

LOISIRS TECHNIQUES D'AUJOURD'HUI

hors série

Led

MICRO

PROGRAMMATION

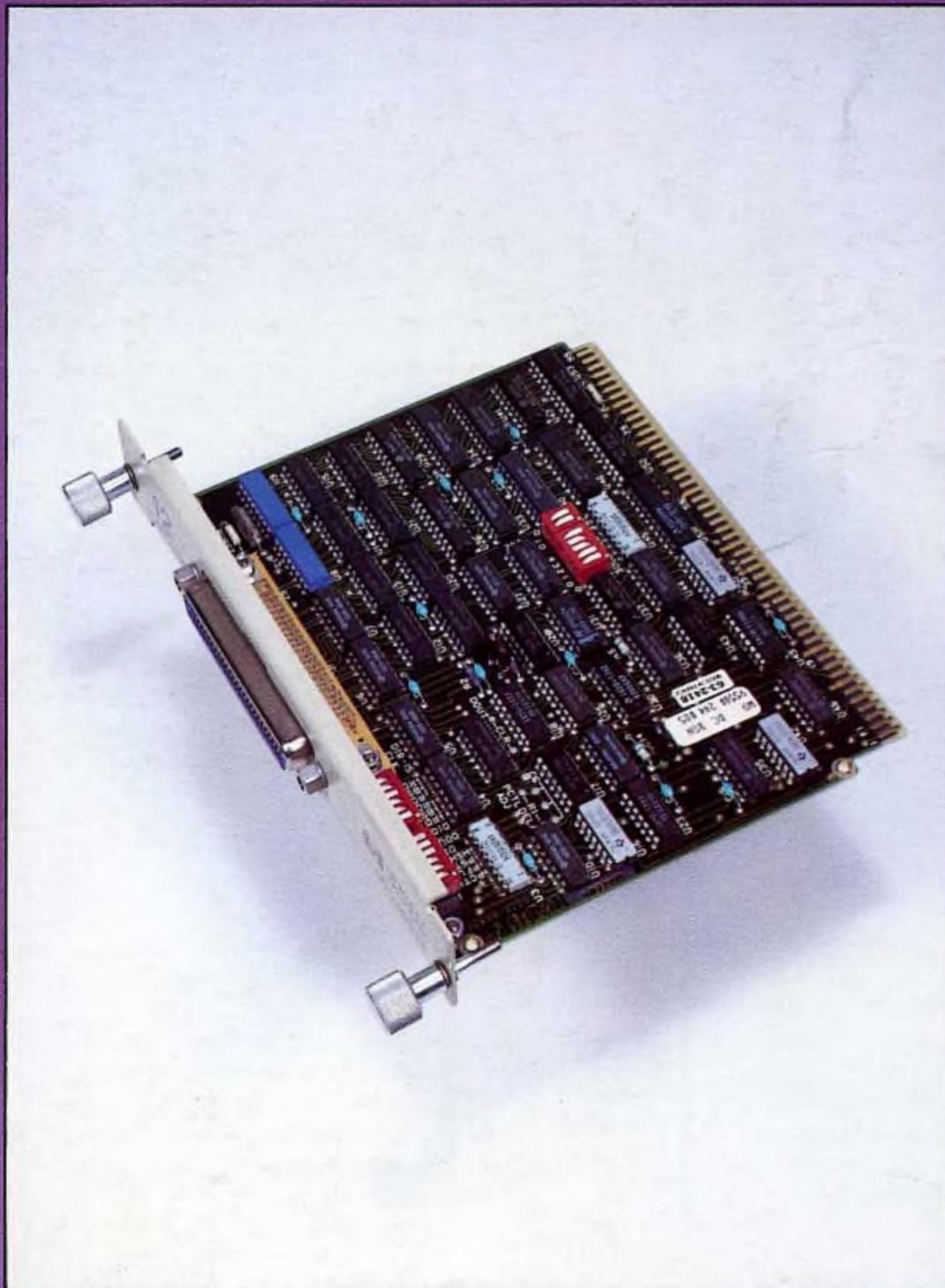
COURS 2^{ème} CYCLE

COURS
N°39
Suite
2^e cycle
N°19

COURS DE
PASCAL
la récursivité

COURS DE
PROGRAM-
MATION
APPROFONDIE :
mouvements
du curseur

dBase III



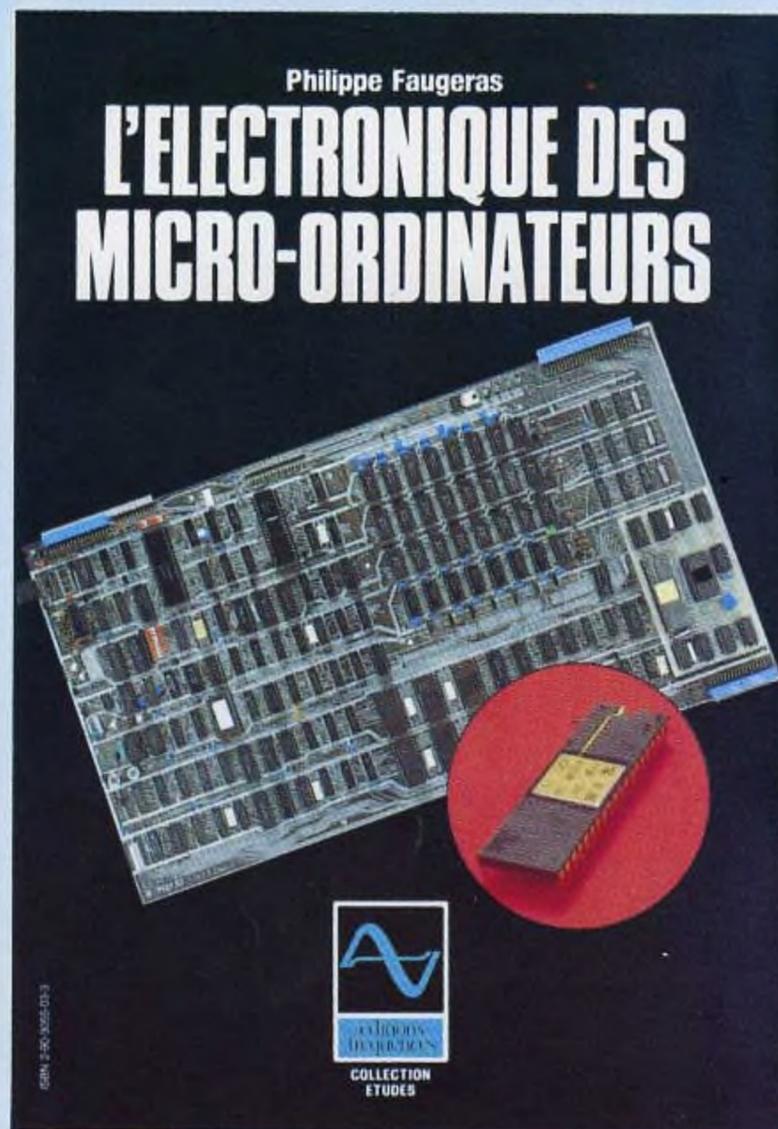
ISSN 0757-6889

M 1988 - 39 - 18,00 F



VOYAGE AU COEUR DES MICRO-ORDINATEURS

dans la
COLLECTION
«ETUDES»
aux
éditions
fréquences



une véritable schémathèque

- 128 pages
 - 101 schémas
 - 34 tableaux
- Prix : 150 F

Que ce soit pour concevoir des interfaces ou optimiser un programme (utilisation des périphériques, encombrement mémoire...) «un micro-informaticien performant» doit posséder une bonne connaissance de son matériel.

Ce livre s'adresse donc à tous les électroniciens qui désirent découvrir les différents

composants constituant un micro-ordinateur. Articulé autour du microprocesseur Z80, cet ouvrage contient de nombreux schémas (plan mémoire, interfaces série et parallèle, interface clavier, interface vidéo, CAN, CNA...) qui pourraient être le thème... de nouvelles extensions.

En vente chez votre libraire et aux Editions Fréquences

BON DE COMMANDE

Je désire recevoir l'ouvrage **L'électronique des micro-ordinateurs** au prix de **160 F** (150 F + 10 F de port).

Nom

Adresse

.....

A adresser aux **EDITIONS FREQUENCES 1 boulevard Ney, 75018 Paris**

Règlement ci-joint :

Par chèque bancaire par chèque postal par mandat

Philippe Faugeras, Docteur-ingénieur en électronique a acquis son expérience dans de grandes entreprises françaises où pendant cinq ans, il a travaillé sur des systèmes d'automatismes à base de microprocesseurs. Philippe Faugeras est responsable de la rubrique «Raconte-moi la micro-informatique» dans la revue LED.

LOM...BIQUES D'AUJOURD'HUI

hors série

LED

MICRO

PROGRAMMATION COURS 2^e CYCLE

Société éditrice :
Editions Fréquences
 Siège social :
 1, bd Ney, 75018 Paris
 Tél. : (1) 46.07.01.97 +
 SA au capital de 1 000 000 F
 Président-Directeur Général :
 Edouard Pastor

LED MICRO
 (cours 2^e cycle)
 Mensuel : 18 F
 Commission paritaire : 64949
 Directeur de la Publication :
 Edouard Pastor
 Tous droits de reproduction réservés
 textes et photos pour tous pays
 LED MICRO est
 une marque déposée ISSN 0757-6888

**Services Rédaction-Publicité-
 Abonnements :**
 1 bd Ney, 75018 Paris
 Tél. (1) 46.07.01.97
 Lignes groupées

Comité de rédaction :
 Dominique Chastagnier
 Jean-François Coblentz
 Charles-Henry Delaleu
 Patrick Gueneau

Secrétaire de Rédaction
 Chantal Cauchois

Publicité A la revue
 Tél. : 46.07.01.97
 Secrétaire responsable
 Annie Perbal

Abonnements
 10 numéros par an
 France : 160 F
 Etranger : 240 F

Réalisation
 Composition-Photogravure
 Edi'Systèmes
 Impression
 Berger-Levrault - Nancy

Les Editions Fréquences éditent
 La Nouvelle Revue du Son
 Son Vidéo Magazine, L'Audiophile
 Led Micro, Zéro-VU magazine
 Music Vidéo Systèmes



AVRIL 87

COURS DE PASCAL de la page 5 à la page 16

- Le retour arrière, de l'anglais
 backtracking p. 6
- Application : les dames en prises p. 7
- Un problème graphique p. 9
- Un exemple mathématique p. 13
- Conclusion p. 14
- Des exercices p. 14
 - Les tours de Hanoi
 - Calculer la limite de l'expression
 - Un programme de tri récursif

**Dominique Chastagnier
 Jean-François Coblentz
 Patrick Gueneau**

COURS DE PROGRAMMATION APPROFONDIE de la page 17 à la page 21

- Mouvements du curseur (suite) p. 18
 - Remarque sur gauche et droite
- Conclusion p. 21

**Dominique Chastagnier
 Jean-François Coblentz
 Patrick Gueneau**

DIALOGUE AVEC NOS LECTEURS de la page 22 à la page 29

- Valeurs propres et
 vecteurs propres p. 22
- Mise en œuvre p. 22
 - Quelques remarques préliminaires
 - Commentaires de programmation
- Benchmarks graphiques p. 27
 - Epicycloïdes
 - Hypocycloïdes
 - Cycloïdes
- Les Fractals p. 28

C'EST ARRIVÉ DEMAIN de la page 30 à la page 32

ET LA TECHNIQUE page 36

dBASE III de la page 37 à la page 49

- Rappel des principales caractéristiques de dBase III p. 38
- Un système de gestion de
 bases de données p. 38
- Les fichiers dBase III p. 39
- La commande Assist p. 39
- Les principales commandes
 de dBase III p. 39
- Création d'un fichier p. 40
- Opérateurs relationnels
 et logiques p. 40
 - Opérateurs relationnels
 - Opérateurs logiques
 - La commande FOR
- Le tri p. 40
- Indexation p. 41
- Recherche rapide d'une
 information (FIND) p. 41
- Fonctions sur les chaînes
 de caractères p. 41
- Réalisation d'un programme p. 41
 - Programme principal
 - Bibliothèque des procédures
- Quelques explications sur
 les ordres utilisés p. 49

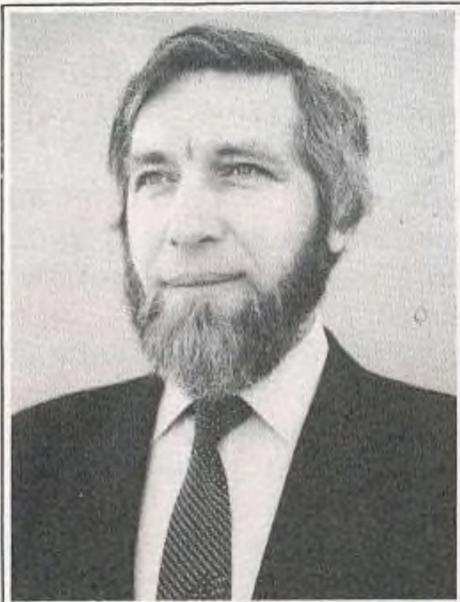
Charles-Henry Delaleu

NOTRE COUVERTURE : Une carte d'entrées-sorties 16 bits parallèles possédant toutes les caractéristiques techniques requises pour des applications industrielles. Hormis les lignes de data, elle dispose de toutes les lignes d'état et de status.

Electronique digitale ?

Notre temps aura témoigné d'une nouvelle technique, une autre façon de communiquer avec l'électronique digitale.

Philippe Duquesne, professeur chargé de cours au CNAM a su dans cet ouvrage en expliquer clairement les fondements.



Philippe Duquesne, ingénieur électronicien (I.S.E.N.) est chargé du cours de microprocesseurs au C.N.A.M. de Paris. Depuis plus de dix ans, il a pris goût à l'enseignement et il est l'auteur d'un ouvrage didactique sur l'électronique digitale et notamment d'un cours pratique de microprocesseurs. Fervent pratiquant du « dialogue » école/industrie, après avoir exercé les fonctions de chef de département électronique chez Burroughs, second constructeur mondial en informatique, il est actuellement chef du service Etudes Electroniques au sein de la direction technique chez Messier Hispano Bugatti (groupe SNECMA) avec, pour principal objectif l'introduction des microprocesseurs dans les trains d'atterrissage.



En vente chez votre libraire et aux Editions Fréquences

Bon de commande

Je désire recevoir le livre : INITIATION A L'ELECTRONIQUE DIGITALE au prix de 105 F (95 F + 10 F de port).

Adresser ce bon aux EDITIONS FREQUENCES 1, bd Ney, 75018 PARIS

Nom Prénom

Adresse

Code postal

Règlement effectué : par C.C.P.

par chèque bancaire

par mandat

COURS DE PASCAL

Dominique Chastagnier
Jean-François Coblenz
Patrick Gueneau

Le mois dernier, nous avons fait un petit tour descriptif de la récursivité. Nous avons montré comment fonctionnait le principe, et surtout que, si cette technique est parfois irremplaçable, elle n'est pas toujours une panacée. Il nous reste à insister sur ces deux points, fondamentaux lorsque la récursivité est évoquée. Ce mois-ci, nous allons donner des exemples intéressants d'applications de la récursivité, à travers des techniques qui, pour certaines, sont à la pointe en matière de programmation.

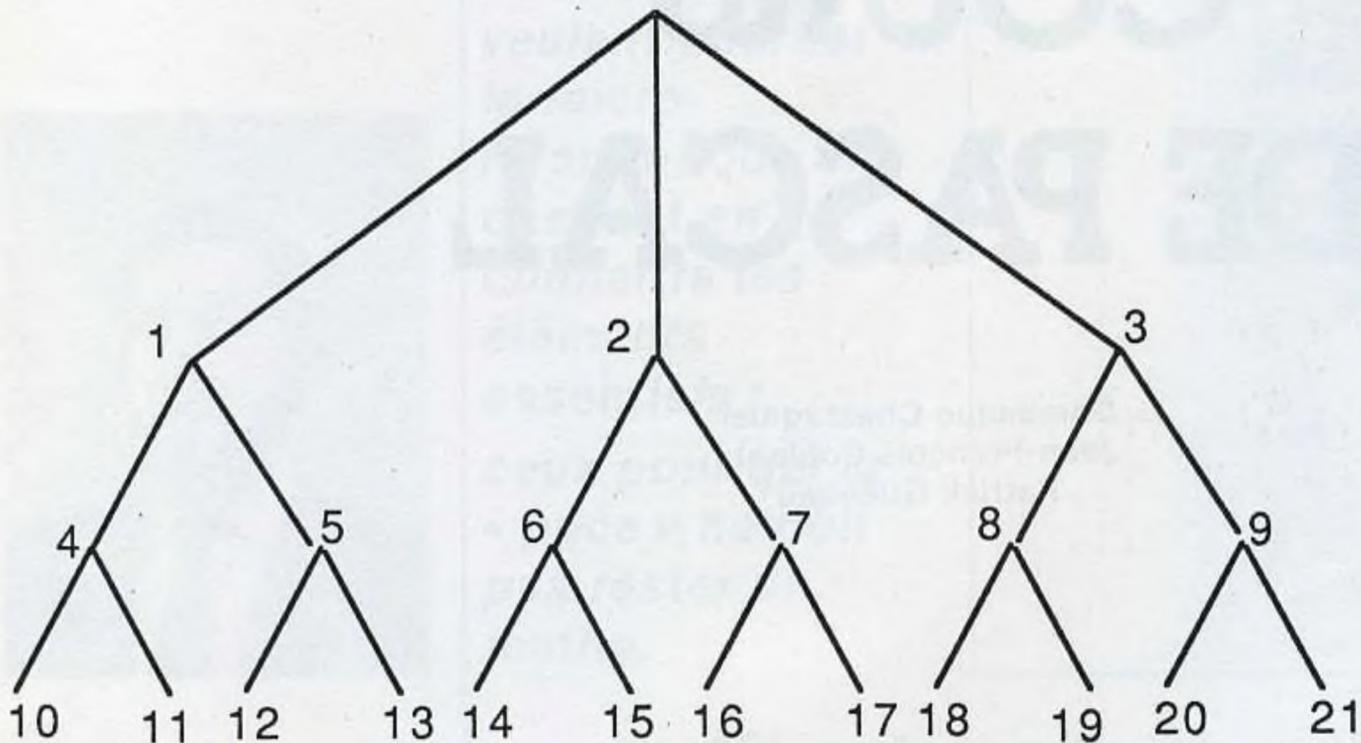
COURS N° 10

PLAN DU COURS

1. Le retour arrière, de l'anglais backtracking
2. Application : les dames en prises
3. Un problème graphique
4. Un exemple mathématique
5. Conclusion
6. Des exercices
 - Les tours de Hanoi
 - Calculer la limite de l'expression
 - Un programme de tri récursif

1. LE RETOUR ARRIERE, DE L'ANGLAIS BACKTRACKING

Voici le problème. Au milieu d'une forêt de possibilités, votre programme doit en choisir une, et pousser au bout ce choix, en en faisant d'autres par la suite. Si le choix en question est mauvais (on ne parvient pas à la solution), il faut revenir en arrière, pour essayer une autre solution. Comment faire ? La récursivité, avec son stockage automatique des paramètres semble bien adaptée.



