

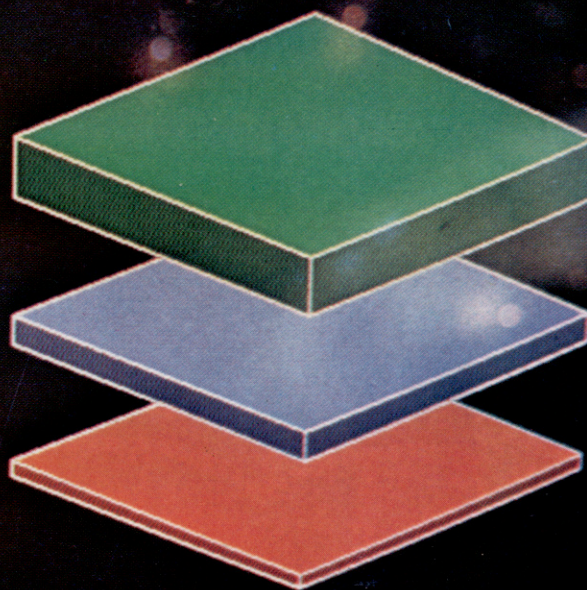
abc

N° 77

COURS
D'INFORMATIQUE
PRATIQUE
ET FAMILIALE

INFORMATIQUE

CDS 5000



Révision : la fonction RND

La cuisine de vos octets

Sur le pont de Christophe Colomb

La traduction automatique

EDITIONS
ATLAS



Besoins spéciaux

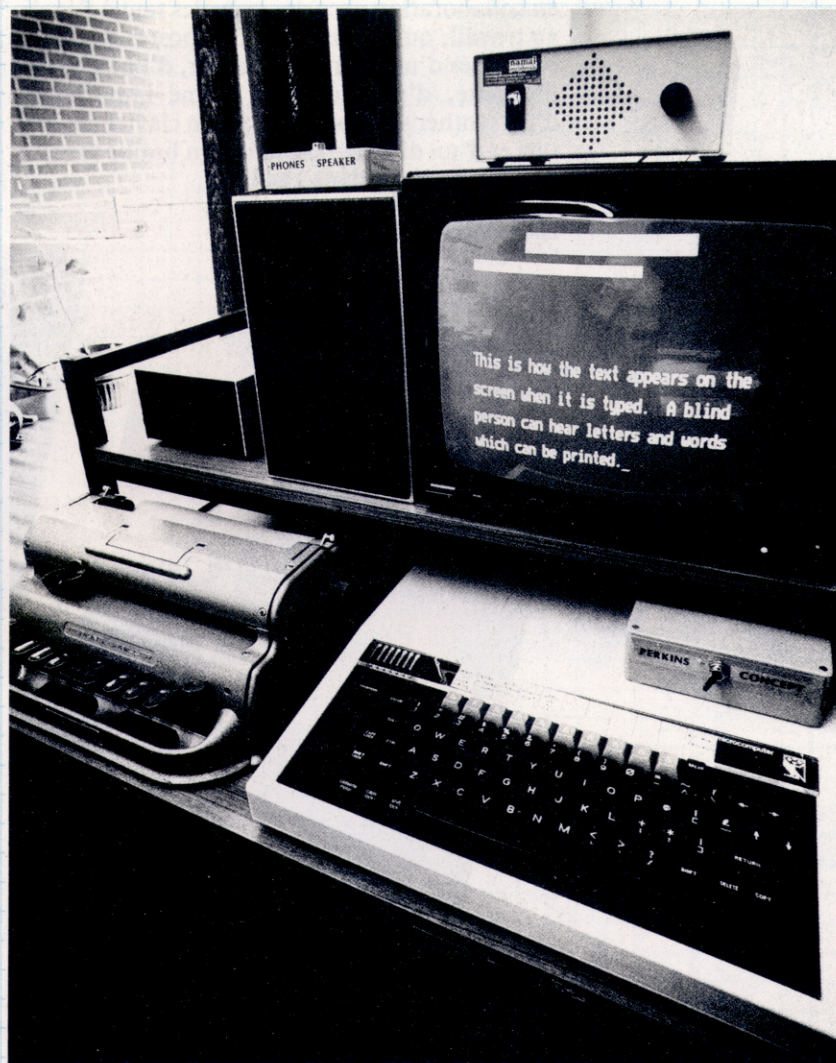
Pour les handicapés, l'ordinateur ne sert pas qu'à accélérer un processus d'apprentissage; il leur permet aussi de briser les barrières qui le séparent du monde normal.

Des enfants, mais aussi des adultes, ont des besoins spéciaux en matière d'enseignement. Cette situation peut venir d'un handicap physique, comme la surdité ou la cécité, ou d'un handicap mental. Certains besoins paraissent évidents, d'autres sont difficiles à diagnostiquer et à comprendre. Chaque incapacité présente des problèmes particuliers. Par manque de moyens et de soutiens dans la plupart des pays, ce marché n'est pas lucratif pour les créateurs de matériel et de logiciels. Néanmoins, l'énergie et l'engagement total des personnes qui travaillent dans ce domaine donnent un bon exemple au reste du corps enseignant. Un groupe de volontaires a par exemple réuni 120 000 F pour mettre au point un téléphone peu coûteux spécialement conçu pour les sourds. Ce type de travail est généralement mené à la base par des individus et par des groupes de bénévoles rejoints par des enseignants, des programmeurs et des ingénieurs, qui mettent au point du matériel et des logiciels destinés à cet enseignement spécialisé.

En Grande-Bretagne, des centres de recherche en microélectronique pour l'enseignement spécialisé (SEMERC) ont été créés. L'une de leurs principales fonctions consiste à former des enseignants dans des établissements spécialisés et à les mettre en contact avec les derniers développements technologiques pouvant faciliter leur travail. Les ressources de ces centres sont peu importantes malgré le travail énorme qui est demandé.

La complexité de la situation est accentuée par la diversité des problèmes rencontrés en enseignement spécialisé. Dans certains cas d'incapacité, les plus grands problèmes se situent au niveau de la communication. Une personne vive et intelligente peut par exemple être prisonnière d'un corps que la paralysie empêche de communiquer avec le monde extérieur. La microtechnologie a permis d'ouvrir de nouvelles voies de communication pour de nombreuses personnes. Par exemple, un capteur optique (une baguette) monté sur un casque en plastique et commandé par les mouvements de la tête permet aux personnes muettes et qui ne peuvent commander leurs membres d'utiliser un micro-ordinateur. La baguette est reliée à l'ordinateur au moyen du port d'entrée analogique. Le capteur optique est semblable à un crayon optique et sert à déplacer un curseur sur l'écran en réponse aux mouvements de la tête.

Un programme de traitement de texte nommé Write peut être utilisé au moyen de la baguette. Il affiche l'alphabet à l'écran, et des lettres indi-



viduelles (majuscules et minuscules) sont sélectionnées en les désignant. Il existe une fonction d'édition, et l'utilisateur dispose de listes de mots. Le texte peut être imprimé, sauvegardé ou effacé. Un programme nommé Paint permet de dessiner en six couleurs, et un autre, nommé Music, affiche un clavier. Les notes sont choisies avec le curseur et sont jouées.

La baguette peut être utilisée par des personnes dont les mouvements manuels sont très peu assurés et par des enfants dès l'âge de huit ans. On travaille actuellement sur un projet visant à relier le capteur optique à un synthétiseur de parole. Des réponses conçues pour des conversations téléphoniques, comme « Pouvez-vous répéter cette phrase? » sont sélectionnées à

Poste de travail

Ce poste de travail britannique est destiné à des personnes souffrant de diverses incapacités. Le dispositif de saisie en braille permet à une personne aveugle de saisir l'information. Le synthétiseur de parole peut ensuite produire une réaction audible. Le texte affiché à l'écran a été élargi pour avantager les personnes à faible acuité visuelle. (Cl. Tony Sleep.)



l'écran et sont reproduites par un synthétiseur de parole. On espère étendre le vocabulaire et mettre au point un type de liaison simple entre le synthétiseur et le téléphone.

Les ordinateurs sont conçus pour les voyants. Les moniteurs, imprimantes et autres traceurs produisent tous des sorties visuelles; tous les programmes utilisent un affichage visuel. Un projet universitaire a examiné comment adapter le matériel peu coûteux existant pour permettre son utilisation par des aveugles. Le projet a été mené en collaboration avec des aveugles tant à l'école, au travail, qu'à la maison. Des postes de travail composés d'un micro-ordinateur, d'une unité de disquette, d'un moniteur, d'une imprimante, d'un synthétiseur de parole, d'un clavier conceptuel et d'un dispositif de saisie en braille modifié ont été installés dans ces lieux.

Une touche magique

La baguette optique permet aux personnes sévèrement handicapées, incapables de parler ou de mouvoir leurs membres, de commander un micro-ordinateur. Le mouvement de la baguette est détecté par un capteur optique. (Cl. Chris Barker.)



Le dispositif de saisie en braille a été connecté à l'ordinateur pour permettre à un aveugle de lire la copie imprimée. Un logiciel permet de traduire le braille en texte normal et de le sauvegarder, de l'éditer et de l'imprimer.

Plusieurs programmes de traitement de texte « parlant » ont été mis au point pour utiliser un clavier classique. Les caractères et les touches spéciales tapées sont confirmés au moyen d'une sortie vocale. La touche Delete annonce également le nom du caractère effacé. Le texte est édité au moyen d'un curseur sonore. Au cours de son déplacement, le texte, les caractères, les mots sont reproduits vocalement. Le curseur peut être arrêté pour effectuer des additions ou des effacements. Des programmes plus évolués permettent d'appeler des fonctions de mise en page avant d'imprimer le texte. Les marges, les en-têtes et les espaces sont définis pour produire une copie finale impeccable.

Les mouvements de certains enfants handicapés ne sont pas suffisamment coordonnés pour utiliser un clavier standard. Un clavier plat, à touches à effleurement, a été mis au point pour de tels utilisateurs. Nommé « clavier conceptuel », il permet au professeur de superposer des grilles interchangeables qu'il aura lui-même conçues. Le clavier peut, par exemple, être divisé en quatre sections, où une image est dessinée dans chaque carré, ce qui permet de commander une tortue-robot. Certains logiciels utilisent ce clavier conceptuel.

La souris utilisée avec l'ordinateur Macintosh de chez Apple élimine également la « barrière du clavier » et offre de grandes facilités pour l'enseignement spécialisé. Il existe déjà un groupe d'utili-



lisateurs qui explore les possibilités de la machine dans ce domaine.

Des interrupteurs spéciaux, actionnés par diverses parties du corps, ont été mis au point. Par exemple, on a conçu un interrupteur destiné aux personnes ne pouvant exécuter un mouvement volontaire. Deux petits disques métalliques placés près des yeux détectent le mouvement horizontal du globe oculaire et les signaux électriques qu'ils produisent sont amplifiés pour commander des interrupteurs.

A l'université de Bradford, un clignotant orange a été placé sur une tortue pour permettre à une personne physiquement handicapée et partiellement aveugle de voir ce dispositif. A l'aide d'un clavier conceptuel, elle était capable de commander les mouvements de la tortue au sol et d'apprécier les résultats de ses interventions. Pour cette personne, comme pour de nombreux enfants handicapés, des expériences de ce genre constituent une première étape les conduisant à

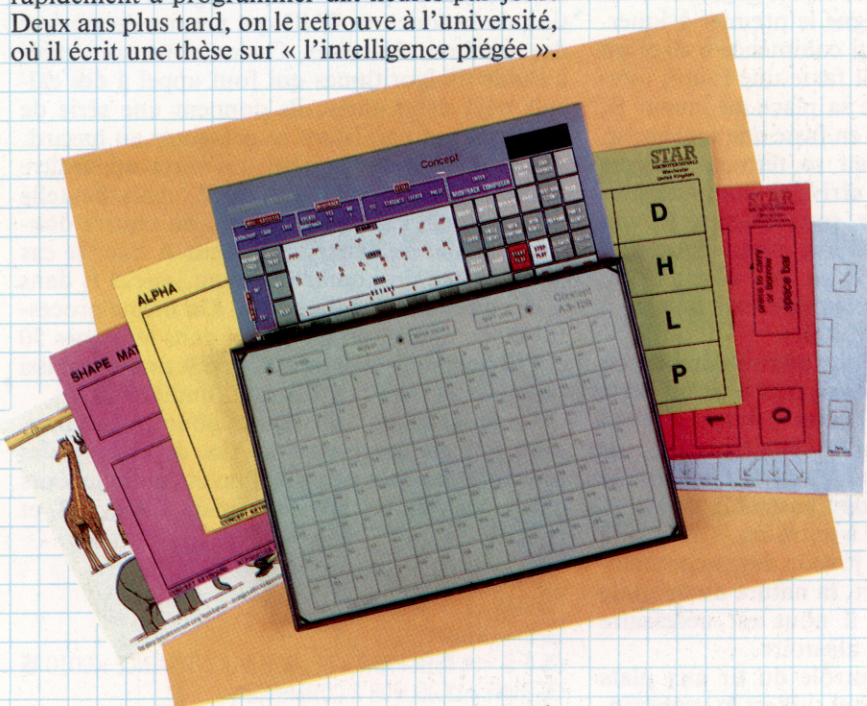


quitter le rôle d'observateur passif pour tendre vers une influence consciente sur leur environnement.

