lucé. Tél. : (37) 35-40-23.

Belgique : ÉDITIONS ATLEN s.a., 55, avenue Huart-Hamoir, 1030 Bruxelles. Tél. : (02) 242-39-00. Banque Bruxelles-Lambert, compte n° 310-0018465-24 Bruxelles.

Canada : ÉDITIONS ATLAS CANADA Ltée, 11450 boulevard Albert-Hudon, Montréal Nord, H 1G 3J9.

Suisse : FINABUCH s.a., ÉDITIONS TRANSALPINES, zona industriale 6849 Mezzovico-Lugano. Tél. : (091) 95-27-44.

RELIEZ VOS FASCICULES

Des reliures mobiles, permettant de relier 12 fascicules, seront en vente en permanence chez votre marchand de journaux.

ATTENTION : ces reliures, présentées sans numérotation, sont valables indifféremment pour tous les volumes de votre collection. Vous les numéroterez vous-même à l'aide du décalque qui est fourni (avec les instructions nécessaires) dans chaque reliure.

En vente tous les vendredis. Volume I, n° 10.

ABC INFORMATIQUE est réalisé avec la collaboration de Trystan Mordrel *(secrétariat de rédaction)*, S.I. André Larochelle *(traduction)*, Ghislaine Goullier *(fabrication)*, Marie-Claire Jacquet *(iconographie)*, Patrick Boman *(correction)*. Crédit photographique, couverture : Photo Sivéa.

Directeur de la publication : Paul Bernabeu. Imprimé en Italie par I.G.D.A., Officine Grafiche, Novara. Distribution en France : N.M.P.P. Tax. Dépôt légal : mars 1984. 9843. Dépôt légal en Belgique : D/84/2783/27.

A NOS LECTEURS

En achetant chaque semaine votre fascicule chez le même marchand de journaux, vous serez certain d'être immédiatement servi, en nous facilitant la précision de la distribution. Nous vous en remercions d'avance.

Les Éditions Atlas

Dessins animés

Prenez le potentiel graphique de votre ordinateur, multipliez-le par mille, et vous avez un système d'animation sur ordinateur.

Le processus de réalisation de dessins animés, que ce soit sur film ou sur bande magnétoscopique, repose sur le fait que le cerveau ne peut figer une image. En présentant à l'œil une succession rapide d'images, une impression de mouvement est créée.

La première tentative d'illusion de mouvement à l'aide d'images fut l'utilisation d'un tambour à fentes multiples où l'on avait dessiné une bande d'images à l'intérieur. Une impression de mouvement était ainsi créée pour l'observateur qui regardait à travers les fentes. Cet appareil était antérieur à la photographie, mais bientôt des photographies remplacèrent les dessins à l'intérieur du tambour. L'étape suivante, le cinéma, nécessitait l'emploi d'une émulsion photographique relativement sensible, pouvant enregistrer une image en moins de un seizième de seconde, puisque les premiers films étaient projetés à 16 images par seconde.

Simulation du mouvement

Curieusement, il fallut un certain temps avant que l'industrie cinématographique n'ait l'idée de dessiner des images, les photographier et projeter le résultat pour obtenir des dessins animés. Puisque chaque seconde de projection demandait la création de 24 images (la vitesse de projection des films modernes), il est clair que la production d'un film de cinq minutes impliquait un travail colossal — 7 200 images dans ce cas. Il n'est pas surprenant que le style des illustrations n'était pas très élaboré. L'apparence de Bugs Bunny ne devait pas trop varier d'une seconde à l'autre!

De telles opérations précises et répétitives peuvent être effectuées par des machines. Lorsque l'ordinateur se charge du travail d'animation — c'est-à-dire ajuster la vitesse du mouvement, modifier les perspectives et la géométrie, définir les éclairages et les ombres —, l'artiste peut alors se concentrer sur la qualité de l'image. L'animation devient ainsi un véritable art graphique, l'artiste consacre tout son temps à la création de l'image pendant que l'ordinateur crée le mouvement.

Dans sa forme la plus simple, ce processus utilise des graphismes à plans-objets (voir page 152) pour créer les personnages, qui sont ensuite transférés sur l'écran et déplacés, ce qui produit le type d'animation utilisé dans des jeux vidéo très simples. Pour créer l'illusion du changement ainsi que celle du mouvement (par exemple un personnage qui marche), il est nécessaire de substituer de façon répétitive un objet à un autre. Comme nous l'avons vu, la création de plans-objets est plutôt lente, et la qualité graphique n'est pas excellente.

Pour une animation plus évoluée, le dessinateur/programmeur doit construire un algorithme qui introduit une profondeur dans l'image en respectant les règles de la perspective. Les objets peuvent être définis à l'écran selon leurs coordonnées X, Y et Z. Ici le programme ne doit pas reproduire les lignes

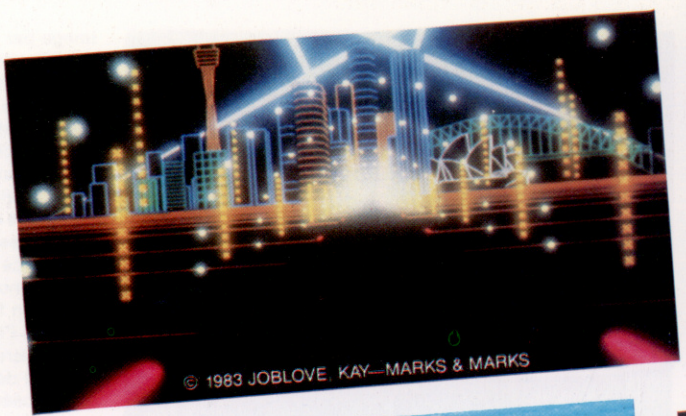
Image par image
Dans un dessin animé conventionnel, comme ces images de « La Panthère rose », l'artiste doit dessiner chaque image séparément. De plus, certains éléments communs ne sont redessinés que si leur apparence ou leur position ont été modifiées. On utilise un film transparent afin que l'image finale puisse être composée d'une série de dessins superposés. L'artiste se concentre sur les images importantes de la séquence, les autres images étant réalisées par des assistants. Les dessins terminés sont alors photographiés à l'aide d'une caméra spéciale, dans l'ordre de projection. (Cl. Richard Williams Animation Ltée.)

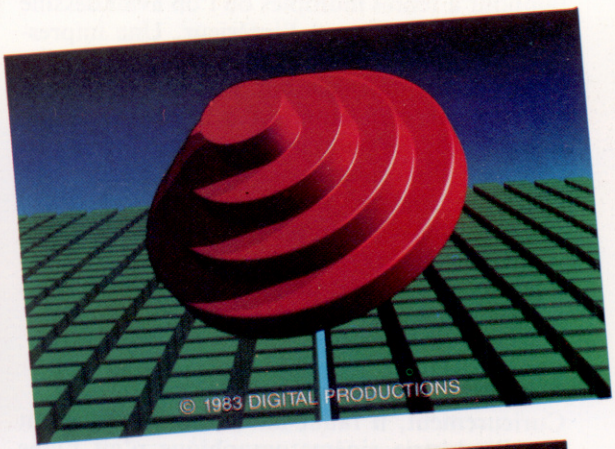
« cachées », ce qui lui permet d'introduire l'opacité dans ce qui n'était jusqu'à maintenant qu'une image transparente.

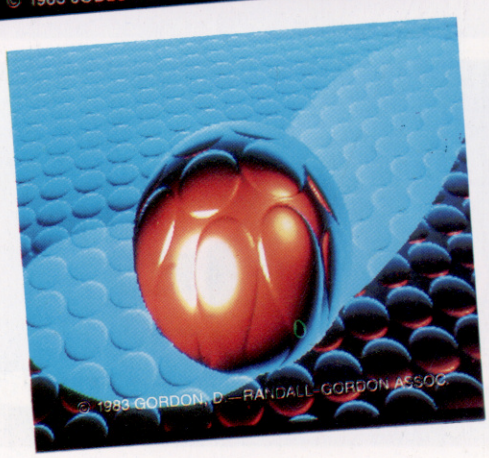
Il est ensuite nécessaire d'obtenir des courbes uniformes. Une ligne courbe n'est spécifiée qu'à l'aide de trois points, ses deux extrémités et le point le plus écarté de la droite joignant ces deux points. Évidemment, une courbe complexe (un S par exemple) doit être fractionnée en éléments plus simples, et il est important de pouvoir indiquer à la machine que la ligne en question est une courbe qui doit être arrondie et non un simple angle droit.

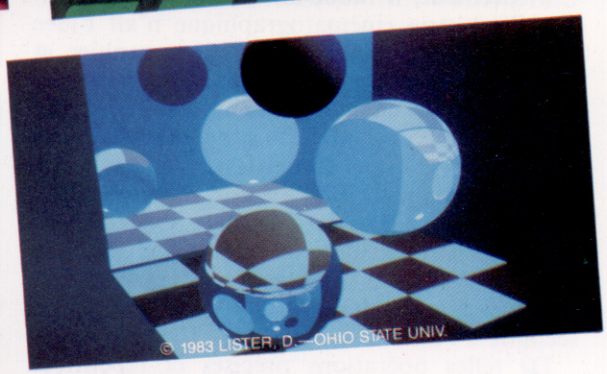
La machine doit aussi pouvoir introduire la lumière et les ombres dans le dessin. L'illustrateur doit d'abord spécifier la position de la source lumineuse. La face de l'objet dirigée vers la source lumineuse sera éclairée et ses contours progressivement ombrés. Un logiciel sophistiqué peut permettre la présence de plusieurs sources lumineuses et tenir compte de la réflexion d'un objet sur un autre.

Examinons maintenant le problème posé par la simulation de mouvement. Il est relativement facile de ramener un mouvement à ses composantes individuelles, même si l'objet représenté est aussi complexe qu'une main humaine. Le facteur déterminant est la capacité et la puissance de traitement de l'ordinateur utilisé. Rappelons que pour produire une image de qualité, il est nécessaire de disposer d'un moniteur d'une résolution approchant les 1 000 × 1 200 points. Chacun de ces points requiert au moins un octet pour définir sa couleur et sa brillance. Cela implique plus de 1 million d'octets par écran. La génération de dessins animés de haute qualité est donc impossible sur des ordinateurs domestiques. Les illustrateurs professionnels utilisent certains des ordinateurs les plus puissants et leurs tarifs le démontrent : le film final peut valoir jusqu'à 12 000 F la seconde.









Boîte magique
Il y a peu de temps, la méthode de production de séquences fixes et animées pour la télévision et le cinéma ressemblait à celle utilisée pour les revues ; le dessin était exécuté sur transparent ou sur papier et était ensuite photographié. Le système Quantel's Paint Box supprime l'utilisation de ces supports, le dessin est composé numériquement à l'aide de l'ordinateur et ensuite enregistré directement sur bande magnétoscopique.

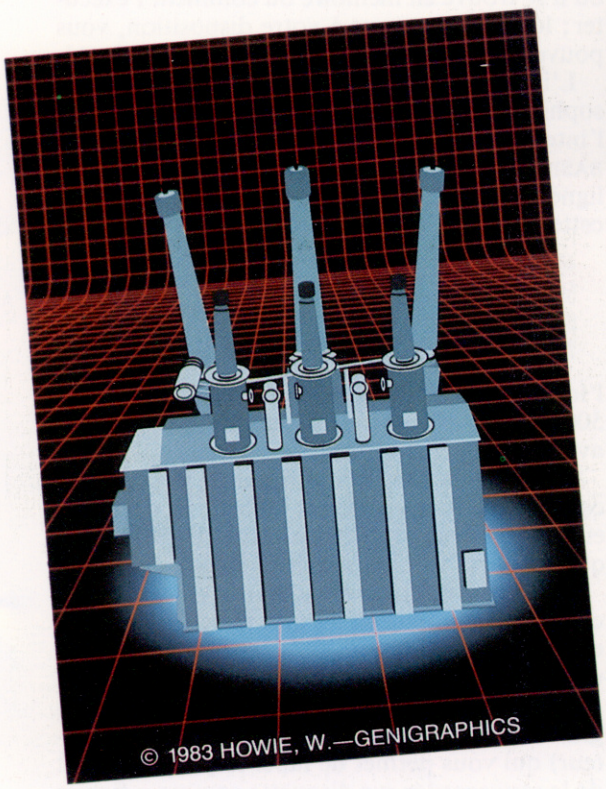
La couleur doit elle aussi être définie. Les ordinateurs domestiques les plus simples offrent jusqu'à 16 couleurs, mais les ordinateurs graphiques professionnels offrent généralement au moins 4 096 couleurs. Certains ne sont limités que par le nombre de bits de l'ordinateur. Pour 24 bits, l'ordinateur a quelque 16,7 millions d'options de couleur. Les fonctions de couleur et d'ombre sont regroupées. Ce procédé permet en outre de mettre en couleurs les vieux films en noir et blanc à l'aide d'un pointeur qui se déplace sur un écran vidéo.

Si nous prenons comme point de départ un objet simple comme un cube, il est relativement facile de comprendre comment il est possible de le déplacer sur l'écran. Un cube peut être défini par les coordonnées de ses huit coins, mais le même principe s'applique à des objets plus complexes. La seule différence est la quantité de mémoire requise pour stocker toutes les coordonnées et la puissance de traitement nécessaire pour manipuler cette information assez rapidement afin de créer un mouvement « en temps réel ». Dans cette application, comme dans toute autre, en augmentant la qualité de l'image on diminue la mémoire disponible et la puissance de traitement. Une résolution élevée implique l'utilisation d'un espace mémoire important. Mais il est essentiel d'inclure le

maximum de détails afin d'obtenir une image de bonne qualité.

Un téléviseur domestique produit (avec 625 × 1 000 points) des images d'une résolution inférieure à celle des moniteurs professionnels mentionnés précédemment. Nous pouvons donc être certains que tout travail comportant ce type de résolution sera d'une qualité égale, voire supérieure à une image télévisée ordinaire. Avec les techniques disponibles actuellement, il est possible de créer une véritable impression de réalité avec des images animées.