Sur le graphique représentant les choix successifs d'un programme, supposons que la réussite soit l'obtention du chiffre 17 et que le critère de choix soit, par ordre de priorité obligatoire (ce qui veut dire qu'à chaque fois, on essaie la première instruction, puis si elle n'est pas possible, la seconde... et on recommence ainsi à chaque nouvelle étape) :

- 1 : descendre vers le nœud gauche
- 2 : descendre vers le nœud droit
- 3 : remonter.

Dans le cas du graphe précédent, on aurait la suite de nœuds :

R - 1 - 4 - 10 - 4 - 11 - 4 - 1 - 5 - 12 - 5 - 13 - 5 - 1 - R - 2 - 6 - 14 - 6 - 15 - 6 - 2 - 7 - 16 - 7 - 17 - FIN

Il est évident que, au lieu de descendre l'arbre jusqu'au bas de chaque branche, nous aurions pu descendre d'un niveau seulement, et explorer les nœuds de ce niveau : il se serait alors produit la suite de nœuds :

R - 1 - R - 2 - R - 3 - R - 1 - 4 - 1 - 5 - 1 - R - 2 - 6 - 2 - 7 - 2 - R - 3 - 8 - 3 - 9 - 3 - R - 1 - 4 - 10 - 4 - 11 - 4 - 1 - 5 - 12 - 5 - 13 - 5 - 1 - R - 2 - 6 - 14 - 6 - 15 - 6 - 2 - 7 - 16 - 7 - 17 - FIN

Il semble ici que le procédé est nettement plus long. Si maintenant le critère de succès est de trouver le 3, avec la première méthode, on aurait :

R - 1 - 4 - 10 - 4 - 11 - 4 - 1 - 5 - 12 - 5 - 13 - 5 - 1 - R - 2 - 6 - 14 - 6 - 15 - 6 - 2 - 7 - 16 - 7 - 17 - 7 - 2 - R - 3 - FIN

Avec la seconde :

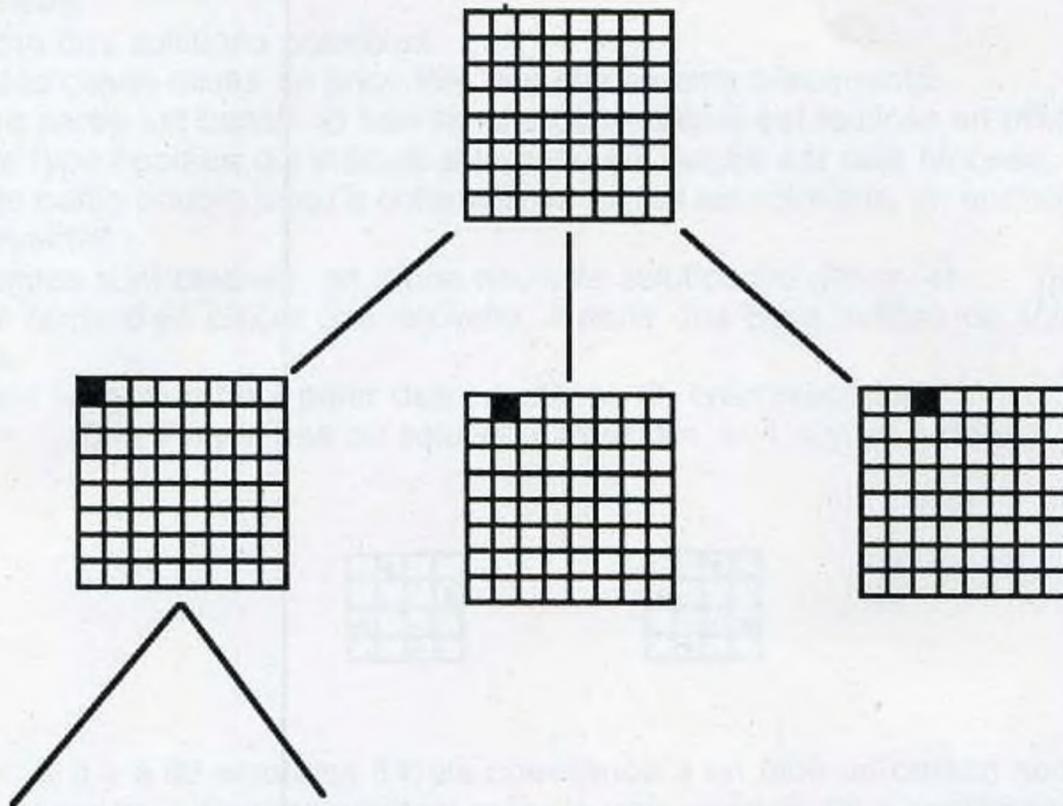
R' - 1 - R - 2 - R - 3 - FIN

La preuve est donc faite que chacune des deux méthodes a ses avantages et ses inconvénients. La première est appelée «parcours en profondeur d'abord», la seconde «en largeur d'abord».

Ceci peut vous sembler un peu artificiel. Donc, nous allons proposer un jeu, pour clarifier les idées. Ce jeu consiste à faire déplacer un cheval sur un échiquier de telle

sorte que toutes les cases soient utilisées, une fois seulement. Il existe un algorithme mais nous ne le considérerons pas, puisque vous ne le connaissez peut-être pas. Comment faire ?

Nous pouvons ramener le problème à la recherche du coup suivant ou, s'il n'existe pas, à l'étude permettant de savoir si on a gagné ou s'il faut recommencer, car le jeu est bloqué. Un exemple graphique d'abord :



On le voit, la recherche peut être vraiment longue et délicate. Mais, heureusement, l'homme a créé l'ordinateur, et nous allons donc nous en servir !!

2. APPLICATION : LES DAMES EN PRISES

Dans le même ordre d'idées, voici un programme qui exécute la recherche complète de toutes les solutions, par une méthode récursive du problème suivant : trouver toutes les positions possibles de n reines sur un échiquier $n \times n$, de telle sorte que les reines ne soient en position de prises sur aucune autre. Cela signifie qu'il positionne une pièce, puis une autre sur la première case libre qu'il trouve, et ainsi de suite. S'il se trouve bloqué, il revient un cran en arrière, et tente de trouver une autre solution. Si c'est impossible, il revient encore un cran en arrière, et ainsi de suite. Une alternative est possible, qui arrête le programme dès qu'une solution est trouvée. Ici, nous choisirons de les trouver toutes. Bien sûr, il ne s'agit pas de les imprimer mais de les compter. Remarquons que lorsqu'une dame est sur une case, sa ligne et sa colonne ne peuvent plus servir. Nous utilisons cette propriété pour débiter sur la case (1,1) la recherche et augmenter l'indice de ligne et celui de colonne à chaque fois. Voici le programme, et les commentaires.

```

program Reines;
const
  dim = 4;
type
  tab_sol = array[1..dim] of integer;
var
  tab : tab_sol;
  i, j, nb_sol, k : integer;
  solution : boolean;

```

```
function bloquee (l, c : integer) : boolean;
var
  ind_l, i : integer;
begin
  bloquee := false;
  for i := 1 to c - 1 do
    begin
      ind_l := abs(tab[i] - l);
      if ind_l * (ind_l - c + i) = 0 then
        bloquee := true;
      end;
    end;
end;

procedure trouve_sol (l, c : integer);
var
  i : integer;
  ok : boolean;
begin
  ok := false;
  if c > dim then
    begin
      nb_sol := nb_sol + 1;
      writeln(nb_sol);
    end
  else
    for i := l to dim do
      if not bloquee(i, c) then
        begin
          tab[c] := i;
          trouve_sol(1, c + 1);
          ok := true;
        end;
    end;

    if ((not ok) and (c > 1)) then
      begin
        i := tab[c - 1];
        tab[c - 1] := 0;
        trouve_sol(i + 1, c - 1);
      end;
  end;

procedure init (var i, j, nb_sol : integer;
  var tab : tab_sol);
var
  k : integer;
begin
  i := 1;
  j := 1;
  nb_sol := 0;
  for k := 1 to dim do
    tab[k] := 0;
  end;
```

```

begin
  init(i, j, nb_sol, tab);
  trouve_sol(i, j);
end.

```

Le programme comporte trois parties distinctes :

- Programme principal qui se contente de lancer la recherche, après une initialisation des données.
- Recherche des solutions possibles.
- Etudes des cases mises en prise lors des placements précédents.

La première partie est banale et très simple. La troisième est réalisée en utilisant une fonction de type booléen qui indique si la case envisagée est déjà bloquée.

La seconde partie boucle jusqu'à obtention de toutes les solutions, en enchaînant les actions suivantes :

- si *dim* dames sont placées, on a une nouvelle solution (ici $dim = 4$),
- sinon, on tente d'en placer une nouvelle, à partir des bons indices de ligne et de colonnes,
- on reprend la recherche à partir des conditions du cran précédent.

Pour $dim = 2, 3$, il n'y a pas de solutions. Pour $dim = 4$, il y en a deux :

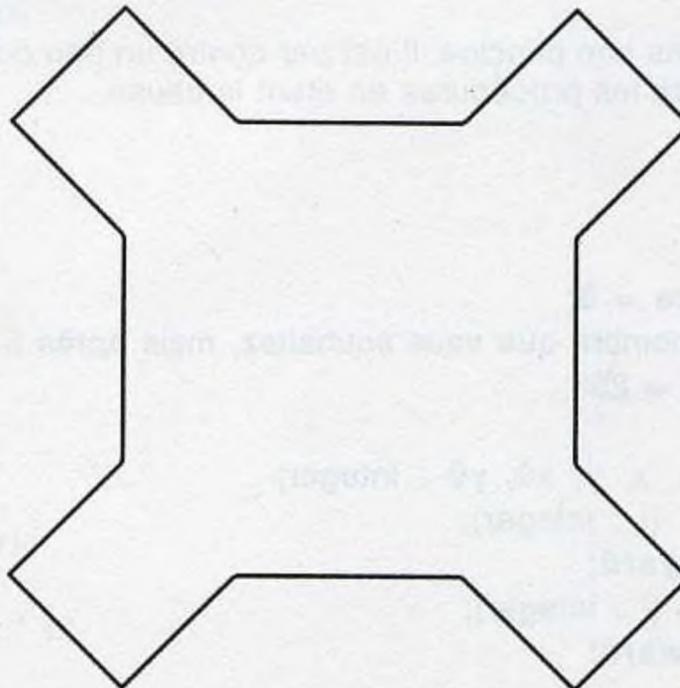
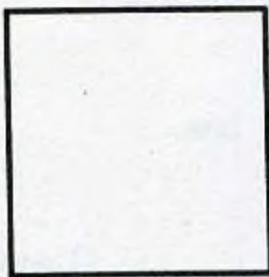


Pour $dim = 8$, il y a 92 solutions !! Cela commence à en faire un certain nombre, et notre programme travaille alors pendant près de trois minutes. En comparaison, il faut 19 secondes pour $dim = 4$.

3. UN PROBLEME GRAPHIQUE

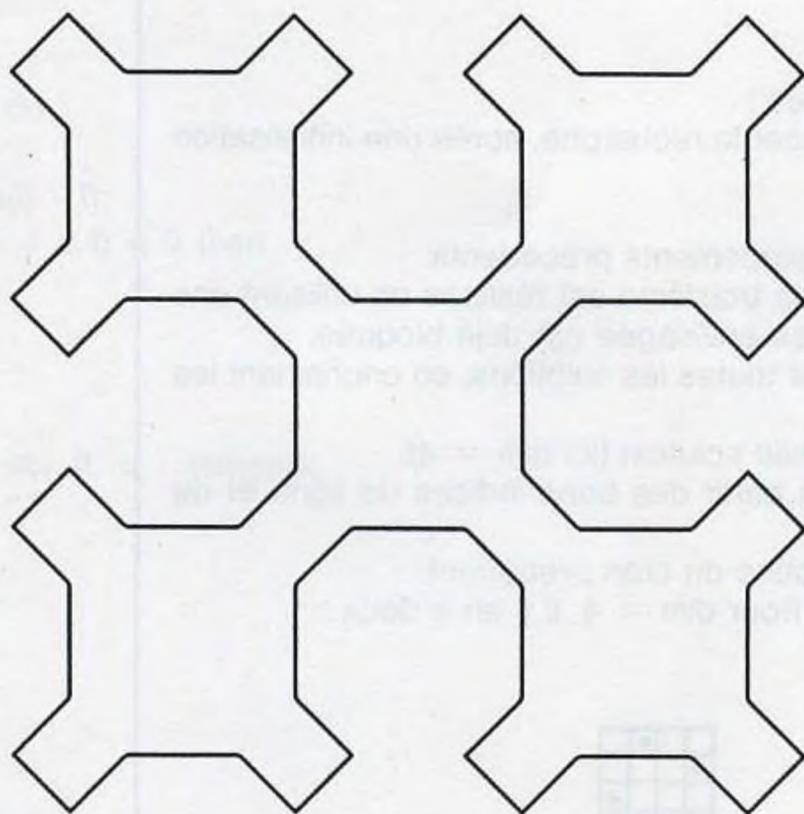
Un problème complètement différent, utilisant les possibilités graphiques des ordinateurs récents. Il est évident que ceux qui ne pourront pas le faire tourner trouveront cela frustrant, mais nous avons préféré le proposer car il illustre avant tout le propos de ce cours. Il est cependant possible d'en faire une version réduite en basse résolution. Tout algorithme non récursif pour ce tracé est très long, très complexe. Le programme trace des courbes dites de Sierpinsky. Ce sont des courbes récursives, c'est-à-dire que chaque élément de la courbe dépend des éléments précédents.

Le principe est le suivant : considérons l'élément graphique de base :



On peut l'utiliser pour former un dessin :

Par une récurrence que nous allons décrire, nous pouvons recommencer cette courbe sur elle-même, et l'on obtient :



On voit que chaque élément est celui de base, diminué de moitié, et accroché à l'élément central. On peut à partir de là proposer une définition récursive.

Notons que nous utilisons dans le programme la commande **Forward** qui permet de ne déclarer une procédure qu'après son appel. C'est obligatoire ici, car les procédures A, B, C et D s'appellent les unes les autres, donc il y aurait forcément un problème sans cette possibilité. La commande FORWARD indique simplement au système que la procédure existe, est bien déclarée, mais est décrite plus loin. Cette commande a déjà été décrite, mais vous en voyez ici la première utilisation. Voici le programme :

Ce programme est écrit sur un Macintosh. Il utilise les procédures graphiques de cet ordinateur, à savoir :

- SHOWDRAWING : permet de visualiser la fenêtre graphique,
- MOVETO(x,y) : aller au point x,y, sans rien tracer,
- LINETO(x,y) : aller au point x,y, en traçant une ligne depuis le point où l'on se trouve.

Ce programme est simple dans son principe. Il est par contre un peu complexe dans son écriture, l'imbrication entre les procédures en étant la cause.

```

program Sierpinski_LED;
  const
    ordre = 3;
    (* a vous de mettre le nombre que vous souhaitez, mais après 5 c est long *)
    h0 = 256;
  var
    i, h, x, y, x0, y0 : integer;
  procedure A (i : integer);
    forward;
  procedure B (i : integer);
    forward;

```

```
procedure C (i : integer);
  forward;
procedure D (i : integer);
  forward;
procedure A;
begin
  if i > 0 then
    begin
      A(i - 1);
      x := x + h;
      y := y - h;
      lineto(x, y);
      B(i - 1);
      x := x + 2 * h;
      lineto(x, y);
      D(i - 1);
      x := x + h;
      y := y + h;
      lineto(x, y);
      A(i - 1)
    end
  end;
end;

procedure B;
begin
  if i > 0 then
    begin
      B(i - 1);
      x := x - h;
      y := y - h;
      lineto(x, y);
      C(i - 1);
      y := y - 2 * h;
      lineto(x, y);
      A(i - 1);
      x := x + h;
      y := y - h;
      lineto(x, y);
      B(i - 1)
    end
  end;
end;

procedure C;
begin
  if i > 0 then
    begin
      C(i - 1);
      x := x - h;
      y := y + h;
      lineto(x, y);
      D(i - 1);
      x := x - 2 * h;
```

```

lineto(x, y);
B(i - 1);
x := x - h;
y := y - h;
lineto(x, y);
C(i - 1)
end
end;

procedure D;
begin
  if i > 0 then
    begin
      D(i - 1);
      x := x + h;
      y := y + h;
      lineto(x, y);
      A(i - 1);
      y := y + 2 * h;
      lineto(x, y);
      C(i - 1);
      x := x - h;
      y := y + h;
      lineto(x, y);
      D(i - 1)
    end
  end;
end;

begin
  ShowDrawing;
  i := 0;
  h := h0 div 4;
  x0 := 2 * h;
  y0 := 3 * h;
  repeat
    begin
      i := i + 1;
      x0 := x0 - h;
      h := h div 2;
      y0 := y0 + h;
      x := x0;
      y := y0;
      moveto(x, y);
      A(i);
      x := x + h;
      y := y - h;
      lineto(x, y);
      B(i);
      x := x - h;
      y := y - h;
      lineto(x, y);
      C(i);
    end
  until i = 4;
end

```

```

        x := x - h;
        y := y + h;
        lineto(x, y);
        D(i);
        x := x + h;
        y := y + h;
        lineto(x, y);
    end;
until i = ordre
end.

```

4. UN EXEMPLE MATHEMATIQUE

Il en fallait un, ne nous le reprochez pas. Nous allons donner un exemple court : la génération de nombres premiers. Un nombre est premier s'il ne peut être divisé que par 1 et par lui-même. A partir de cette définition, il vient le programme suivant :

```

program Premier_recuratif;
var
  p : integer;

function teste (n, p : integer) : boolean;
begin
  if sqr(n) > p then
    teste := true
  else
    begin
      if p mod n = 0 then
        teste := false
      else if n > 2 then
        teste := teste(n + 2, p)
      else
        teste := teste(n + 1, p);
      end;
    end;
end;

begin
  repeat
    write('introduire un entier : ');
    readln(p);
    if p <> 0 then
      begin
        if teste(2, p) then
          writeln('ce nombre est premier')
        else
          writeln(' ce nombre n est pas premier ');
          writeln;
          writeln;
        end;
      until p = 0;
    end.

```

Ce programme calcule donc si un nombre est premier ou non, en le divisant par 2 et par tous les nombres impairs inférieurs à sa racine carrée. Ce critère est suffisant puisque

si un nombre plus grand que sa racine carrée le divise, un nombre plus petit le divisera aussi. En effet, soit p le nombre. S'il existe b , plus grand que \sqrt{p} , qui divise p , alors il existe a tel que :

$$a \cdot b = p.$$

Si a est plus grand que \sqrt{p} , $a \cdot b$ est forcément plus grand que p , donc a est plus petit que p et la recherche l'aura trouvée avant de trouver b . Pour un nombre p , on regarde s'il est divisible par 2, puis par 3, puis par la suite des nombres impairs. Les nombres pairs sont éliminés par le fait que si un nombre pair divise p , 2 divise p . Il est possible d'en faire autant avec les multiples de 3, 5, ... mais la relation donnant les nombres à tester devient plus compliquée.