Lors d'une réunion d'un groupe d'utilisateurs de LOGO tenue en 1984, il fut décrit de remarquables exemples de progrès réalisés à l'aide de LOGO et de la tortue chez des enfants handicapés physiquement et mentalement.

Ainsi, Michel, âgé de dix-sept ans, séjournant dans un établissement spécialisé, n'avait jamais écrit un mot. Ses professeurs pressentaient chez lui une intelligence qu'il n'avait encore pu exprimer. Après avoir découvert l'ordinateur, il se mit rapidement à programmer dix heures par jour. Deux ans plus tard, on le retrouve à l'université, où il écrit une thèse sur « l'intelligence piégée ».

que de très peu de fonds pour acheter du matériel et il ne provoque que très peu de recherches puisque les sociétés de logiciels et de matériel réalisent leurs profits dans la gestion et l'industrie plutôt que dans l'enseignement. Le manque d'argent et de personnel compétent ralentit la mise au point des équipements nécessaires et les faibles budgets des écoles freinent l'acquisition de matériel informatique. Malgré ces lacunes au niveau des ressources, un travail prometteur est déjà entrepris.

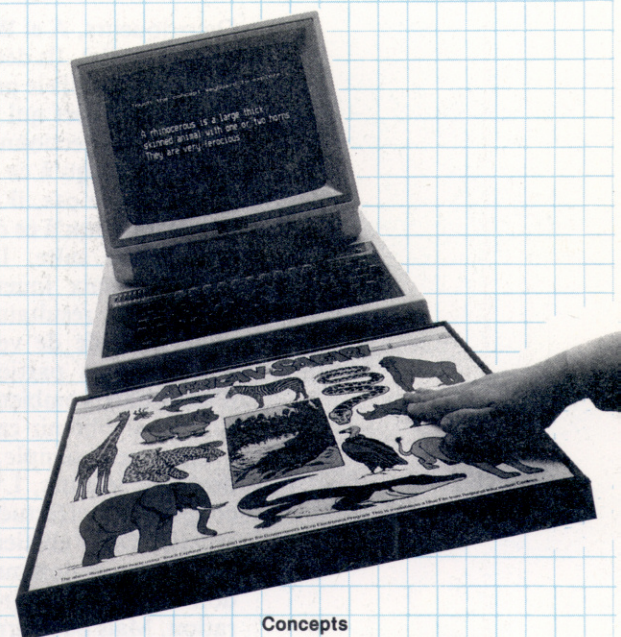


Un autre cas intéressant est celui d'un garçon de sept ans atteint d'autisme qui réussit à commander une tortue au moyen d'une boîte à boutons, dispositif similaire au clavier conceptuel. L'expérience suscita chez lui un tel enthousiasme, qu'il commença à parler.

A la voix et à l'œil

D'autres efforts ont permis la conception d'un dispositif servant à aider les enfants souffrant de difficultés d'élocution. Un microphone modifié de radio CB connecté à un micro-ordinateur permet aux enfants de commander diverses activités à l'écran au moyen de leur voix. Le logiciel mis au point pour ce dispositif leur permet de dessiner le relief d'une ville. La hauteur et la largeur des immeubles sont déterminées par l'intensité et la durée de la voix. D'autres programmes permettent de faire exploser des vaisseaux spatiaux sur l'écran, de diriger un bateau dans des rapides et de piloter un hélicoptère pour secourir un canoéiste. Ils donnent aux enfants l'opportunité d'utiliser leur voix de façon pratique.

Comme l'enseignement normal, l'enseignement spécialisé est en retard par rapport aux nouvelles découvertes technologiques. Il ne dispose



Concepts

Le clavier conceptuel est fondé sur les principes de la tablette graphique. La résolution de la tablette a été réduite; on a ajouté un masque qui permet aux utilisateurs souffrant de difficultés de coordination des mouvements de pouvoir utiliser efficacement l'ordinateur sans avoir à utiliser le clavier normal. La conception des masques peut aider à réduire la crainte qu'inspire le premier contact avec l'ordinateur chez un nouvel utilisateur.

(Cl. A.B. Division Europe.)

Ordinateurs et élèves doués

Les enfants doués et talentueux ont des besoins éducatifs particuliers. L'utilisation de micros offre pour eux de larges possibilités. Il en va de même chez les enfants handicapés. En fait, de nombreux enfants souffrant de handicaps physiques, comme la dyslexie, sont doués intellectuellement.

Comme pour la personne handicapée, le micro-ordinateur offre à l'enfant surdoué un moyen d'intervenir dans son environnement et d'écarter les barrières naturelles et artificielles au développement et à l'épanouissement individuel. Plus spécifiquement, les ordinateurs peuvent offrir à l'enfant surdoué :

- L'opportunité d'acquérir des connaissances et de développer de nouveaux intérêts à son propre rythme.
- De nouvelles approches créatives pour résoudre des problèmes.
- Un médium de communication et de coopération avec d'autres élèves doués.
- L'opportunité de faire des exercices fastidieux et fatigants d'une façon qui élimine l'ennui et qui entretient l'intérêt pour le sujet associé.
- Un nouveau médium à découvrir et pour la découverte.



Compteurs et aléas

Un jeu de « tir à la cible » se prête très bien à un contrôle par programme. Nous en profitons pour introduire la fonction aléatoire RND.

Deux joueurs, A et B, pratiquent le tir au pigeon (plat d'argile). Ils décident que celui qui réussira vingt tirs positifs le premier sera le gagnant. C'est le tirage au sort qui désigne le premier à jouer. Le joueur sélectionné qui commence à tirer est A. Il tire jusqu'à ce qu'il fasse une faute; lorsque tel est le cas, il cède sa place au joueur B, qui tirera jusqu'à ce qu'il en fasse une également, et ainsi de suite. Le jeu se termine lorsque l'un des deux joueurs arrive à comptabiliser vingt bons résultats sur son tableau. Il sera alors possible de voir, d'une part, lequel des deux joueurs participant au jeu a gagné et, d'autre part, combien de coups de feu il lui a fallu tirer pour obtenir cette victoire.

Dans cet exemple, il faut quatre compteurs, deux par joueurs. L'un comptera le nombre de coups de feu effectués et l'autre le nombre de tirs réussis. On s'abstiendra, à cette occasion, de toute intervention manuelle; seul, le programme doit assurer le contrôle de l'ensemble de l'opération. Mais pour arriver à notre fin, c'est-à-dire savoir, d'une part, quel joueur devra commencer à tirer et, d'autre part, la nature du coup de feu (manqué ou non), il nous est nécessaire d'introduire un nombre aléatoire.

Le programme de contrôle du tir aux plats d'argile avec compteurs qui suit est la transcription en BASIC du diagramme ci-contre.

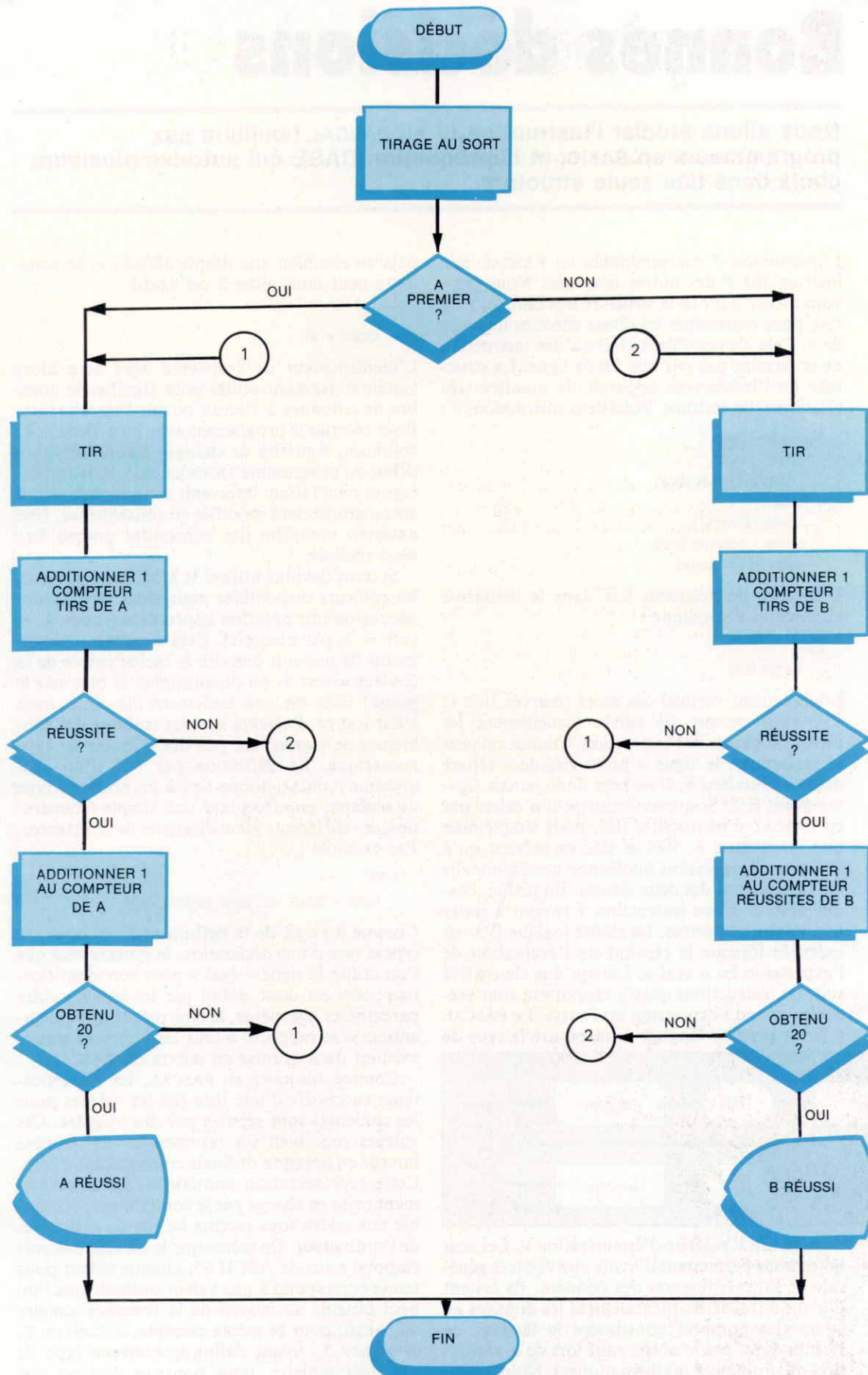
La partie qui peut paraître la plus difficile au moment de programmer correspond au tirage au sort initial. Ce point est résolu grâce à l'utilisation de la fonction RND qui permet de générer des nombres aléatoires. Ces nombres sont créés à l'aide d'algorithmes qui font appel à des calculs déjà déterminés; ils donnent une série de nombres qui paraissent se présenter au hasard. Ainsi, dans le cas présent, on verra apparaître seulement les nombres 1 ou 2, mais de telle manière que, par comparaison, on pourra donner le signal de départ au joueur A dans le cas où il s'agira du nombre 1, et au joueur B dans le cas du nombre 2 (ligne 20). Un même processus de comparaison intervient dans les lignes 50 et 110 quand il s'agit de savoir si le tir a été réussi (numéro 1) ou s'il a été raté (numéro 2).