Pour créer une image de cette qualité, un logiciel sophistiqué et un matériel spécialement conçu sont nécessaires. La méthode de conception la plus populaire utilise une tablette graphique, petite planche à dessin qui renferme une

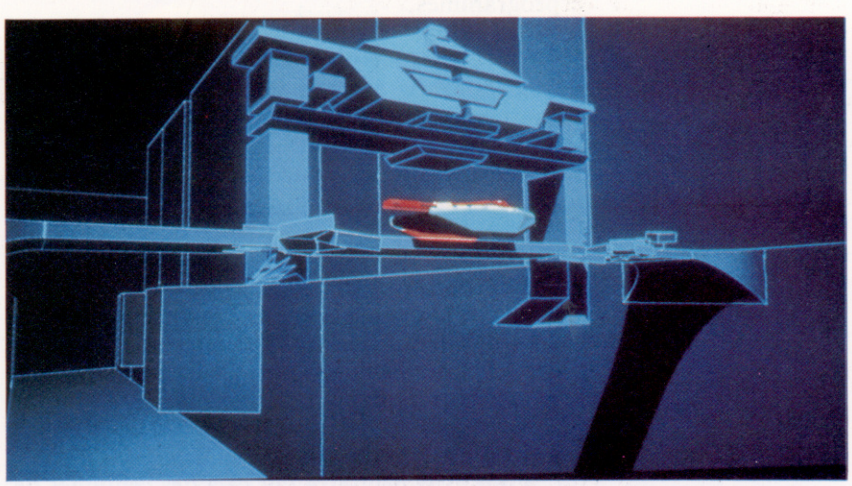


matrice de fils. Ce réseau sert à détecter la position d'un stylo. L'ordinateur affiche alors la ligne ou le point résultant. Il est possible de calquer un dessin, de dessiner librement, ou d'utiliser des instruments de dessin, comme sur du papier ordinaire. L'image est numérisée (traduite en coordonnées X et Y), écrite en mémoire et affichée par l'ordinateur. La nature de la ligne tracée à l'écran peut être définie par l'utilisateur, tout comme le dessinateur peut choisir d'utiliser un crayon, une plume ou un pinceau. La couleur peut aussi être définie en appelant une palette de couleurs au bas de l'écran. Si la couleur nécessaire n'est pas standard, elle peut être obtenue en mixant différentes couleurs, exactement comme de la peinture. Le stylo peut servir de « gomme à effacer », et les dessins peuvent être superposés.

Comment faire bouger une image que l'on vient de créer? Il suffit simplement d'automatiser le processus conventionnel, en utilisant l'ordinateur pour stocker les images créées, pour les colorer et pour illustrer des ébauches de séquences. Cette approche accélère déjà beaucoup le travail, mais des techniques de programmation plus évoluées permettent d'aller encore plus loin. De la même manière que les courbes sont automatiquement tracées, des séquences de mouvement peuvent être produites en donnant les première et dernière images. Ce processus, dans un studio d'animation conventionnel, est exécuté par un assistant. En fait, la majeure partie du travail d'animation est effectuée par des assistants, et c'est eux que l'ordinateur remplace. Comme nous l'avons déjà mentionné, l'introduction de l'ordinateur dans le processus d'animation permet à l'artiste de se concentrer sur la qualité de l'image. La majeure partie du travail de l'animateur consiste à créer l'illusion du mouvement, mais cette tâche est réalisable sur ordinateur. Il suffit d'énoncer des règles, et la machine n'a qu'à les suivre pour produire le résultat désiré, ce qui prouve que ce travail est particulièrement destiné à l'informatisation.

Voir c'est croire
Avec la rapidité et l'énorme capacité de stockage des ordinateurs modernes, il est possible de créer sur film ou sur écran de téléviseur une image indiscernable d'une photographie. Puis, à l'aide de techniques de programmation, il est possible de manipuler ces images de telle façon qu'elles semblent réelles à l'observateur.
(Cl. Ian Dobbie.)

Tron
Tron, produit par Walt Disney Productions, fut le premier film à utiliser la production d'images assistée par ordinateur. Le réalisateur intégra un mélange d'images produites par ordinateurs et d'effets photographiques spéciaux afin de créer un univers particulièrement étonnant.
(Cl. Walt Disney Ltée.)



Traductions

Les ordinateurs « pensent » en code machine; les programmeurs préfèrent écrire en un langage comme le BASIC. Les compilateurs et les interpréteurs offrent diverses méthodes de traduction.

Les premiers ordinateurs ne possédaient pas de clavier. Les instructions des programmes devaient être définies en réglant chacun des huit interrupteurs pour représenter une simple opération. Ces réglages d'interrupteur sont en fait un code machine.

Il était donc logique de remplacer ces interrupteurs par un clavier de machine à écrire, et de remplacer les réglages d'interrupteurs par des mots du langage humain. Cela donna des langages évolués comme le BASIC, remplaçant ainsi les codes machine.

Cependant les processeurs des ordinateurs n'ont pas changé, ils ont continué à travailler avec les mêmes configurations de 0 et de 1, les programmeurs ont donc dû développer des programmes écrits en code machine pour traduire ces programmes évolués en configurations avec lesquelles les ordinateurs peuvent travailler. On nomme ces programmes interpréteurs ou compilateurs, selon leur mode de traduction.

En informatique (comme dans tout domaine), tout gain en puissance ou en vitesse doit être payé, soit en argent, en temps, soit en liberté d'action. Il en est ainsi pour les interpréteurs et les compilateurs. Ils donnent au programmeur toutes les fonctions nécessaires de traduction de programme. Les interpréteurs et les compilateurs ont chacun leurs avantages et leurs inconvénients.

Les interpréteurs, généralement intégrés dans l'ordinateur domestique, permettent de traduire économiquement des programmes de langages évolués dans un code pouvant être compris par l'ordinateur. Ils n'utilisent pas beaucoup de mémoire, ce qui laisse plus d'espace pour vos programmes.

La plupart des micros qui coûtent environ 5 000 F offrent un interpréteur BASIC : vous tapez un programme BASIC, puis vous tapez la commande RUN, et le programme fonctionne ou est interrompu avec un message provenant du système qui vous annonce quelque chose du genre :

```
SYNTAX ERROR ON LINE 123
```

Vous devez alors taper la commande LIST, trouver l'erreur, la corriger, taper la commande RUN; à ce moment le programme fonctionne ou est interrompu de nouveau, et ainsi de suite. Notez que certains interpréteurs BASIC plus sophistiqués recherchent des erreurs de syntaxe lors de l'entrée de chaque ligne.

Vous avez peut-être fait cela des centaines de fois sans jamais songer à l'interpréteur. Sa principale qualité est précisément d'être un dispositif invisible qui vous permet de travailler sur votre programme sans avoir à vous demander où il se trouve en mémoire ou comment l'exécuter; le programme est à votre disposition, vous pouvez l'exécuter, le lister ou l'éditer à volonté.

L'interpréteur est facile à utiliser, mais très sophistiqué : chaque fois que vous tapez RUN, l'interpréteur doit trouver votre programme BASIC en mémoire, puis le traduire et l'exécuter ligne par ligne. Si votre programme renferme cette boucle :

```
400 LET N=0
500 PRINT N
600 LET N=N+1
700 IF N < 100 THEN GOTO 500
```

l'interpréteur doit traduire et exécuter les lignes 500 à 700 une centaine de fois, comme s'il ne les avait jamais rencontrées.

Les compilateurs sont différents. Ils sont plus coûteux, difficiles à écrire; de plus, ils occupent et utilisent beaucoup de mémoire. Ils sont presque toujours stockés sur disquettes.

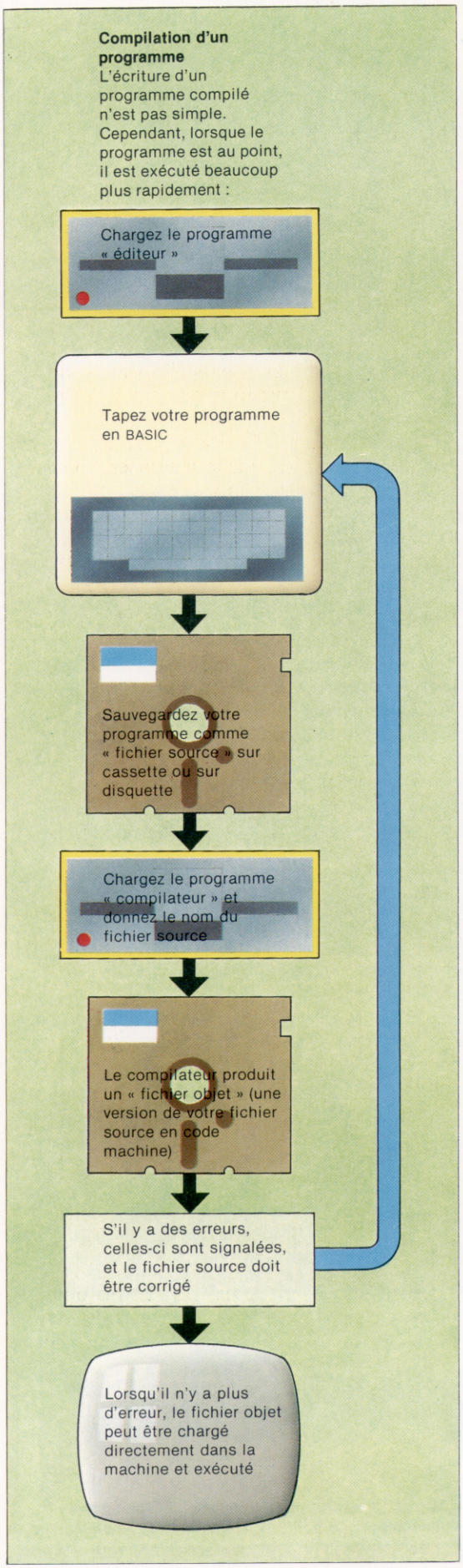
Ils offrent souplesse, puissance et vitesse; un compilateur traduirait les quatre lignes ci-dessus en une seule opération, puis exécuterait ce code une centaine de fois.

Vous chargez et exécutez d'abord le programme de création de fichier (nommé l'éditeur) qui vous permet de taper le programme et de le sauvegarder sur disquette comme « fichier source ».

Les fichiers doivent être nommés de façon à les retrouver après les avoir créés (tout comme des dossiers dans une armoire de classement), l'éditeur vous demande donc de nommer le fichier source. Les noms de fichier sont souvent composés de deux parties : la première est une étiquette, tout nom que vous choisissez — PROGRAMME par exemple — et la seconde partie est généralement un code de trois lettres qui indique la nature du contenu du fichier; ce code est l'« extension ». Un fichier BASIC peut avoir le code BAS comme extension de nom de fichier. Votre fichier source est maintenant stocké sur la disquette sous le nom PROGRAMME.BAS. Maintenant, en réponse à la commande :

```
COMPILE PROGRAMME.BAS
```

l'ordinateur chargera et exécutera le compilateur BASIC sur le fichier source BASIC nommé PROGRAMME.BAS.



Compilation d'un programme
L'écriture d'un programme compilé n'est pas simple. Cependant, lorsque le programme est au point, il est exécuté beaucoup plus rapidement :

Vous devez attendre quelques secondes, selon la longueur de votre programme pendant que le compilateur traduit votre programme en un « fichier objet » qu'il sauvegarde sur la disquette sous le nom PROGRAMME.OBJ — l'extension OBJ indique que ce fichier est un fichier objet, traduction en code machine d'un fichier source.

Pendant que le compilateur traduit votre fichier, il recherche des erreurs de syntaxe. S'il en trouve, vous obtiendrez un message comme celui-ci :

```
100 REED X:IF X = 3(N+2) LET P=Q
    1              2       3
FATAL ERROR:-
1) //REED// UNRECOGNISED COMMAND
2) ///(//ILLEGAL OPERATOR HERE
3) ??"THEN" OR "GOTO" EXPECTED HERE
```

Vous obtenez ce genre de message pour chaque ligne qui renferme une erreur. En d'autres mots, la signalisation d'erreur est beaucoup plus complète qu'avec un interpréteur. Vous devez charger et exécuter l'éditeur de nouveau, rappeler en mémoire le fichier source, apporter les modifications nécessaires, et essayer de compiler le programme de nouveau. S'il n'y a plus d'erreur, vous pouvez taper :

```
RUN PROGRAMME
```

Le programme à ce moment fonctionne, ou non, comme vous l'aviez prévu. Il n'y a plus d'erreur de syntaxe puisque vous les avez corrigées, mais il est possible que vous désiriez toutefois apporter certaines modifications; dans ce cas, vous devez charger et exécuter l'éditeur, modifier le fichier source, le recompiler... et ainsi de suite.

Les qualités du compilateur ne sont pas évidentes lors du développement du programme, quoique la signalisation d'erreur soit très pratique. Vous ne découvrirez les véritables avantages du compilateur qu'au moment où vous exécuterez un programme mis au point. Au moment où les interpréteurs révèlent leur faiblesse.

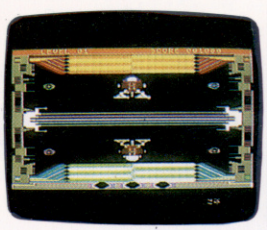
Les programmes compilés sont rapides, jusqu'à 50 fois plus rapides que les programmes interprétés, selon l'efficacité du compilateur, mais la vitesse d'exécution du programme compilé est compensée par un temps de développement plus long.

Comparer les compilateurs et les interpréteurs à l'aide de séquences de commandes utilisateur comme celle ci-dessus est injuste pour les compilateurs, puisqu'ils sont généralement écrits pour des machines plus puissantes, moins spécialisées, dont les utilisateurs peuvent écrire et exécuter des programmes dans différents langages de programmation.

Le fait que les fichiers objets compilés sont écrits en code machine illisible peut être un avantage. Si vous vendez un logiciel, vous ne vendez pas le fichier source, mais uniquement le fichier objet, ce qui le rend beaucoup plus difficile à copier ou à modifier.

Lent
Dans un jeu d'aventures, le programme sert surtout à manipuler des chaînes de texte. Il peut donc être écrit en BASIC et interprété lors de son exécution.

Rapide
Plusieurs programmes de gestion (particulièrement les feuilles de calcul électronique) sont difficiles à écrire en code machine puisqu'ils impliquent de nombreuses opérations mathématiques.



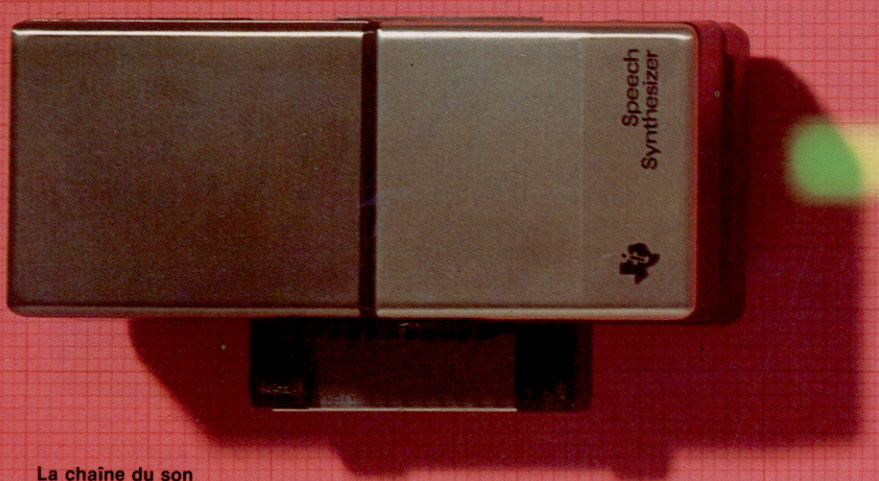
Plus rapide
Pour les jeux de type arcades à action rapide qui impliquent des mouvements, même un programme compilé ne serait pas assez rapide. De tels programmes doivent donc être écrits en code machine.