5. CONCLUSION

On le voit, la notion de récursivité est d'une grande importance, tant pratique que théorique. Elle est à la base de nombreux algorithmes très importants, tels celui dit de «backtracking» ou de retour arrière. C'est l'algorithme utilisé pour le programme des dames sur l'échiquier. Il consiste à avancer dans la solution tant que c'est possible, puis, en cas de blocage, de revenir en arrière d'un cran, et de recommencer la recherche depuis ce point, dans une autre direction. Cet algorithme est particulièrement utilisé dans les applications actuelles en intelligence artificielle. Il est, en règle générale, très commode pour la rédaction de programmes clairs. Par contre, il devient plus délicat de suivre le déroulement précis du programme. Tout comme la récursivité, il est surtout utile lorsque la solution d'un problème n'est pas directement accessible par un algorithme. Ce qui signifie que la récursivité est particulièrement pratique lorsqu'on n'a aucune idée de ce que l'on cherche.

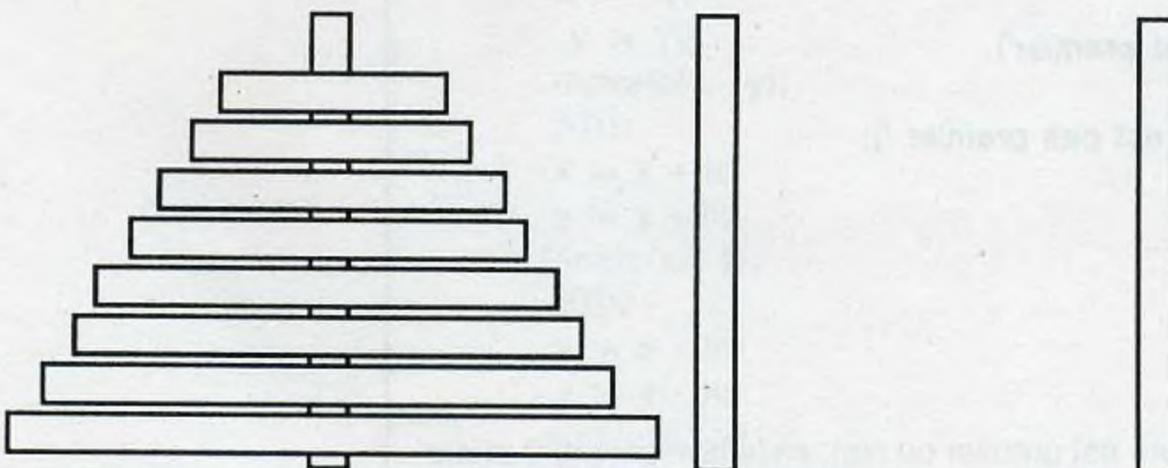
Le mois prochain, nous aborderons une autre notion capitale en Pascal, les pointeurs, et l'allocation dynamique de mémoire. Ces éléments forment la pierre angulaire des langages évolués et leur implantation première a été faite pour Pascal, puis sur d'autres, mais avec plus ou moins de succès quant à la facilité d'emploi ou la gestion de la mémoire.

6. DES EXERCICES

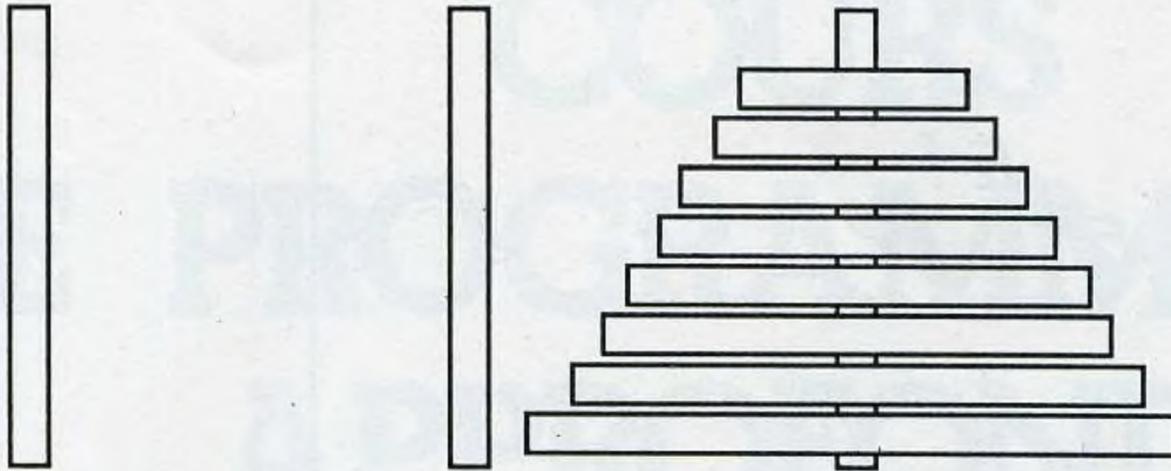
1. Les tours de Hanoi

Voilà un problème passionnant, dérivé d'une très saine occupation des moines d'un monastère vietnamien. Vous disposez de trois poteaux et de n disques percés, tous de taille différente. Vous pouvez placer un disque à chaque fois et ne le reposer que sur un disque plus grand. Si vous partez avec tous les disques sur un poteau d'un côté, il faut les amener tous sur le poteau opposé.

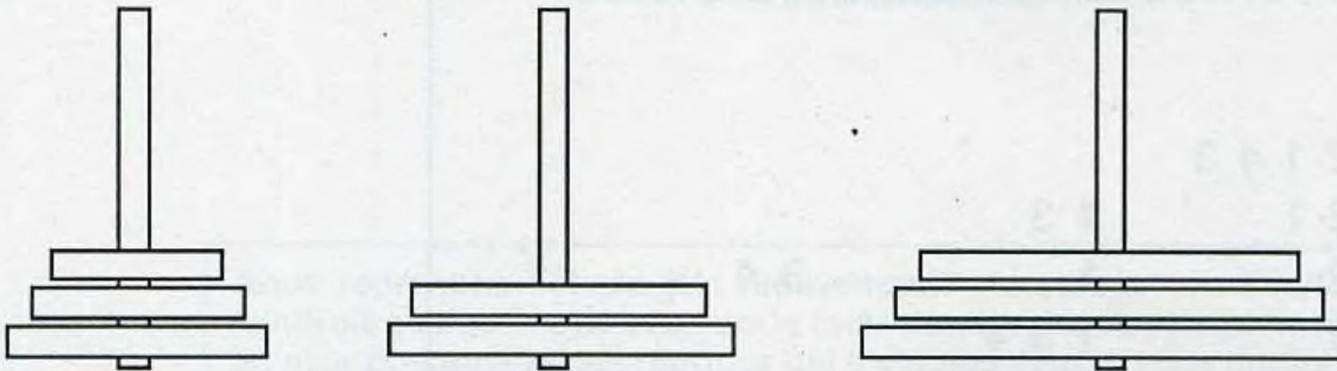
La position de départ :



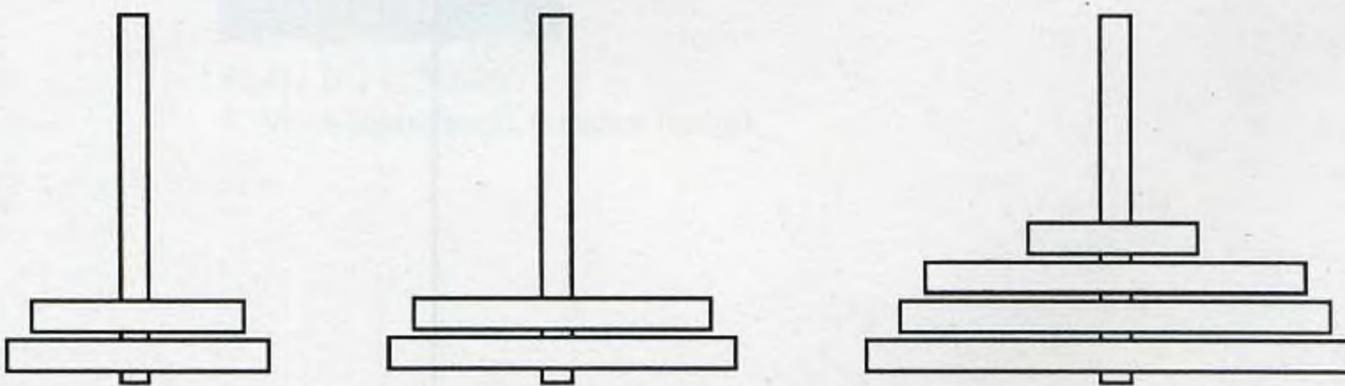
et la position d'arrivée :



Un mouvement possible : de la position



à la position :



L'exercice est de programmer cet amusant jeu à l'aide d'un algorithme récursif, pour un nombre de disque qui sera mis en constante dans le programme, ou demandé au début de l'exécution.

Un détail : les moines déplacent 64 disques, et la légende veut que lorsque ce travail sera terminé, le monde verra sa fin. Il faut dire que dans le cas de 64 disques, le temps (manuel !) est tout à fait considérable. Avec 8 disques, un Pascal compilé demande une bonne demi-douzaine de minutes.

2. Calculer la limite de l'expression :

$$\sqrt{2} \sqrt{2} \sqrt{2} \sqrt{2} \sqrt{2} \sqrt{2} \dots$$

autrement dit, racine de deux, à la puissance racine de deux,...

3. Un programme de tri récursif

Une méthode de tri consiste à couper la liste en deux, au centre, puis à refaire de même sur chaque sous-liste, jusqu'à ce qu'il n'y ait qu'un ou deux éléments dans la liste courante que l'on ordonne, puis on réunit ces listes deux par deux, et on les ordonne, etc. Faire le programme correspondant. Un exemple sur cinq valeurs :

```

5 2 1 4 3
5 2 1      4 3
5 2        1      3 4
2 5        1 3 4
2 5        1 3 4
2 5 1 3 4
1 2 3 4 5

```

Bonne chance.

COURS DE PROGRAMMATION APPROFONDIE

Dominique Chastagnier
Jean-François Coblentz
Patrick Gueneau

Nous reprenons l'étude des mouvements du curseur de l'éditeur de texte à l'endroit où nous l'avons laissé le mois dernier. Nous y avons inclus le descriptif en pseudo-langage des routines qui s'y rapportent. Il vous faudra encore un peu de patience pour venir à bout de ce projet. Il nous reste en effet à décrire quelques routines essentielles et surtout à effectuer l'assemblage des divers éléments pour enfin obtenir l'application souhaitée (on peut l'espérer).

COURS N° 19

PLAN DU COURS

4. Mouvements du curseur (suite)

4. MOUVEMENTS DU CURSEUR (SUITE)

GAUCHE :

début_procedure

X: = X - 1 ; (un cran à gauche)

si (X = 0) alors

(on est sur le bord de l'écran)

début_si

Y: = Y - 1 ; (il faut passer à la ligne précédente)

si (Y = 0) alors

(on est en haut, il faut donc décaler)

début_si

DEF_HAUT_BAS(err) ;

Y: = 1 ;

fin_si

si (err est faux) alors

(si on n'est pas en haut du buffer)

- (il faut donc bouger le curseur)

début_si

(on se replace en fin de ligne)

X: = long (Buffer(prem_ligne + Y - 1)) ;

sinon

X: = 1 ;

ERREUR : (beep et pas de mouvement)

*fin_si**fin_si**fin_procedure*

DROITE :

début_procedure

X: = X + 1 ; (un cran à droite)

si (X > long (buffer(prem_ligne + Y - 1))) alors

(on est à la fin de la ligne)

début_si

Y: = Y + 1 ; (il faut passer à la ligne suivante)

si (Y > nb_ligne) alors

(on est en bas, il faut donc décaler)

début_si

DEF_BAS_HAUT(err) ;

Y: = nb_ligne ;

fin_si

si (err est vrai) alors

(si on est en bas du buffer)

(il faut donc reprendre l'ancienne place du curseur)

début_si

X: = X - 1 ; (on se replace en fin de ligne)

ERREUR ; (beep et pas de mouvement)

```

                sinon
                X:= 1 ; (on est en début de ligne suivante)
            fin_si
        fin_si
    fin_procedure

```

Remarque sur GAUCHE ET DROITE :

On s'aperçoit que les routines DEF_HAUT_BAS et DEF_BAS_HAUT ne traitent pas directement l'erreur. Il faut donc le faire dans les routines appelantes. L'avantage de cette solution est de différencier les opérations à effectuer en retour de l'appel : nous verrons pour les procédures HAUT et BAS, l'intérêt de ce traitement différé.

MONTE :

```

début_procedure
    Y:= Y - 1 ;
    si (Y = 0) alors

        (il faut directement appeler DEF_HAUT_BAS)

```

```

début_si
    DEF_HAUT_BAS(err) ;
    Y:= 1 ;      (on reste en haut de l'écran)
    si (err est vrai) alors

```

(on ne fait rien et on génère une erreur)

```

        début_si
            ERREUR ;
        fin_si
    fin_si

```

(calcul de la nouvelle position du curseur)

```

X0:= long (buffer(prem_ligne + Y - 1) ) ;
si (X > X0) alors

```

(il faut mettre alors le curseur en fin de ligne)
(c'est en fait une convention, mais elle simplifie la gestion)

```

    début_si
        X:= X0 + 1 ;      (après le dernier caractère)
    fin_si
fin_procedure

```

DESCENDRE :

```

début_procedure
    Y:= Y + 1 ;
    si (Y > nb_ligne) alors

```

(il faut directement appeler DEF_BAS_HAUT)

```

début_si
    DEF_BAS_HAUT(err) ;
    Y:= nb_ligne      (on reste en bas de l'écran)
    si (err est vrai) alors

```

(on ne fait rien et on génère une erreur)

```

    début_si
      ERREUR ;
    fin_si
  fin_si

```

(calcul de la nouvelle position du curseur)

```

X0: = long (buffer(prem_ligne + Y - 1) ) ;
si (X > X0) alors

```

(il faut mettre alors le curseur en fin de ligne)
(c'est en fait une convention, mais elle simplifie la gestion)

```

    début_si
      X: = X0 + 1 ;      (après le dernier caractère)
    fin_si
  fin_procedure

```

(Rappel : nb_ligne est le nombre de lignes de l'écran et prem_ligne le n° de la première ligne affichée du buffer.)

Il ne nous reste plus qu'à écrire les deux routines de défilement.

DEF_HAUT_BAS :

```

début_procedure
  prem_ligne: = prem_ligne - 1 ;
  si (premier_ligne = 0) alors

```

(on est en haut du buffer donc on ne fait rien)

```

    début_si
      prem_ligne: = 1 ;      (on restitue l'ancienne valeur)
      err: = vrai ;
    sinon
      err: = faux

```

(on affiche tout l'écran)

```

    effacement ;      (routine standard d'effacement d'écran)

```

(boucle sur les nb_ligne lignes à afficher)

```

    pour i allant_de prem_ligne à (premier_ligne + nb_ligne - 1)

```

```

      début_pour
        position_écran(1, i - premier_ligne + 1) ;
        écrit(buffer(i) ) ;
      fin_pour

```

```

    fin_si

```

DEF_BAS_HAUT :

```

début_procedure
  prem_ligne: = premier_ligne + 1 ;
  si (premier_ligne > nb_occup) alors

```

(on est bas du buffer donc on ne fait rien)

```

début_si
  prem_ligne := nb_occup ;    (on restitue l'ancienne valeur)
  err := vrai ;

                               sinon
  err := faux ;

  (on affiche tout l'écran)

  effacement ;    (routine standard d'effacement d'écran)

  (boucle sur les nb_ligne lignes à afficher)

  pour i allant_de prem_ligne à (prem_ligne + nb_ligne - 1)

    début_pour
      position_écran(1, i - prem_ligne + 1)
      écrit(buffer(i)) ;
    fin_pour

  fin_si

```

rappel : nb_occup est le nombre total de lignes utilisées du buffer.

CONCLUSION

Il nous reste donc à étudier quelques routines comme celles de destructions (caractères, lignes), mais aussi celle fondamentale d'insertion d'une nouvelle ligne ; elle sera affectée bien évidemment à la touche «return». C'est cette dernière procédure qui sera notamment chargée de détecter un dépassement de capacité du buffer. Nous verrons aussi qu'elle provoque de nombreuses opérations tant sur ce buffer que sur l'affichage (défilement partiel de l'écran, modification des lignes, etc.). Nous pourrions ensuite décrire en détail le fonctionnement général du programme (nous fournirons le programme source en Pascal et en Basic de la routine CORPS). Enfin, nous reviendrons quelque peu sur les détails des différentes routines que nous présentons ce mois-ci pour permettre de mieux comprendre le comment et le pourquoi de nos choix.

DIALOGUE AVEC NOS LECTEURS

Ce mois-ci, nous ne traiterons que la résolution du troisième degré. Nous l'avons adaptée à un problème mathématique souvent rencontré dans des applications pratiques : la détermination des valeurs propres et vecteurs propres d'une matrice 3×3 symétrique. Pour des raisons de place, nous ne vous soumettrons que la solution Pascal, mais la traduction en Basic n'est pas d'une grande difficulté.

I. VALEURS PROPRES ET VECTEURS PROPRES

Nous vous prions de nous excuser du caractère exagérément «matheux» de cet exercice et nous bornerons à rappeler les propriétés des vecteurs propres et valeurs propres, considérant que la théorie mathématique sous-jacente n'est pas de notre domaine.

Toute matrice M peut se mettre sous la forme d'un produit de trois matrices.

$$M = P^{-1} A P \quad \text{où } P^{-1} \cdot P = I$$

I est la matrice identité car elle a des zéros partout sauf sur la diagonale où il y a des 1 et le produit de $M \cdot I = M$.

A est une matrice nulle sauf sur la diagonale où se trouvent toutes ses valeurs propres. Pour déterminer les valeurs propres d'une matrice, on résout l'équation $M - \lambda I = 0$, équation en λ du degré de la taille de la matrice.

Ici, c'est bien notre équation du troisième degré.

Après cela, à chaque valeur propre correspond un vecteur propre, vecteur qui reste invariant à la transformation par $M - \lambda_i I$ où λ_i est la $i^{\text{ème}}$ valeur propre.

II. MISE EN ŒUVRE

1. Quelques remarques préliminaires

Ce programme a été prévu pour calculer vite et bien, c'est-à-dire qu'il ne peut se permettre des résultats fantaisistes ni pousser très loin la précision, ceci nécessitant généralement beaucoup trop de calculs.

Ainsi, on ne teste pas par rapport à 0 mais à un epsilon donné. De plus, la résolution du troisième degré est hybride entre la solution purement algébrique et la méthode trigonométrique.