Ce programme a été réalisé pour un Commodore 64; les compteurs de tirs de chacun des joueurs et les compteurs de l'un et l'autre joueurs ont été dénommés respectivement CA et CB, et A et B.

```
10 REM ... Tir au plat d'argile ...
15 REM ... Tirage au sort ...
20 S = INT (RND(1)*2) + 1
25 IF S = 1 THEN PRINT « LE JOUEUR A COMMENCE » GOTO 45
30 PRINT « LE JOUEUR B COMMENCE »
35 GOTO 100
40 REM .... ACTION DE TIR DU JOUEUR A ....
45 PRINT « TIR A »
50 X = INT (RND(1)*2) + 1
55 CA = CA + 1
60 REM .... TIR RÉUSSI OU RATÉ
65 IF X = 1 THEN GOTO 80
70 PRINT « RATÉ »
75 GOTO 100
80 PRINT « RÉUSSI »
83 A = A + 1
85 IF A <> 20 THEN GOTO 45
90 PRINT « JOUEUR A GAGNANT DANS »; CA; « TIRS »
95 END
99 REM .... JOUEUR B DOIT TIRER
100 PRINT « TIR B »
110 R = INT(RND(1)*2) + 1
120 CB = CB + 1
130 REM .... TIR RÉUSSI OU RATÉ
140 IF R = 1 THEN GOTO 170
150 PRINT « RATÉ »
160 GOTO 45
170 PRINT « RÉUSSI »
175 B = B + 1
180 IF B <> 20 THEN GOTO 100
190 PRINT « JOUEUR B GAGNANT DANS »; CB; « TIRS »
200 END
```



Patrick Rouchon



Bonnes décisions

Nous allons étudier l'instruction IF en PASCAL familière aux programmeurs en BASIC, et l'introduction CASE qui autorise plusieurs choix dans une seule structure.

L'instruction IF est semblable en PASCAL aux instructions IF des autres langages. Nous pouvons mettre à profit la structure très libre du PASCAL pour représenter les divers chemins logiques de IF. Cela est possible du fait qu'une instruction ne se termine pas par une fin de ligne. La structure de l'instruction apparaît de manière très claire par son écriture. Voici deux instructions IF :

```
IF compte = limite
  THEN
    WriteLn ('Pas de place')
  ELSE
    write ('Suivant?');
IF nombre > maximum THEN
  maximum := nombre
```

L'absence de l'élément ELSE dans la deuxième instruction IF implique :

```
ELSE
  (ne rien faire)
```

L'alignement vertical des mots réservés THEN et ELSE vous permet de suivre visuellement les niveaux logiques de l'instruction. Comme on peut le remarquer, le signe « point-virgule » sépare deux instructions IF. Il ne peut donc jamais figurer avant ELSE. Souvenez-vous qu'il n'existe pas en PASCAL d'instruction ELSE, mais simplement une instruction IF. THEN et ELSE ne servent qu'à délimiter l'expression booléenne conditionnelle des instructions des deux clauses. En réalité, chaque emploi d'une instruction IF revient à tester une valeur booléenne. La clause logique THEN est exécutée lorsque le résultat de l'évaluation de l'expression est « vrai ». Lorsqu'une clause ELSE suit, les instructions qui s'y rapportent sont exécutées quand l'expression est fausse. Le PASCAL a été le premier langage à introduire le type de

déjà vu combien une simple définition de constante peut nous aider à cet égard :

```
CONST
  largeur = 80;
```

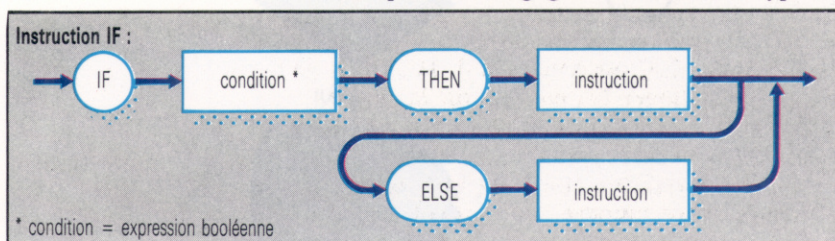
L'identificateur de constante largeur sera alors systématiquement utilisé pour signifier le nombre de colonnes à l'écran ou sur l'imprimante. Pour réécrire le programme avec un affichage 40 colonnes, il suffira de changer de définition au début du programme : tous les calculs de formatage et ceux faisant intervenir cette donnée seront automatiquement modifiés en conséquence. Une notation complète des constantes pourra être ainsi réalisée.

Si nous devons utiliser le graphisme couleur, les couleurs disponibles pourraient être étalonnées selon une notation appropriée (rouge = 1, vert = 2, par exemple). Cela supposerait néanmoins de pouvoir prendre la racine carrée de la couleur « vert », ou de multiplier le bleu par le jaune ! Cela est non seulement illogique, mais aussi source d'erreurs si nous traitons des problèmes ne concernant pas des données de type numérique. La définition par TYPE d'un programme PASCAL donne lieu à un nouveau type de scalaire, en effectuant une simple énumération des différents identificateurs de constantes. Par exemple :

```
TYPE
  teinte = (rouge, vert, jaune, magenta, cyan);
```

Comme il s'agit de la définition de ce nouveau type et non d'une déclaration, la syntaxe veut que l'on utilise le signe « égal » pour son identificateur ; teinte est donc défini par les valeurs entre parenthèses. De même, le type prédéfini « valeur-entière » se rapporte à tous les nombres entiers existant dans la mise en œuvre du PASCAL.

Comme toujours en PASCAL, les identificateurs successifs d'une liste (ici les valeurs pour les couleurs) sont séparés par des virgules. Ces valeurs sont bien sûr représentées de manière interne en notation ordinale commençant à zéro. Cette représentation numérique, automatiquement prise en charge par le compilateur, ressemble aux codes sous-jacents au jeu de caractères de l'ordinateur. De même que le caractère A correspond au code ASCII 65, chaque valeur pour teinte correspond à une valeur ordinale que l'on peut obtenir au moyen de la fonction scalaire ord. Ainsi, pour ce même exemple, ord(rouge) est 0, et ord(cyan) 5. Ayant défini ce nouveau type de grandeur scalaire, nous pouvons déclarer une



données dit « scalaire d'énumération ». Les scalaires nous permettent d'avoir une vue très générale de la classification des données. Ils évitent d'avoir à transcrire mentalement les données en codes (les nombres constituent le langage de l'ordinateur, pas le nôtre, sauf lors de la résolution de problèmes mathématiques). Nous avons

variable de manière habituelle :

```
VAR
  couleur : teinte;
```

Cette déclaration attribue un identificateur `couleur` à un élément de donnée du type `teinte`. De même :

```
VAR
  lettre : car;
```

donne la nature de l'élément de donnée appelé `lettre`. Les seules opérations que l'on peut utiliser sur des scalaires de numération sont les tests de relation et les fonctions scalaires. Nous pourrions écrire :

```
IF couleur < cyan THEN
  couleur := succ (couleur)
```

Souvenez-vous que `pred (rouge)` et `succ (cyan)` n'existent pas... La variable `couleur` est incompatible avec toute autre variable, scalaire ou non. Cela veut dire que nous ne pouvons plus effectuer des opérations illégales, prendre la racine carrée ou écrire :

```
couleur := couleur + 1
```

Il s'agit d'une restriction évidente. Caractères et nombres peuvent être utilisés comme paramètres d'instructions `WriteLn`, mais

```
WriteLn (couleur)
```

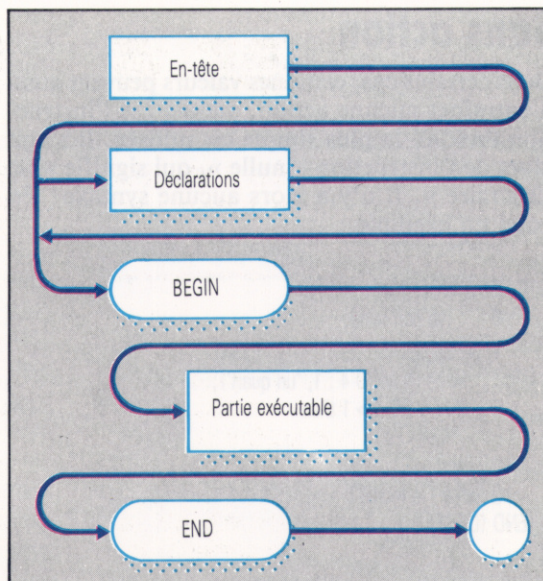
serait illégal. La raison en est que les valeurs de ce type sont purement conceptuelles. Si nous voulons afficher leur identité, il nous faudra faire correspondre les valeurs des couleurs aux chaînes de caractères. C'est là une application toute trouvée pour une autre construction syntaxique remarquable du PASCAL : `CASE`.

Nous avons vu que l'instruction PASCAL `IF` est tout à fait familière et gagne en lisibilité grâce aux possibilités de formatage du PASCAL. Il arrive cependant que des décisions à choix multiples doivent être prises. Elles supposent alors quelque chose comme :

```
IF N = 1
  THEN
    write ('1er')
ELSE
  IF N = 2
    THEN
      write ('2e')
    ELSE
      IF N = 3
        THEN
          write ('3e')
        ELSE
          write ('N°')
```

Lorsque l'instruction à exécuter dépend de la valeur d'une seule grandeur scalaire appartenant à une fourchette bien déterminée et limitée, nous avons intérêt à utiliser `CASE`. Dans le cas présent :

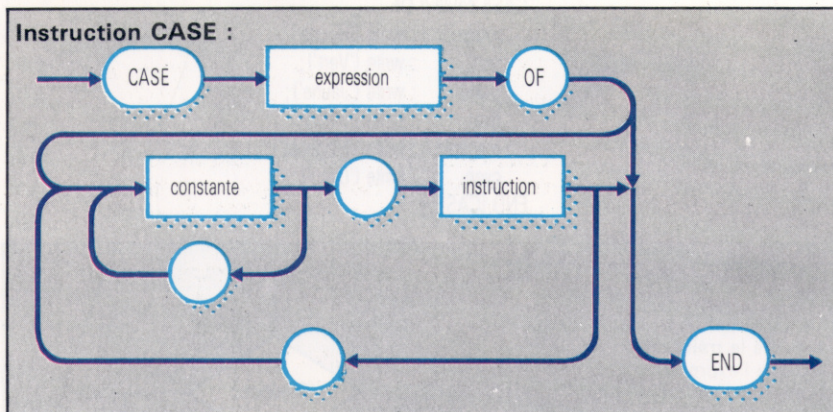
```
CASE N OF
  1 : write ('1er')
  2 : write ('2e')
  3 : write ('3e')
```



```
4, 5, 6,
7, 8, 9 : write ('N°')
END (CASE)
```

Vous remarquerez que cela n'est valable que pour des valeurs de `N` comprises dans la fourchette spécifiée dans le « corps » de l'instruction `CASE`, de 1 à 9 par exemple.

Toutes les valeurs que peut prendre `N` à l'exécution doivent être listées individuellement, et `N = 0` serait alors illégal.



De nombreux compilateurs PASCAL comprennent le mot réservé supplémentaire `OTHERWISE` ou `OTHERS`, qui sert à lister une décision par défaut. `CASE` est la seule instruction PASCAL à utiliser, `END` pour terminer une structure qui n'a pas été saisie par `BEGIN`. C'est pourquoi il est correct en PASCAL d'indiquer `END` comme sur la figure.

La syntaxe de `CASE` évalue l'expression scalaire située entre les mots réservés `CASE` et `OF`. Un simple nom de variable constitue une expression minimale qui ne suppose pas de calcul. La valeur obtenue est alors comparée à chaque constante listée dans les listes-étiquettes de situation (`case`). Lorsque la valeur est identifiée, l'instruction suivant le signe « : » est exécutée, et elle seule. Le chemin logique reste donc cohérent, et la syntaxe, rigoureuse. Lorsque plusieurs opérations sont nécessaires, une instruction composite comprise entre `BEGIN` et `END` peut intervenir.

Sans action

Dans certains cas, certaines valeurs peuvent aussi n'entraîner aucune action, auquel cas l'instruction PASCAL la plus simple est utilisée. Il s'agit alors de l'instruction « nulle », qui signifie « ne rien faire ». Il n'y a alors aucune syntaxe ! En voici un exemple :

```
CASE N MOD 4 OF
  0 : (ne rien faire);
  1, 3 : begin
    write (N MOD 4 : 1, 'un quart');
    if N MOD 4 > 1 then
      write ('s')
    end;
  2 : write ('la moitié')
END (CASE)
```

Vous remarquez que le signe « : » reste nécessaire pour séparer cette instruction « absente » de celle qui suit. La dernière instruction de l'intitulé n'en a pas besoin du fait qu'il est suivi d'un mot réservé (END) et non d'un autre intitulé ou instruction. L'emploi de l'opérateur MOD dans l'expression est destiné à faire en sorte que la valeur soit comprise dans la fourchette 0 – 3. MOD donne le reste des divisions entre entiers, comme avec le BASIC.

```
CASE couleur OF
  rouge : write ('Rouge');
  vert : write ('Vert');
  jaune : write ('Jaune');
  bleu : write ('Bleu');
  magenta : write ('Magenta');
  cyan : write ('Cyan');
END (CASE)
```

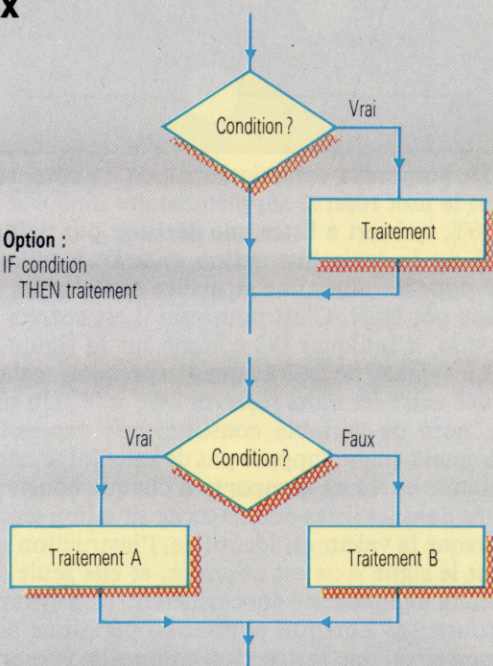
Constructions et choix

La construction IF ... THEN exécute le traitement lorsque l'expression donne un résultat « vrai ». Dans le cas contraire, le programme passe à l'instruction suivante. Si la construction IF ... THEN présente un embranchement possible, IF ... THEN ... ELSE permet un choix. Selon le résultat du test de début, un des deux traitements est exécuté.