L'ordinateur parle

Jusqu'ici c'était de la pure science-fiction. Maintenant, avec un synthétiseur de parole, d'un prix très raisonnable, votre ordinateur peut réellement vous parler.

Alors que la reconnaissance de la parole n'est pas encore au point, la synthèse de la parole a maintenant été maîtrisée. Il y a peu de temps encore, la puissance et la capacité de traitement dont devait disposer un ordinateur pour produire une voix humaine étaient énormes. Maintenant, grâce à certains dispositifs complémentaires, presque tous les ordinateurs sont en mesure de parler. Les rapides progrès technologiques et la chute des coûts de production ont donné la parole à de plus en plus de machines.

Dans la parole humaine, on peut distinguer trois genres de sons. Le premier est du type voyelle — *o, a, e,* et ainsi de suite. Ces sons sont produits par la vibration des cordes vocales, la fréquence de cette vibration détermine le son produit. Le deuxième est le son fricatif, comme *ss, ch, t* et *ff*. Ici l'air provenant des poumons évite les cordes vocales, et la fréquence du son est commandée par la position des lèvres et de la langue. Le troisième « son » est le silence ou — pour être plus précis — les intervalles à l'intérieur de mots comme huit, puisque et ainsi de suite. La présence d'intervalles dans ces mots n'est peut-être pas évidente, mais si vous essayez de les prononcer très lentement, vous noterez qu'il est impossible de passer de façon continue du son *i* au son *t*.



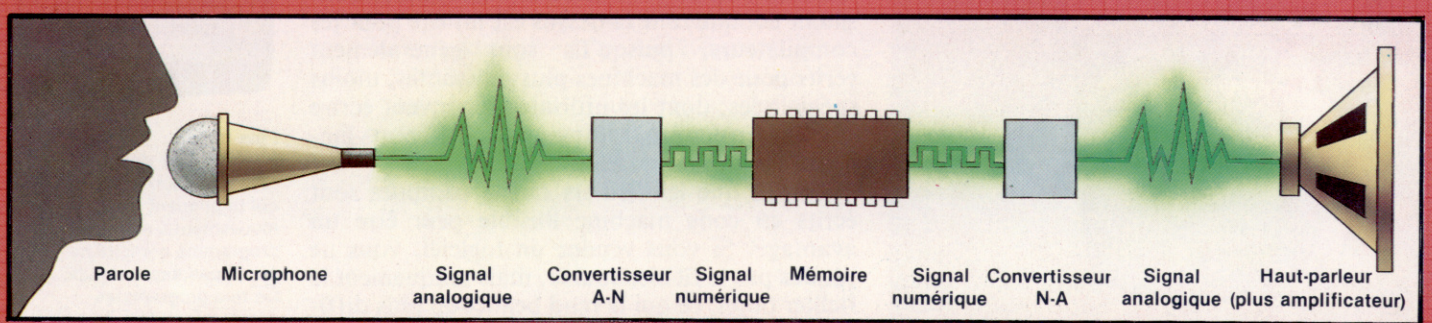
La chaîne du son
La parole peut être numérisée et stockée en mémoire, soit en RAM ou en ROM. La sortie électrique d'un microphone traverse un convertisseur analogique-numérique. La parole peut alors être recréée à l'aide d'un convertisseur numérique-analogique, d'un amplificateur et d'un haut-parleur. (Cl. Kevin Jones.)

Composantes du son

Il y a deux méthodes pour produire électroniquement des sons qui ressemblent à la parole. Avec la première, jusqu'à tout récemment la plus courante, en analysant les fréquences qui composent la parole il est possible de concevoir un système de règles qui nous permet de reproduire tout son à partir de ses composantes. Par exemple, le mot « toux » peut être défini comme la combinaison des fréquences qui composent le sont, suivie immédiatement des fréquences ou.

Ces composantes individuelles sont appelées « phonèmes » et en les utilisant dans diverses combinaisons des mots peuvent être construits. Les caractéristiques individuelles de la voix humaine ont tendance à disparaître lorsque la parole est produite de cette façon, mais les mots peuvent être reconnus et compris. Puisque les règles utilisées pour produire les phonèmes sont intégrées dans l'équipement, l'utilisateur peut entrer une liste de phonèmes dans le système. Ceux-ci sont reproduits à l'aide d'un petit haut-parleur. Avec un peu d'expérience, il est possible de produire des phrases complètes en appelant des séquences de phonèmes, qui sont généralement stockées dans des chaînes BASIC.

La seconde méthode repose sur la possibilité qu'a l'oreille humaine de remplir les intervalles.



Par exemple, la gamme de fréquences qui peut être transmise par des lignes téléphoniques ne donne que le cinquième de la qualité que nous pouvons attendre d'un système hi-fi, et le son est quand même parfaitement intelligible. C'est parce que notre cerveau compense les intervalles.

La seconde méthode de synthèse, que l'on nomme « parole numérisée », utilise le même phénomène. Les mémoires étant moins coûteuses, il est maintenant possible de convertir la parole en information numérique à l'aide d'un convertisseur analogique-numérique. Les données résultantes sont alors compressées plusieurs centaines de fois et stockées en ROM — créant ainsi les intervalles que notre oreille peut compenser.

Pour produire les mots stockés, il suffit d'indiquer à l'ordinateur leur adresse dans la ROM, l'information numérique est alors extraite et reconvertie en son. Puisque les mots sont stockés, les caractéristiques individuelles demeurent inchangées. Des puces peuvent ainsi reproduire très fidèlement la voix de personnages célèbres.

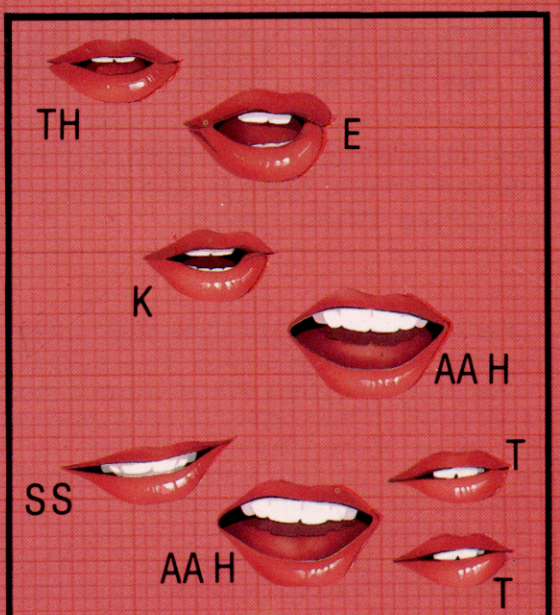
Certains ordinateurs, le Sirius notamment, possèdent un matériel intégré ainsi qu'un logiciel qui permet à l'utilisateur de numériser sa voix à l'aide d'un microphone. Les données résultantes sont stockées sur disquette — une seconde de discours occupe environ un Koctet — et peuvent être rappelées par un programme d'application et servir de messages.

Les utilisations de synthétiseurs de parole sont si nombreuses et si variées qu'il est impossible de toutes les énumérer. Les synthétiseurs de parole peuvent effectuer des annonces dans les gares et dans les aérogares. Aux États-Unis, ils sont largement utilisés dans le système téléphonique pour informer les utilisateurs de mauvais numéros, de services interrompus, etc. Plusieurs services de commande informatisés utilisent maintenant la synthèse de la parole. Un numéro de commande est tapé au clavier, et l'ordinateur le répète afin de vérifier cette entrée. L'ordinateur peut aussi indiquer à l'utilisateur le niveau des stocks et les délais de livraison, la commande peut ainsi être modifiée au moment de sa saisie.

Des unités de synthèse de parole sont aussi intégrées dans des voitures — la Renault 11 TSE par exemple — en équipement standard. Plus qu'un simple gadget, le synthétiseur produit des avertissements que le conducteur peut entendre sans quitter la route des yeux.

Dans les ordinateurs domestiques, la synthèse de la parole sert à améliorer les programmes de jeu : le programme peut annoncer verbalement les pointages ou l'approche d'un ennemi, ce qui permet au joueur de se concentrer sur les tactiques du jeu plutôt que d'avoir à consulter des messages au bas de l'écran.

Finalement, des appareils éducatifs comme « Dictée magique » de Texas Instrument prononcent un mot qui doit être épelé correctement, ou des mots étrangers tout en les affichant.



Composantes de la parole

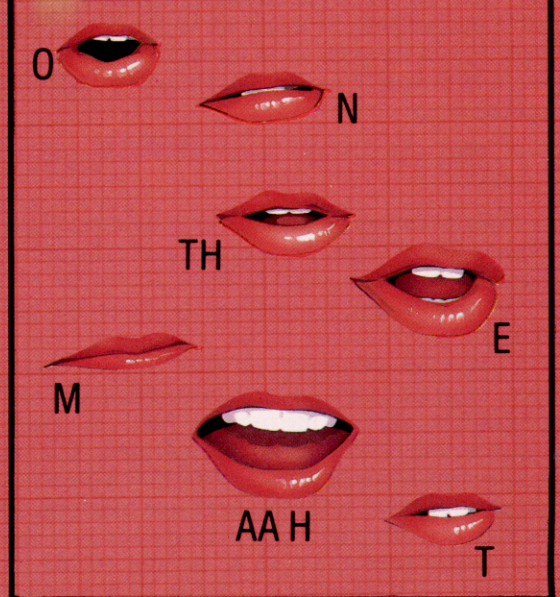
Si nous prenons une courte phrase anglaise : « The cat sat on the mat », il est possible de la fractionner en une série de phonèmes, comme ceci :

TH EE / K AA T / S AAT / O HN / TH EE / M AA HT

Les phrases doivent être fractionnées différemment selon la puce utilisée. Une autre puce utiliserait les phonèmes suivants pour la même phrase :

THV E / K A T / S A T / UH_3 N / THV E / MAT

La puce Voltrax, par exemple, renferme quelque soixante phonèmes qui peuvent être adressés directement par un simple nombre. Pour faciliter l'utilisation du logiciel, un jeu de programmes est fourni et permet à l'utilisateur de taper l'expression requise sous la forme des phonèmes qu'elle renferme, comme dans l'exemple ci-dessus. Cependant, des appareils comme le Braid Speech Synthetiser ou le Voltrax Personal Speech System, qui renferment un microprocesseur dédié et un logiciel sophistiqué, permettent à l'utilisateur de taper l'expression en langage courant et d'obtenir la prononciation équivalente de la phrase. (Cl. Tony Lodge.)



Peek et Poke

Ces deux commandes sont utilisées lorsque vous désirez programmer quelque chose dont le BASIC ne peut se charger, mais chaque machine les utilise différemment.

PEEK et POKE sont deux instructions BASIC utilisées dans des programmes évolués lorsque des bits ou des octets individuels doivent être manipulés en mémoire. L'instruction PEEK sert à examiner le contenu d'une adresse spécifique de la mémoire, et POKE sert à stocker un nombre (allant de 0 à 255) dans une adresse.

Les instructions PEEK et POKE permettent au programmeur d'accéder au fonctionnement interne de l'ordinateur. Normalement, le BASIC intégré de votre ordinateur se charge des adresses où sont stockées les variables et les données qui définissent les caractères affichés à l'écran. Bien que nous n'ayons généralement pas à nous soucier de leur emplacement en mémoire, nous devons parfois le connaître. Pour ce faire nous utilisons la commande PEEK.

Voici un court programme permettant d'examiner toute adresse en mémoire :

```
10 REM EXAMEN D'ADRESSES DE MÉMOIRE
20 PRINT « ENTREZ EN DÉCIMAL L'ADRESSE A EXAMINER »
30 INPUT M
40 P = PEEK(M)
50 PRINT « LE CONTENU DE L'ADRESSE »; M; « EST »; P
60 GOTO 20
70 END
```

Cela imprimera en décimal le contenu de l'adresse spécifiée (évidemment, l'ordinateur le stocke en binaire). Si vous désirez voir le contenu de l'adresse exprimé en caractères d'impression, le BASIC possède une fonction qui convertit les nombres décimaux en leurs caractères équivalents. Il s'agit de la fonction CHR$; modifier la ligne 50 de la façon suivante imprimera le caractère stocké en mémoire à l'adresse spécifiée :

```
50 PRINT « LE CONTENU DE L'ADRESSE »; M; « EST »;
CHR$(P)
```

Pour examiner la totalité de la mémoire, une boucle FOR...NEXT peut être ajoutée en effaçant la ligne 30, en changeant la ligne 20 en FOR X = 1 TO 65535 et en remplaçant la ligne 60 par NEXT X.

Pour vous permettre de voir chaque caractère, vous pourriez intégrer une boucle de temporisation après l'instruction PRINT et avant l'instruction NEXT X. Notez également que la limite supérieure de la boucle FOR NEXT a été choisie en supposant que vous avez une mémoire de 64 K. Ce nombre sera plus petit si vous avez une plus petite mémoire : 16 K signifie 16 383 en décimal, 32 K requiert 32 767 et 48 K requiert 49 151. Voici le listing complet du programme :

```
10 REM IMPRESSION DU CONTENU DE TOUTES LES
ADRESSES DE LA MÉMOIRE
20 FOR X = 0 TO 65535
30 LET Y = PEEK(X)
40 PRINT « ADRESSE »; X; « = »; Y; « = »;
50 PRINT CHR$(Y)
60 FOR D = 1 TO 200
70 NEXT D
80 NEXT X
90 END
```

Bien que la fonction CHR$ convertisse les nombres décimaux en caractères équivalents, les caractères qui peuvent être imprimés sont représentés par les nombres 32 à 127. La plupart des ordinateurs utilisent les nombres compris entre 128 et 255 (le plus grand nombre représenté dans un seul octet) pour des caractères graphiques spéciaux. Plusieurs des nombres entre 0 et 31 désignent des fonctions d'écran spéciales. Lorsque le programme rencontre ces nombres, ils sont convertis par CHR$ en de curieux effets d'écran. Ils peuvent effacer l'écran, ou encore placer le curseur dans le coin supérieur gauche de l'écran.