2. Commentaires de programmation

- Il manque de commentaires ! Exemple-type de programme de «matheux».
- On détermine l'équation $\lambda^3 + B\lambda^2 + C\lambda + D$ d'où l'on sort $x^3 + px + q$.
- La matrice est symétrique donc toutes les racines sont réelles.
- Le type REEL = REAL peut faire rire mais si vous voulez passer en double précision, vous n'avez qu'à le changer une seule fois dans le programme, c'est très agréable.

```
{ Module de calcul de vecteurs propres et valeurs propres }
```

```
PROGRAM CALCUL_VECTEURS ;
```

```
Const
```

```
  d_pi_s_3 = 2.0943951 ; { deux * pi / trois }  
  epsilon = 0.00001 ;   { zero de precision }  
  n_vol_max = 100 ;
```

```
Type
```

```
  solutions = (simples,doubles,triples) ;
```

```
  REEL = REAL ;
```

```
  T_MATRICE = ARRAY [1..3,1..3] OF REEL ;
```

```
  T_VECTEUR = ARRAY [1..3] OF REEL ;
```

```
  T_REEL_VOL = ARRAY [0..N_VOL_MAX] OF REEL ;
```

```
  T_INTEGER_VOL = ARRAY [0..N_VOL_MAX] OF INTEGER ;
```

```
  TEXTE = ARRAY [1..8] OF CHAR ;
```

```
Var
```

```
  ok : boolean ;  
  i : integer ;  
  matrice , vect_prop : T_MATRICE ;  
  val_prop : T_VECTEUR ;
```

```
{-----}
```

```
Function Arc_cos ( X : REEL ) : REEL ;
```

```
  Begin  
    Arc_cos := ARCCOS ( X ) ;  
  End ;
```

```
{-----}
```

```
Function Cub_root ( X : REEL ) : REEL ;
```

```
{ Calcul de la racine cubique d' un reel ;  
  si celui-ci est negatif il rend la valeur negative }.
```

```
Var  
  signe : BOOLEAN ;  
  result : REEL ;
```

```
Begin  
  signe := false ;  
  IF X < 0 THEN  
    begin  
      X := - X ;  
      signe := true ;  
    end ;  
  result := EXP ( LN ( X ) / 3.0 ) ;  
  IF signe THEN result := - result ;  
  Cub_root := result ;  
End ;
```

```
{-----}
```

```

Function Valeurs_propres ( IN matr_A      : T_MATRICE ;
                          OUT x          : T_VECTEUR ;
                          OUT val_propres : solutions ) : BOOLEAN ;

```

```

Var

```

```

i , j : integer ;
B , C , D , p , q , racine , theta , discriminant , test : REEL ;
ok : BOOLEAN ;
matr_t : ARRAY [1..6,1..6] OF REEL ;

```

```

Begin

```

```

ok := true ;
for i := 1 to 3 do
  for j := 1 to 3 do
    begin
      matr_t[i,j] := matr_A[i,j] ;
      matr_t[i+3,j] := matr_A[i,j] ;
      matr_t[i,j+3] := matr_A[i,j] ;
      matr_t[i+3,j+3] := matr_A[i,j] ;
    end ;

```

```

B := 0.0 ;
for i := 1 to 3 do B := B - matr_t[i,i] ;
C := 0.0 ;
for i := 1 to 3 do
  C := C + matr_t[2 * i - 1,2 * i - 1] * matr_t[2 * i,2 * i]
        - matr_t[i + 1,i] * matr_t[i,i + 1] ;

```

```

D := 0.0 ;
for i := 1 to 3 do
  begin
    p := 1 ;
    for j := 1 to 3 do
      p := p * matr_t[i + j - 1,j] ;
    D := D - p ;
    p := 1 ;
    for j := 3 downto 1 do
      p := p * matr_t[i - j + 3,j] ;
    D := D + p ;
  end ;

```

```

p := ( 3.0 * C - SQR ( B ) ) / 3.0 ;
q := ( 2.0 * B * B * B - 9.0 * B * C + 27.0 * D ) / 27.0 ;
discriminant := q * q / 4.0 + p * p * p / 27.0 ;
IF ABS ( q ) < 1 THEN test := epsilon
  ELSE test := 0.0000001 * q * q ;
IF discriminant > test THEN ok := false

```

```

ELSE IF discriminant > - test THEN
  begin
    racine := 2 * cub_root( - q / 2.0 ) ;
    B := - B / 3.0 ;
    x[1] := racine + B ;
    x[2] := ( - racine / 2.0 ) + B ;
    x[3] := x[2] ;
    IF racine < test THEN val_propres := triples
      ELSE val_propres := doubles ;
  end

```

```

ELSE { discriminant > 0 }
  begin
    racine := 2.0 * SQRT( - p / 3.0 ) ;
    theta := arc_cos( 3.0 / p * q / racine ) / 3.0 ;
    x[1] := racine * COS( theta ) - B / 3.0 ;
    x[2] := racine * COS( theta + d_pi_s_3 ) - B / 3.0 ;
    x[3] := racine * COS( theta + 2 * d_pi_s_3 ) - B / 3.0 ;
    val_propres := simples ;
  end ;

```

```
Valeurs_propres := ok ;
End ;
```

```
{-----}
```

```
Function Calcul_vect_propres (   matr_A      : T_MATRICE ;
                                OUT val_prop  : T_VECTEUR ;
                                OUT vect_prop  : T_MATRICE ) : BOOLEAN ; EXTERN ;
```

```
Var
  i , j , k : integer ;
  determinant , norme , verif , seuil : REEL ;
  val_propres : solutions ;
  matr_t : T_MATRICE ;
  ok : BOOLEAN ;
```

```
Begin
  ok := valeurs_propres ( matr_A , val_prop , val_propres ) ;
  seuil := 0.0 ;
  for i := 1 to 3 do seuil := seuil + ABS ( val_prop[i] ) ;
  seuil := 0.0001 * seuil ;
  IF val_propres = simples THEN
    for i := 1 to 3 do
      begin
        for j := 1 to 3 do
          for k := 1 to 3 do
            matr_t [ j , k ] := matr_A [ j , k ] ;
          for k := 1 to 3 do
            matr_t [ k , k ] := matr_t [ k , k ] - val_prop [ i ] ;
          vect_prop [ 3 , i ] := 1.0 ;
          determinant := matr_t [ 1 , 1 ] * matr_t [ 2 , 2 ]
                        - matr_t [ 2 , 1 ] * matr_t [ 1 , 2 ] ;
          IF ABS ( determinant ) > epsilon THEN
            begin
              vect_prop [ 1 , i ] := ( matr_t [ 2 , 3 ] * matr_t [ 1 , 2 ]
                                     - matr_t [ 1 , 3 ] * matr_t [ 2 , 2 ] ) / determinant ;
              vect_prop [ 2 , i ] := ( matr_t [ 1 , 3 ] * matr_t [ 2 , 1 ]
                                     - matr_t [ 2 , 3 ] * matr_t [ 1 , 1 ] ) / determinant ;
              verif := 0.0 ;
              for j := 1 to 3 do verif := verif + matr_t [ 3 , j ] * vect_prop [ j , i ] ;
            end
          ELSE
            verif := 1.0 ;
          IF ABS ( verif ) > seuil THEN
            begin
              vect_prop [ 2 , i ] := 1.0 ;
              determinant := matr_t [ 1 , 1 ] * matr_t [ 3 , 3 ]
                          - matr_t [ 3 , 1 ] * matr_t [ 1 , 3 ] ;
              IF ABS ( determinant ) > epsilon THEN
                begin
                  vect_prop [ 1 , i ] := ( matr_t [ 3 , 2 ] * matr_t [ 1 , 3 ]
                                           - matr_t [ 3 , 3 ] * matr_t [ 1 , 2 ] ) / determina
                  vect_prop [ 3 , i ] := ( matr_t [ 1 , 2 ] * matr_t [ 3 , 1 ]
                                           - matr_t [ 3 , 2 ] * matr_t [ 1 , 1 ] ) / determina
                  verif := 0.0 ;
                  for j := 1 to 3 do verif := verif + matr_t [ 2 , j ] * vect_prop [ j , i ] ;
                end ;
              end ;
            IF ABS ( verif ) > seuil THEN
              begin
                vect_prop [ 1 , i ] := 1.0 ;
                determinant := matr_t [ 2 , 2 ] * matr_t [ 3 , 3 ]
```

```

- matr_t [ 3 , 2 ] * matr_t [ 2 , 3 ] ;
IF ABS ( determinant ) > epsilon THEN
  begin
    vect_prop [ 2 , i ] := ( matr_t [ 3 , 1 ] * matr_t [ 2 , 3 ]
                          - matr_t [ 3 , 3 ] * matr_t [ 2 , 1 ] ) / determina
    vect_prop [ 3 , i ] := ( matr_t [ 3 , 1 ] * matr_t [ 2 , 2 ]
                          - matr_t [ 3 , 2 ] * matr_t [ 2 , 1 ] ) / determina
    verif := 0.0 ;
    for j := 1 to 3 do verif := verif + matr_t [ 1 , j ] * vect_prop [ j , i
  end ;
end ;
IF ABS ( verif ) < seuil THEN
  begin
    norme := 0.0 ;
    for j := 1 to 3 do norme := norme + SQR ( vect_prop [ j , i ] ) ;
    norme := SQRT ( norme ) ;
    for j := 1 to 3 do vect_prop [ j , i ] := vect_prop [ j , i ] / norme ;
  end
ELSE
  begin
    writeln ( ' pas de vecteur propre ' ) ;
    ok := false ;
  end ;
end
ELSE IF val_propres = doubles THEN
  begin
    for j := 1 to 2 do
      for k := 1 to 3 do
        matr_t [ j , k ] := matr_A [ j , k ] ;
      vect_prop [ 3 , 1 ] := 1.0 ;
      for k := 1 to 2 do
        matr_t [ k , k ] := matr_t [ k , k ] - val_prop [ 1 ] ;
      determinant := matr_t [ 1 , 1 ] * matr_t [ 2 , 2 ]
                    - matr_t [ 2 , 1 ] * matr_t [ 1 , 2 ] ;
      vect_prop [ 1 , 1 ] := ( matr_t [ 2 , 3 ] * matr_t [ 1 , 2 ]
                          - matr_t [ 1 , 3 ] * matr_t [ 2 , 2 ] ) / determinant ;
      vect_prop [ 2 , 1 ] := ( matr_t [ 1 , 3 ] * matr_t [ 2 , 1 ]
                          - matr_t [ 2 , 3 ] * matr_t [ 1 , 1 ] ) / determinant ;
      for k := 1 to 2 do
        matr_t [ k , k ] := matr_A [ k , k ] - val_prop [ 2 ] ;
      vect_prop [ 1 , 2 ] := 1.0 ;
      vect_prop [ 1 , 3 ] := 0.0 ;
      vect_prop [ 2 , 2 ] := 0.0 ;
      vect_prop [ 2 , 3 ] := 1.0 ;
      i := 1 ;
      WHILE ( i < 4 ) and ( matr_t [ i , 1 ] * matr_t [ i , 3 ] = 0 ) DO
        i := i + 1 ;
      IF i < 4 THEN
        vect_prop [ 3 , 2 ] := - matr_t [ i , 1 ] / matr_t [ i , 3 ]
      ELSE
        vect_prop [ 3 , 2 ] := 0 ;
      i := 1 ;
      WHILE ( i < 4 ) and ( matr_t [ i , 2 ] * matr_t [ i , 3 ] = 0 ) DO
        i := i + 1 ;
      IF i < 4 THEN
        vect_prop [ 3 , 3 ] := - matr_t [ i , 2 ] / matr_t [ i , 3 ]
      ELSE
        vect_prop [ 3 , 3 ] := 0 ;
      for i := 1 to 3 do
        begin
          norme := 0.0 ;
          for j := 1 to 3 do norme := norme + SQR ( vect_prop [ j , i ] ) ;
          norme := SQRT ( norme ) ;
          for j := 1 to 3 do vect_prop [ j , i ] := vect_prop [ j , i ] / norme ;
        end ;
      end ;
    end
  end
end

```

```

ELSE
  begin
    for i := 1 to 3 do
      for j := 1 to 3 do
        IF i = j THEN vect_prop [ i , j ] := 1.0
          ELSE vect_prop [ i , j ] := 0.0 ;
      end ;
    Calcul_vect_propres := ok ;
  End ;
Begin      { Programme Principal }

  ok := Calcul_vect_propres ( matrice , val_prop , vect_prop ) ;

End.

```

M. Michel Auchère nous a envoyé trois petits programmes de tracés de courbes qui sont très intéressants en tant que «benchmarks» graphiques pour évaluer les performances de votre machine.

III. BENCHMARKS GRAPHIQUES

I. EPICYCLOÏDES

```

5 REM RAYON DU GALET R
10 R=1
20 INPUT "RAYON DE LA PISTE" ; P
30 CLS
40 FOR T=0 TO 2 * PI * R STEP 0 , 02
50 RAD
60 X=(P+R) * SIN(T)-R * SIN(T * (1+P/R))
70 Y=(P+R) * COS(T)-R * COS(T * (1+P/R))
80 PLOT 15 * X , 15 * Y
90 NEXT
100 PLOT 0 , -200
110 DRAW 0 , 200
120 PLOT -320 , 0
130 DRAW 320 , 0
140 GOTO 140

```

II. HYPOCYCLOÏDES

```

5 REM RAYON DU GALET R
10 R=1
20 INPUT "RAYON DE LA PISTE" ; P
30 CLS
40 FOR T=0 TO 2 * PI * R STEP 0 , 02
50 RAD
60 X=(P-R) * COS(T)+R * COS(T * (P/R-1))
70 Y=(P-R) * SIN(T)-R * SIN(T * (P/R-1))
80 PLOT 20 X , 20 Y
90 NEXT
100 PLOT 0 , -200
110 DRAW 0 , 200
120 PLOT -320 , 0
130 DRAW 320 , 0
140 GOTO 140

```

III. CYCLOÏDES

```

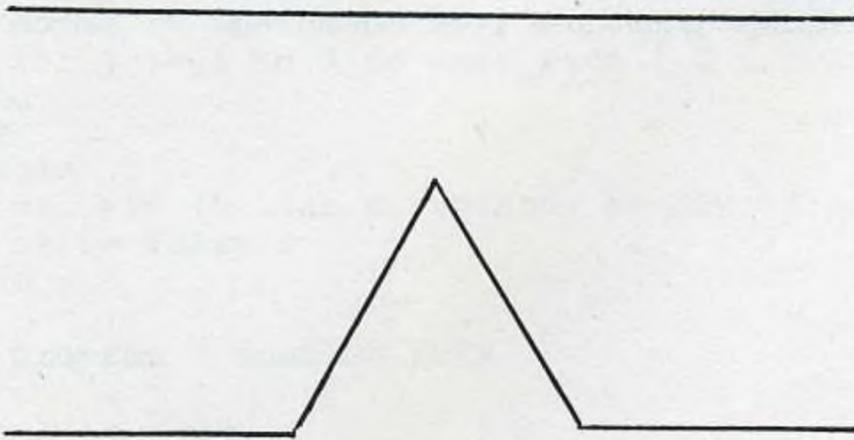
5 INPUT "RAYON" ; R
10 CLS
20 PLOT 0 , 100
30 DRAW 640 , 100
40 FOR T=0 TO 8 * PI * R STEP 0 , 1
50 RAD
60 X=T-SIN(T)
70 Y=1-COS(T)
80 PLOT 25 * X , 100 * Y + 100
90 NEXT T
100 GOTO 100

```

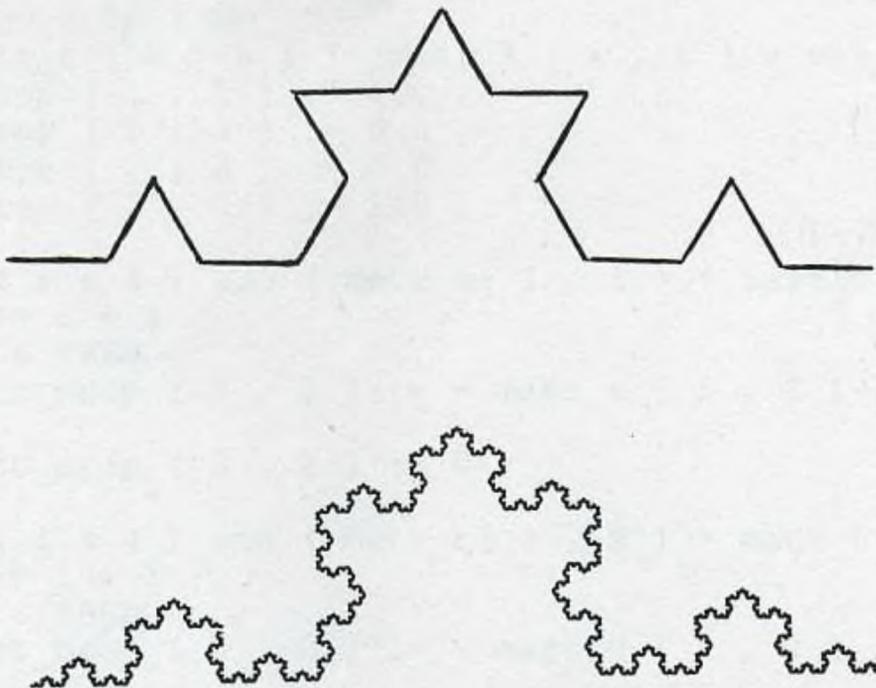
IV. LES FRACTALS

Nous allons aujourd'hui vous présenter un sujet qui passionne beaucoup de scientifiques et autant de non-scientifiques : les Fractals (et non pas «fractaux» car ce nom est issu de l'anglais).