Choix

```
IF condition
  THEN traitement A
  ELSE traitement B
```

Option :
IF condition
THEN traitement



Instructions mensuelles

Ce programme nous fournit un bon exemple de programmation en PASCAL. L'instruction CASE est utilisée pour transformer les données numériques entrées en scalaires de numération PASCAL. Ces derniers étant utilisés en données-chaînes résultantes aux fins de traitement interne, puis codés à nouveau.

```
PROGRAMME      Date (Entrées / Sorties):

CONST
  Point = '.';

TYPE
  Calendrier = (Jan, Fev, Mar, Avr, MAI,
               Jun, Jul, Aou, Sep, Oct, Nov, DEC);

VAR
  NomDuMOIS : Calendrier;
  jour,
  mois,
  année,
  Symbole de gauche : entier;
  : cart;
  AnnéeBisextile : booléen;

BEGIN
  WriteLn ( 'Donnez la date sous la forme : ' );
  WriteLn ( 'JJ/MM/AA : 48 );
  WriteLn;
  Write ( 'Date ? ' );
  read ( jour, symbole, mois, symbole, année );
  AnnéeBisextile := année MOD 4 = 0;
  (négliger les siècles )

  IF ( mois < 0 ) AND ( mois >= 12 )
  THEN
    CASE mois OF
      1 : NomDuMOIS := Jan;
      2 : NomDuMOIS := Fev;
      3 : NomDuMOIS := Mar;
      4 : NomDuMOIS := Avr;
      5 : NomDuMOIS := Mai;
      6 : NomDuMOIS := Jun;
      7 : NomDuMOIS := Jul;
      8 : NomDuMOIS := Aou;
      9 : NomDuMOIS := Sep;
      10 : NomDuMOIS := Oct;
      11 : NomDuMOIS := Nov;
      12 : NomDuMOIS := Dec;
    END (CASE);
  ELSE
    BEGIN
      WriteLn ( 'Eh ? ');
      WriteLn ( ' - le programme est, "sur le point'
              de se planter !' );
      END;
      ( NomDuMOIS n'est pas initialisé )

    CASE NomDuMOIS OF
      Jan, Mar,
      Mai, Jul : Restant := 31 - jour;

      Avr, Jun,
      Sep, Nov : Restant := 30 - jour;

      Fev : SI Bisextile
        THEN
          Restant := 29 - jour
        ELSE
          Restant := 28 - jour;
    END (CASE);

    WriteLn;
    WriteLn ( 'Il y a ' : Restant : 1 ' dans ' );

    CASE NomDuMOIS OF
      Jan : write ( 'Janvier' );
      Fev : write ( 'Février' );
      Mar : write ( 'Mars' );
      Avr : write ( 'Avril' );
      Mai : write ( 'Mai' );
      Jun : write ( 'Juin' );
      Jul : write ( 'Juillet' );
      Aou : write ( 'Août' );
      Sep : write ( 'Septembre' );
      Oct : write ( 'Octobre' );
      Nov : write ( 'Novembre' );
      Dec : write ( 'Décembre' );
    END (CASE);

    WriteLn ( Point );
  END.
```


Cuisines pour tous

Pour ceux qui sont chargés de concevoir l'agencement des cuisines, Ordicuisine a créé un système simple et performant, assurant à la fois la gestion administrative et commerciale des « cuisinistes ».

La cuisine, ce n'est pas seulement l'affaire des cuisiniers. C'est aussi celle des cuisinistes.

Qu'est-ce qu'un cuisiniste ? C'est un architecte spécialisé dans l'agencement des cuisines. La cuisine est aujourd'hui un produit de grande consommation destiné à tous les foyers. Alors qu'autrefois les particuliers acquéraient un à un les éléments d'équipement de leur cuisine, aujourd'hui celles-ci sont livrées tout équipées dans les appartements neufs. Le prêt-à-porter s'est adapté à l'immobilier. Mais il est aussi possible de faire réaliser une cuisine « sur mesure ». Dans les deux cas, le cuisiniste intervient.

Or, ce métier, surgi récemment et soudainement, n'a ni passé ni histoire. Aussi les cuisinistes sont-ils souvent dépourvus de qualification professionnelle.

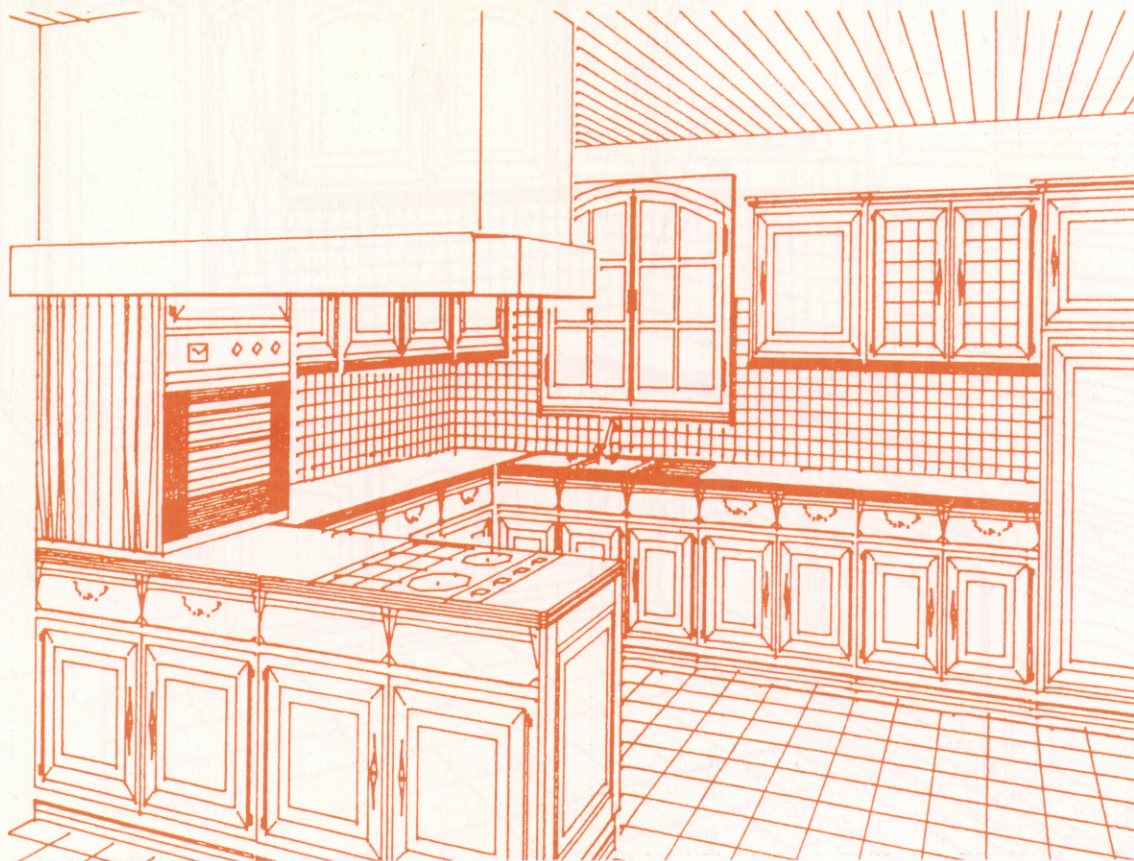
C'est pourquoi une équipe de cuisinistes a résolu de mettre à votre disposition un outil qui fera de vous un professionnel, moyennant un logiciel simple, « Ordicuisine », et un équipement informatique très répandu, puisqu'il fonctionne sur l'IBM PC ou compatible.

Cette réalisation a pu voir le jour grâce à une collaboration très étroite entre une équipe hautement qualifiée, composée de cuisinistes et d'ingénieurs, et des organismes professionnels de la cuisine, rassemblant fabricants, groupements de cuisinistes, syndicats et utilisateurs, au cours de cinq années de recherche et de développement au plus haut niveau.

Les programmes « Ordicuisine » peuvent conserver une qualité constante et être remis à jour régulièrement grâce aux relations entretenues par la société qui les conçoit avec les fabricants et les cuisinistes, notamment dans le cadre de séminaires des utilisateurs.

L'élaboration actuelle du système « Ordicuisine » permet ainsi la réalisation du dossier complet d'un client en trente minutes, avec la possibilité de programmer jusqu'à vingt devis différents par implantation avec votre coefficient de vente.

Pour cela, vous disposez de trois cent cinquante meubles dans vingt collections, ainsi que d'un programme annexe particulier à votre entre-



Cuisiniste

En France, la pièce la plus importante d'une habitation est la cuisine, bien sûr. Et les « cuisinistes » sont là... (Doc. Ordicuisine.)



prise — par exemple électroménager, etc. — et éventuellement d'un programme de gestion comptable, de publipostage. Bientôt, un logiciel de télétransmission vous permettra de faire votre devis chez le client et de le comptabiliser directement au siège de votre société par l'intermédiaire du réseau télématique.

L'informatique pour le succès des repas

Pour gérer l'administratif, un programme de gestion comptable est conseillé. Il tient à jour vos trois comptabilités en temps réel : la comptabilité clients, la comptabilité fournisseurs et la comptabilité générale.

Grâce au programme de gestion comptable, vous aurez à tout moment la disponibilité de votre balance fournisseurs (le montant de vos dettes détaillées) et de votre balance clients (le montant de vos créances).

Par ailleurs, il existe un choix important de programmes de gestion, car leur conception répond à des lois connues et pas nécessairement liées à la cuisine. La bonne gestion est universelle, elle ne dépend pas seulement des « sauces » ! Au gain de temps très appréciable — trente

minutes pour un dossier complet au lieu d'une demi-journée — s'ajoute le fait que les risques d'erreurs sont diminués substantiellement par « Ordicuisine » : cela est dû au devis précis, clair, qui facilite la communication avec le client. Le temps du client, lui aussi, n'est pas gaspillé inutilement, et ce dernier a l'avantage de voir tout de suite, grâce à la visualisation sur ordinateur et au dessin sur table traçante, l'allure finale de sa cuisine.

La société Ordicuisine a choisi l'ordinateur IBM PC pour piloter son système. Celui-ci s'accompagne d'un écran graphique, d'une imprimante Epson et d'une table traçante Bryans Gould.