L'instruction POKE est en fait l'inverse de l'instruction PEEK. Elle vous permet d'« écrire » un octet de données (tout nombre compris entre 0 et 255) dans une adresse de mémoire. POKE doit être utilisée avec précaution : si vous écrivez un nombre dans une mauvaise section de mémoire, vous pouvez interrompre le fonctionnement de l'ordinateur en écrasant une partie d'un programme essentiel. Si ce problème survient, vous devez remettre l'ordinateur à zéro (en le mettant hors tension puis de nouveau sous tension, ou en utilisant le bouton RESET, s'il en possède un), mais cela risque de détruire l'un de vos programmes. Avant d'utiliser POKE, consultez la topographie mémoire donnée dans le manuel de votre ordinateur, afin de savoir où se trouve la « zone utilisateur ».

La plupart des ordinateurs domestiques permettent à l'utilisateur d'accéder à la mémoire vidéo (la mémoire servant à stocker les caractères affichés à l'écran). Normalement, l'ordinateur lit la forme des différents caractères dans une ROM spéciale nommée générateur de caractères, où sont stockées les configurations de points de chaque caractère. Mais il est aussi généralement possible d'utiliser la RAM. Lorsque les codes de configuration des caractères sont stockés en RAM, de nouvelles configurations peuvent être spécifiées en nombres décimaux à l'aide de l'instruction POKE.

L'instruction POKE doit être utilisée avec précaution puisqu'elle modifie le contenu de certaines adresses de la mémoire et peut avoir un effet sur le fonctionnement de l'ordinateur. Elle ne peut endommager le matériel, mais elle peut entraîner la perte d'un programme. Voici quelques instructions : sur l'Atari 400 et 800 écrire un 1 dans l'adresse 751 fait disparaître le curseur; essayez POKE 751,1. Sur le Commodore 64, essayez POKE 1024,1. 1024 est la première adresse de l'écran. Sur le Sinclair Spectrum, essayez :

```
100 FOR N = 0 TO 6 STEP 2
110 POKE USR « A » + N +
    1,BIN01010101
120 POKE USR « A » + N +
    1,BIN10101010
130 NEXT N
140 PRINT « AAAAAAAA »
```

Les A de la ligne 140 doivent être tapés dans le mode graphique.

Alice le merveilleux

Premier micro-ordinateur domestique du GIE Matra et Hachette, Alice ambitionne de faire parler le BASIC à un public non initié, en particulier les enfants.

Frère jumeau de l'ordinateur Tandy TRS 80 MC 10, le micro-ordinateur Alice ne permettra pas à un programmeur confirmé d'atteindre le pays des merveilles, mais il en donnera la clef à tous ceux qui ambitionnent de s'initier à l'informatique, sans se condamner à des fins de mois difficiles.

La présentation d'Alice est le résultat d'une évidente volonté de séduire. Matra et Hachette ont voulu mettre tous leurs atouts dans cette jolie petite boîte rouge, bien proportionnée, qui est une invite permanente à l'utilisation.

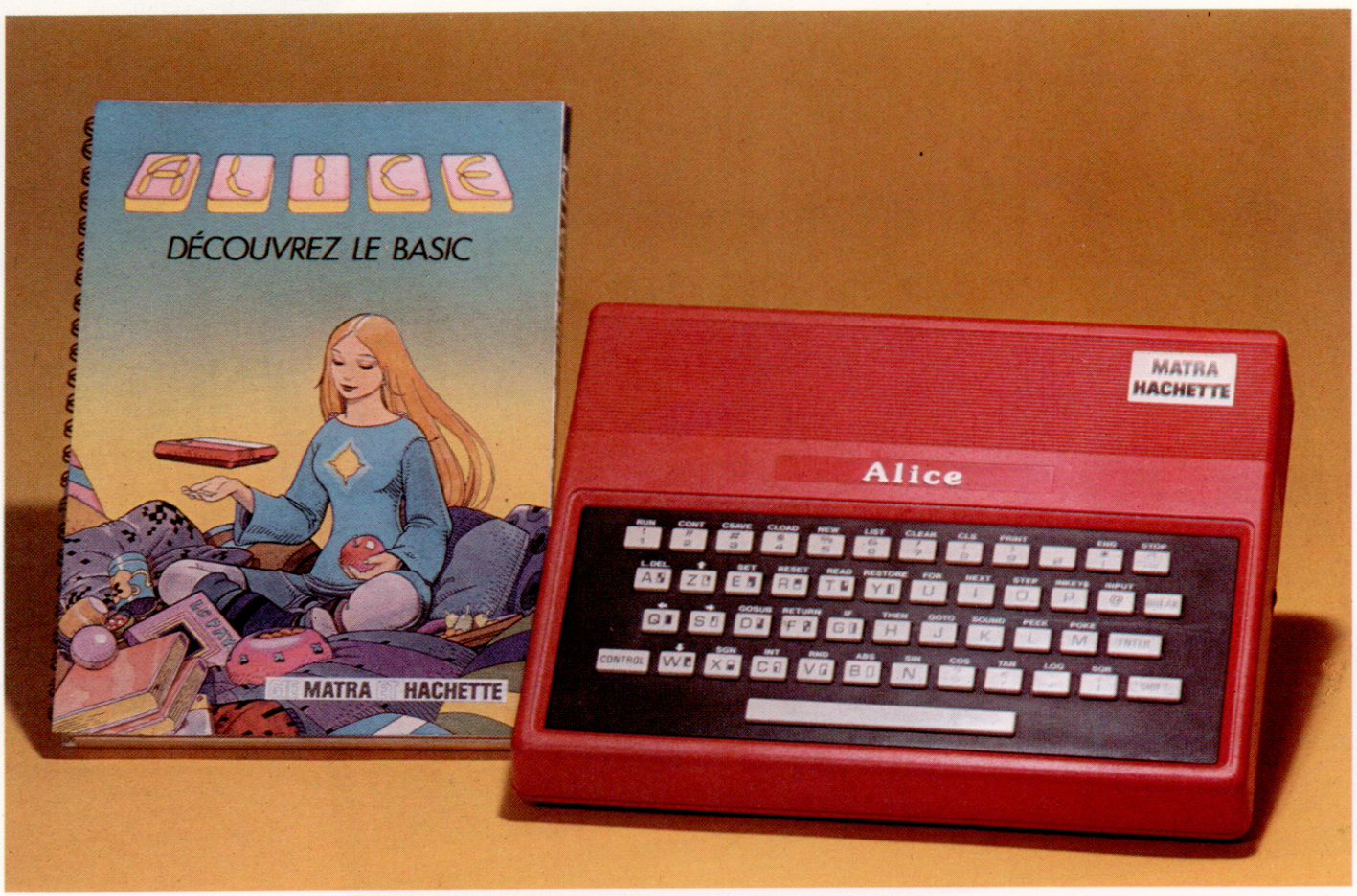
Le clavier AZERTY est très réussi et offre des sensations tactiles qui se rapprochent de celles produites par le clavier d'une machine à écrire électronique. Les risques d'erreur de frappe sont ainsi considérablement réduits et l'utilisateur peut entrer confortablement les données d'un programme.

Le souci d'efficacité et de clarté se retrouve dans l'agencement général de la machine. A l'arrière, la présentation ne permet aucun risque d'erreur dans les diverses possibilités de connexions offertes par Alice. La sortie multi-broches pour la prise Peritel, le lecteur de cassettes (tout modèle équipé de prises microphone et écouteur fera l'affaire), la prise de courant et la sortie imprimante, tout est clairement indiqué — en français — sans ambiguïtés d'aucune sorte.

Cet outil d'initiation vous permettra de créer des images stables à neuf couleurs, de bonne qualité, et de produire facilement de la musique. Ces avantages compensent les quelques inconvénients de l'Alice : un graphisme à basse résolution, une faible mémoire de base (4K), un seul son produit à la fois et peu de possibilités d'extension avec une seule sortie série RS232.

Comme possibilités externes, notons que l'une des imprimantes au catalogue est celle de son jumeau Tandy, l'imprimante DMP 100, permettant l'écriture de minuscules (obtenues

Enfin les grands groupes français se lancent dans la course à l'informatique individuelle. Alice, résultat des efforts combinés de Matra et de Hachette, est un ordinateur peu onéreux, permettant une initiation progressive à l'informatique et au langage BASIC.



Matra

sur l'écran par une touche de vidéo-inversion) sur une ligne de 20 cm de large. D'autres périphériques seraient bientôt annoncés, de même qu'une extension mémoire. Mais ces aspects apparemment négatifs ne sont que la conséquence d'une volonté de simplification et de démocratisation de l'informatique. Alice se veut l'ordinateur d'initiation à la portée du plus grand nombre.

En résumé, Alice est l'ordinateur d'apprentissage par excellence qui ne devrait pas tarder à se développer rapidement dans les écoles ou les clubs. A ce sujet, il est beaucoup attendu de l'alliance de Matra avec Hachette pour la mise au point de nombreux didacticiels de qualité.

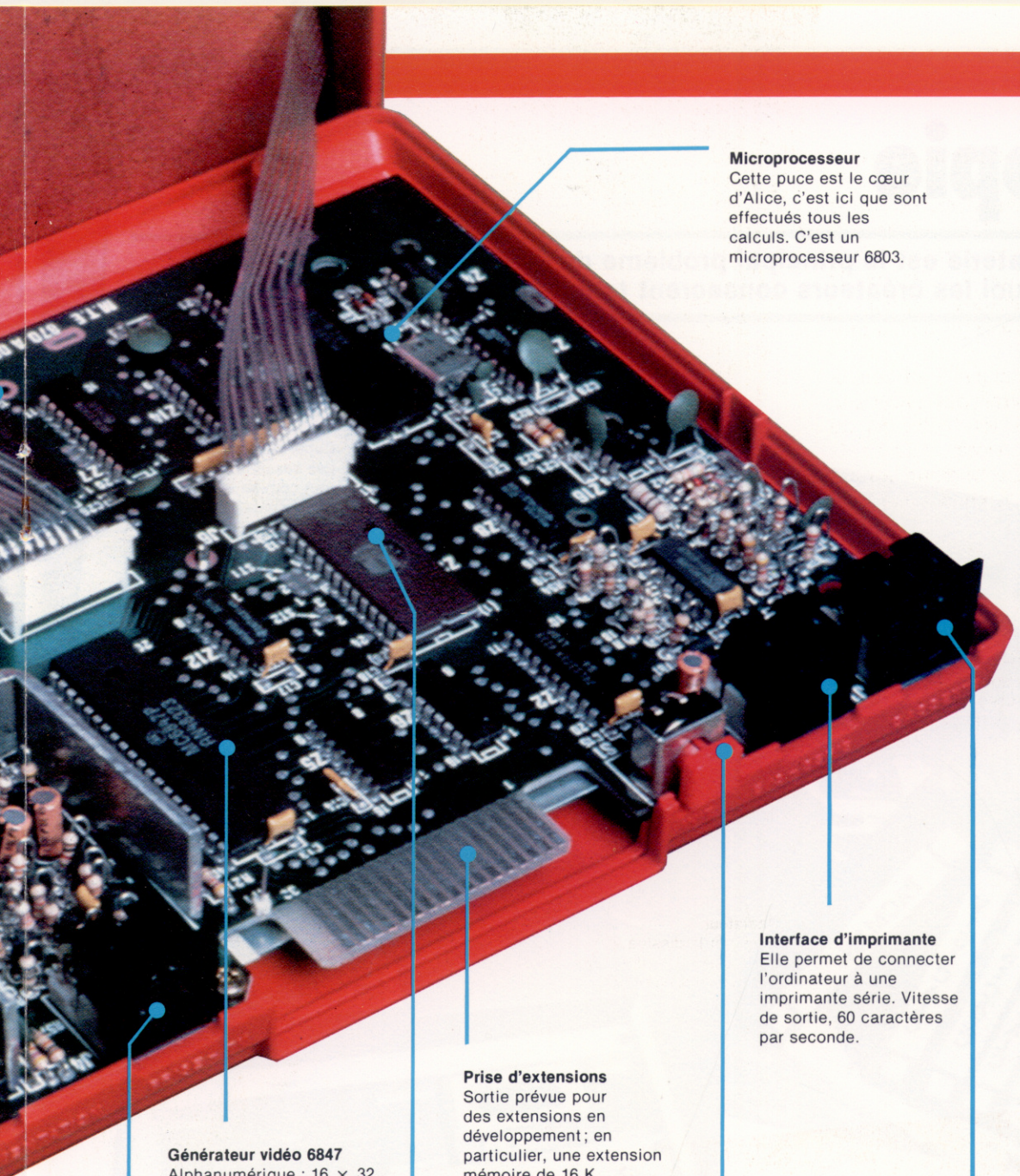
Matra

RAM
Mémoire vive utilisateur, 2 puces 4016 de 2 K chacune.

Radiateur
Il permet la dissipation de la chaleur produite par les circuits électriques.

Prise d'alimentation
L'alimentation électrique provient du transformateur fourni avec l'appareil.

Prise péritélévision
Grâce à son câble comportant une prise péritel, Alice peut se connecter directement aux appareils de télévision couleur modernes.



Microprocesseur
Cette puce est le cœur d'Alice, c'est ici que sont effectués tous les calculs. C'est un microprocesseur 6803.

Interface d'imprimante
Elle permet de connecter l'ordinateur à une imprimante série. Vitesse de sortie, 60 caractères par seconde.

Prise d'extensions
Sortie prévue pour des extensions en développement ; en particulier, une extension mémoire de 16 K utilisateur, des poignées de jeux, des cartouches ROM, une capacité de synthèse vocale.

Générateur vidéo 6847
Alphanumérique : 16 × 32 caractères ; mode semi-graphique, 64 × 32 et neuf couleurs. Une plus haute résolution est possible : 128 × 64, 4 couleurs.

Touche d'initialisation
Permet d'arrêter le déroulement d'un programme sans effacer la mémoire vive.

Prise cassette
L'unité cassette est reliée à l'ordinateur par cette prise. Elle permet une vitesse de sortie de 150 caractères par seconde.

ROM
La mémoire morte 8 K, BASIC MICROSOFT résident. Comprend l'interpréteur basic.

ALICE

PRIX

* *

DIMENSIONS

51 × 216 × 178 mm.

UC

Microprocesseur à 8 bits 6803.

VITESSE DE L'HORLOGE

0,89 MHz.

MÉMOIRE

4 à 16 K.

AFFICHAGE VIDÉO

16 lignes de 32 caractères de type ASII offrant un maximum de 64 × 32 points qui peuvent être colorés en 9 couleurs.

INTERFACES

Lecteur de cassettes, téléviseur doté d'une prise péritel, toute imprimante 600 bauds 7 bits équipée de l'interface RS232C.