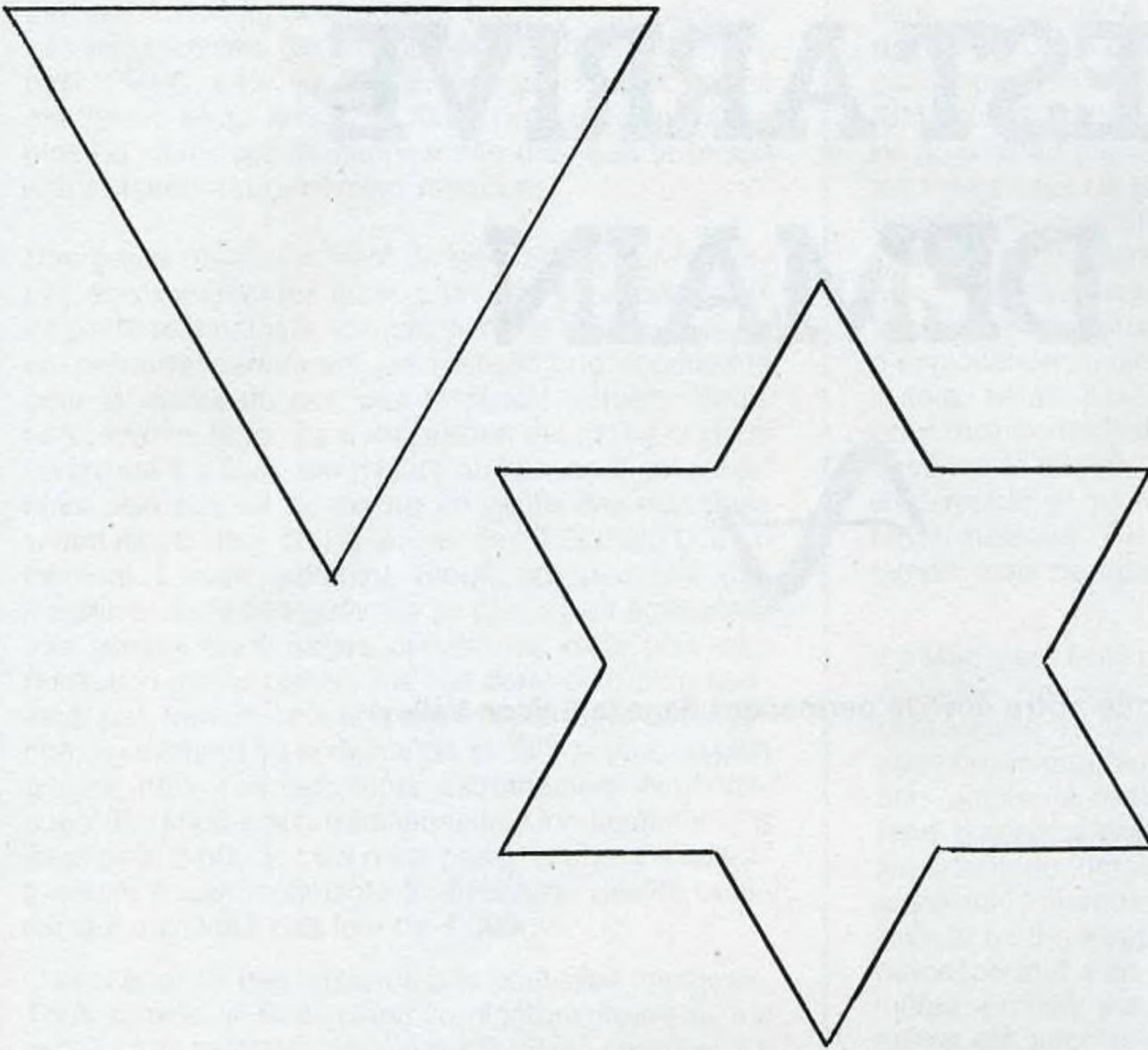
En quoi consiste un fractal ? C'est simplement un segment qui est remplacé par un motif de même longueur que le dit segment ; ce motif étant lui-même uniquement constitué de segments de longueur identique, cette longueur pouvant toutefois différer de la longueur du segment originel.



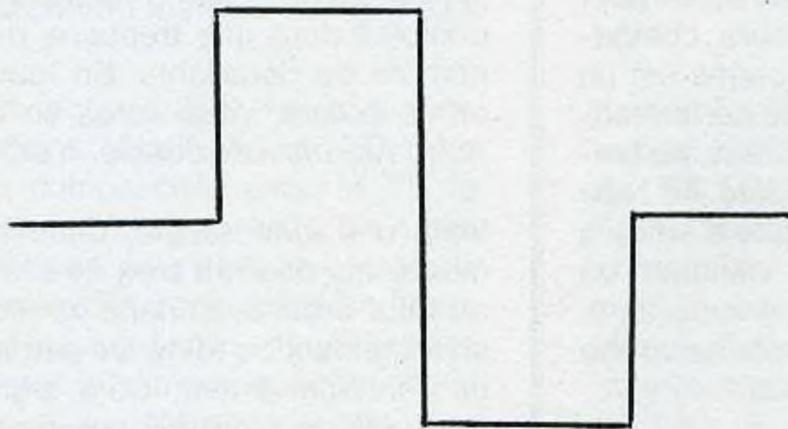
Une fois effectuée la substitution, on la recommence sur les nouveaux segments.



De proche en proche, on obtient un dessin de plus en plus sophistiqué. Il est alors intéressant de définir la figure de départ non pas comme un seul segment mais comme un groupe de segments constituant une figure simple (triangle équilatéral, carré, pentagone, etc.) ne possédant comme seule contrainte que la longueur invariante de ses composants.

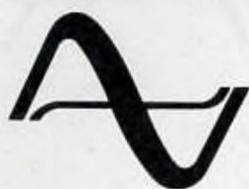


Le dernier sujet d'étude reste la définition du modèle de substitution. Au contraire de ce que l'on pourrait croire, les plus complexes ne sont pas toujours les plus jolis. Si vous en trouvez un paraissant original, envoyez-le nous.



Du point de vue programmation, la récursivité du processus est évidente, si le programme est aisé en BASIC, il devient enfantin en PASCAL. Bon courage.

C'EST ARRIVE DEMAIN



(en direct de notre envoyé permanent dans la Silicon Valley)

A l'aide de matériels informatiques somme toute assez rudimentaires, a été menée une étude fort intéressante sur la vigilance des pilotes de lignes lors des grands trajets intercontinentaux. Les conclusions sont à la fois instructives et inquiétantes. Il ressort de ce travail que les pilotes sont en état de sous-vigilance la plupart du temps, en raison de l'inadaptation du corps aux décalages horaires, et surtout à la multiplication de ces décalages dans les deux sens. Jours raccourcis puis allongés sont épuisants pour l'homme. L'étude qui portait sur plusieurs compagnies, montre que la jeunesse des équipages est un critère primordial de sécurité, la baisse de performances étant particulièrement nette après 35 ans. Au palmarès de la sécurité vient d'ailleurs en tête Air New Zealand, qui met ses équipages à la retraite d'office à 40 ans. Les compagnies américaines viennent en queue de classement, car aux USA le principe de la retraite obligatoire est illégal. La France est dans une moyenne que certains trouveront honorables.

Un développeur propose un petit bijou de programme et d'électronique permettant de faire tourner sur un Atari ST la plupart des programmes du MAC. Pour qu'un tel programme tourne sur Atari, il suffit qu'il soit programmé «correctement» sur le Mac, c'est-à-dire qu'il suive à la lettre l'interface Apple. Si c'est le cas,

vous aurez le bonheur de le voir tourner sur votre ST et en plus bien plus vite que sur un Mac. En effet, l'horloge interne du processeur de l'Atari est plus rapide. Comment une telle prouesse est-elle possible ? C'est simple en théorie, puisque les deux ordinateurs ont le même cœur, le 68000. En pratique, c'est moins immédiat et, en particulier, il vous faudra acheter les ROM de l'Apple, le fabricant de l'émulateur ne voulant pas prendre le risque de poursuites judiciaires de la part d'Apple. Mais ces ROM sont disponibles pour une trentaine de dollars chez un bon nombre de détaillants. En tout, pour moins de deux cents dollars, vous aurez un Mac en plus de votre Atari. Atari favori, désolé, n'est pas une rime riche.

Voilà une idée simple, dans sa formulation tout au moins, qui pourrait bien être la clé de la bureautique du futur proche. Borland va prochainement proposer un traitement de texte qui permet de choisir son mode de fonctionnement. Cela signifie que les usagers habituels de Wordstar pourront choisir un environnement Wordstar complet, et il en va de même pour la plupart des grands traitements de textes populaires (Wordstar, Word Perfect, Word et Multimate dans un premier temps). Il est clair que c'est là une bonne idée puisque cela permet à une société d'acquiescer ce programme sans s'inquiéter de la formation qu'ont reçue

ses employés. Ils connaissent tous l'un au moins de ces programmes. Ce produit est prévu pour les compatibles PC, cela va de soi, puisque sur les autres machines, Mac, Amiga et Atari, l'interface est figée plus ou moins par la machine elle-même et le temps d'adaptation est nettement raccourci.

Une petite révolution vient de se produire sur le marché des imprimantes laser. L'un des deux fabricants de cette technologie, Canon, vient de mettre au point un procédé permettant de réaliser une imprimante pour la moitié du prix des matériels actuels. Seule contrepartie, le temps d'impression est moins bon, de l'ordre de 6 pages par minute au lieu de 8 actuelles. Mais cela permet de mettre en vente des machines valant moins de 2 000 \$ au lieu des 4 000 à 5 000 du moment. L'autre fabricant, Ricoh, propose déjà des machines dans ces gammes de prix, ayant également une vitesse de 6 pages par minute, mais pour une résolution moins bonne. Il s'agit donc bien d'un nouveau pas franchi vers une imprimante laser bon marché, au moment où la demande se fait de plus en plus grande pour ces substituts extrêmement économiques de l'impression traditionnelle. Une imprimante à laser pour 2 000 \$, cela n'est pas trop cher comparativement à une imprimante traditionnelle qualité courrier qui n'en vaut pas loin de 1 000.

C'est l'époque des annonces de nouvelles machines. Tout comme le Beaujolais, l'ordinateur nouveau est arrivé chez tous les fabricants. Tous ou presque. En effet, ATT, grand rival américain d'IBM sur le créneau des compatibles, doit renoncer à la sortie de sa nouvelle gamme pour cause de... non compatibilité. On pourrait croire que ce type de ridicule péripétie ne pourrait se produire que chez Bull. Pas du tout. Mais, il est quand même surprenant qu'un tel fabricant de compatibles, bien rodé, rencontre un problème aussi grotesque, peu de jours avant la mise en vente de sa nouvelle gamme. En fait la compatibilité de ses produits serait de l'ordre de 60 %, ce qui est absolument inacceptable sur un marché où les vrais compatibles sont légions.

Pour parler des autres nouveautés, en voici une rapide liste des principales. Atari sort un PC pour 500 \$, disposant de la compatibilité avec le XT, le milieu de gamme IBM. Seule limitation réelle, il n'a pas de possibilité d'extension, un peu comme l'Atari ST. Compaq propose une nouvelle version de portable, plus léger et un peu plus puissant, utilisant un processeur 80286, le processeur à la mode pour les nouveaux PC. Apple propose un «file server» basé sur Appletalk, le réseau maison, permettant également de se connecter sur un réseau de PC. Un seul problème, mais un peu gros, il demande un Mac uniquement pour le faire tourner, ce qui signifie quand même un système conséquent pour un réseau. Grid a montré la semaine dernière un portable vraiment beau et puis-

sant, compatible AT, disposant en standard de plus de 1 Mo et d'un disque 3,5 pouce de 720 Ko. Le tout pour moins de 1 700 \$, ce qui est compétitif. Enfin, IBM annonce trois nouvelles machines, dont un bas de gamme au prix de 700 \$, ce qui est plus cher que les milieu-haut de gamme de certains compatibles et deux autres ordinateurs très intéressants. Le premier est basé sur le 80286 et devrait remplacer le AT sous peu, il dispose en standard de la couleur et possède tous les plus actuels, dont les meilleurs systèmes d'exploitation du moment pour PC, en mémoire morte. Il sera, hélas, sans doute très cher, trop peut-être pour être compétitif. Enfin, IBM propose un appareil basé sur le 80386, la dernière petite merveille d'Intel, ultra-rapide et permettant de gérer une très importante mémoire. Ce dernier sera près dans peu de temps, mais peu de détails sont encore connus.

Le Mac s'est taillé pour l'instant la part du lion dans un domaine nouveau, la micro-impression, traduction personnelle et libre de «Desktop Publishing». Ses possibilités graphiques liées à un système d'exploitation simple, le rendaient particulièrement apte à ce type d'applications. Mais, devant le développement important du domaine, les compatibles PC entrent aussi sur le marché. En particulier, un programme réalisé par un développeur habitué au Mac, Aldus Corporation, permet d'en faire autant ou presque qu'avec le même produit sur le Mac. Des améliorations ont même été apportées, qui se retrouveront bientôt sur la version Mac. Mais surtout, ce type de programmes peut tirer un grand profit de la vitesse des PC-AT et autres haut de gamme des compatibles PC. Ils utilisent en général Windows, une interface logicielle de Microsoft, qui transforme un PC en sorte de Mac. Windows est lent mais, sur un AT, cela n'est pas trop sensible. Et sur un PC, vous pouvez avoir la couleur. Tous ces programmes sont disponibles et il faudra déboursier 700 \$, ce qui est énorme pour un particulier, mais peu pour une petite société qui évitera ainsi les frais d'impression élevés pour des brochures simples.

Il se propage une rumeur intéressante ces derniers temps. IBM proposerait bientôt une nouvelle gamme de PC. Rien de bien passionnant, pensez-vous. Erreur, car cette nouvelle gamme ne serait pas PC-compatible. Le monde à l'envers. Alors que croire ? Il semble que la réalité soit bien mesquine une fois de plus. A chaque fois qu'une rumeur semble prendre sa source chez IBM, c'est pour aider cette société à prendre des parts de marché, ou tout au moins en perdre le moins possible avant la sortie d'un nouveau matériel. La dernière fois, c'était avant la sortie du PC Jr. Tout le monde savait ou croyait savoir qu'un merveilleux ordinateur familial allait sortir et qu'il fallait attendre, plutôt qu'acheter un autre produit. A cette époque, la montagne avait accouché de la souris que

l'on sait, et le Junior fut un échec retentissant. Aujourd'hui, la plupart des sociétés de grande et moyenne tailles semblent vouloir renouveler leur parc de compatibles PC, car les progrès en termes de vitesses et de puissances ont été importants en un an. Que fait IBM ? Il leur dit d'attendre, par le biais de cette rumeur. Mais aussi, une contre-rumeur maléfique se propage. Elle dit que le futur PC sera totalement incompatible non seulement avec les PC mais aussi avec les extensions existantes. Ceci veut dire qu'IBM souhaite ré-établir un monopole de fait sur les ordinateurs et sur les extensions en tous genres. En bref, tenter de réussir ce qui a été manqué la première fois, mais cette manœuvre attaque de front toutes les sociétés qui se sont équipées de matériels lourds et coûteux sous la foi de la stabilisation qu'apportait le standard actuel PC, de même que les sociétés produisant des interfaces et extensions pour ce standard. Affaire à suivre.

Une grande évolution semble se dessiner en ce qui concerne les tableurs. Pendant dix ans, les enfants de VisiCalc, le premier du genre, se sont contentés de plagiat simple et peu inventif. Puis sont venus les intégrés, mais la partie calcul n'évoluait toujours pas. Il semble que maintenant, la créativité soit à nouveau présente dans ce secteur de l'activité logicielle. Parmi les nouveautés, il faut signaler le grand nombre de programmes qui permettent d'ajouter des possibilités aux programmes existants. Ainsi, Hal, de Lotus, qui complète 1-2-3 de la même société, ou Note It, de Turner Hall, de nouveaux venus sur le marché.

Hal est une interface «intelligente» pour 1-2-3, permettant d'ajouter des commandes, de gérer le tout avec une plus grande simplicité, ce qui n'est pas un mal, et facilitant le contact avec l'utilisateur. C'est sans doute la raison de son nom, celui que portait l'ordinateur intelligent et capable de raisonnement du film «2001, Odyssée de l'Espace». Rappelons que Hal, dans le film, se détraque et finit par tuer.

Mais il y a aussi des programmes qui proposent tout cela directement, plus bien d'autres choses encore, toutes bien nouvelles. Parmi ces programmes, le nouveau Silk, de Daybreak Technologies, ou Trapeze, ce dernier pour le Mac. Ces programmes permettent d'automatiser un grand nombre de calculs, en propageant des formules à l'aide d'une simple touche, de surveiller l'évolution de certains résultats tout au long de l'élaboration de la feuille de calculs, ou de simuler ces résultats à l'aide d'une feuille spéciale, destinée à des essais numériques. Il est possible de choisir un résultat, puis de laisser le programme trouver les meilleures données de départ permettant de réellement obtenir la solution «imposée». Donc, un tableur devient un concurrent direct de programmes de simulations, ou de gestions numériques. Silk permet même de modifier (un peu) la structure des données pour récupérer sous un format différent des fichiers venant d'un gros système. Il est aussi possible de définir des

structures plus complexes que des nombres, par exemple des couples de nombres, des tableaux, et autres formes utiles dans divers types d'applications. Il est possible de pister les problèmes par suivi des calculs, ce qui permet d'éliminer la bête noire de ces programmes, les références circulaires, c'est-à-dire des cases que l'on calcule à partir d'autres, ces dernières ayant besoin des premières, ce qui arrive sur les grosses feuilles, où il arrive que la logique des enchaînements soit délicate à décortiquer et où toute modification devient délicate à effectuer. Ces programmes permettent aussi de sauver en permanence sur disque les commandes tapées et si un problème fait planter le système, il est possible de reprendre où l'on en était. Enfin, Trapeze, tirant parti des possibilités de graphisme du Mac, permet de créer des cases ou des zones graphiques. C'est là une amélioration formidable, car avant, il fallait récupérer les résultats du tableur, les mettre dans un traitement de texte ou de graphiques, arranger le tout, puis imprimer. Avec Trapeze, tout peut être fait d'un seul coup. C'est en particulier plus puissant qu'un logiciel intégré, où tout cohabite mais, en France, vous savez maintenant que cohabiter n'est pas toujours une panacée.

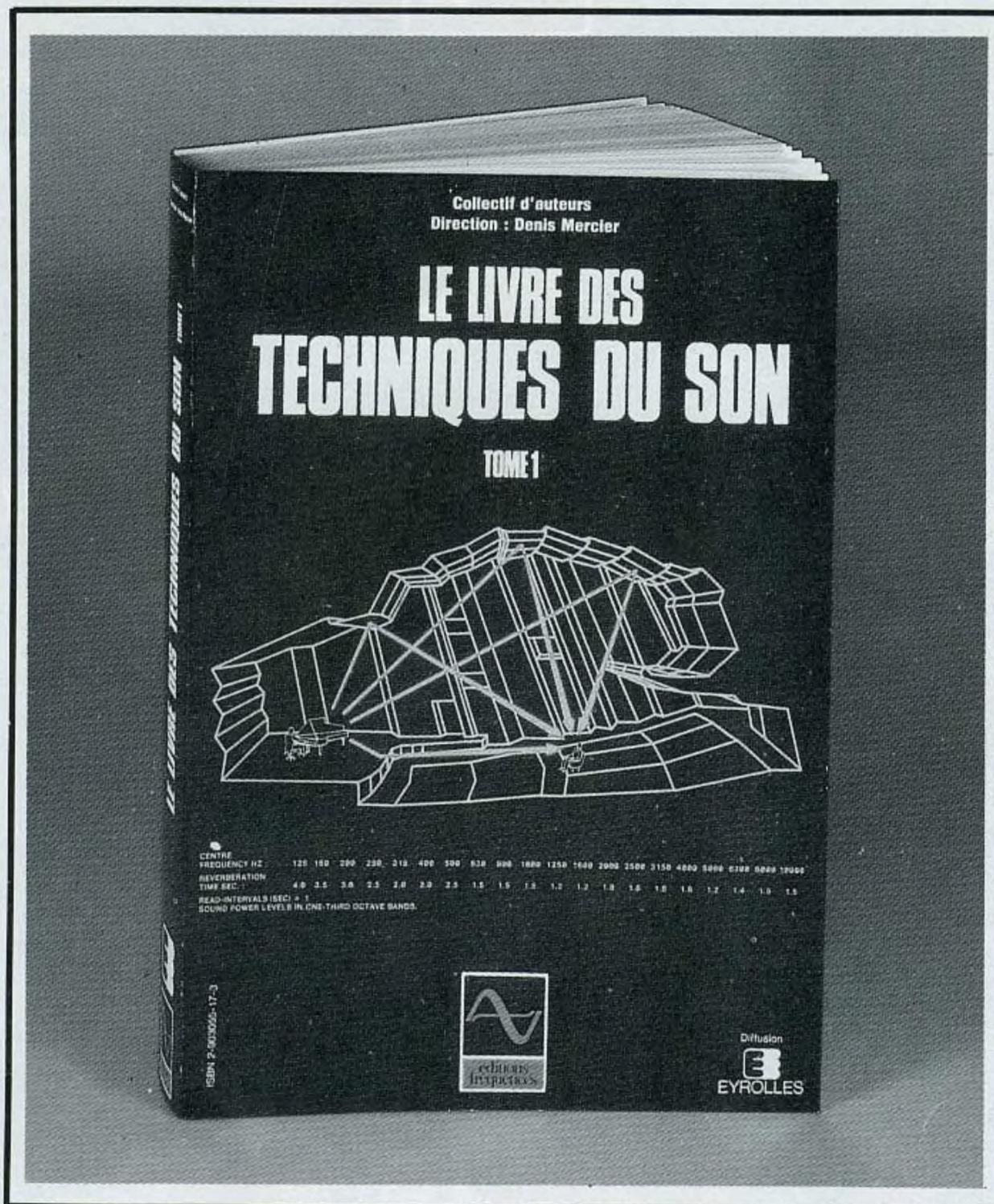
Un utilisateur de programme comme vous et moi en a eu assez, un jour, de se battre avec. Las de chercher en vain après avoir essayé dix tableurs et une bonne vingtaine de traitements de textes, il vient de publier une sorte de code du programmeur. Sept points à retenir :

- pas de protection,
- pas de programme d'installation de programme, ces programmes qui permettent de configurer le logiciel à votre système. Au logiciel de s'adapter,
- Mise à jour des parties interfaces du logiciel par des commandes du programme et non par un programme annexe. Ceci permet de modifier les spécifications sans sortir du programme,
- Indépendance des gestions d'entrées-sorties vis-à-vis du programme, et ajout dans la documentation du programme de la description des commandes permettant de créer de nouvelles entrées-sorties pour des périphériques à venir,
- Un guide référence complet, c'est-à-dire un livre séparé réservé à la description de chaque commande,
- Aide à l'écran en permanence, avec adaptation automatique à la situation du moment,
- Interface systématique pour une souris pour les ordinateurs qui ne la proposent pas en standard.