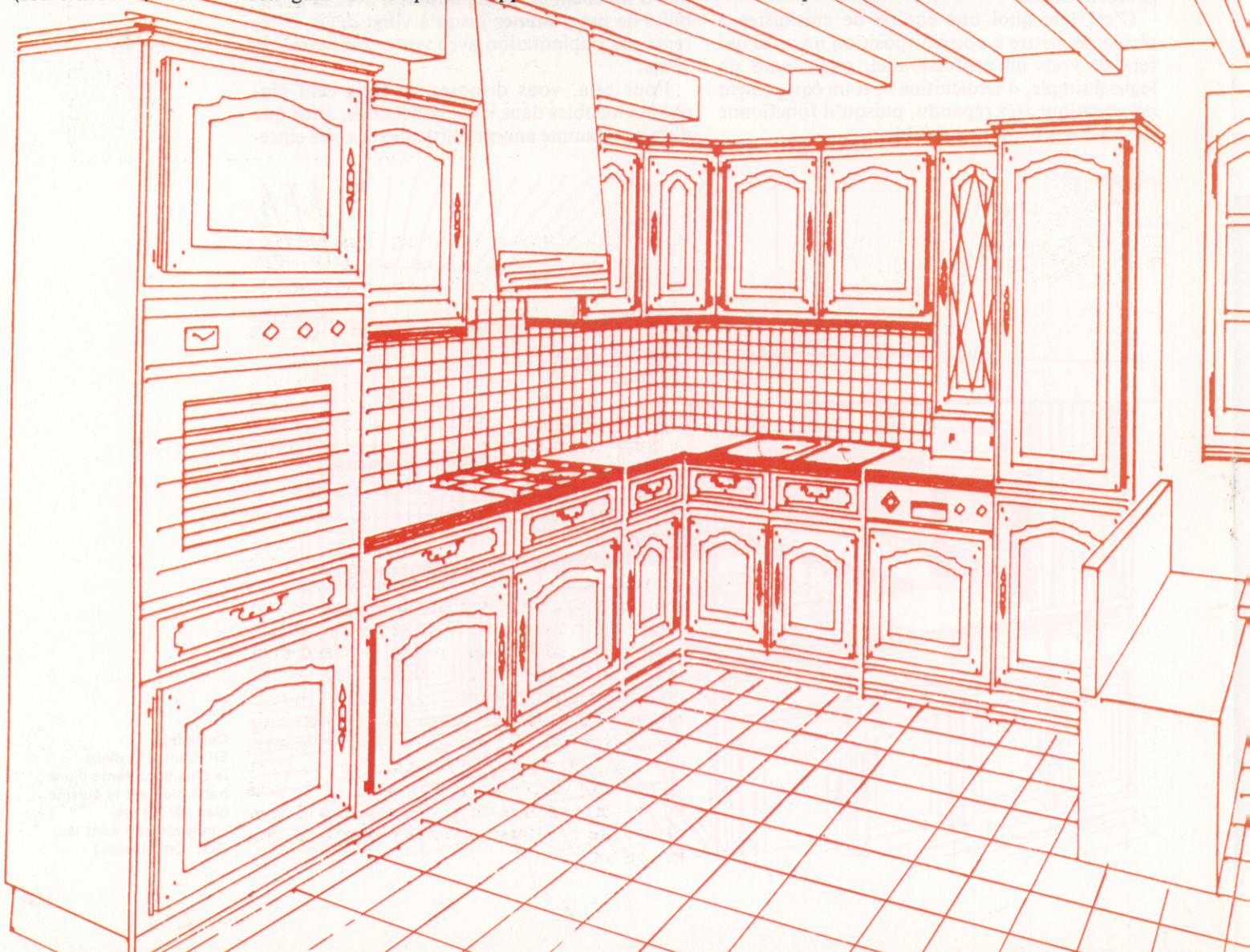
Les caractéristiques requises sont une capacité de mémoire de masse de 1 280 000 caractères et une mémoire vive (RAM) de 384 K. L'imprimante matricielle a une vitesse de cent soixante caractères à la seconde, en écriture bidirectionnelle. La table traçante est de série Colorwriter avec microprocesseur intégré et 2 K de mémoire tampon.

Le contrat de maintenance, matériel et logiciel, vous assure un service après-vente efficace par l'intermédiaire de « stations-service » et d'un réseau de techniciens agréés dans toutes les grandes villes. D'une manière générale, l'intervention n'excède pas trente-six heures.

Net et propre

Un devis précis et clair facilite la communication avec le client, qui voit immédiatement sur écran l'allure de sa future cuisine.

(Doc. Ordicuisine.)



Sur le pont

Nous allons réaliser un programme de simulation qui nous mènera dans l'Amérique du XVI^e siècle. Il est rédigé en BASIC Microsoft, et conçu sous forme de modules indépendants.

Tout le monde sur le pont

Quatre tableaux gèrent les données relatives à l'équipage. C\$(), W6() et CC() traitent des caractéristiques propres à chaque catégorie — description, salaires, nombre de marins correspondant. TS(,) renferme la composition réelle de l'équipe, qui peut comporter seize hommes au total. Il est préférable de traiter chaque catégorie par référence à un numéro plutôt qu'à un nom. Ce numéro correspond à une adresse dans les trois premiers tableaux. (Cl. Ian McKinnell.)



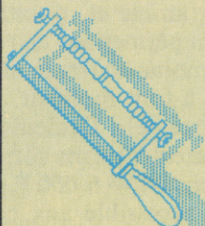
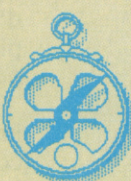











Dans ce jeu de simulation, vous allez tenir le rôle d'un financier du XVI^e siècle désireux d'organiser une expédition dans le Nouveau Monde. Vous devrez préparer le voyage, le mener à bien, et enfin commercer — de façon aussi profitable que possible — avec les indigènes.

Au début du jeu, le joueur dispose d'un navire de commerce et de deux mille pièces d'or. Il lui faudra rassembler un équipage, en lui offrant un certain salaire hebdomadaire, dont le montant dépendra de la qualification de chacun des membres. Bien entendu, les marins embauchés doivent représenter un éventail assez vaste de professions, sans pour autant accaparer à eux seuls toutes les dépenses. Le navire lui-même comporte seize membres d'équipage. S'il y en a davantage, ils auront besoin de nourriture et de boisson en plus, d'où des frais. Mais, inversement, des matelots en nombre réduit peuvent se montrer incapables de faire face à certaines circonstances imprévues (attaque de pirates, par exemple). Votre vaisseau ira moins vite, et vous dépenserez plus d'argent.

Le premier module du jeu initialise les variables et les tableaux dont le programme fera

usage; il gère également l'embauche des hommes d'équipage. Les tableaux sont DIMensionnés pour accepter toutes les données relatives aux matelots. C\$() renferme les descriptions des différentes tâches que se répartiront vos loups de mer. Il y a en tout cinq catégories; aussi le tableau est-il dimensionné, ligne 16, de façon à contenir cinq éléments. Les descriptions réelles sont ensuite affectées individuellement à chacun d'eux. Il est plus pratique — et moins coûteux en espace mémoire — de se référer à ces descriptions grâce à leur numéro dans le tableau. On l'appelle souvent l'adresse, parce qu'il donne la position d'un élément particulier au sein d'un tableau.

Les capacités de l'équipage augmentent ou diminuent tout au long du jeu en fonction des circonstances. Afin de garder trace de la composition du groupe, ainsi que de la force de chacun de ses membres, nous recourrons à un tableau à deux dimensions, TS(,), DIMensionné de façon à comporter seize cases et deux rangées. Chaque ligne verticale est consacrée à un marin; la case du haut gère son appartenance professionnelle, celle du bas indique sa force individuelle (fixée à 100 en début de partie).

	(1)	(2)	(3)	(4)	(5)																																															
C\$()						Description équipage																																														
	Matelot	Médecin	Charpentier	Navigateur	Cuisinier																																															
W6()						Salaires																																														
CC()						Décompte équipage																																														
TS(,)	<table><tr><td>1</td><td>1</td><td>2</td></tr><tr><td>100</td><td>100</td><td>100</td></tr></table>	1	1	2	100	100	100	<table><tr><td>1</td><td>4</td></tr><tr><td>100</td><td>100</td></tr></table>	1	4	100	100	<table><tr><td>1</td><td>3</td><td>5</td><td>1</td></tr><tr><td>100</td><td>100</td><td>100</td><td>100</td></tr></table>	1	3	5	1	100	100	100	100	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Catégorie Force individuelle
1	1	2																																																		
100	100	100																																																		
1	4																																																			
100	100																																																			
1	3	5	1																																																	
100	100	100	100																																																	
0	0	0	0	0	0	0																																														
0	0	0	0	0	0	0																																														
0	0	0	0	0	0	0																																														
0	0	0	0	0	0	0																																														
	Équipage embauché jusque-là				Espace pour matelots supplémentaires																																															



Le salaire alloué aux diverses catégories sera contenu dans un tableau à cinq éléments, WG(). Si nous désirons connaître quel est le gain hebdomadaire d'un médecin (catégorie 2), il nous suffira de chercher à WG(2). Les matelots touchent dix pièces d'or par semaine, les médecins vingt-cinq, les navigateurs vingt, les cuisiniers et les charpentiers quinze. Ligne 12, une variable MO est initialisée à 2000. Elle représente l'argent possédé par le joueur et diminuera à chaque paiement.

Il sera plus tard indispensable de savoir combien chaque catégorie de marins compte d'hommes, de façon à voir comment ils se comportent face à l'adversité. On pourrait pour cela consulter la ligne supérieure du tableau TS, mais il est plus simple, et plus rapide, d'en créer un autre, qui contiendra ce type de renseignements. Lui aussi n'aura besoin que de cinq éléments. Il est dimensionné à la ligne 18.

Guide de voyage

Le module ayant engrangé toutes les informations nécessaires, il affiche à l'écran diverses informations : but du voyage, durée approximative, argent à mettre de côté pour les provisions. Un sous-programme particulier assure l'impression de ces messages, qui apparaissent lettre par

lettre : ligne 9100, une variable S\$ accueille la phrase correspondante, et affiche un caractère après l'autre, avec, à chaque fois, une brève pause. Celle-ci se fait plus longue à chaque nouvelle formule (ligne 9200).

Le sous-programme de la ligne 1000 vous permettra d'embaucher votre équipage. Il faut sélectionner une catégorie particulière, ou appuyer sur F quand vous aurez fini.

Toutes les entrées valables sont stockées dans le tableau T\$(,), ligne 1156, tandis que CN garde en mémoire le nombre de marins embauchés jusque-là. A chaque nouvel arrivant, le montant total des salaires hebdomadaires WT est affiché ; on le calcule en ajoutant chaque élément du tableau WG() à l'ancienne valeur de WT (voir ligne 1158). A chaque phase, le programme affiche la composition actuelle de l'équipage (lignes 1160 à 1190). La ligne 1185 permet de s'assurer qu'une catégorie particulière de membres d'équipage comprend plus d'un homme ; dans ce cas, la description correspondante se voit ajouter un « S ».

Il arrivera plusieurs fois que le joueur se voie demander d'appuyer sur une touche pour que le programme continue. La variable K\$ se réduit au message « Appuyez sur une touche pour continuer », et elle est transmise à S\$ pour affichage. Notre module prend fin avec l'embauche. La prochaine fois, nous verrons comment acheter des provisions...

Préparatifs de départ
Le premier module de notre jeu gère l'embauche de l'équipage. Le joueur peut ainsi emmener jusqu'à seize hommes ; les données qui les concernent sont stockées dans plusieurs tableaux différents.

Module 1

```

9 PK$=" APPUYEZ SUR UNE TOUCHE POUR CONTINUER"
10 DIM TS(16,2): REM CARACTERISTIQUES EQUIPAGE
11 CN=0: REM NB HOMMES EQUIPAGE
12 MO=2000: REM ARGENT
13 DIM WG(5): WG(1)=10: WG(2)=25: WG(3)=15: WG(4)=20: WG(5)=15: REM SALAIRES
14 WT=0: REM TOTAL SALAIRES HEBDO
15 CM=16: REM EQUIPAGE MAX.
16 DIM CD$(5): CD$(1)="MATELOT": CD$(2)="MEDECIN": CD$(3)="CHARPENTIER"
17 CD$(4)="NAVIGATEUR": CD$(5)="CUISINIER"
18 DIM CC(5): REM NB.DS CATEGORIES
90 PRINTCHR$(147): ST$=" VOYAGE VERS LE NOUVEAU MONDE*": GOSUB 9100: PRINT
91 GOSUB 9200
92 ST$="VOUS ETES LE CAPITAIN D'UN VAISSEAU*": GOSUB 9100: PRINT
93 ST$="VOUS PARTEZ VERS LE NOUVEAU MONDE*": GOSUB 9100: PRINT
94 ST$="LE VOYAGE DEVRAIT PRENDRE HUIT SEMAINES, MAIS*": GOSUB 9100: PRINT
95 ST$="IL PEUT ETRE PLUS LONG*": GOSUB 9100: PRINT
96 ST$="VOUS DEVREZ RECRUTER UN EQUIPAGE, LE PAYER, ACHETER*": GOSUB 9100: PRINT
97 ST$="DES PROVISIONS ET DES MARCHANDISES*": GOSUB 9100: PRINT
98 ST$="POUR COMMERCER, VOUS AVEZ*": GOSUB 9100: PRINT
99 ST$="2 000 PIECES D'OR A DEPENDRE*": GOSUB 9100: PRINT: GOSUB 9200
100 PRINT: ST$=" BONNE CHANCE!": GOSUB 9100: GOSUB 9200: PRINT
91 ST$=PK$: GOSUB 9100
94 GETIP$: IF IP$="" THEN 94
95 GOSUB 9200
500 GOSUB 1000
999 END
1000 PRINTCHR$(147): PRINT " PHASE 1: RECRUTEMENT"
1010 PRINT "-----"
1012 PRINT
1015 GOSUB 9200
1020 PRINT: PRINT "CATEGORIES DE MARIN:"
1025 GOSUB 9200
1030 PRINT
1040 PRINT "SALAIRE HEBDO SELON QUALITE:"
1050 PRINT "-----"
1060 PRINT "1 MARIN 10 PIECES D'OR"
1070 PRINT "2 MEDECIN 25 PIECES D'OR"
1080 PRINT "3 CHARPENTIER 15 PIECES D'OR"