LANGAGE

BASIC microsoft résident. Interpréteur BASIC très puissant quant à l'exécution des programmes et la variété des instructions disponibles.

AUTRE LANGAGE DISPONIBLE

Aucun.

ACCESSOIRE FOURNI

Prise péritel.

CLAVIER

Mécanique à 48 touches mécaniques AZERTY avec majuscules. Pas plus de trois fonctions par touche.

DOCUMENTATION

Le guide de présentation d'Alice est une véritable réussite. Cet ouvrage accompagne pas à pas l'utilisateur dans sa découverte de l'informatique BASIC. Les rédacteurs de ce guide ont su faire preuve d'humour et ont cherché à éviter les deux pièges de ce type de littérature, le didactisme et le pédantisme.

Copie

La piraterie est le principal problème de l'industrie du logiciel. Voilà pourquoi les créateurs consacrent tant d'efforts à se protéger.

Copie accélérée
Les cassettes de logiciel, tout comme les cassettes de musique, peuvent être copiées à l'aide d'un copieur de cassettes rapide. Cet appareil est muni d'un lecteur maître où l'original est placé, et d'un certain nombre d'unités esclaves qui réalisent les copies simultanément. Copier les deux côtés d'une cassette ne prend que quelques secondes. Les disquettes doivent être copiées individuellement à l'aide de lecteurs normaux. (Cl. Tony Sleep.)

Un pour cent
Tout comme il est théoriquement illégal de copier des cassettes musicales, la copie de programmes est ce qu'on appelle de la piraterie. Malheureusement pour les éditeurs, la piraterie n'est pas simplement difficile à empêcher, mais il est également difficile de la détecter et d'y donner des suites judiciaires. Certains éditeurs déclarent que pour chaque copie vendue, une centaine de copies illégales sont produites. (Cl. Tony Sleep.)

La piraterie de logiciel est simplement la copie non autorisée de programmes. Comme dans l'industrie musicale, l'industrie du logiciel se voit confronter à la piraterie de diverses manières et à des niveaux différents. Au niveau le plus bas, il y a piraterie chaque fois qu'un utilisateur d'ordinateur domestique copie un programme qu'il a emprunté à un ami. Même si certains programmes (surtout ceux écrits en code machine) ne peuvent être sauvegardés à l'aide des commandes BASIC normales, il est toujours possible de relier deux unités à cassette et de copier le programme sans l'aide de l'ordinateur.

Certains distributeurs de jeux déclarent que chaque copie vendue crée une centaine de copies illégales. Bien qu'on puisse dire qu'ils peuvent facilement encaisser cette perte, on doit se rappeler qu'il y a de nombreuses personnes qui vivent des droits des programmes vendus et qui ne roulent pas en Rolls-Royce!

Il y a eu une vive controverse au sujet des vendeurs qui louaient des programmes ou qui permettaient leur essai avant l'achat, et facilitaient ainsi la copie de programmes. Des revendeurs moins scrupuleux vont même plus loin en donnant des copies illégales des programmes les plus populaires pour faciliter la vente d'un ordinateur domestique.

Il y a même des distributeurs qui reproduisent des programmes en grande quantité et qui les

revendent à d'autres revendeurs, et ce n'est pas risqué si la transaction se fait dans un autre pays.

Au-delà, la piraterie devient plus sophistiquée et plus difficile à repérer. Une personne peut prendre un programme existant, y apporter des modifications et le vendre comme si elle avait créé. La nouvelle version peut offrir des améliorations considérables au niveau performances et fonctions additionnelles, ou simplement comporter des menus et des messages différents de façon à rendre le programme méconnaissable. Cette pratique est plus répandue dans les programmes professionnels que dans les jeux.

Les modifications mineures apportées à un programme permettent d'échapper à l'accusation de piraterie qui s'applique aux simples copies, c'est probablement pourquoi tant d'informaticiens peu scrupuleux s'y intéressent. Les éditeurs de logiciel sont très peu protégés sur le plan légal et les lois de droits d'auteur existantes ne protègent pas les programmes contre la piraterie ou contre la modification. Il semble que les droits d'auteur ne concernent que l'imprimé (avec certaines exceptions pour la musique), par conséquent les programmes qui sont stockés sur disquette ou sur cassette ne sont pas couverts. Comme pour toute question légale, une jurisprudence doit être établie, et cela prend du temps.

Il y a aussi des cas plus confus où une société reproduit sa propre version d'un programme populaire. Ici le code du programme n'est pas copié, les concepteurs ne font que noter précisément comment se déroule le programme, et ils écrivent alors un programme en partant de zéro et essaient d'obtenir le même effet. Le meilleur exemple en a été le programme PacMan, le jeu d'arcades qu'Atari a diffusé pour ses ordinateurs domestiques et pour ses cartouches vidéo et qui par la suite est apparu en diverses versions chez de nombreux éditeurs. Chacune de ces versions semble un peu différente, mais chacune comportait la petite créature familière qui gobe tout sur son passage dans un labyrinthe. Après quelques mois, Atari réussit à se défaire de ces compétiteurs soit par un procès ou, dans des cas de moindre importance, par la menace d'une procédure judiciaire.

Les auteurs de logiciels ne peuvent généralement pas compter sur les dispositions légales pour protéger leurs programmes et leurs droits d'auteur.

Certains fournisseurs décident de vendre leur logiciel à un prix assez bas pour que la copie ne soit pas intéressante. Des programmes plus sophistiqués sont accompagnés d'un excellent manuel et d'une présentation attrayante encourageant ainsi l'achat de la copie légale.

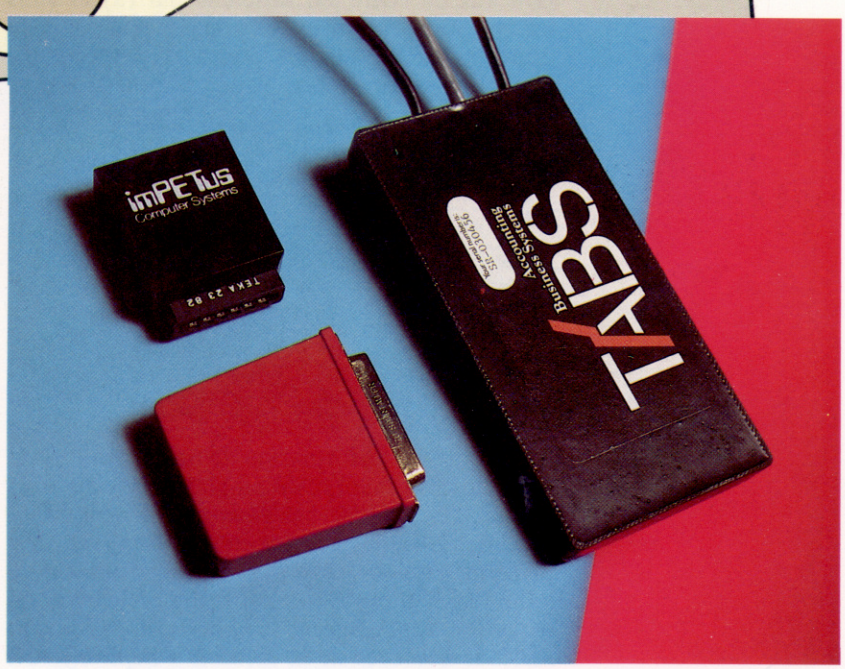
L'enregistrement de l'utilisateur est une façon de protéger des programmes encore plus coûteux : pour obtenir de l'assistance et d'éventuelles mises à jour, vous devez renvoyer une carte d'enregistrement.

Il existe aussi des méthodes de protection impliquant du matériel ; un dispositif de la taille d'une boîte d'allumettes est inséré dans l'un des ports d'interface de l'ordinateur et permet l'exécution du programme. Les circuits de ce dispositif définissent un code électronique, normalement une configuration de 0 et de 1 écrite dans une ROM. Le programme d'application adresse fréquemment ce dispositif ; s'il ne reçoit pas le code attendu, il est interrompu. Le code peut être différent d'un dispositif à l'autre, ce qui signifie que chaque copie du programme doit être assortie au dispositif avec lequel elle est vendue. La seule façon de faire des copies illégales est de construire le dispositif, ou d'écrire à nouveau le programme en enlevant les sections qui appellent le dispositif, ce qui n'est pas impossible mais qui est quand même au-delà des possibilités de la plupart des programmeurs d'ordinateurs domestiques.

Certaines sociétés de logiciel essaient d'obtenir le même type de protection sans utiliser de matériel additionnel. Un code magnétique est superposé sur la disquette ou sur la cassette « derrière » l'enregistrement du programme lui-même, ce code ne sera pas transféré lors d'une copie, le programme ne peut alors fonctionner que sur la disquette ou sur la cassette d'origine.

La seule protection matérielle véritablement rentable est la cartouche ROM, dont le prix plus élevé est compensé par un chargement plus rapide que sur cassette. Néanmoins, même la cartouche n'est pas à l'abri des copies, il existe des dispositifs qui peuvent copier une cartouche soit sur cassette, soit sur un nouveau type de cartouche qui peut être programmée ou reprogrammée par l'utilisateur.

La piraterie de logiciel est un combat permanent où chacun rivalise d'ingéniosité. Elle ne sera jamais éliminée ; la copie peut par contre être assez complexe et coûteuse pour ne devenir qu'un phénomène marginal. Sans oublier les attraits du manuel d'accompagnement, un des atouts essentiels du fabricant.



Dispositifs de protection
Ces petits dispositifs protègent certains programmes contre la copie illicite. De tels programmes ne pourront être exécutés que si le bon dispositif de protection est enfiché dans l'une des interfaces de l'ordinateur. Les circuits électroniques internes sont généralement scellés, ils sont donc très difficiles à modifier.
(Cl. Ian McKinnell.)

Une autre dimension

Les tableaux à une dimension, comme nous l'avons vu, stockent un ensemble de données. Les tableaux à deux dimensions sont utilisés pour les tableaux et les diagrammes.

Jusqu'ici, nous avons étudié deux types de variables, les variables simples et les variables indicées. Les variables simples sont comparables à des adresses de mémoire où les nombres et les chaînes de caractères peuvent être stockés et manipulés en mentionnant l'« étiquette » de la variable. Les variables simples ne peuvent stocker qu'une valeur ou qu'une chaîne et portent des noms de variables simples, comme N, B2, X, Y3. Les variables indicées, appelées quelquefois tableaux à une dimension, peuvent stocker une liste de valeurs ou de chaînes. Le nombre de valeurs ou de chaînes pouvant être stockées est spécifié au début du programme à l'aide d'une instruction DIM. Par exemple, DIM A(16) déclare un tableau A qui peut contenir seize valeurs différentes. On doit cependant signaler que certains BASIC acceptent A(0) comme premier élément, DIM A(16) peut alors définir dix-sept éléments. Ces éléments sont désignés en utilisant l'indice approprié. PRINT A(1) imprime le premier élément du tableau; LET B = A(12) attribue la valeur du douzième élément du tableau A à la variable B; LET A(3) = A(5) attribue la valeur du cinquième élément au troisième élément.

Nous devons cependant être quelquefois en mesure de manipuler des données présentées sous forme de tableau. Les données ressemblent alors à une feuille de calcul électronique (voir page 158). On retrouve de telles dispositions de données dans des résultats sportifs ou dans des tableaux de vente par article et par rayon dans un magasin. Voici un exemple de tableau de données, présentant les dépenses domestiques pendant une année :

	Loyer	Téléphone	Électricité	Nourriture	Voiture
Janv.	2 600	251	415	1 613	505
Févr.	2 600	354	437	1 459	462
Mars	2 600	290	507	1 512	434
Avril	2 600	262	446	1 553	492
Mai	2 600	193	398	1 509	483
Juin	2 600	204	326	1 476	523
Juil.	2 600	305	261	1 503	584
Août	2 600	295	224	1 480	612
Sept.	2 600	282	244	1 486	594
Oct.	2 600	311	345	1 549	235
Nov.	2 600	310	395	1 600	459
Déc.	2 600	289	422	2 106	512

En disposant l'information de cette façon, il est possible de la manipuler simplement de diverses façons. Il est facile par exemple de trouver les dépenses totales de mars en additionnant simplement tous les nombres apparaissant sur la ligne Mars. Il est tout aussi facile de trouver les dépenses annuelles de téléphone en additionnant les nombres de la colonne Téléphone. On peut également calculer très simplement des moyennes annuelles ou mensuelles. Ce tableau est un tableau à deux dimensions. Il a 12 lignes et 5 colonnes.

Les tableaux à deux dimensions comme celui-ci peuvent aussi être représentés en BASIC de la même façon que les tableaux à une dimension. Mais ici la variable nécessite deux indices.

Si nous écrivions un programme BASIC utilisant ce tableau d'informations, le plus simple serait de traiter ce tableau comme un tableau à deux dimensions. Nous lui donnons encore un nom de variable. Appelons-le A. Tout comme un tableau à une dimension, il doit être déclaré. Comme il possède douze lignes et cinq colonnes, il doit être déclaré de la façon suivante : DIM A(12,5). L'ordre des deux indices est important; la convention adoptée est la suivante : les lignes sont d'abord définies, puis les colonnes. Notre tableau compte douze lignes (une pour chaque mois) et cinq colonnes (une pour chaque catégorie de dépenses), ce qui en fait un tableau 12 par 5.

L'instruction DIM a deux fonctions principales. Elle réserve l'espace mémoire nécessaire pour le tableau, et elle permet de désigner chaque adresse par un nom de variable suivi entre parenthèses des positions de ligne et de colonne. L'instruction DIM X(3,5) crée une variable X pouvant représenter un tableau de trois lignes et de cinq colonnes.

Revenons au tableau et supposons que ces données ont été entrées dans un tableau de deux dimensions nommé A. Trouvez les valeurs contenues dans A(1,1), A(1,5), A(2,1), A(3,3) et A(12,3).