Avec toutes ces propositions, un programme pourra devenir un vrai plaisir, et non plus un cauchemar de l'utilisateur occasionnel. Reste à savoir si les développeurs suivront le chemin qui semble pourtant s'imposer. Le passé proche autorise à l'optimisme.

Au mois prochain.

VIENT DE PARAÎTRE:



- 11 auteurs
 - 360 pages
 - 300 schémas et illustrations
- Prix : 350 F

Il y a bientôt trois ans démarrait ce travail de fond. Plus de vingt auteurs étaient sollicités pour concentrer en trois tomes les techniques du son. Le premier tome vient de paraître. Il traite de l'**Acoustique fondamentale**, des **Sources acoustiques**, de l'**Acoustique architecturale**, de la **Perception auditive**, des **Notions fondamentales de l'Electricité**, de l'**Enregistrement magnétique** ainsi que de la **Technologie audio-numérique**.

L'équipe des plus grands spécialistes actuels a été animée par Denis Mercier. Ensemble, ils ont mis sur pied un ouvrage actuellement unique au monde.

PROCHAINEMENT :

Collection jaune Etude autour du 6809 (constructions et logiciels) de Claude Vicidomini
L'image numérique de Jean-Marc Nasr
Le Basic structuré de Jean-François Coblenz
Divertissements en Basic de Franck Brown

Collection noire La création musicale par ordinateur de Frédéric Levé
Pratique de l'Amiga de Henri Cohen et François Dress

Collection noire (format 165 x 240)

LES SYNTHETISEURS, UNE NOUVELLE LUTHERIE de Claude Gendre - 184 p. - Face au développement spectaculaire des synthétiseurs, grâce à l'électronique numérique, le besoin d'un ouvrage complet accessible et surtout bien informé des dernières ou futures techniques, se faisait ressentir. Le vœu est comblé, en 180 pages.

LES HAUT-PARLEURS de Jean Hiraga - 320 p. - Un gros volume qui connaît un succès constant : bien plus qu'un traité, il s'agit d'une véritable encyclopédie, alliant théorie et pratique, histoire en une mine inépuisable d'informations, reconnue dans le monde entier.

INTRODUCTION A L'AUDIO-NUMERIQUE de Jean-Pierre Picot - 160 p. - C'est le premier ouvrage paru en langue française traitant de l'audio numérique : écrit par un professionnel, avec rigueur et simplicité, il explique brillamment les bases de cette technique : quantification, conversion, formats, codes d'erreurs.

L'OPTIMISATION DES HAUT-PARLEURS ET ENCEINTES ACOUSTIQUES de Charles-Henry Delaleu - 240 p. - Seconde édition améliorée d'un ouvrage fort attendu des passionnés d'électroacoustique. Ce livre permet aux amateurs et aux professionnels de se familiariser avec les rigoureuses techniques de modélisation des haut-parleurs et enceintes acoustiques et d'en mener à bien la réalisation.

LES MAGNETOPHONES de Claude Gendre - 160 p. - Pour tout savoir sur le magnétophone depuis l'avènement de cette mémoire des temps modernes, jusqu'aux enregistreurs numériques, en passant par la cassette «Les magnétophones» est un ouvrage pratique, complet, indispensable à l'amateur d'enregistrement magnétique.

LES MAGNETOSCOPES ET LA TELEVISION de Claude Gendre - 256 p. - Complément direct des «Magnétophones» «Les magnétoscopes et la télévision» débute par un bel historique de la télévision et la description des premiers magnétoscopes. La théorie et la pratique de la capture et de l'enregistrement moderne des images vidéo en sont la teneur essentielle.

L'ELECTRONIQUE DES MICRO-ORDINATEURS de Philippe Faugeras - 128 p. - Cet ouvrage est destiné aux électroniciens désireux d'aborder l'étude du «hard» des micro-ordinateurs. Cette étude s'articule autour du microprocesseur Z-80 très répandu, et en décrit les éléments périphériques : mémoire, clavier, écran, interfaces de toutes sortes.

PERIPHERIQUES : INTERFACES ET TECHNOLOGIE de Philippe Faugeras - 136 p. - Faisant suite à la parution de «L'électronique des micro-ordinateurs», cet ouvrage s'adresse aux électroniciens désireux de s'initier aux montages périphériques des micro-ordinateurs, interfaces en particulier, qui permettent la communication avec monde extérieur.

SELECTION DE L'AUDIOPHILE - TOME 1 : L'ELECTRONIQUE 256 p.

SELECTION DE L'AUDIOPHILE - TOME 2 : LES TRANSDUCTEURS 256 p.

Introuvable aujourd'hui, une sélection des meilleurs articles de la célèbre revue «L'Audiophile». Le tome 1 traite de l'électronique audio à tubes et transistors. Dans un esprit identique, le tome 2 traite du domaine passionnant que constituent les transducteurs en audio.

LE MINI STUDIO de Denis Fortier - 160 p. - Le monde de l'audio évolue... Un secteur d'activité entièrement neuf vient d'apparaître : les mini-studios. L'ouvrage de Denis Fortier, ingénieur du son, aborde le sujet de la manière la plus globale. Après les données physiques indispensables, le choix des maillons, la manière d'installer et d'exploiter.

LES TECHNIQUES DU SON Collectif d'auteurs sous la direction de Denis Mercier - 360 p. - Le Livre des Techniques du Son est le premier ouvrage interdisciplinaire en langue française s'adressant aux professionnels du son.

| Réf. | Prix TTC |
|------|----------|
| E 15 | 140 F |
| E 01 | 165 F |
| E 05 | 155 F |
| E 04 | 154 F |
| E 02 | 92 F |
| E 03 | 155 F |
| E 06 | 150 F |
| E 22 | 150 F |
| E 13 | 155 F |
| E 12 | 165 F |
| E 25 | 140 F |
| E 33 | 350 F |

Collection rouge (format 135 x 210)

CONSEILS ET TOURS DE MAIN EN ELECTRONIQUE de Jean Hiraga 160 p. - Le «dernier coup de patte» apporté à un montage, celui qui fait la différence entre la réalisation approximative et le kit bien fini, ce savoir-faire s'acquiert au fil des ans... ou en parcourant «Conseils et tours de main en électronique».

LES LECTEURS DE COMPACT-DISCS 200 p. - Tout beau, tout nouveau, le lecteur laser. Qu'en est-il réellement ? Pour en savoir plus, un livre traitant du sujet s'imposait. «Les lecteurs de compact-discs» permet de faire son choix parmi 37 modèles testés, analysés, examinés et écoutés.

LEXIQUE DE L'ELECTRONIQUE ANGLAIS-FRANÇAIS de Jean Hiraga - 72 p. - Pour la première fois en électronique, un lexique anglais-français est présenté sous une forme pratique avec en plus des explications techniques, succinctes mais précises. Ce sont plus de 1 500 mots ou termes anglais qui n'auront plus de secret pour vous.

FILTRES ACTIFS ET PASSIFS POUR ENCEINTES ACOUSTIQUES de Charles-Henry Delaleu - 160 p. - Finis les calculs fastidieux et erronés ! Grâce à cet ouvrage, les concepteurs d'enceintes acoustiques gagneront un temps appréciable durant la phase d'étude et de mise au point : 120 abaques et tableaux pour tous types de filtres et d'impédances de HP !

17 MONTAGES ELECTRONIQUES de Bernard Duval - 128 p. - Voici enfin réunies dans un même ouvrage, dix-sept descriptions complètes et précises de montages électroniques simples. Il s'agit de réalisations à la portée de tous, dont bon nombre d'exemplaires fonctionnent régulièrement. Les schémas d'implantation et de circuits imprimés sont systématiquement publiés.

WEEK-END PHOTO de Philippe Folie-Dupart - 208 p. - Accessible à tous, «Week-end photo» permet de découvrir de façon simple les différents aspects de la photographie actuelle. Vous y trouverez les bases indispensables pour vous perfectionner, un guide de choix des appareils 24 x 36 et des illustrations abondamment commentées.

| | |
|------|-------|
| L 07 | 68 F |
| L 10 | 130 F |
| L 09 | 65 F |
| L 11 | 85 F |
| L 14 | 95 F |
| L 20 | 130 F |

Collection jaune (format 210 x 270)

INITIATION A LA ROBOTIQUE 96 p. - Cet ouvrage eut un succès retentissant dès sa sortie. Bien plus qu'un cours d'initiation, il s'agit aussi du premier recueil d'informations données par les concepteurs, les utilisateurs et les fans de cybernétique enfin réunis !

INITIATION A LA MICRO-INFORMATIQUE COURS 1^{er} CYCLE - LE VOLUME 1 de Claude Polgar - 272 p. (épuisé)

INITIATION A LA MICRO-INFORMATIQUE COURS 1^{er} CYCLE - LE VOLUME 2 de Claude Polgar - 208 p.

INITIATION A LA MICRO-INFORMATIQUE COURS 1^{er} CYCLE - LE VOLUME 3 de Claude Polgar - 250 p.

Passé les premiers remous de la révolution que fut l'avènement de la micro-informatique, il fallut bien tenter d'en réunir les enseignements. Une lacune apparut : celle d'un ouvrage d'initiation à la programmation, universel et complet.

INITIATION A L'ELECTRONIQUE DIGITALE de Philippe Duquesne - 104 p. - Ce cours d'initiation à l'électronique digital est dû à Ph. Duquesne, chargé de cours de microprocesseurs au CNAM. L'objet de cet ouvrage est de présenter les opérateurs logiques et leurs associations. La technologie est évoquée, brièvement, elle aussi.

INITIATION AUX MICROPROCESSEURS de Philippe Duquesne - 136 p. - Du même auteur, Ph. Duquesne, on nous propose cette fois-ci, de pénétrer au cœur même de l'ordinateur, de comprendre le fonctionnement de l'élément vital qu'est le microprocesseur et enfin de maîtriser l'assembleur, langage du microprocesseur.

INITIATION TV : RECEPTION, PRATIQUE, MESURES, CIRCUITS de Roger-Charles Houzé - 136 p. - Issu d'un cours régulièrement remis à jour, ce livre permet à l'amateur comme au professionnel de se tenir au courant de l'état actuel de la technologie en télévision. De nombreux schémas explicatifs illustrent le contenu du livre.

INITIATION A LA MESURE ELECTRONIQUE de Michel Casabo - 120 p. - Il n'existait pas, jusqu'à présent, un ouvrage couvrant de manière générale mais précise, l'ensemble des problèmes relatifs à l'instrumentation et à la méthodologie du laboratoire électronique. C'est chose faite aujourd'hui avec ce volume récemment paru.

INITIATION AUX AMPLIS A TRANSISTORS de Gilles Le Doré - 96 p. - Après un bref historique du transistor, cet ouvrage traite essentiellement de la conception des amplificateurs modernes à transistors. La théorie est décrite de manière simple et abordable, illustrée d'exemples de réalisations commerciales. Le but du livre est de donner à chacun la possibilité de réaliser soi-même son amplificateur.

INITIATION AUX AMPLIS A TUBES de Jean Hiraga - 152 p. - Complémentaires des «Amplis à transistors» «les Amplis à tubes» sera certainement une petite encyclopédie sur ce sujet : historique, mais aussi polémique puisque les tubes sont encore d'actualité et parce que les arguments en faveur de cette technique et ses défenseurs sont encore nombreux.

INITIATION A L'ELECTRICITE ET A L'ELECTROTECHNIQUE de Roger Friederich - 110 p. - Vous trouverez aisément en librairie des ouvrages d'initiation à l'électronique ou aux techniques les plus avancées des circuits intégrés, etc. Mais si vous désirez une initiation aux bases de l'électricité et de l'électrotechnique sans vous en remettre à des ouvrages scolaires, alors vous ne trouverez pas !

INITIATION A LA VIDEO LEGERE - THEORIE ET PRATIQUE de Claude Gendre - 72 p. - Choix d'un standard ? Caméscopes VHS, VHS-C ou 8 mm ? Connexion ? Compatibilité ? Accessoires ? Montage ? Enfin... comment filmer ? Le nouveau livre de Claude Gendre répond à toutes ces questions. Cet ouvrage essentiellement pratique n'a pas d'équivalent en librairie aujourd'hui.

LES MONTAGES ELECTRONIQUES de Jean-Pierre Lemoine - 276 p. - Véritable encyclopédie. Plus de 1 000 dessins, 25 montages originaux.

LE TELEPHONE ET LES RADIOTELEPHONES de Roger-Charles Houzé - 96 p., 73 schémas.

LES BASES DE L'ELECTRONIQUE de Raymond Breton - 84 p., 162 schémas. Vous ne connaissez pas l'électronique : ce livre vous permet d'accéder aux bases nécessaires mais néanmoins d'atteindre un niveau vous permettant d'aborder des constructions de bases.

| | |
|------|-------|
| P 08 | 115 F |
| P 16 | 130 F |
| P 17 | 130 F |
| P 27 | 190 |
| P 19 | 95 F |
| P 18 | 95 F |
| P 21 | 135 F |
| P 23 | 140 F |
| P 24 | 130 F |
| P 26 | 155 F |
| P 28 | 150 F |
| P 29 | 100 F |
| P 30 | 250 F |
| P 31 | 130 F |
| P 32 | 120 F |

Diffusion auprès des libraires assurée exclusivement par les Editions Eyrolles.

Bon de commande à retourner aux Editions Fréquences 1, boulevard Ney 75018 Paris.

Je désire recevoir le(s) ouvrage(s) ci-dessous référencé(s) que je coche d'une croix :

| | | | | | | | | | |
|-------------------------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|
| E 01 <input type="checkbox"/> | E 02 <input type="checkbox"/> | E 03 <input type="checkbox"/> | E 04 <input type="checkbox"/> | E 05 <input type="checkbox"/> | E 06 <input type="checkbox"/> | L 07 <input type="checkbox"/> | P 08 <input type="checkbox"/> | L 09 <input type="checkbox"/> | L 10 <input type="checkbox"/> |
| L 11 <input type="checkbox"/> | E 12 <input type="checkbox"/> | E 13 <input type="checkbox"/> | L 14 <input type="checkbox"/> | E 15 <input type="checkbox"/> | P 16 <input type="checkbox"/> | P 17 <input type="checkbox"/> | P 18 <input type="checkbox"/> | P 19 <input type="checkbox"/> | L 20 <input type="checkbox"/> |
| P 21 <input type="checkbox"/> | E 22 <input type="checkbox"/> | P 23 <input type="checkbox"/> | P 24 <input type="checkbox"/> | E 25 <input type="checkbox"/> | P 26 <input type="checkbox"/> | P 27 <input type="checkbox"/> | P 28 <input type="checkbox"/> | P 29 <input type="checkbox"/> | P 30 <input type="checkbox"/> |
| P 31 <input type="checkbox"/> | P 32 <input type="checkbox"/> | E 33 <input type="checkbox"/> | | | | | | | |

Frais de port : + 12 F par livre commandé, soit la somme totale ci-jointe, de Frs par CCP Chèque bancaire Mandat-lettre

Nom Prénom

Adresse

Ville Code Postal

ET LA TECHNIQUE ?

Si l'informatique a été, à l'origine, créée pour des besoins scientifiques ; il faut bien admettre que les applications de gestion ont très nettement pris le dessus. Or depuis peu, il semble que nous arrivions vers une certaine saturation du marché. En effet, de gros clients sont déjà bien servis ! Ainsi les grands constructeurs reviennent-ils à leurs premières amours.

Enfin, si les bureaux sont équipés, le moins que l'on puisse dire, c'est que les ateliers sont sous-équipés. C'est bien connu. Une société de fabrication de matériel informatique ne peut faire des efforts que pour les marchés à fort potentiel de développement économique. Aussi, après quelques légers problèmes, nombreux sont les fabricants qui ont décidé de porter tous leurs efforts vers l'informatique industrielle. L'informatique peut s'implanter, dans une très large mesure, dans le cycle de fabrication d'un objet quelconque. Qu'il s'agisse de C.A.O., de C.F.A.O. ou de contrôle de la qualité en passant par la gestion des stocks, il y a beaucoup à faire dans ce domaine. Comme toujours, c'est le militaire et l'aérospatiale qui entraînent ce long processus. Il convient donc, en partant des connaissances acquises dans ces activités, de démocratiser les systèmes informatiques techniques pour les rendre plus abordables envers l'industrie classique. Il est évident que devant les modifications apportées par les marchés internationaux, le contrôle de la qualité et les coûts de revient sont devenus de véritables outils de guerre économique. Ce n'est que grâce à une forte technicité que les pays dits « développés » pourront continuer à garder leur avance. L'informatique industrielle est prête pour jouer son rôle.

dBase III

Charles-Henry Delaleu

Au hit-parade des progiciels vendus en France, dBase III est le premier de gestion de base de données. DBase III suit en effet une brillante carrière. Rappelons que dBase II fut un des premiers logiciels à être diffusé à une grande échelle. Ses limitations et quelques problèmes de jeunesse furent corrigés dans la version dBase III. Aujourd'hui, dBase III Plus est une version améliorée de dBase III. Elle autorise l'utilisation de fichiers en réseaux. De plus, certains algorithmes de tri ont été ré-étudiés afin d'optimiser la vitesse d'exécution sur gros fichiers.