```

```

1090 PRINT "4 NAVIGATEUR 20 PIECES D'OR"
1100 PRINT "5 CUISINIER 15 PIECES D'OR"
1105 GOSUB 9200
1110 PRINT: PRINT
1120 ST$="1 ENTREZ CATEGORIE CHOISIE (1-5)*": GOSUB 9100
1122 ST$="2 OU 'F' POUR TERMINER*": GOSUB 9100: PRINT: INPUT IP$
1125 CT=VAL(IP$)
1128 IF LET$(IP$,1)="F" THEN PRINT: PRINT "FIN DU RECRUTEMENT.": GOSUB 9200: GOT01310
1130 IF CT < 0 AND CT > 5 THEN 1150
1139 PRINT: PRINT
1140 PRINT IP$: ST$=" NON VALABLE": GOSUB 9100
1142 GOSUB 9200
1145 ST$="RECOMMENCEZ"
1146 GOSUB 9100
1147 GOT01300
1150 PRINT: PRINT
1155 CN=CN+1: REM "EQUIPAGE DEJA RECRUTE"
1156 TS(CN,1)=CT: REM CATEGORIE
1157 TS(CN,2)=100: REM FORCE AU DEPART
1158 WT=WT+WG(CT): REM SALAIRES TOTAUX
1159 CC(CT)=CC(CT)+1: REM DECOMPTE CATEGORIE
1160 ST$="EQUIPAGE RECRUTE."
1170 FOR T=1 TO 5
1180 PRINT ST$: CC(T): " " : CD$(T):
1185 IF CC(T)=1 OR CC(T)=0 THEN PRINT "S": GOT01189
1186 PRINT " "
1189 ST$="
1190 NEXT
1195 PRINT: PRINT "SALAIRE HEBDOMADAIRE TOTAL "WT
1200 IF CN=CN-1 THEN PRINT: ST$="PLUS QU'UN SEUL !": GOSUB 9100: GOT0
1202 IF CN=CN THEN PRINT: ST$="EMBAUCHE TERMINEE !": GOSUB 9100: GOT01310
1295 REM
1300 GOT01015
1310 PRINT: ST$=PK$: GOSUB 9100: PRINT: GOSUB 9200
1320 GETIP$: IF IP$="" THEN 1320
1998 RETURN
9100 REM IMPRESSION DE ST$
9110 FOR S3=1 TO 32
9115 IF MID$(ST$,S3,1)="" THEN S3=32: GOT09140
9120 PRINT MID$(ST$,S3,1):
9130 FOR S4=1 TO 25: NEXT
9140 NEXT: PRINT
9189 RETURN
9200 REM RETARD BOUCLE
9210 FOR S5=1 TO 1000: NEXT
9299 RETURN

```

Variantes de basic

Spectrum :

Procédez aux modifications suivantes :

```

16 DIM C$(5,9): C$(1)="MATELOT": C$(2)="MEDECIN": C$(3)="CHARPENTIER"
80 CLS: LET S$=" VOYAGE VERS LE NOUVEAU MONDE*": GOSUB 9100: PRINT
94 LET P$=INKEY$: IF P$=" " THEN GO TO 94
1005 CLS: PRINT " PHASE 1 - RECRUTER L'EQUIPAGE "
1128 IF P$(1 TO 1)="" THEN PRINT "EMBAUCHE TERMINEE."
GOSUB 9200: GOT01310
9115 IF S$(3 TO 3)="" THEN S3=32: GOT0 9140
9120 PRINT S$(3 TO 3);

```

BBC Micro :

Procédez aux modifications suivantes :

```

80 CLS: S$=" VOYAGE VERS LE NOUVEAU MONDE*": GOSUB 9100: PRINT
94 P$=GET$
1005 CLS: PRINT " PHASE 1 - RECRUTER L'EQUIPAGE "

```




Traduction du progrès

La société Gachot n'est pas une P.M.E. comme les autres. Elle est en train de devenir le leader mondial de la traduction automatique, grâce au système Systran.

L'une des premières P.M.E. en France à s'informatiser est aujourd'hui aussi celle qui développe et commercialise une application révolutionnaire de l'informatique : la traduction automatique.

En 1947, Jean Gachot crée, à Soisy-sous-Montmorency, dans la banlieue parisienne, la société qui porte son nom et qu'il dirige actuellement. Dès le début, la personnalité de son fondateur a imprimé à Gachot S.A. ses caractéristiques marquantes : innovation technologique et réalisations exceptionnelles.

Toma, un Américain d'origine hongroise, qui a inventé un système de traduction automatique appelé Systran. Conçu au départ dans un cadre militaire, en vue de traduire des documents scientifiques et techniques du russe en anglais, et inversement, ce système s'est peu à peu élargi à la plupart des langues européennes (allemand, français, italien, portugais, espagnol).

L'invention du Dr Toma est également exploitée depuis plusieurs années par la General Motors, la Xerox et de nombreux autres « géants » industriels du continent nord-américain pour la traduction de documents techniques.

Au début de 1982, Gachot S.A. s'assure l'exclusivité de la commercialisation du système Systran pour les pays arabes, et finance différents projets pour développer les programmes et les dictionnaires anglais-arabe.

Pourquoi l'arabe ? Tout simplement parce que c'est la sixième langue parlée dans le monde (après le chinois, l'anglais, le russe, l'espagnol et l'hindi), et qu'en tant que langue cible de la traduction, il vient actuellement au cinquième rang mondial, tout de suite après l'anglais, et il représente 9 % de l'ensemble des traductions écrites mondiales. Soit près de 300 millions de dollars — ce qui équivaut à 3 milliards de francs français ! — et une croissance annuelle de plus de 10 %.

« Les installations proposées sont, en fait, des ensembles complets, clés en main, qui sont de véritables usines à traduire », déclare à cet égard Jean Gachot.

L'ensemble des opérations de traduction automatique s'effectue à l'aide d'ordinateurs de grande capacité, de lecteurs optiques pour la saisie des textes imprimés ou dactylographiés et leur transfert sur support magnétique, d'imprimantes à laser et de photocomposeuses pour l'édition des textes traduits, de terminaux spécialisés, de contrôleurs de réseaux et, bien entendu, de logiciels considérables et complexes, de traducteurs de gestion et de traitement de texte.

En novembre 1984, la Communauté économique européenne (C.E.E.), qui a aussi adopté le Systran pour traiter les montagnes de traductions anglais-français, français-anglais, etc., signe avec la société Gachot un accord par lequel le système Systran sera mis à la disposition des organismes du secteur public des pays de la Communauté.

Il va sans dire que Jean Gachot compte beaucoup sur cette nouvelle activité pour générer de substantiels profits et permettre ainsi à sa société de rester à la pointe du progrès technologique dans tous ces domaines cités plus haut.

De l'avant

Partir de l'industrie alimentaire pour aboutir — et ce n'est pas une finalité — à l'informatique, c'est pour Jean Gachot un chemin qui n'est pas surprenant lorsqu'on recherche avant tout l'innovation technologique. (Cl. Gachot S.A.)



Partie de l'industrie alimentaire et du traitement des jus de fruits pour passer ensuite à l'industrie chimique, à la pétrochimie, au pétrole, la société Gachot s'est ensuite lancée dans le domaine de la robinetterie industrielle, où elle a su maintenir un délicat équilibre entre la création de vannes et de robinets, les investissements nécessaires à leur fabrication et ceux indispensables à leur distribution commerciale. Elle se classe ainsi parmi les mille premières entreprises françaises, et occupe le troisième rang pour la robinetterie.

Disposant d'un département informatique depuis 1962, Gachot fut l'une des premières sociétés en France à s'informatiser. Dès 1970-1971, elle met au point un logiciel destiné à la programmation de problèmes de gestion intitulé « langage Ulysse ». Celui-ci permet une mise au point en quatre mois, au lieu d'un an et demi, d'une nouvelle chaîne de facturation.

Face aux difficultés conjoncturelles qui touchent l'ensemble de l'industrie depuis la fin des années 1970, Jean Gachot s'interroge : pourquoi ne pourrait-on pas appliquer à la robinetterie les méthodes ayant permis d'obtenir en électronique les résultats explosifs que l'on connaît ?

Son intérêt pour l'électronique et l'informatique amène Jean Gachot à rencontrer Peter



Actuellement, Gachot S.A. propose ses services à trois types de clients : elle effectue la saisie optique de documents imprimés, convertissant ainsi des éléments aussi divers que lettres dactylographiées, notices techniques, livres ou articles, en enregistrements magnétiques (disques, disquettes, bandes magnétiques, etc.) au format désiré par le client. Celui-ci peut dès lors soumettre ces enregistrements à tout type de traitement de texte, en vue, entre autres, d'une traduction ultérieure ou de l'incorporation dans un autre document imprimé.

A partir de ces enregistrements, Gachot S.A. propose deux types de traduction : soit une traduction dite « brute », soit une traduction « révisée ». Le premier cas consiste à faire subir au texte source le traitement par Systran dans l'un des couples langue source/langue cible suivants : français-anglais, anglais-français, anglais-allemand, anglais-italien ou anglais-arabe.

Ces traductions brutes n'étant généralement pas parfaites, elles nécessitent une mise au point par un traducteur humain. C'est pourquoi Gachot S.A. propose également, en collaboration avec la Chambre nationale des entreprises de traduction, une traduction finie.

Si le coût de la traduction automatique est encore relativement élevé — elle est facturée à quelques centimes le mot —, sa rapidité vertigineuse dépasse de loin les moyens humains : le Systran traite trois cent mille mots à l'heure et peut travailler vingt-quatre heures par jour sans congés, alors qu'un traducteur humain ne traduit que deux cent cinquante mots à l'heure environ, et ce huit heures par jour. Mieux encore, le Systran est capable de conserver une homogénéité terminologique remarquable d'un bout à l'autre des ouvrages traduits.

C'est ainsi que Gachot S.A., qui a déjà « inventé » notamment les robinets à boule, le V16, la discovanne, etc., bien connus de tous les spécialistes et des utilisateurs de robinetterie industrielle, va rapprocher l'Europe du monde arabe, et permettre ainsi à la France de réaliser une « première » mondiale dans le secteur de l'informatique.



A partir de sa console, le client peut faire traduire ses textes par Systran, en se reliant au centre serveur, via un modem, par liaison téléphonique PTT.

L'écran du terminal visualise côte à côte les textes en langue source et en langue cible, facilitant ainsi l'intervention interactive de l'opérateur.

Le marché mondial de la traduction

Voici quelques chiffres sur le marché mondial de la traduction (statistiques communiquées par la C.E.E.) :

Marché actuel.

- 150 millions de pages par an.
- 175 000 traducteurs permanents.
- 3 milliards de dollars de chiffre d'affaires par an.

Marché prévisionnel.

9 à 10 % d'expansion par an.

Autres possibilités.

Le marché potentiel supplémentaire représenterait au moins + 30 % du marché actuel, si l'on pouvait assurer une baisse des délais et des prix.

Il est à remarquer que la traduction est un poste important du budget de fonctionnement d'organismes tels que les Nations unies (70 millions de dollars, soit 7 % du budget) ou la C.E.E. (330 millions d'ECU, soit 40 % des dépenses administratives).

Il est donc évident qu'un abaissement notable des coûts et des délais de traduction par l'utilisation des possibilités de l'informatique permettra la traduction d'une masse importante de documentation et transformera le marché actuel.



Avant d'être traduits automatiquement, les textes imprimés sont saisis par un lecteur optique qui les transforme en données numériques stockées sur un support magnétique.

Maestro

Avant de « connaître » la musique, sachez que le jeu que nous vous présentons ici utilise des fonctions propres aux micro-ordinateurs conçus selon le standard MSX.