Il est possible d'entrer un tableau de données dans un programme à l'aide d'instruction LET, par exemple :

```
30 LET A(1,2) = 251
40 LET A(1,3) = 415
50 LET A(1,4) = 1613
:
:
610 LET A(12,5) = 512
```

Mais cela est une façon laborieuse de procéder. Il serait plus simple d'utiliser des instructions READ et DATA ou l'instruction INPUT imbriquée dans des boucles FOR-NEXT. Examinons comment procéder à l'aide de l'instruction READ :

```
10 DIM A(12,5)
20 FOR L = 1 TO 12
30 FOR C = 1 TO 5
40 READ A(L,C)
```

```
50 NEXT C
60 NEXT L
70 DATA 2 600, 251, 415, 1 613, 505, 2 600, 354, 437
80 DATA 1 459, 462, 2 600, 290, 507, 1 512, 434, 2 600
90 DATA 262, 446, 1 553, 492, 2 600, 193, 398, 1 509
100 DATA 483, 2 600, 204, 326, 1 476, 523, 2 600, 305
110 DATA 261, 1 503, 584, 2 600, 295, 224, 1 480, 612, 2 600
120 DATA 282, 244, 1 486, 594, 2 600, 311, 345
130 DATA 1 549, 235, 2 600, 310, 395, 1 600, 459
140 DATA 2 600, 289, 422, 2 106, 512
150 END
```

Dans ce programme, de nombreux points importants doivent être signalés. Le premier est que l'instruction DIM est située au début du programme. Une instruction DIM ne doit être exécutée qu'une seule fois dans un programme, il est donc préférable de la placer au début du programme avant toute boucle. Le second point est la présence de deux boucles, une pour définir l'indice de ligne et l'autre pour définir l'indice de colonne. Ces deux boucles ne se suivent pas, elles sont imbriquées. Notez les limites choisies. FOR L = 1 TO 12 incrémente la valeur de ligne de 1 à 12; for C = 1 TO 5 incrémente la valeur de colonne de 1 à 5.

Une instruction READ a été placée au milieu de ces boucles imbriquées. Voici la partie importante du programme :

```
20 FOR L = 1 TO 12
30 FOR C = 1 TO 5
40 READ A(L,C)
50 NEXT C
60 NEXT L
```

Lors de la première boucle, après l'exécution des lignes 20 et 30, les valeurs de L et de C sont toutes deux 1, la ligne 40 équivaut donc à READ A(1,1). Le premier élément de données dans l'instruction DATA est 2 600, cette valeur est donc affectée à la première ligne, première colonne du tableau. Le choix de huit éléments dans chaque instruction DATA est purement arbitraire.

Puis l''instruction NEXT C renvoie le programme à la ligne 30 et la valeur de C est incrémentée à 2. La ligne 40 équivaut donc à READ A(1,2) et l'élément suivant de données, 251, est affecté à la première ligne, deuxième colonne du tableau. Ce processus est répété jusqu'à ce que C ait été incrémenté à 5. Après quoi, l'instruction NEXT L de la ligne 60 renvoie le programme à la ligne 20 et R est incrémenté à 2. La ligne 30 remet C à 1 de nouveau et la ligne 40 équivaut maintenant à READ A(2,1).

Ces boucles imbriquées sont très pratiques, mais il est nécessaire d'être prudent. Chaque boucle doit être imbriquée entièrement à l'intérieur d'une autre boucle et l'ordre des instructions NEXT doit être soigneusement respecté. Notez où se trouve l'instruction NEXT L de la première boucle FOR L. Quand deux boucles sont imbriquées, la première boucle est nommée la boucle externe et la seconde boucle est nommée la boucle interne. La boucle interne doit être exécutée au complet avant que ne soit incrémenté l'indice de la boucle externe. Il est possible d'imbriquer autant de boucles que nécessaire, mais de tels programmes peuvent devenir difficiles à suivre et à mettre au point. Effectuer des branchements à l'intérieur d'une boucle est une mauvaise pratique de programmation et les instructions GOTO doivent être évitées.

Examinons les instructions DATA. Notez la présence de virgules entre les éléments de données et l'absence de virgule avant le premier élément et après le dernier de chaque ligne. Les erreurs de saisie de données sont faciles à faire et difficiles à repérer par la suite. Vous pouvez utiliser autant d'instructions DATA que vous le désirez. Chaque nouvelle ligne doit commencer par l'instruction DATA. Les données sont lues un élément à la fois, en commençant au début de la première instruction DATA et en continuant jusqu'au dernier élément. Veillez à entrer un nombre suffisant de données, sinon vous obtiendrez un message d'erreur lors de l'exécution du programme.

Le programme présenté jusqu'ici ne fait qu'affecter des valeurs à un tableau à deux dimensions. Après l'exécution du programme aucune intervention ne sera apparente, et l'écran n'affichera que le message de sollicitation BASIC. Pour vérifier l'entrée en mémoire des données, essayez quelques commandes PRINT. (Une commande en BASIC est un mot qui peut être exécuté en mode direct sans avoir à être placé dans un programme, il n'est pas nécessaire de spécifier de numéro de ligne. Voici des exemples : LIST, RUN, SAVE, AUTO, EDIT et PRINT.) PRINT A(1,1) <R> devrait faire apparaître le nombre 2 600 à l'écran. Quel serait l'affichage des commandes suivantes?

```
PRINT A(12,1)
PRINT A(1,5)
PRINT A(5,1)
PRINT A(5,5)
```

Pour que le programme serve à quelque chose, il doit être étendu. Tel quel, il constitue un bon programme principal. Pour l'utiliser à l'intérieur d'un programme plus complexe et plus utile, des sous-programmes peuvent être écrits et appelés par des instructions GOSUB insérées aux endroits appropriés avant l'instruction END.

Lors des premières étapes de conception d'un programme de comptabilité domestique, il est préférable de commencer par une simple description écrite des tâches à effectuer. Nous pourrions décider de calculer les totaux et moyennes des dépenses mensuelles ou d'une catégorie (électricité par exemple). Nous pourrons définir ultérieurement les instructions du programme. Si le programme comporte des choix à faire concernant le sous-programme à exécuter, il serait préférable de proposer à l'utilisateur un menu où il lui serait possible de choisir ce sous-programme. Voici une première esquisse de notre programme :

```
PROGRAMME PRINCIPAL
(ENTRÉE DE DONNÉES)
```

```
MENU
(SÉLECTION DES SOUS-PROGRAMMES)

FIN
```

Pour perfectionner le programme, nous aurons besoin de sous-programmes qui calculent les totaux mensuels ou des diverses catégories (MOISTOTAL et CATTOTAL), les dépenses moyennes mensuelles (MOISMOY) et les moyennes annuelles par catégorie (CATMOY). En utilisant un nom pour désigner les sous-programmes, nous n'aurons pas à nous soucier des numéros de ligne pour l'instant. Après réflexion, nous pourrions décider que même la sélection du menu principal devrait être traitée dans un sous-programme de façon à définir le programme principal comme un module distinct. Voici une ébauche un peu plus élaborée du programme :

```
PROGRAMME PRINCIPAL (ENTRÉE DE DONNÉES)
MENU (APPEL D'UN SOUS-PROGRAMME)
END

** SOUS-PROGRAMMES **

1 MENU
2 TOTAUX
3 MOYENNES

(2) TOTAUX
4 MOISTOTAL
5 CATTOTAL

(3) MOYENNES
6 MOISMOY
7 CATMOY
```

Nous voyons dans cette esquisse du programme que le sous-programme MENU nous permet de choisir les options TOTAUX ou MOYENNES. Ces deux options sont des sous-programmes. Le sous-programme TOTAUX nous propose un autre choix : MOISTOTAL ou CATTOTAL. Ces deux sous-programmes effectuent les calculs.

Le sous-programme MOYENNES nous propose le choix suivant : MOISMOY ou CATMOY, qui sont eux aussi des sous-programmes effectuant les calculs appropriés. Ici nous pouvons voir si notre « programme » fonctionne ou non sans avoir encore à commencer la véritable programmation en BASIC. Dès que nous jugeons cette ébauche satisfaisante, nous sommes prêts à commencer l'écriture des modules (sous-programmes). La seule modification au programme principal sera l'insertion d'un appel de sous-programme avant l'instruction END; nous pouvons donc ajouter :

```
145 GOSUB ** MENU **
```

Notez que nous utilisons encore des « noms » pour désigner les sous-programmes et non des numéros de ligne. Plusieurs langages, dont le PASCAL, permettent d'appeler les sous-programmes par leur nom, mais la plupart des versions du BASIC ne le permettent pas; dans ce cas, le numéro de ligne doit être spécifié.

Voyons comment le sous-programme MENU peut être écrit (les numéros de lignes ont été omis, vous devrez donc les ajouter si vous désirez utiliser ce programme).

```
REM LE SOUS-PROGRAMME ** MENU **
PRINT « DÉSIREZ-VOUS CALCULER DES (T)OTAUX OU DES
  (M)OYENNES ? »
PRINT « TAPEZ T OU M »
INPUT L$
IF L$ = « T » THEN GOSUB * TOTAUX *
IF L$ = « M » THEN GOSUB * MOYENNES *
RETURN
```

Nous plaçons les noms des sous-programmes appelés entre les symboles *_____*. Vous devrez utiliser des numéros de ligne. Ceux-ci pourront être insérés lorsque vous connaîtrez la position occupée par les sous-programmes.

Supposons que vous tapiez T pour TOTAUX. Le programme appelle alors le sous-programme TOTAUX. Ce sous-programme propose un menu et devrait se présenter comme ceci :

```
REM LE SOUS-PROGRAMME ** TOTAUX **
PRINT « DÉSIREZ-VOUS LES TOTAUX »
PRINT « DU (M)OIS OU D'UNE (C)ATÉGORIE ? »
PRINT « TAPEZ M OU C »
INPUT L$
IF L$ = « M » THEN GOSUB * MOISTOTAL *
IF L$ = « C » THEN GOSUB * CATTOTAL *
RETURN
```

Supposons que vous ayez sélectionné M pour MOISTOTAL. Voyons comment nous pourrions écrire un module qui calcule les dépenses d'un mois de l'année.

```
REM LE SOUS-PROGRAMME ** MOISTOTAL **
REM CECI CALCULE LES DÉPENSES TOTALES
REM D'UN MOIS
PRINT « CHOISISSEZ UN MOIS »
PRINT « 1-JAN 2-FÉV 3-MARS 4-AVR 5-MAI »
PRINT « 6-JUIN 7-JUIL 8-AOUT 9-SEP »
PRINT « 10-OCT 11-NOV 12-DÉC »
PRINT « TAPEZ LE NUMÉRO DU MOIS »
LET T = 0
INPUT M
FOR C = 1 TO 5
LET T = T + A(M,C)
NEXT C
PRINT « LES DÉPENSES TOTALES DU MOIS »
PRINT « NUMÉRO »; M; « SONT »; T
RETURN
```

Le nombre représentant le mois est tapé et l'instruction INPUT l'affecte à la variable M (MOIS). M sert à spécifier la « ligne » du tableau à deux dimensions A. La boucle FOR-NEXT incrémente la valeur de C (colonne) de 1 à 5; ainsi, lors de la première boucle, si nous avons tapé 3 pour mars, l'instruction LET équivaut à LET T = T + A(3,1). Lors de la deuxième boucle, elle équivaut à LET T = T + A(3,2) et ainsi de suite.

Nous vous laissons écrire les autres sous-programmes, ou essayer les autres exercices. Les tableaux à deux dimensions sont parfaits pour tout programme qui traite des tableaux de données, quelle que soit leur nature.

Réponses de la page 175

Fonction RND

```
40 IF R>6 THEN LET R = 1
```

Boucle et moyenne

```
5 FOR L = 1 TO 100
:
:
80 LET T = T + R
90 NEXT L
100 LET A = T/100
110 END
```

Remplacer par un sous-programme

Effacez les lignes 5, 80, 90, 100 et 110 de la solution précédente. Changez les lignes 10 à 70 en 1000 et 1070. La ligne 40 doit être identique à celle de la solution de la fonction RND ci-dessus. Puis ajoutez 1080 RETURN. Insérez le résultat dans le programme principal. Les lignes 50 et 130 doivent être 50 GOSUB 1000 et 130 GOSUB 1000.

INKEY$

```
10 PRINT « PRESSEZ UNE TOUCHE »
20 LET A$ = INKEY$
30 IF A$ = « » THEN GOTO 20
40 PRINT « LA TOUCHE QUE VOUS AVEZ TAPÉE EST »; A$
50 END
```

Boucle de synchronisation

```
5 PRINT « PRESSEZ LA BARRE D'ESPACEMENT APRÈS
  DIX SECONDES »
10 FOR L = 0 TO 1
20 LET R = R + 1
30 IF INKEY$ = « »THEN GOTO 60
40 LET L = 0
50 NEXT L
60 PRINT « LA VALEUR DE R APRÈS DIX SECONDES EST »;
  R
70 END
```

IF THEN

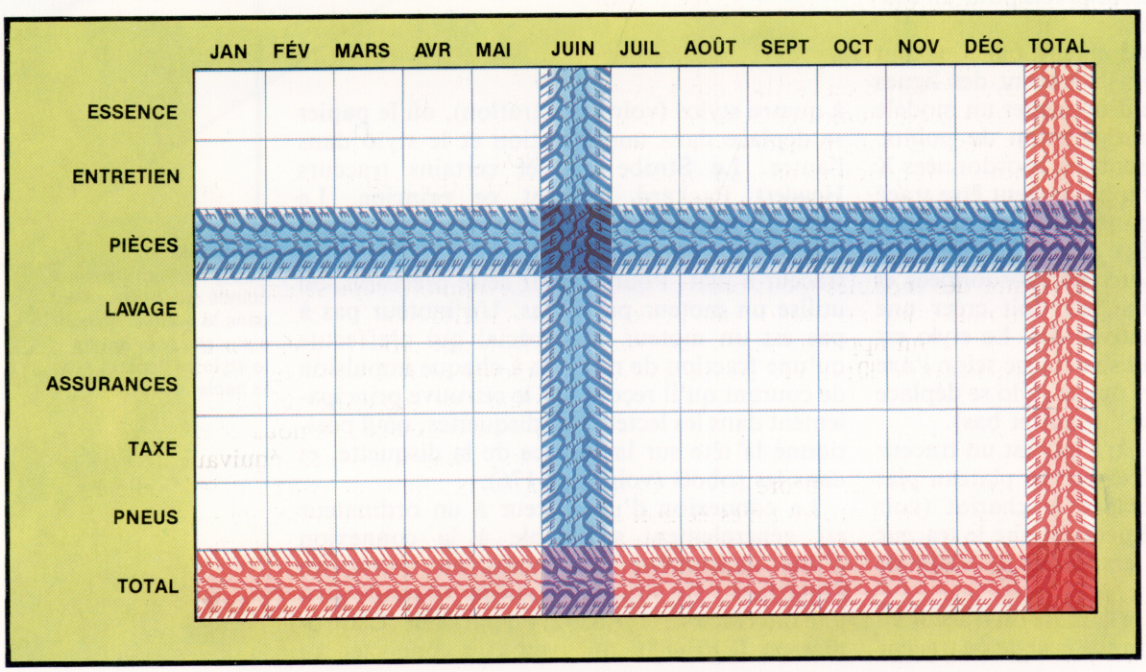
```
10 GOSUB 1000
20 PRINT « DEVINEZ LE NOMBRE »
30 FOR G = 1 TO 5
40 INPUT N
50 IF N>R THEN GOTO 110
60 IF N<R THEN GOTO 130
70 IF N=R THEN GOTO 150
80 NEXT G
90 PRINT « PLUS D'ESSAIS VOUS AVEZ PERDU »
100 GOTO 500
110 PRINT « VOTRE RÉPONSE EST TROP ÉLEVÉE »
120 GOTO 80
130 PRINT « VOTRE RÉPONSE EST TROP BASSE »
140 GOTO 80
150 PRINT « C'EST EXACT, FÉLICITATIONS »
500 END
1000 REM ** SOUS-PROGRAMME ALÉATOIRE **
(Insérez votre sous-programme ici)
1020 RETURN
```

Exercices

• ***Affectation de valeurs.*** Écrivez un programme qui affecte des valeurs aux éléments (essence, service, etc.) du tableau suivant. Puis écrivez un sous-programme qui demande un mois et une catégorie de dépenses, et qui imprime le contenu de la case ainsi spécifiée. Finalement, écrivez un sous-programme qui calcule la somme de chaque colonne et qui place le résultat dans la case du bas, qui fait la même opération pour les lignes et qui calcule le grand total et le stocke dans la case inférieure droite.