En prologue, Ashton-Tate présente dBase III comme un système de gestion de base de données de type relationnel. C'est exact dans le principe mais il faut se souvenir que nous sommes dans un environnement micro-informatique. Si les possibilités sont énormes, attention aux capacités de la machine ainsi qu'aux temps d'exécution.

dBase III peut être utilisé de trois manières distinctes : soit en mode clavier simple par interrogation, soit en mode ASSIST par système de menu déroulant, soit en mode programmé en utilisant le langage de programmation de dBase III.

Afin de faciliter l'utilisation de dBase III, deux utilitaires ont été conçus pour créer des applications de gestion de fichiers simples (dBoutils...) en automatique.

DBASE

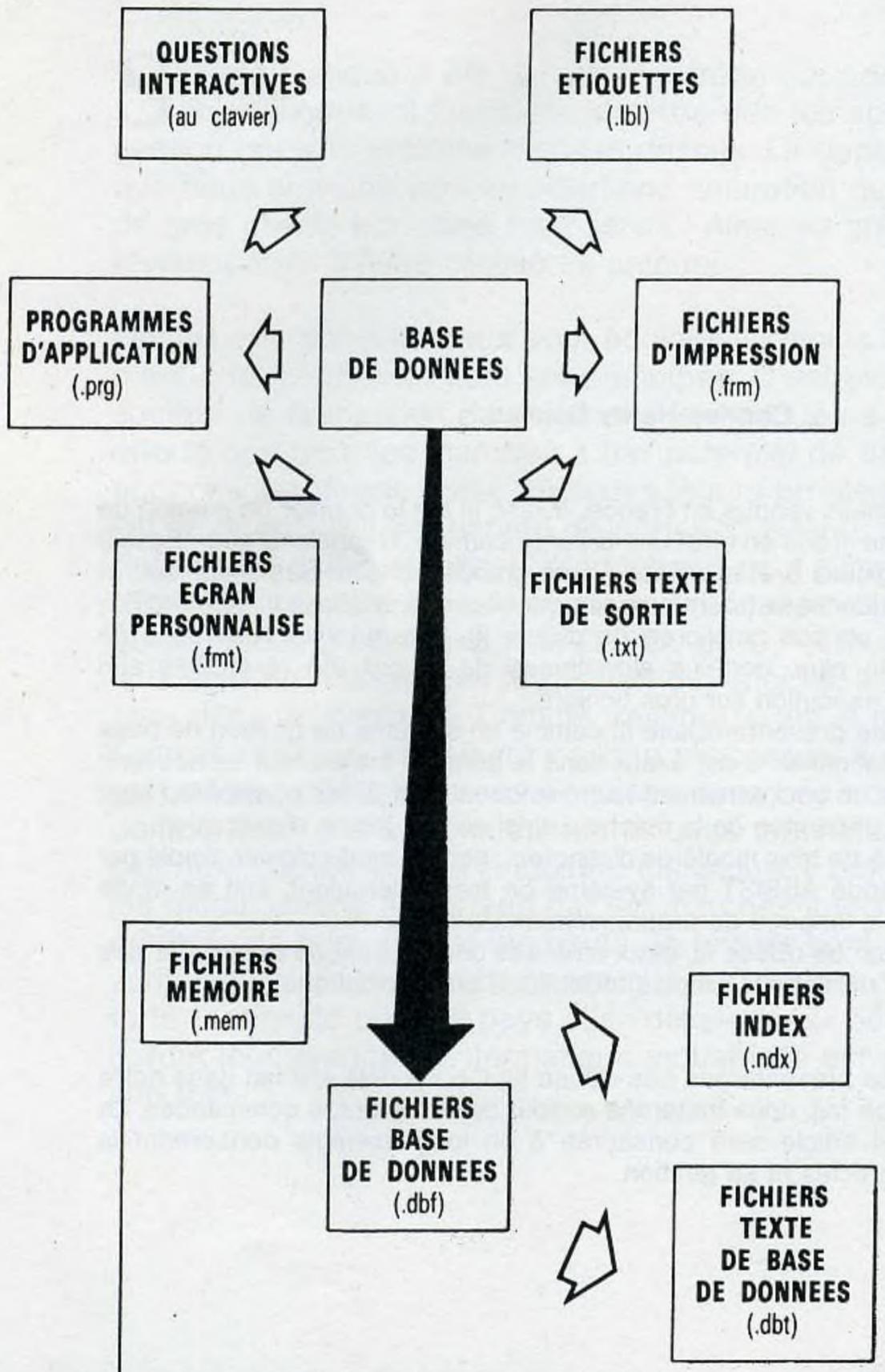
Dans cet article, nous ne présenterons pas dBase III. Ceci a déjà été fait dans notre n° 20 de mai 1985. De ce fait, nous traiterons surtout des différentes commandes. La deuxième partie de cet article sera consacrée à un long exemple concernant la réalisation d'un fichier articles et sa gestion.

RAPPEL DES PRINCIPALES CARACTERISTIQUES DE DBASE III

1. Système de gestion de base de données de type relationnel.
2. Mode interactif ou mode programme.
3. 128 champs et 4 000 caractères maximum par enregistrement.
4. Nouveau type de données MEMO permettant aux enregistrements d'atteindre une longueur de plus de 500 000 caractères.
5. Un milliard d'enregistrements par fichier.
6. Utilisation possible et simultanée de 10 fichiers de données.
7. Tri ultra-rapide sur plusieurs champs.
8. Fichier indexé, recherche très rapide, etc.
9. Langage très complet, avec possibilité d'utiliser des procédures et des passages de paramètres.

UN SYSTEME DE GESTION DE BASES DE DONNEES

dBase III est architecturé autour d'un certain nombre de fichiers réalisant une base de données.



LES FICHIERS DBASE III

dBase III sauvegarde les informations sur des fichiers disques de neuf formats différents. Chacun répond à un besoin particulier.

a. Fichier base de données

Les fichiers de base de données stockent les informations. Ils sont réalisés sous forme d'enregistrements et de champs. Chaque enregistrement contient un seul ensemble d'informations.

b. Fichiers MEMO

Les fichiers MEMO sont des fichiers auxiliaires des fichiers de données. Ils sont utilisés pour stocker les informations de champs MEMO. dBase III peut contenir un champ MEMO pour chaque enregistrement. Il s'agit en fait d'un texte se rattachant à un enregistrement. Ce texte peut servir à stocker des informations autres que les champs usuels.

c. Fichiers index

Les fichiers index permettent d'utiliser un fichier de données dans un ordre logique différent de l'ordre physique. L'ordre physique correspond à l'ordre d'enregistrement lors de la saisie. L'ordre logique peut être un ordre alphabétique ou numérique, basé sur le contenu d'un ou de plusieurs champs.

d. Fichiers commande

Ils contiennent les programmes de gestion ou d'application qui ont été créés grâce à l'éditeur de dBase III. Ils sont réalisés avec la commande MODIFY COMMAND.

e. Fichiers format

Les fichiers format permettent de créer des écrans de saisie personnalisés et des éléments de sortie personnalisés sur imprimante.

f. Fichiers étiquettes

Ceux-ci contiennent les informations nécessaires pour remplir des étiquettes. En effet, grâce à la commande LABEL, il est possible d'imprimer les informations sur un format d'étiquette.

g. Fichiers mémoire

Les fichiers mémoire peuvent contenir jusqu'à 256 variables mémoire qui sont stockées dans une mémoire tampon et peuvent être utilisées par plusieurs programmes ou applications à tout instant. Il s'agit, en d'autres termes, de variables communes.

h. Fichiers d'impression

Les fichiers d'impression contiennent les informations nécessaires à la commande REPORT. Cette dernière autorise la confection de formats d'impression prédéterminés, par exemple une édition de type facture ou formulaire pré-établi.

i. Fichiers texte

Les fichiers texte sont sous format ASCII. Il s'agit uniquement de caractères imprimables. Ils servent d'interface entre dBase III et d'autres programmes utilisateurs.

LA COMMANDE ASSIST

Parmi les trois modes d'utilisation de dBase III, le mode Assist est très intéressant. En effet, en mode simple (interrogation à partir du clavier), il convient de bien connaître toutes les commandes de dBase III ainsi que leurs utilisations. En mode programmé, il faudra bien sûr écrire ses programmes. Le mode Assist simplifie la tâche. Il pourra être utilisé pour des applications simples. Dans ce cas, l'utilisation des fichiers se fait en mode interactif à l'aide de menus déroutants. De plus en plus de logiciels utilisent ce mode de fonctionnement. Le mode Assist est activé en tapant ASSIST puis validation. Chaque partie du logiciel possède un système de menus arborescents. Le déplacement d'un menu à l'autre et le pointage d'une commande se font à l'aide des touches de déplacement du clavier (flèches) et de la touche validation (ENTER). Il est ainsi possible d'accéder à toutes les commandes et autres fonctions.

LES PRINCIPALES COMMANDES DE DBASE III

USE : La commande USE indique à dBase III le fichier que l'on désire utiliser. Ex. : USE NOMS : utiliser le fichier NOMS.

DISPLAY : Cette commande permet de visualiser le contenu du premier enregistrement.

| | |
|-------------------|--|
| DISPLAY Next 4 | : Il s'agit d'une extension de la commande DISPLAY. Ceci signifie visualiser du premier enregistrement jusqu'au quatrième. |
| CLEAR | : Efface l'écran. |
| LIST | : Lister tous les enregistrements. |
| DISPLAY au | : Idem à LIST, mais ici il faudra appuyer à la fin de chaque page écran sur ENTER pour voir la page suivante. |
| DISPLAY STRUCTURE | : Cette commande indique la structure d'un fichier : <ul style="list-style-type: none"> - le nom et le type de champ - la taille de chaque champ - le nombre d'enregistrements - la date de la dernière mise à jour. |
| DISPLAY RECORD 10 | : Affiche le contenu de l'enregistrement n° 10. |
| GOTO 22 | : Aller à l'enregistrement n° 22. |
| QUIT | : Quitter dBase III. |
| HELP | : Appel du menu d'aide. |
| APPEND | : Commande pour ajouter un enregistrement. |
| EDIT 12 | : Cette commande permet d'aller à l'enregistrement n° 12 pour effectuer une modification. Un menu d'aide et de commandes apparaît en haut de l'écran. |
| BROWSE | : Cette commande permet d'accéder en mode modification sur tout le fichier. |
| DELETE RECORD 5 | : Détruire l'enregistrement n° 5 (destruction logique). |
| RECALL ALL | : Rappeler les enregistrements détruits par DELETE. |
| PACK | : Détruire définitivement les enregistrements détruits par DELETE. Il s'agit ici d'une destruction physique. |
| CLEAR ALL | : Cette commande ferme tous les fichiers ouverts. |

CREATION D'UN FICHIER

La création d'un fichier peut se faire soit en mode ASSIST, soit grâce à la commande CREATE. Cette dernière appelle une routine de création qui contient tous les ordres pour la création d'un fichier.

OPERATEURS RELATIONNELS ET LOGIQUES

Opérateurs relationnels

- = égal à
- < inférieur à
- > supérieur à
- <= inférieur ou égal à
- >= supérieur ou égal à
- < > différent de

Opérateurs logiques

- AND et logique
- OR ou logique
- NOT non logique

La commande FOR

La commande FOR permet de donner une ou plusieurs conditions aux commandes DISPLAY et LIST.

Ex. : 1. DISPLAY NOM FOR NOM < ' N '

Afficher les noms dont le premier caractère est supérieur à N.

2. DISPLAY NOM, LOYER FOR LOYER < = 8000

Afficher les noms et les loyers pour chaque enregistrement dont le champ loyer est inférieur ou égal à 8 000 F.

LE TRI

Le tri rapide se fait en deux temps. Il convient de réaliser un double au fichier que l'on veut trier. Puis, dans un second temps, le tri est effectué en partant des données du fichier maître vers les fichiers triés.

La commande est ;

SORT ON < liste de champs > TO < nouveau fichier >

INDEXATION

Dans le cas d'un fichier trié, le fait d'ajouter des enregistrements a toutes les chances de mettre un terme au tri préalablement effectué. Il existe pour cela une autre solution : l'indexation. Un fichier d'indexation est un fichier qui reprend l'ordre dans lequel les enregistrements devront être relus. En d'autres termes, un champ d'un enregistrement se verra affecter d'un numéro d'ordre de classement. Ainsi, l'ensemble du fichier n'a-t-il pas besoin d'être entièrement réorganisé à chaque modification. Il y a donc le fichier principal et un fichier simplifié d'indexation.

Commande :

```
USE NOMS
INDEX ON NOM TO FAMILLE
```

Cela signifie : - utiliser le fichier NOMS
- indiquer le champ NOM sur FAMILLE.

RECHERCHE RAPIDE D'UNE INFORMATION (FIND)

Grâce à l'indexation, il est possible de retrouver rapidement une information.

Commande :

```
SET INDEX TO FAMILLE
FIND ROUX
```

Cela signifie : - se positionner sur l'index FAMILLE
- chercher le nom ROUX.

FONCTIONS SUR LES CHAINES DE CARACTERES

+ Concaténation de chaînes ou de champs.
LEN Evaluation de la longueur d'une chaîne ou d'un champ.
UPPER Conversion de minuscules en majuscules.
TRIM Elimination des espaces de la fin de la chaîne.

LA COMMANDE ERASE

Cette commande permet d'effacer un fichier.

REALISATION D'UN PROGRAMME

Grâce à l'éditeur de texte de dBase III, il est possible d'écrire des programmes d'application partant des fichiers dBase III.

Le langage de programmation fourni avec le progiciel est assez puissant et bien structuré.

En deuxième partie de cet article, nous présentons un programme de gestion d'adresses sous dBase III. La structure du fichier adresses est la suivante :

```
. USE ADRESSE
. DISPLAY STRUCTURE
Structure for database: C:ADRESSE.dbf
Number of data records:      0
Date of last update   : 01/01/80
Field  Field Name  Type      Width  Dec
  1  NOM           Character  20
  2  PRENOM        Character  30
  3  ADRESSE       Character  40
  4  CODE          Character   5
  5  VILLE         Character  30
  6  TEL           Character  20
  7  DIVERS        Character  70
** Total **                216
```

Tous les champs sont de type caractère.

Le programme est architecturé en deux parties :

- le programme principal,
- les procédures de gestion des enregistrements.


```

SET COLOR TO 7/0
@ 23,18 SAY 'Annuler'
SET COLOR TO 0/7
DO annule
SET COLOR TO 0/7
@ 23,18 SAY 'Annuler'
CASE reponse='R'
SET COLOR TO 7/0
@ 23,26 SAY 'Rechercher'
SET COLOR TO 0/7
DO efface
DO cherche
SET COLOR TO 0/7
@ 23,26 SAY 'Rechercher'
CASE reponse='S' .AND. .NOT. EOF()
SET COLOR TO 7/0
@ 23,37 SAY 'Suivant'
SET COLOR TO 0/7
SKIP
DO affiche
SET COLOR TO 0/7
@ 23,37 SAY 'Suivant'
CASE reponse='P' .AND. .NOT. BOF()
SET COLOR TO 7/0
@ 23,45 SAY 'Prudent'
SET COLOR TO 0/7
SKIP-1
DO affiche
SET COLOR TO 0/7
@ 23,45 SAY 'Prudent'
CASE reponse='E'
SET COLOR TO 7/0
@ 23,55 SAY 'Edition'
SET COLOR TO 0/7
DO efface
DO edite
SET COLOR TO 0/7
@ 23,55 SAY 'Edition'
CASE reponse='I'
SET COLOR TO 7/0
@ 23,63 SAY 'Index'
SET COLOR TO 0/7
DO efface
SET COLOR TO 0/7
@ 00,00 SAY SPACE(79)
@ 00,01 SAY 'Ractualisation du fichier en cours...'
PACK
SET COLOR TO 0/7
@ 23,63 SAY 'Index'
SET COLOR TO 7/0
@ 00,00 SAY SPACE(79)
CASE reponse='Q'
SET COLOR TO 7/0
@ 23,69 SAY 'Quitter'
CLOSE PROCEDURE

```

```

        CLOSE DATABASES
        SET COLOR TO 7/0
        CLEAR
        RETURN
    OTHERWISE
        ?? CHR(7)
    ENDCASE
    SET COLOR TO 7/0
    @ 23,77 SAY ' '
ENDIF
ENDDO

```

BIBLIOTHEQUE DE PROCEDURES

```

* Nom.....: ADRESSE.prc
* But.....: Procdures du fichier ADRESSE
* Date.....: Le 31/ 3/1987
* Auteur...: GENCODE (Gnrateur de programmes dBASE III)

```

PROCEDURE cree

```

    STORE 'N' TO present
    SET COLOR TO 7/0,7/0
    STORE SPACE( 15) TO xnom
    @ 4, 4 GET xnom
    READ
    IF xnom=' '
        @ 4, 4 SAY SPACE( 15)
        RETURN
    ELSE
        STORE UPPER(xnom) TO xnom
        CLEAR GETS
        DO store
        APPEND BLANK
        DO saisie
        STORE 'O' TO present
    ENDIF
    RETURN

```

PROCEDURE cherche

```

    STORE 'N' TO present
    SET COLOR TO 7/0,7/0
    STORE SPACE( 15) TO xnom
    @ 4, 4 GET xnom
    READ
    IF xnom=' '
        @ 4, 4 SAY SPACE( 15)
        RETURN
    ELSE
        STORE UPPER(xnom) TO xnom
        CLEAR GETS
        FIND '&xnom'
        IF .NOT. EOF()
            DO AFFICHE
            STORE 'O' TO PRESENT
        ELSE
            SET COLOR TO 0/7
            @ 00,00 SAY SPACE(79)
            @ 00,01 SAY 'FICHE INCONNUE: Pressez une touche...'
        ENDIF
    ENDIF

```

```

        SET CONSOLE OFF
        WAIT
        SET CONSOLE ON
        SET COLOR TO 7/0
        @ 00,00 SAY SPACE(79)
    ENDIF
ENDIF
RETURN
PROCEDURE annule
    SET COLOR TO 0/7
    @ 00,00 SAY ' Etes-vous sur de vouloir supprimer cet'
    @ 00,40 SAY 'te fiche (O/N):'
    @ 00,55 SAY ' '
    STORE ' ' TO reponse
    SET CONSOLE OFF
    WAIT TO reponse
    SET CONSOLE ON
    STORE UPPER(reponse) TO reponse
    @ 00,56 SAY reponse
    IF reponse='O'
        @ 00,59 SAY 'Confirmez (O/N): '
        STORE ' ' TO reponse
        SET CONSOLE OFF
        WAIT TO reponse
        SET CONSOLE ON
        STORE UPPER(reponse) TO reponse
        @ 00,76 SAY reponse
        IF reponse='O'
            DELETE
            STORE 'N' TO present
            DO EFFACE
        ENDIF
    ENDIF
ENDIF
SET COLOR TO 7/0
@ 00,00 SAY SPACE(79)
RETURN
PROCEDURE edite
    STORE 'N' TO present
    @ 23,03 SAY SPACE(73)
    @ 23,03 SAY '(1)=Edition par REPORT (2)=Edition par LABEL (3)=Fin'
    @ 23,57 SAY ' Votre choix:'
    DO WHILE .T.
        @ 23,75 SAY 'O'
        STORE INKEY() TO reponse
        IF reponse < 49 .OR. reponse > 51
            SET COLOR TO 0/7
            @ 23,75 SAY '±'
            SET COLOR TO 7/0
            LOOP
        ELSE
            STORE CHR(reponse) TO reponse
            @ 23,75 SAY reponse
            DO CASE
                CASE reponse = '1'
                    @ 23,03 SAY SPACE(73)
                    @ 23,03 SAY 'Donnez le nom du fichier REPORT: ± ±'