Ce programme n'est pas vraiment un jeu, mais il constitue une bonne démonstration des possibilités musicales du Spectravideo. Deux voies seulement sont utilisées pour cette invention en *la* mineur de J.-S. Bach. Il ne tient qu'à vous d'en ajouter une troisième.

```

10 REM *****
20 REM * MAESTRO *
30 REM *****
40 KEY OFF
50 CLEAR 5000
60 SCREEN 1,2
70 A$="VST32L32D4R32EA05C04BEB05DL16CE04
  G#05E"
80 A1$="VST32L32D2A1603A8G+16AEA04C03BEB
  D4D"
90 A$=A$+"04L32AEA05C04BEB05DL16C04A8R16
  "
100 A1$=A1$+"L16C03AG+EL32AEA04C03BEB04D
  "
110 A$=A$+"05L32R32ECE04A05C04EGL16FA05D
  FL32FD"
120 A1$=A1$+"L16C03A04C03AL32D4D03AFADFD
  2A03C02L16B"
130 B$="04B05D04G0BDFL16EG05CE"
140 B1$="03D6BL32BGECE026BL16A03CL32DFO
  2B03D02L16GB03L32CE02A03C02L16FDL32G03GF
  6C604CED03G04DF"
150 B$=B$+"L32EC04A05C04L16F05DL32D04BGB
  L16E05CL32C04AFAL16DB05C8R"
160 B$=B$+"04L32R32G05CED04G05DFL16EG04B
  05GL32C04G05CED04G"
170 B1$=B1$+"L16EC03BGL32D4C03G04CED03G0
  4DFL16ECB"
180 C$="05DFL16ECBE"
190 C1$="R16R32L32GEGCE03GB"
200 C$=C$+"06L32C05AEACE04A05CL16DF+A06C
  "
210 C1$=C1$+"L16A04CEGL32F+ADF+03A04D03F
  +A"
220 C$=C$+"L3205BGD04B05D04G05L16CEGBL

```

```

32AF+"
230 C1$=C1$+"L16GB04DF+L32EGCE03G04C03EG
  L16F+"
240 D$="D+F+04B05D+04F+AL16G05GL32GECE"
250 D1$="AB04D+L32R32ECE03A04CEG"
260 D$=D$+"04L16A05F+L32F+D04B05D04L16G0
  5EL32EC04A05C"
270 D1$=D1$+"F+D03B04D03GB04DF+EC03A04C0
  3F+A04C16"
280 D$=D$+"04F+05GF+ED+F+04B05D+E8R8L32"
290 D1$=D1$+"C03B04C03AL16B02B03L32E04E0
  3BGE02BGB"
300 E$="R32GB-GEGC+EGEC+E04AGFED05FA-FDF
  04B05DFD04B05D04GFEDC05EGECE04A05CD+C04A
  05C04F+ED+C+03B05D"
310 E1$="L16E03EGB-C+8R804D03DFA-D2B8R80
  4C03CEF+02A8R803B"
320 F$="FD04B05D04G+B05D04BG+BER32R16R32
  EA05C04BEB05DL16C04AG+EL32A05CEC04A05C04
  F+A05C04AF+AD+05C04BAG+B"
330 F1$="02B03DF02G+8R32D04L32D03B04L16C
  03AG+EL32AEA04C03BEB04DCEAECE03A04C03F+A
  04C03AF+AD+F+L16E"
340 G$="05D04BG+BDFG+FD03B04FEDCEAECE03
  A04CD+C03A04C03F+04C03BAL16G+04BG+ER32L3
  2EA05C04BEB05D0C04A"
350 G1$="G+BG+E02BG+EA03CEC02A03C02D+8R3
  2L3203BG+EDBG+DL16CE02G+03E02A"
360 H$="05CED04B05DFCEGEFEDC04B05CDEFD6+
  DBDCAF04B05D04G+B05C04AEABG+AECE03AB"
370 H1$="03F+02B03G+CADB-G+FD02BG+ADEFD
  E03E02ABAB"
380 FOR I=1 TO 10
390 X$=X$+CHR$(0)
400 NEXT I

```

```

410 X$=X$+CHR$(30)+CHR$(63)+CHR$(127)+CH
  R$(127)+CHR$(63)+CHR$(30)
420 X$=X$+CHR$(192)+CHR$(160)+CHR$(144)+
  CHR$(144)+CHR$(144)+CHR$(160)
430 FOR I=1 TO 8
440 X$=X$+CHR$(128)
450 NEXT I
460 X$=X$+CHR$(0)+CHR$(0)
470 SPRITE$(1)=X$
480 GOSUB 660
490 PLAY A$,A1$
500 GOSUB 660
510 PLAY B$,B1$
520 GOSUB 660
530 PLAY C$,C1$
540 GOSUB 660
550 PLAY D$,D1$
560 GOSUB 660
570 PLAY E$,E1$
580 GOSUB 660
590 PLAY F$,F1$
600 GOSUB 660
610 PLAY G$,G1$
620 GOSUB 660
630 PLAY H$,H1$
640 GOSUB 660
650 GOTO 490
660 FOR I=1 TO 15
670 X=RND(1)*240
680 Y=RND(1)*176
690 PUT SPRITE I,(X,Y),RND(1)*15,1
700 FOR J=1 TO 80
710 NEXT J
720 NEXT I
730 RETURN

```




Livres pour débuter

Voici une sélection de livres pour vous initier à ces étranges et merveilleuses machines que sont les micro-ordinateurs. Ils vous feront entrer dans le cercle des amateurs éclairés, puis des passionnés de micro-informatique.



Initiation aux micro-ordinateurs. Niveau 2

L'auteur analyse en détail et de façon très progressive toute la partie interne d'un micro-ordinateur. Il procède ensuite à l'étude de la logique extérieure à l'unité centrale, notamment les entrées/sorties, les différents types d'accès à la mémoire, les modes de transfert de données, etc. Les deux derniers chapitres sont consacrés, l'un à la programmation, l'autre à l'analyse des instructions. Enfin, l'ouvrage est complété par les codes de caractères et deux index, l'un en français, l'autre en anglais.

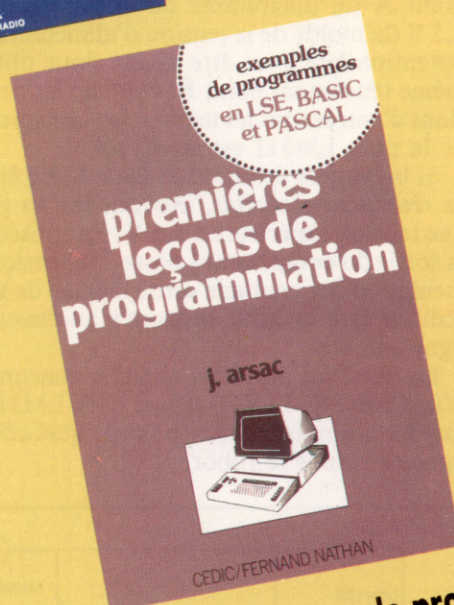
Par A. Osborne.
488 pages, format 16 x 24 cm.
S.E.C.F. Éditions Radio.



Initiation aux micro-ordinateurs. Niveau 1

Extrêmement progressif, cet ouvrage explique minutieusement la constitution d'un système à micro-ordinateur (unité centrale, mémoire et périphérique), son utilisation et les caractéristiques de ses différents composants. Ensuite sont traités les notions fondamentales (nombres, systèmes de numération, logiques, codes...), les langages de programmation et le fonctionnement du système.

Par A. Osborne.
304 pages, format 14 x 20 cm.
S.E.C.F. Éditions Radio.



Premières leçons de programmation

Une initialisation magistrale à la programmation, qui devrait permettre à chacun d'aborder cette science de façon rationnelle. De nombreux exemples de programmes sont analysés, et ils sont fournis chacun dans trois langages : LSE, BASIC et PASCAL.

Par Jacques Arsac.
230 pages, format 16 x 24 cm.
CEDIC-Nathan.



Mission

En munissant notre robot d'une série de résistances photosensibles, nous lui donnons une « vue » assez puissante pour suivre une ligne sombre.

L'unité qui permet au robot de suivre une ligne est composée de deux circuits identiques, chacun alimentant une ligne d'entrée différente sur le port utilisateur.

Chaque circuit est construit autour d'une puce LM311 qui compare les tensions présentes sur ses deux broches d'entrée. Si la tension de la broche d'entrée positive est plus haute que celle de l'entrée négative, la sortie passera de zéro volt à la tension d'alimentation. Du point de vue logique, la sortie provenant de cette puce est zéro lorsque les deux tensions d'entrée sont identiques et un lorsque la tension de la broche d'entrée positive est supérieure à celle de la broche d'entrée négative.

Deux résistances photosensibles (LDR) forment dans chaque circuit un diviseur de tension entre l'alimentation et la prise de terre. Si la lumière tombant sur les deux LDR est la même, leur résistance est aussi la même et la tension au point A du diagramme du circuit est environ 2,5 V (la moitié de la tension d'alimentation). Le potentiomètre peut être ajusté pour obtenir la même tension au point B. Comme les deux tensions d'entrée sont les mêmes, la sortie provenant de la puce LM311 est zéro.

Si la lumière éclairant la ligne LDR diminue, sa résistance s'accroît, provoquant au point A une tension supérieure à 2,5 V. En conséquence, la sortie du comparateur passe à la tension d'alimentation donnant une sortie logique de un ; cela indique que la LDR se trouve au-dessus de la ligne sombre.

La présence des deux circuits, chacun formé d'une paire de LDR et d'une puce LM311, permet à l'ordinateur de détecter de quel côté de la ligne se trouve le robot.

Étalonnage

Après avoir construit et installé la carte, nous devons l'étalonner. Dans des conditions lumineuses semblables à celles dans lesquelles vous désirez exécuter le programme de suivi de tracé, exécutez ce programme et ajustez les deux potentiomètres de la carte qui vient d'être installée. Tracez sur une feuille blanche un trait noir de

Programmes test LDR

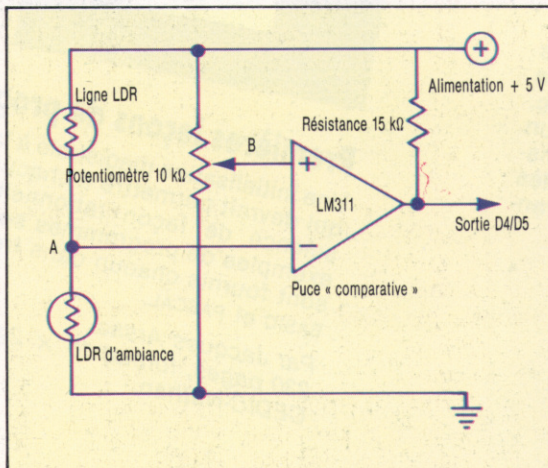
```
1000 REM **** LDR TEST BBC ****
1010 DDR=&FE62:DATREG=&FE60:DDR=15:REM LIGNES
      0-3 OUTPUT
1020 REPEAT
1030 contents=DATREG
1040 IF(contents AND 16)=0 THEN PRINT TAB(5)
      "LEFT":
1050 IF(contents AND 32)=0 THEN PRINT TAB(15)
      "RIGHT":
1060 PRINT
1070 UNTIL FALSE
```

```
1000 REM **** LDR TEST CBM ****
1010 DDR=&56579:DATREG=&56577:POKE DATREG,15
1020 CN=PEEK(DATREG)
1030 IF(CN AND 16)=0 THEN PRINTTAB(5)"GAUCHE":
1040 IF(CN AND 32)=0 THEN PRINTTAB(15)"DROITE":
1050 PRINT
1060 GOTO 1020
```

```
1000 REM **** LDR TEST SPECTRUM ****
1010 CLEAR 32499:LET ST=32500:GOSUB 3000
1020 LET NM=16:GOSUB 2000:IFUSR ST=0 THEN PRINT
      TAB 51"GAUCHE":
1030 LET NM=32:GOSUB 2000:IFUSR ST=0 THEN PRINT
      TAB 51"DROITE":
1040 PRINT
1050 GOTO 1020
1060 :
2000 REM **** EXECUTER ET ****
2010 POKE ST+1,IN 31
2020 POKE ST+3,NM:RETURN
2030 :
3000 REM **** CHARGEUR LANGAGE MACHINE ****
3010 FOR I=ST TO ST+8
3020 READ A:POKE I,A
3030 NEXT I
3040 DATA 62,0,14,0,161,6,0,79,201
3050 RETURN
```