• ***Mise au point.*** Le programme suivant ne fonctionnera pas correctement et affichera un message d'erreur. Trouvez l'erreur et apportez les corrections nécessaires.

```
10 DIM A(3,4)
20 FOR L = 1 TO 3
30 FOR C = 1 TO 4
40 READ A(L,C)
50 NEXT C
60 NEXT L
70 FOR X = 1 TO 3
90 FOR Y = 1 TO 4
100 PRINT A(Y,X)
110 NEXT Y
120 NEXT X
130 DATA 2,4,6,8,10,12,14,16,18,20,22
140 END
```



Dépenses de voiture
Cette figure illustre une grille de 8 cases par 13. Les lignes représentent divers éléments de dépenses pour une voiture et les colonnes représentent les mois de l'année. Effectuez l'exercice « Affectation de valeurs » pour calculer le coût d'utilisation annuel d'une voiture. (Cl. Tony Lodge.)

Traceurs

L'utilisation d'un traceur est le meilleur moyen d'obtenir une sortie graphique de haute qualité avec votre ordinateur. Munis de stylos feutres, certains peuvent changer de couleur automatiquement.

De nombreux utilisateurs d'ordinateur estiment qu'il est essentiel de pouvoir créer des copies imprimées de ce qui est affiché à l'écran. Les ingénieurs, les scientifiques, les dessinateurs techniques et les hommes d'affaires ont tous besoin de diagrammes et de tableaux précis que les imprimantes conventionnelles ne peuvent pas créer. Seul un traceur est en mesure de créer de telles images et, jusqu'à tout récemment, ceux-là étaient trop coûteux pour l'utilisateur d'ordinateur domestique.

Cependant, avec l'introduction de traceurs comme le Tandy/CGP-115 et l'Oric MCP-40, ce type d'impression est maintenant à la portée de toutes les bourses. Une gamme complète de traceurs vient de faire son apparition sur le marché et offre des caractéristiques qu'on ne pouvait trouver précédemment que sur des appareils coûtant des dizaines de milliers de francs.

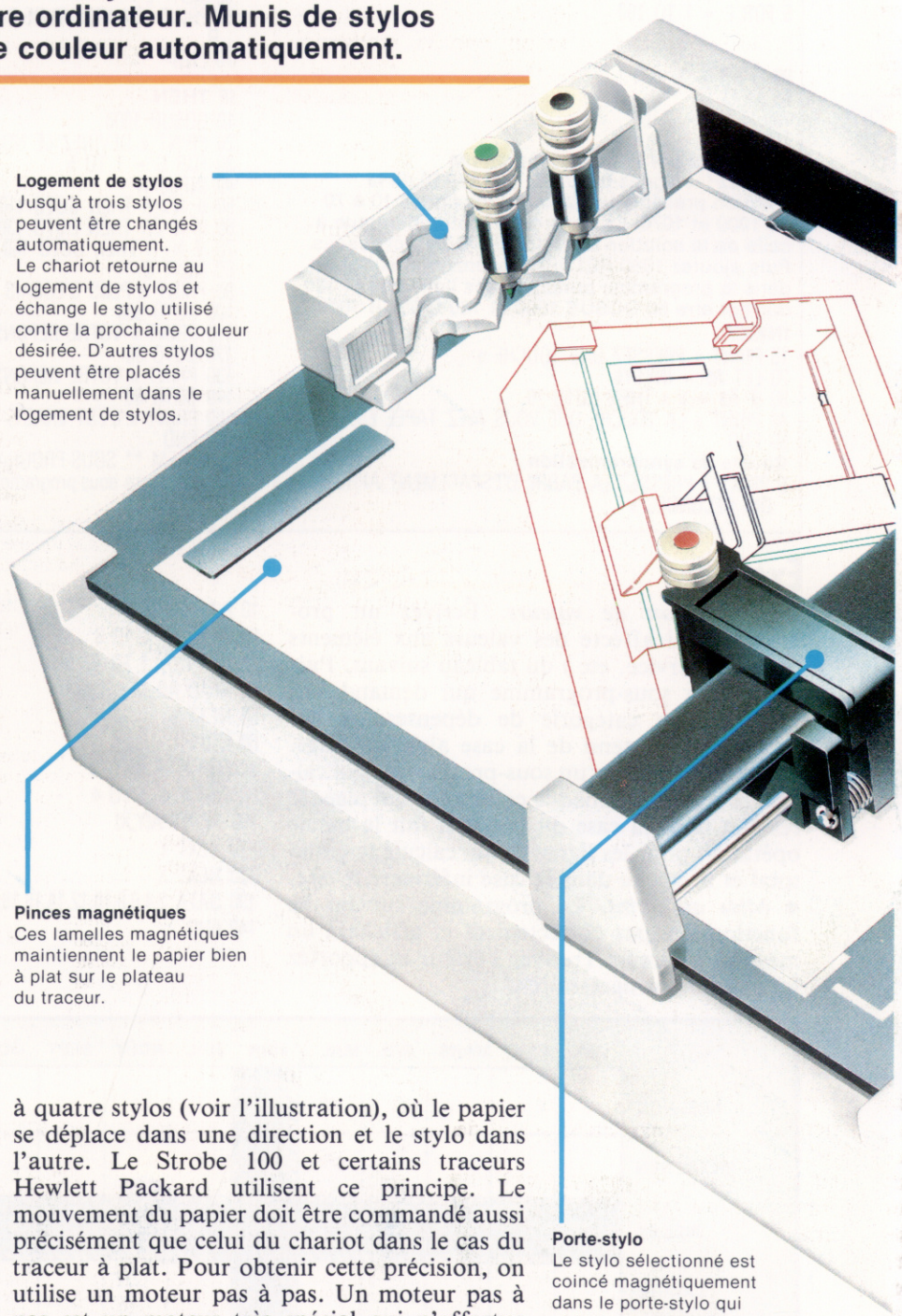
C'est le type de sortie produit par l'ordinateur qui détermine si on a besoin d'un traceur ou non. Un ingénieur ou un dessinateur doivent produire des dessins précis d'équipement et d'installations, un homme d'affaires utilise des tableaux et des graphiques pour illustrer l'évolution des ventes. De tels travaux seraient difficiles et laborieux à réaliser à l'aide d'une imprimante conventionnelle et l'impression serait en noir et blanc. Une autre solution économique serait de photographier l'écran; cela serait peut-être suffisant pour l'homme d'affaires, mais ne serait certainement pas assez précis pour un dessinateur ou un architecte.

Les traceurs ne fonctionnent pas du tout comme les imprimantes : ils tracent des lignes entre deux points au lieu d'imprimer un modèle de caractère ou une configuration de points. Ces divers systèmes utilisent des coordonnées X et Y. Tout comme un graphique peut être tracé en définissant les coordonnées où doit passer la ligne, toute forme peut être décrite en termes de coordonnées. Pour joindre ces coordonnées et ainsi reproduire la forme, on doit créer une forme quelconque de mouvement. Le stylo est donc fixé à un chariot qui se déplace selon l'axe X (gauche et droite) alors que le stylo se déplace sur le chariot selon l'axe Y (haut et bas).

Le type traditionnel de traceur est un traceur à plat où le papier est fixé sur un plateau plat au-dessus duquel se déplace le chariot (voir l'illustration). L'inconvénient est que le traceur doit être au moins aussi grand que la feuille de papier.

Il est possible de réduire la taille du traceur en adoptant une version à grande échelle de traceur à quatre stylos (voir l'illustration), où le papier se déplace dans une direction et le stylo dans l'autre. Le Strobe 100 et certains traceurs Hewlett Packard utilisent ce principe. Le mouvement du papier doit être commandé aussi précisément que celui du chariot dans le cas du traceur à plat. Pour obtenir cette précision, on utilise un moteur pas à pas. Un moteur pas à pas est un moteur très spécial qui n'effectue qu'une fraction de rotation à chaque impulsion de courant qu'il reçoit. On le retrouve principalement dans les lecteurs de disquettes, où il positionne la tête sur la surface de la disquette, et dans les robots (voir page 176).

La connexion d'un traceur à un ordinateur est généralement semblable à la connexion d'une imprimante, du moins au niveau de l'interface. Un traceur possède normalement une interface série (RS232) ou parallèle (Centronics ou IEEE488), qui peut être connectée au



Logement de stylos
Jusqu'à trois stylos peuvent être changés automatiquement. Le chariot retourne au logement de stylos et échange le stylo utilisé contre la prochaine couleur désirée. D'autres stylos peuvent être placés manuellement dans le logement de stylos.

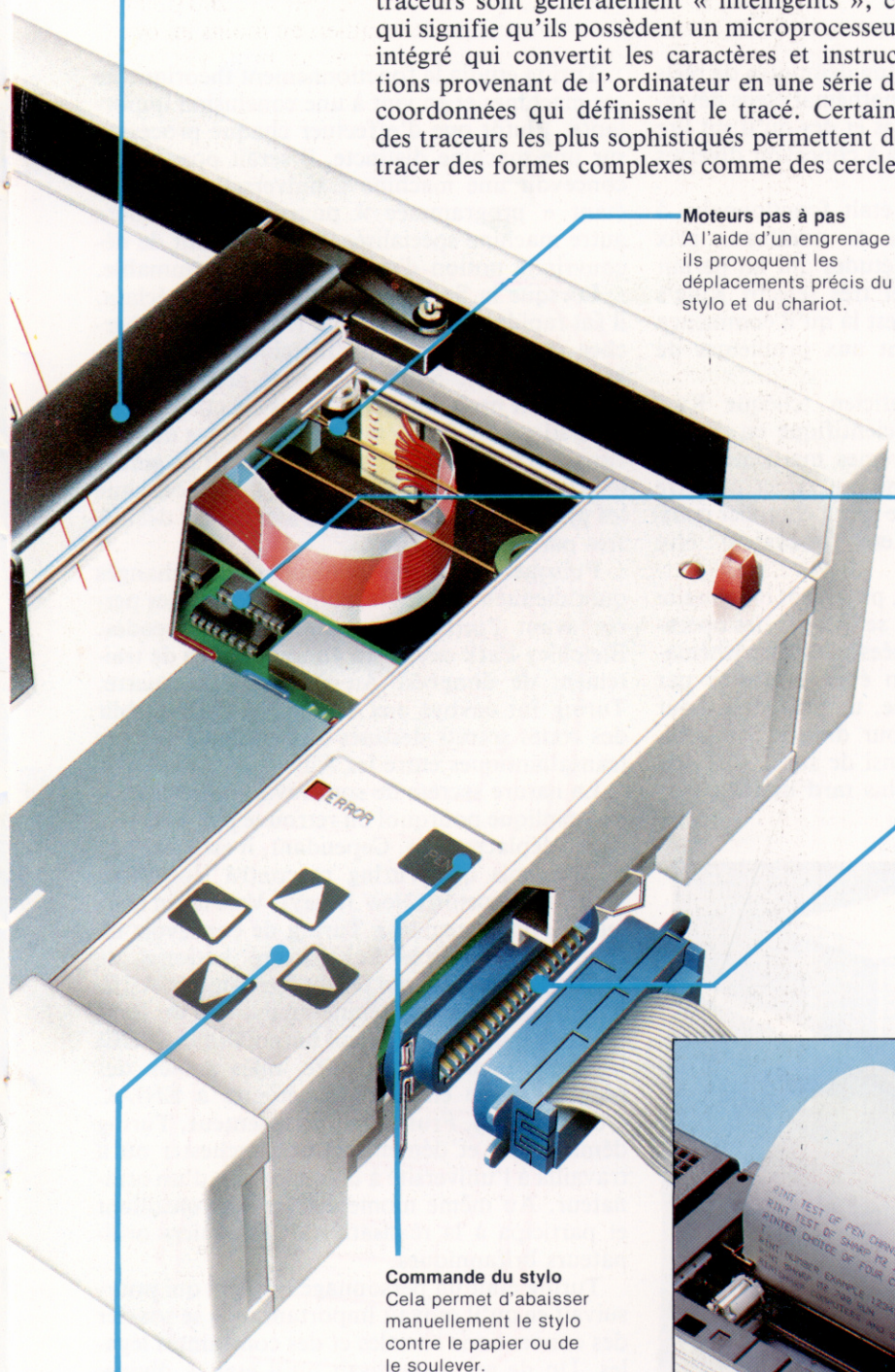
Pinces magnétiques
Ces lamelles magnétiques maintiennent le papier bien à plat sur le plateau du traceur.

Porte-stylo
Le stylo sélectionné est coincé magnétiquement dans le porte-stylo qui s'abaisse pour mettre le stylo en contact avec le papier.

port normalement utilisé par une imprimante. La programmation est souvent un peu plus compliquée, puisque, au lieu de n'envoyer que des résultats, le programme doit aussi fournir des renseignements relatifs à la présentation des résultats. Cela est normalement fait de la même façon que la réalisation d'un diagramme à l'écran.

En raison de la complexité de leur sortie, les traceurs sont généralement « intelligents », ce qui signifie qu'ils possèdent un microprocesseur intégré qui convertit les caractères et instructions provenant de l'ordinateur en une série de coordonnées qui définissent le tracé. Certains des traceurs les plus sophistiqués permettent de tracer des formes complexes comme des cercles ou des courbes en ne donnant que les points de départ; le traceur fait le reste. L'identification des graphiques et des diagrammes ainsi que la coloration des graphiques circulaires et des diagrammes de Gant sont souvent automatiques, ce qui simplifie grandement la programmation.