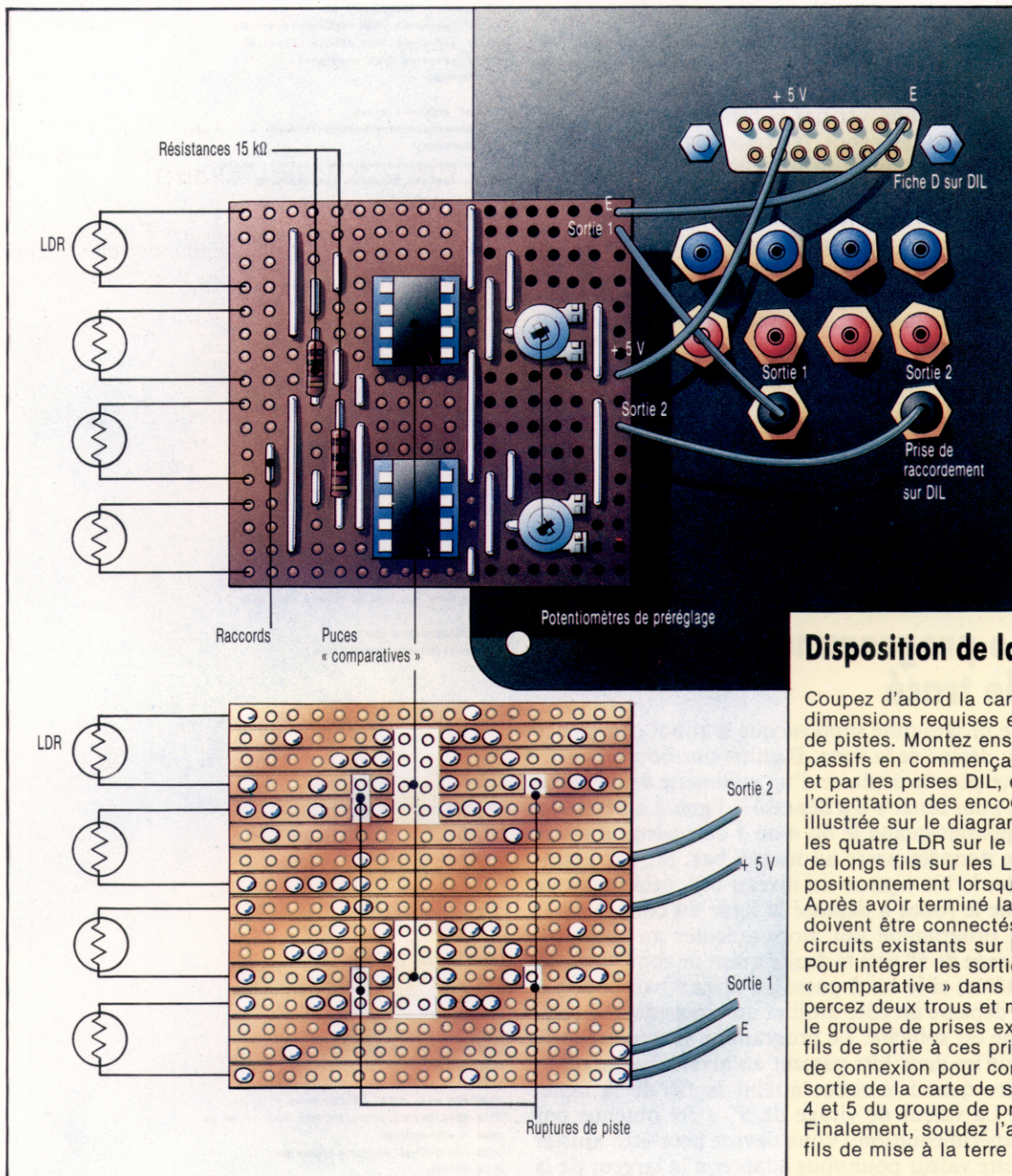
Dispositif optique

Le circuit optique est composé de deux circuits internes identiques utilisant une paire de résistances photosensibles. Une puce « comparative » est utilisée pour produire un signal numérique si des intensités lumineuses différentes sont détectées par les LDR. De cette manière, le robot peut détecter s'il se trouve ou non au-dessus de la ligne.



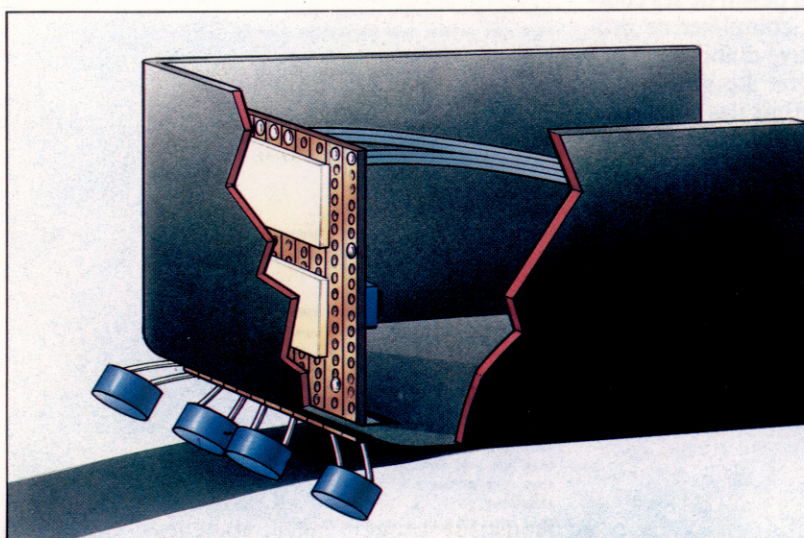
Liste des pièces

Nombre	Article
2	Puces « comparatives » LM311
2	Prises 8 broches DIL
2	Prises 2 mm
2	Résistance 15 kΩ, 0,4 W
2	Potentiomètres pré-réglés 10 kΩ horizontaux
4	LDR ORP12
1	Carte de montage 50 x 24
1 m	Câble-ruban 4 voies
1 m	Fil nu 20 swg



Disposition de la carte

Coupez d'abord la carte de montage aux dimensions requises et effectuez les coupures de pistes. Montez ensuite les composants passifs en commençant par les liaisons câblées et par les prises DIL, en veillant à ce que l'orientation des encoches soit identique à celle illustrée sur le diagramme. Finalement, soudez les quatre LDR sur le bord de la carte. Laissez de longs fils sur les LDR pour permettre leur positionnement lorsque la carte est installée. Après avoir terminé la carte, quatre conducteurs doivent être connectés pour la relier aux circuits existants sur le couvercle du robot. Pour intégrer les sorties de la puce « comparative » dans le système de connexion, percez deux trous et montez les prises derrière le groupe de prises existant. Soudez les deux fils de sortie à ces prises. Utilisez deux cordons de connexion pour connecter les prises de sortie de la carte de suivi de tracé aux lignes 4 et 5 du groupe de prises de connexion. Finalement, soudez l'alimentation de 5 V et les fils de mise à la terre à la fiche de type D.



Les yeux

La carte de suivi de tracé est montée sur la face avant interne du boîtier du robot de façon que les quatre LDR puissent sortir par une fente de 6×1 cm coupée à la base. Pliez délicatement les fils des LDR afin que les LDR soit disposés comme sur l'illustration. Les deux LDR du milieu doivent être placés aussi près que possible l'un de l'autre afin qu'ils se trouvent tous deux au-dessus de la ligne à suivre.

(Cl. Kevin Jones.)



2,5 cm de largeur. Ajustez à tour de rôle les deux potentiomètres à l'aide d'un tournevis ; l'ordinateur ne doit produire aucune sortie lorsque les deux LDR centrales se trouvent au-dessus de la ligne. Ensuite, vérifiez la fonction en déplaçant lentement le robot latéralement de gauche à droite pour lui faire franchir la ligne. Lorsque les quatre LDR se trouvent au-dessus du papier blanc, l'écran doit afficher LEFT RIGHT. Pendant ce déplacement au-dessus de la ligne, le message doit d'abord être LEFT, puis aucun message n'apparaît lorsque le robot est positionné directement au-dessus de la ligne. Lorsque le robot est déplacé un peu plus vers la droite, l'écran doit afficher RIGHT et finalement LEFT RIGHT lorsque les quatre LDR sont de nouveau au-dessus du papier blanc. Le programme suppose que la LDR de gauche est connectée à la ligne 4 et la LDR de droite à la ligne 5.

Les conditions de lumière doivent demeurer constantes pendant l'étalonnage. Il est préférable d'éclairer la zone de fonctionnement au projecteur.

Le programme de suivi de tracé

Le programme suppose que le robot commence au-dessus de la ligne. Il utilise une boucle répétitive pour le déplacer d'un millimètre vers l'avant et pour tester s'il a dépassé la ligne. La détection apparaît quand le bit 4 ou 5 du registre du port de données passe au niveau bas. Si, par exemple, le bit 4 passe au niveau bas, cela implique que le robot a dépassé la ligne du côté gauche ; le programme fait alors exécuter au robot un virage de 5° vers la droite avant de continuer. De même, si le bit 5 passe au niveau bas, le robot a dépassé du côté droit et un virage de 5° à gauche est sollicité. Le programme se termine lorsque les deux bits passent au niveau bas, c'est-à-dire quand le robot atteint la fin de la ligne.

La valeur de virage de 5° a été obtenue par expérimentation. Vous devrez peut-être ajuster cette valeur pour vous adapter à la largeur de la ligne que vous avez tracée et au dessin de ses courbes. Vous pouvez également compléter ce programme pour que le robot trouve d'abord la ligne et commence alors à la suivre. Le système de connexion vous permet de définir des combinaisons de capteurs connectés aux quatre lignes d'entrée du registre de données afin de pouvoir mettre au point vos propres applications.

Programme de suivi de tracés

BBC Micro

```
10 REM **** SUIVEUR DE TRACES BBC ****
20 :
30 PROCinitialise
40 PROCfollow line
50 END
60 :
70 DEF PROCfollow line
80 REPEAT
90 PROCmove(forwards,1)
100 PROCTest idr
110 UNTIL endflag=1
120 ENDPROC
130 :
140 DEF PROCTest idr
```

```
150 idrval=?DATREG AND 48
160 IF idrval=32 THEN PROCturn(right,5)
170 IF idrval=16 THEN PROCturn(left,5)
180 IF idrval=0 THEN endflag=1
190 ENDPROC
200 :
210 DEF PROCinitialise
220 DDR=&FEE2:DATREG=&FEE0:DDR=15
230 ?DATREG=1
240 forwards=4:backwards=2:left=6:right=0
250 pd ratio=3,34446:pa ratio=379/90
260 endflag=0
270 ENDPROC
280 :
290 DEF PROCmove(dir,distance)
300 ?DATREG=(?DATREG AND 1) OR dir
310 puses=pd ratio*ABS(distance)
320 FOR I=1 TO puses:PROCpuise:NEXT I
330 ENDPROC
340 :
350 DEF PROCturn(dir,angle)
360 ?DATREG=(?DATREG AND 1) OR dir
370 puses=pa ratio*angle
380 FOR I=1 TO puses:PROCpuise:NEXT I
390 ENDPROC
400 :
410 DEF PROCpuise
420 ?DATREG=(?DATREG OR 8)
430 ?DATREG=(?DATREG AND 247)
440 ENDPROC
```

Commodore 64

```
10 REM **** SUIVEUR DE TRACES CBM ****
20 :
30 GOSUB2000:REM INIT
40 GOSUB1000:REM SUIVRE TRACES
50 END
60 :
1000 REM **** SUIVRE TRACES ****
1010 DR=FW:DS=1:GOSUB2500:REM DEPLACEMENT
1020 GOSUB1500:REM TEST LDRS
1030 IF EF 1 THEN 1000
1040 RETURN
1050 :
1500 REM **** TEST LDRS ****
1510 LV=PEEK(DATREG) AND 48
1520 IF LV=32 THEN DR=RT:DS=5:
GOSUB3000:REM TOURNER
1530 IF LV=16 THEN DR=LF:DS=5:
GOSUB3000:REM TOURNER
1540 IF LV=0 THEN EF=1
1550 RETURN
1560 :
2000 REM **** INIT ****
2010 DDR=56579:DATREG=56577:POKEDDR,15
2020 POKEDATREG,1
2030 FW=4:BW=2:LF=6:RT=0
2040 PD=3,34446:PA=379/90
2050 EF=0:REM FIN DRAPEAU
2060 RETURN
2070 :
2500 REM **** MOVE (DR,DS) ****
2510 POKE DATREG,(PEEK(DATREG) AND 1) OR DR
2520 PL=PD*ABS(DS)
2530 FOR I=1 TO PL:GOSUB 3500:NEXT I
2550 RETURN
2560 :
3000 REM **** TOURNER (DR,DS) ****
3010 POKE DATREG,(PEEK(DATREG) AND 1) OR DR
3020 PL=PA*ABS(DS)
3030 FOR I=1 TO PL:GOSUB 3500:NEXT I
3040 RETURN
3050 :
3500 REM **** IMPULSION ****
```

Sinclair Spectrum

```
10 REM **** SUIVEUR DE TRACES SPECTRUM ****
12 REM SUPPRIMER LIGNES 2010,2020,2510,3010 DE LA
VERSION CBM
13 REM ET REMPLACER PAR
15 CLEAR 32499:ST=32500:GOSUB 4500
1510 LET NM=48:GOSUB 4000:LET LV=USR ST
3510 OUT 31,DR+9
3520 OUT 31,DR+1
4000 REM **** EXECUTER ET ****
4010 POKE ST+1,IN 31
4020 POKE ST+3,NM:RETURN
4500 REM **** CHARGEUR LANGAGE MACHINE ****
4510 FOR I=ST TO ST+8
4520 READ A:POKE I,A
4530 NEXT I
4540 DATA 62,0,14,0,161,6,0,79,201
4550 RETURN
```