De nombreux traceurs sont livrés avec un logiciel qui leur permet de copier directement l'écran. Si ce type de programme n'est pas fourni, l'utilisateur doit écrire les routines nécessaires pour traduire l'information sur écran en codes appropriés pour piloter le traceur. Certains traceurs ne possèdent pas de jeu de caractères intégré, et le programmeur doit donc également définir les codes des lettres et des chiffres qu'il veut utiliser. Par contre, ce fait permet à l'utilisateur de concevoir ses propres caractères. Dès qu'une forme est conçue, elle peut être tracée à toute position, dans toute orientation ou dimension; une bibliothèque de formes peut ainsi être composée et utilisée à volonté. Des routines servant à tracer des cercles et des courbes sur des sections de graphiques sont souvent très utiles, particulièrement dans le domaine du graphisme de gestion, et celles-ci doivent aussi être créées.



Chariot du stylo
Le chariot peut être positionné en tout point sur l'axe X et le porte-stylo est alors positionné sur l'axe Y. La combinaison de ces deux mouvements permet de définir tout tracé sur la feuille.

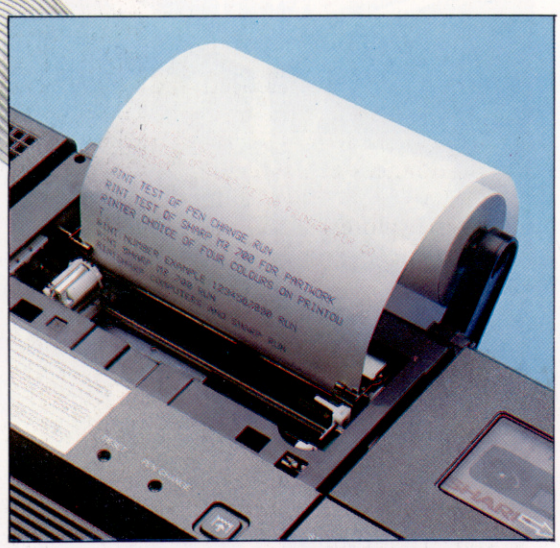
Moteurs pas à pas
A l'aide d'un engrenage ils provoquent les déplacements précis du stylo et du chariot.

Carte
Les traceurs sont généralement des dispositifs « intelligents », c'est-à-dire qu'ils peuvent obéir à des commandes évoluées comme « dessiner un cercle d'un centre A et d'un rayon B ». La carte renferme ses propres microprocesseurs RAM et ROM.

Connexion d'interface
Les traceurs sont connectés à un ordinateur au moyen d'une interface standard comme RS232 (série) ou comme Centronics (parallèle). Pour l'ordinateur, le traceur apparaît comme une imprimante.

Commande du stylo
Cela permet d'abaisser manuellement le stylo contre le papier ou de le soulever.

Commande de déplacement du stylo
Le stylo peut être positionné manuellement sur la feuille.



Traceur/ imprimante à quatre stylos

Ce mécanisme intéressa l'industrie de la micro-informatique lorsqu'il apparut sur le Sharp CE-150. Ses prédécesseurs le Tandy CGP-115 et l'Oric MCP-40 ont contribué à offrir à l'utilisateur de micro-ordinateur une impression couleur économique.
Comme dans toute conception bien pensée, le système est étonnamment simple. Un rouleau de papier est entraîné par un hérisson. Le papier est avancé et reculé par pas très précis pendant qu'un chariot muni de quatre stylos miniatures se déplace horizontalement.
Pour imprimer, que ce soit du texte ou du graphisme, le porte-stylo tourne jusqu'à ce que la bonne couleur soit positionnée et le stylo est pressé contre le papier. Les lignes horizontales sont créées par le mouvement du stylo pendant que le papier est immobile, les lignes verticales sont produites par le mouvement du papier contre le stylo.
(Cl. Chris Stevens.)

Alan Turing

Ce mathématicien britannique a donné son nom à un test d'intelligence de machine. Il travailla surtout à des fins militaires pendant la guerre.

Dès sa jeunesse, Alan Turing démontra de réels talents scientifiques. Les mathématiciens dévoilent leurs talents très jeunes et dès qu'il sut lire et écrire Turing commença à concevoir diverses inventions.

Pendant que son père était fonctionnaire à Madras, Turing remporta de nombreux prix scolaires et une bourse d'études qui lui permit de s'inscrire au célèbre et très sélectif King's College de Cambridge. C'est là qu'il commença à s'intéresser sérieusement aux problèmes de logique mathématique.

En 1931, le mathématicien tchèque Kurt Gödel surprit le monde scientifique en découvrant l'existence de théorèmes mathématiques qui, tout en étant vrais, ne pouvaient pas être démontrés. Alan Turing se mit à chercher comment certains d'entre eux pouvaient être démontrés.

Il proposa une machine, purement imaginaire à ce moment, qui pourrait se charger des opérations normalement effectuées par des mathématiciens. Chaque opération était effectuée par une machine; par exemple, une machine pour additionner, une autre pour diviser, une troisième pour intégrer, et ainsi de suite. Ces machines furent nommées plus tard machines de Turing.

Turing étudia le fonctionnement théorique de ces machines et en vint à une conclusion importante. Plutôt que d'effectuer chaque processus sur une machine distincte, il serait possible de concevoir une machine « universelle » qui en étant « programmée » pourrait imiter toute autre machine spécialisée. Turing venait de découvrir la notion d'ordinateur programmable.

Lorsque la Seconde Guerre mondiale éclata, il fut rapidement recruté par le centre de recherches gouvernemental de Bletchley Park (Buckinghamshire). Sans la guerre, ses machines n'auraient probablement jamais été construites, mais Bletchley Park était impliqué dans des travaux secrets et urgents ayant pour but de percer les codes militaires allemands. Rappelons que les premiers codes allemands avaient été déchiffrés par des Français.

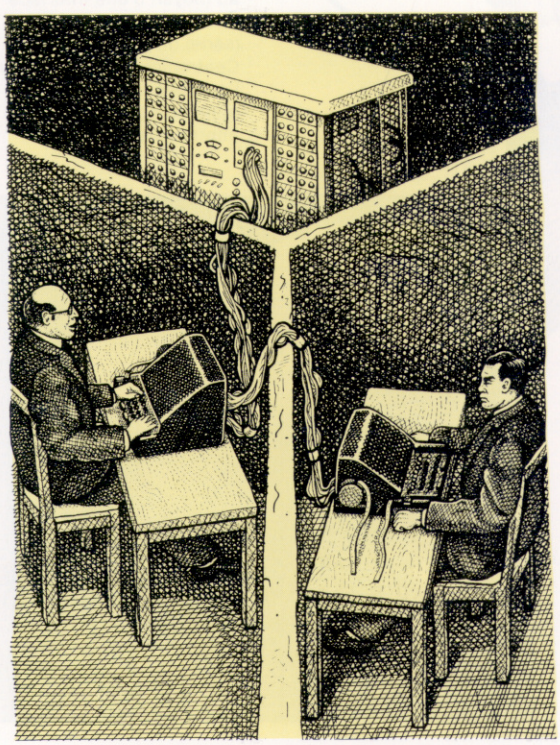
Puisque ces codes pouvaient être changés quotidiennement, les machines devaient les percer avant l'introduction de nouveaux codes. Bletchley Park devint un énorme centre de traitement de données. Au milieu de la guerre, Turing fut envoyé aux États-Unis pour établir des codes secrets destinés aux communications transatlantiques entre les Alliés.

La nature secrète de son travail de cette époque explique pourquoi on retrouve peu de traces de ses déplacements. Cependant, il est généralement admis que Turing rencontra Von Neumann à Princeton (New Jersey). Vers la fin de la guerre, on demanda à Turing de concevoir un ordinateur pour le Laboratoire national de physique, machine qui devait être appelée ACE.

Tout comme la machine analytique de Babbage, la machine de calcul automatique (ACE) fut très longue à construire, mais à bien des points de vue elle était supérieure à ENIAC (voir page 46). Frustré par cette lenteur, Turing démissionna et déménagea à Manchester où il travailla à l'université à la conception d'un ordinateur. Au même moment il devint consultant et participa à la réalisation des premiers ordinateurs britanniques.

Turing était un personnage original qui poursuivait ce qu'il croyait important sans se soucier des conventions sociales et des contraintes légales. Un de ses amis disait qu'il était « divinement retardé » pour reconnaître les défauts chez les autres, mais que son génie scientifique était incomparable. Il fut condamné pour homosexualité en 1952 et se suicida deux ans plus tard. Qui peut dire quelle contribution il aurait pu apporter à l'intelligence artificielle s'il vivait encore aujourd'hui?

Un mathématicien sportif
Alan Turing (1912-1954) recherchait inspiration et détente en courant sur de longues distances. Il avait noté l'effet bénéfique de l'effort physique sur la créativité.



Les machines peuvent-elles penser?
Pour répondre à cette question, Turing proposa son fameux test qu'il appela « jeu d'imitation », mais qu'on nomma par la suite « test de Turing ». Un homme est placé dans une pièce avec un téléscripteur. Celui-ci est relié à un téléscripteur commandé par un autre homme dans une autre pièce ainsi qu'à l'ordinateur que l'on désire tester. Le premier homme pose des questions à l'autre homme et à l'ordinateur. S'il est incapable de déterminer de quel côté de la cloison se trouve la machine, on peut alors dire que cette machine est intelligente. Cette approche est logique, nous ne pouvons reconnaître l'intelligence chez les autres qu'en comparant leurs réactions avec les nôtres.
(Cl. Bob Venables.)

PROGRAMME N° 3

INTRODUCTION AUX BOUCLES

Vous avez précédemment étudié un certain nombre d'instructions qui vous ont permis de résoudre des opérations d'addition simple. Pour aller plus loin dans les calculs que vous allez être appelé à effectuer, il est nécessaire d'introduire la notion de « boucle ». Celle-ci va vous permettre de confier à votre ordinateur des tâches qui simplifieront énormément l'entrée des informations et leurs traitements.
Supposons que vous vouliez imprimer des nombres entiers de 1 à 20 en plaçant un nombre par ligne. La manière la plus évidente est :

```
10   HOME
20   PRINT 1
30   PRINT 2
40   PRINT 3
50   PRINT 4
60   PRINT 5
70   PRINT 6
80   PRINT 7
90   PRINT 8
100  PRINT 9
110  PRINT 10
120  PRINT 11
130  PRINT 12
140  PRINT 13
150  PRINT 14
160  PRINT 15
170  PRINT 16
180  PRINT 17
190  PRINT 18
200  PRINT 19
210  PRINT 20
```

Exemple 1

Toutefois, vous devrez introduire 20 instructions, et si vous vouliez imprimer les nombres entiers de 1 à 1 000, il vous en faudrait 1 000. Sur la base de vos connaissances actuelles, vous pouvez imprimer (PRINT) des nombres entiers à partir de 1 en quatre instructions seulement au moyen d'une boucle.
L'instruction GOTO 20 renvoie à l'exécution de la ligne 20. Ceci s'appelle une boucle.

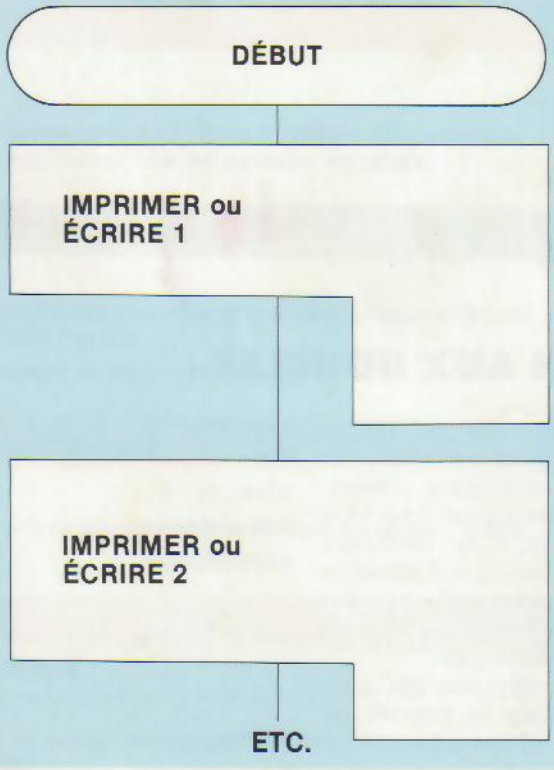
```
§10HOME
§15N=1
§?20 ?N
§30 N=N+1
§4050GOTO20
```

Exemple 2

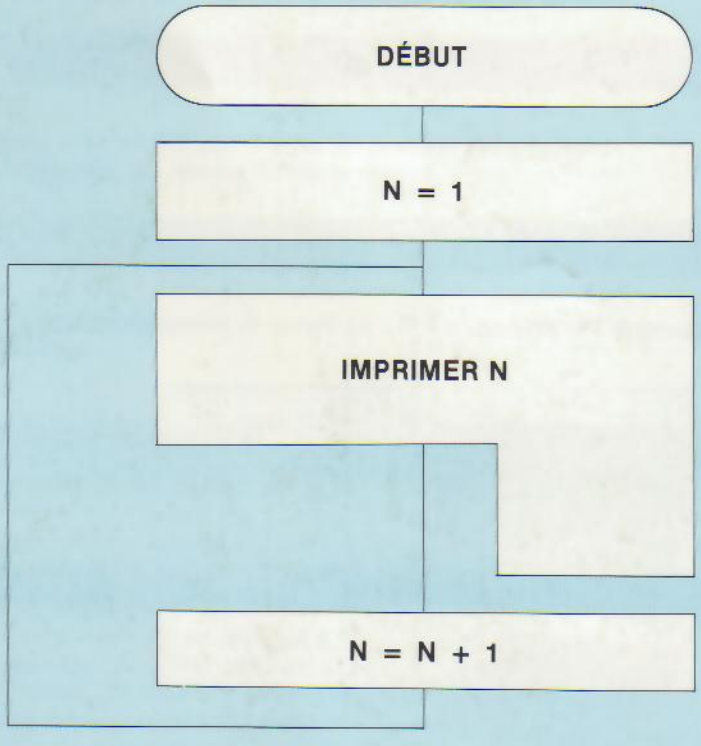
Exécution

```
§RUN
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
```

ORGANIGRAMMES



Exemple 1



Exemple 2

Remarques : le programme ne s'arrête pas.
Pour stopper l'exécution, taper CTRL-C RETURN

Touche CTRL
Simultanément
Touche C

REMARQUE :

L'exécution de ce programme ne s'arrête pas là. Il existe un procédé permettant de contrôler la longueur et la durée d'une boucle. Vous voulez par exemple que l'instruction exécute un GOTO si N est à 20 et qu'elle ne le fasse pas si N est à 21 ? Nous verrons la réponse la prochaine fois.