```

```

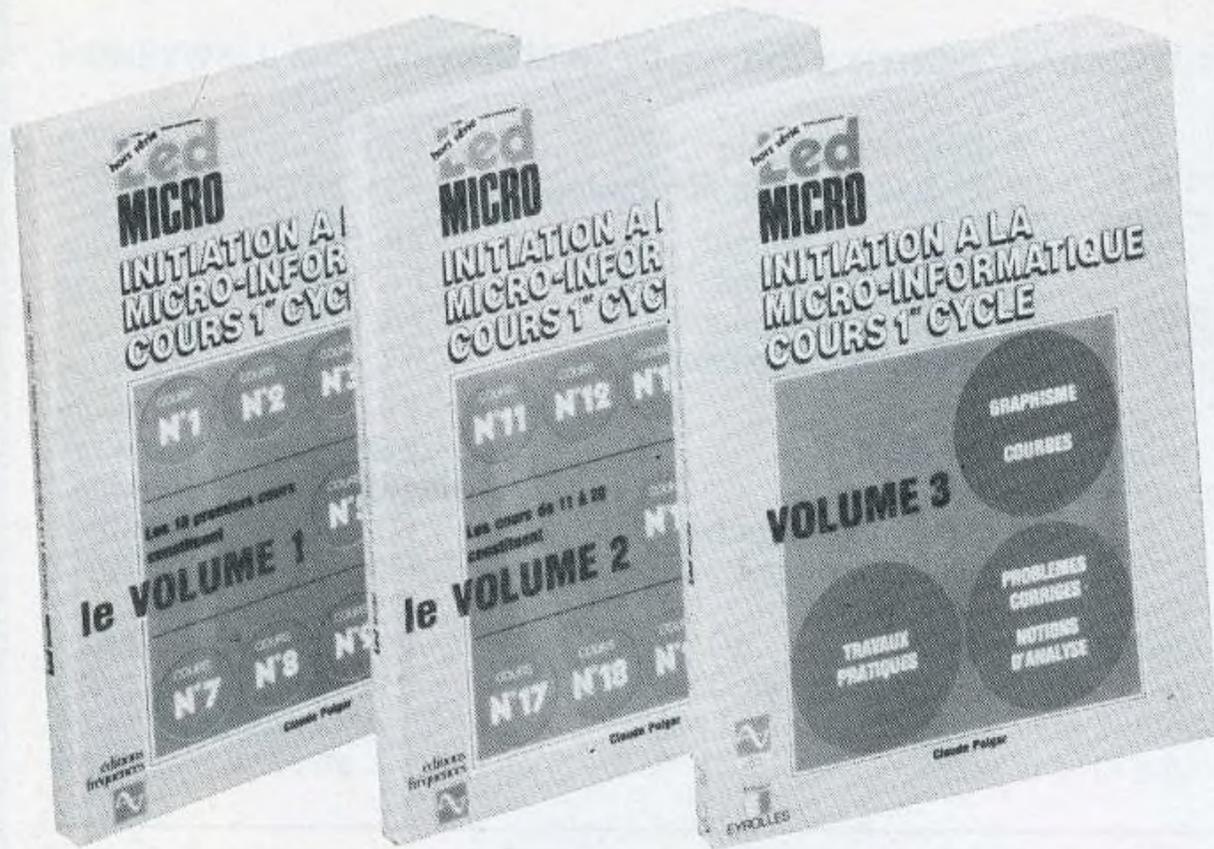
SET COLOR TO 7/0,7/0
STORE SPACE(8) TO fichier
@ 23,37 GET fichier
READ
CLEAR GETS
IF fichier # ' '
    STORE TRIM(fichier)+'.frm' TO fichier
    IF .NOT. FILE('&fichier')
        @ 23,03 SAY "Ce fichier n'existe pas ou ne se "
        @ 23,36 SAY 'trouve pas dans le rpertoire courant...'
        SET CONSOLE OFF
        WAIT
        SET CONSOLE ON
    ELSE
        SET CONSOLE OFF
        REPORT FORM &fichier TO PRINT NOEJECT
        SET CONSOLE ON
    ENDIF
ENDIF
EXIT
CASE reponse = '2'
@ 23,03 SAY SPACE(73)
@ 23,03 SAY 'Donnez le nom du fichier LABEL: ± ±'
SET COLOR TO 7/0,7/0
STORE SPACE(8) TO fichier
@ 23,37 GET fichier
READ
CLEAR GETS
IF fichier # ' '
    STORE TRIM(fichier)+'.lbl' TO fichier
    IF .NOT. FILE('&fichier')
        @ 23,03 SAY "Ce fichier n'existe pas ou ne se "
        @ 23,36 SAY 'trouve pas dans le rpertoire courant...'
        SET CONSOLE OFF
        WAIT
        SET CONSOLE ON
    ELSE
        SET CONSOLE OFF
        LABEL FORM &fichier TO PRINT
        SET CONSOLE ON
    ENDIF
ENDIF
EXIT
CASE reponse = '3'
EXIT
ENDCASE
ENDIF
ENDDO
SET COLOR TO 0/7
@ 23,03 SAY 'CrerÜ'
@ 23,09 SAY 'ModifierÜ'
@ 23,18 SAY 'AnnulerÜ'
@ 23,26 SAY 'RechercherÜ'
@ 23,37 SAY 'SuivantÜ'
@ 23,45 SAY 'PrudentÜ'

```

```

@ 23,55 SAY 'Edition0'
@ 23,63 SAY 'Index0'
@ 23,69 SAY 'Quitter0'
SET COLOR TO 7/0
@ 23,77 SAY ' '
RETURN
PROCEDURE saisie
SET COLOR TO 7/0,7/0
@ 4,38 GET xrenom
@ 7, 8 GET xadresse
@ 9,12 GET xcp
@ 9,29 GET xville
@ 11, 4 GET xtel
@ 13, 7 GET xdivers
READ
CLEAR GETS
REPLACE nom          WITH xnom
REPLACE prenom       WITH xrenom
REPLACE adresse      WITH xadresse
REPLACE cp           WITH xcp
REPLACE ville        WITH xville
REPLACE tel          WITH xtel
REPLACE divers       WITH xdivers
RETURN
PROCEDURE affiche
SET COLOR TO 7/0
@ 13, 7 SAY divers
@ 11, 4 SAY tel
@ 9,29 SAY ville
@ 9,12 SAY cp
@ 7, 8 SAY adresse
@ 4,38 SAY prenom
@ 4, 4 SAY nom
IF nom<>' '
    STORE '0' TO PRESENT
ENDIF
RETURN
PROCEDURE efface
SET COLOR TO 7/0
@ 13, 7 SAY SPACE( 20)
@ 11, 4 SAY SPACE( 11)
@ 9,29 SAY SPACE( 10)
@ 9,12 SAY SPACE( 5)
@ 7, 8 SAY SPACE( 20)
@ 4,38 SAY SPACE( 15)
@ 4, 4 SAY SPACE( 15)
STORE 'N' TO present
RETURN
PROCEDURE store
STORE SPACE( 15) TO xrenom
STORE SPACE( 20) TO xadresse
STORE 0 TO xcp
STORE SPACE( 10) TO xville
STORE SPACE( 11) TO xtel
STORE SPACE( 20) TO xdivers
RETURN

```

**Le cours
d'initiation
le plus
complet
+ de 700 pages**

Non, on ne s'initie pas à la micro-informatique en 5 leçons !

Si vous croyez au Père Noël vous pouvez espérer apprendre l'Informatique en lisant les innombrables «Cours de BASIC pour débutants» qui ont poussé comme des champignons dans les années 1980. Votre ordinateur risque de finir ses jours au-dessus de votre armoire.

Mais si vous voulez vraiment apprendre à programmer il faut avoir le courage de commencer par A pour arriver à Z. Programmer est un loisir intelligent et peut devenir un métier passionnant, mais l'étude de la programmation nécessite un minimum de travail et de méthode.

Etre sérieux – c'est le pari que fit la revue LED-MICRO en publiant à partir de 1985 les 20 premiers cours de C. Polgar. Plus de 40 000 lecteurs les ont suivis. Ce succès nous a conduit à demander à C. Polgar de remettre son cours à jour et de le compléter. Le résultat : un ouvrage épais (3 tomes, plus de 700 pages format 21 x 27), permettant d'acquérir agréablement des connaissances solides.



Diffusion auprès des libraires assurée exclusivement par les Editions Eyrolles.

Initiation à la micro-informatique C. Polgar

Bon de commande à retourner aux Editions Fréquences
1, boulevard Ney 75018 Paris.

Je désire recevoir le tome 1 140 F (130 F + 10 F de frais de port)
le tome 2 140 F (130 F + 10 F de frais de port)
le tome 3 200 F (190 F + 10 F de frais de port)

Ci-joint mon règlement par :

CCP Chèque bancaire Mandat

Nom

Prénom

Adresse

Code postal Ville

Une seule
parmi près de 600 lettres
de lecteurs :

J'enseigne les mathématiques dans une Université de Sciences Humaines et j'ai été amenée, alors que n'avais moi-même reçu aucune formation à la micro-informatique, à initier des étudiants de 1^{re} année de Mathématiques et Sciences Sociales (MASS) à la programmation en S-BASIC (sur Goupil-3), dans le but de faire avec eux de l'analyse numérique élémentaire. Ce que j'ai fait, tant bien que mal, cette année, en collaboration avec deux autres collègues. Nous sommes conscientes d'avoir commis un certain nombre d'erreurs pédagogiques et nous souhaitons tenter d'y remédier l'an prochain.
J'ai découvert votre revue tout récemment, alors que j'arrivais quasiment au bout de mon enseignement. J'ai été très sensible à votre démarche pédagogique et je me sens personnellement tout à fait en accord avec votre manière de procéder. Je me suis procurée l'ensemble des n^{os} de la revue et me permettrai de puiser dans votre cours certains exemples ou certaines façons de présenter les choses l'an prochain. Donc merci à vous...
C.L. St Cloud, le 22/5/85

Initiation à la Micro-Informatique 1^{er} Cycle Tome 3 (enfin paru !)

3.16 (Suite et fin) L'affichage

- ★ Etude des instructions permettant d'effectuer des présentations « évoluées » : PRINT TAB - PRINT USING - LOCATE - COLOR en mode texte.
- ★ Présentation en tableaux de toutes sortes grâce à la pratique des opérateurs MODULO et DIVISION ENTIÈRE.
- ★ Beaucoup de programmes utilisent des assemblages de ces instructions et opérateurs... dont la combinaison n'est pas toujours facile.

3.17 Compléments

- ★ Etude des dernières instructions, fonctions et variables du cycle 1 : FILES, KILL, AUTO, ON ERROR GOTO, RESUME, ERR, ERL, DELETE, EDIT, RENUM TRON, TROFF, STOP, CONT, KEY ON, KEY OFF, FIX, BEEP.
- ★ Compléments de cycle 1 qui sont maintenant accessibles aux élèves : sur la précision et les erreurs dues à l'arrondi, sur la sélection, les boucles.

3.18 Graphisme

- ★ Une étude complète et détaillée sur les instructions graphiques en haute résolution : SCREEN, PSET, PRESET, STEP, LINE, CIRCLE, COLOR, POINT, PAINT, sans éluder aucune des difficultés et « pièges » classiques : l'incrustation de texte dans le dessin, les « bavures » dues au PAINT mal utilisé.
- ★ Une étude détaillée du langage graphique DRAW, avec ses subtilités et ses pièges (sous-chaînes X, paramètres variables dans le DRAW, etc.).
- ★ De nombreux exercices avec leurs solutions (80) et leurs illustrations sur des photos d'écran en couleur (48 photos).

3.19. Dessin des courbes

- ★ Un chapitre séparé du graphisme général (chapitre 3.18) de façon à ce que les « non matheux » puissent le sauter sans remords : ils ne seront pas punis !
- ★ Pour les matheux : une excellente révision et illustration des courbes de toutes sortes : $Y = f(x)$, courbes paramétrées, courbes en coordonnées polaires, avec des exemples utiles : courbes d'amortissement, astroïde, cardioïde, décomposition d'une fonction périodique par une série de Fourier.

3.20. Révision générale

- ★ L'enchaînement des notions selon l'ordre « pédagogique » qui a été utilisé jusqu'ici est bien différent de l'ordre « logique ». Autant qu'un cours d'anglais suit un ordre différent de celui (plus logique !) d'une grammaire anglaise.
- ★ Tout ce qui a été enseigné jusqu'ici résumé en 30 pages. Une référence pour retrouver la notion dont on a besoin à travers le cours et ses exercices. Mais aussi une réflexion sur la structure d'un langage informatique, d'où une préparation à la lecture des cours de PASCAL (par exemple !).

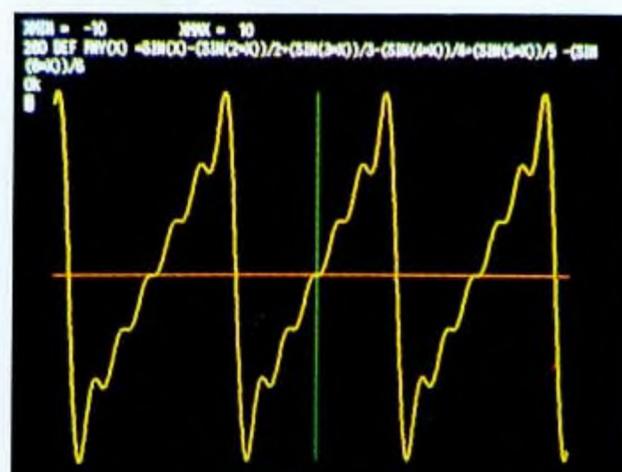
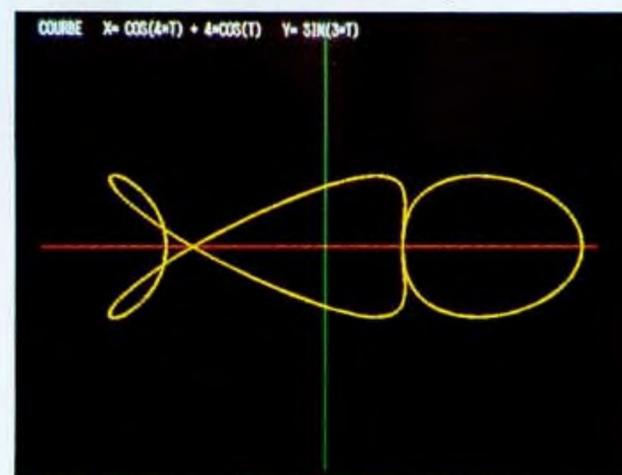
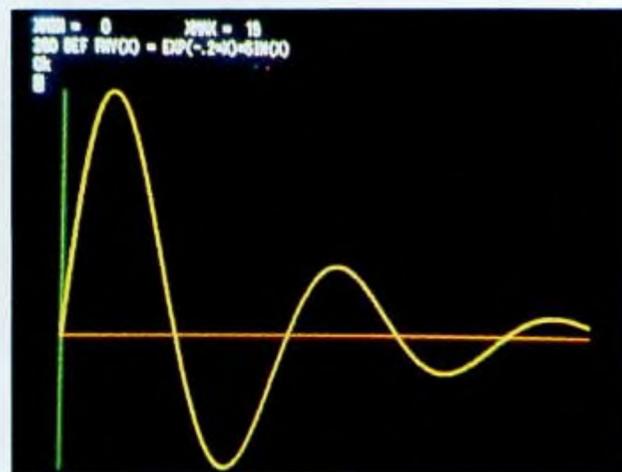
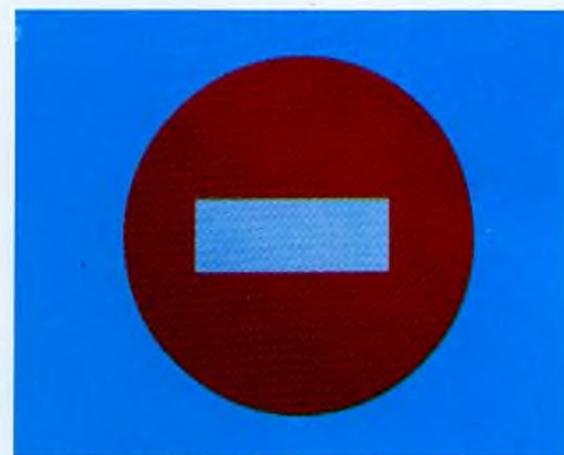
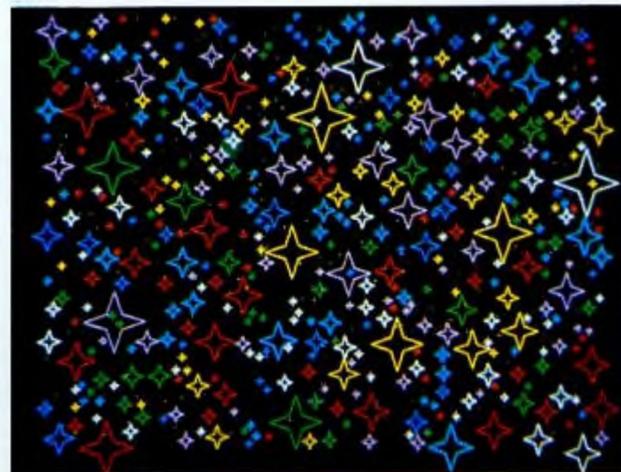
3.21. Techniques de mise au point

- ★ Les outils de base : étude des éditeurs de texte, connaissance et interprétation des messages d'erreur.
- ★ Comment rechercher et corriger ses erreurs.
- ★ La représentation du dialogue homme-machine, pour noter l'expérience que vous acquérez par la pratique.

3.22. Problèmes de synthèse - Notions d'analyse

C'est à la fois la conclusion, la partie la plus originale et la plus utile de ce cours. L'auteur ne se contente pas de fournir une liste de problèmes avec leur solution : il se met à la place du programmeur débutant en essayant de décortiquer le « processus de réflexion » qui fait passer de l'énoncé d'un problème à sa solution : une initiation pratique à l'analyse.

1 livre broché de 248 pages pages 21 x 27, dont 8 pages en couleur



nouveau!



- *exploiter toutes les possibilités des systèmes MIDI*
- *réaliser vous-mêmes un clip vidéo*
- *tirer le maximum de vos synthétiseurs*
- *installer chez vous votre studio d'enregistrement*
- *tout savoir sur les nouveautés musique et vidéo créatives*

**Tout cela chaque mois
dans Music Vidéo Systèmes**

une publication des Editions Fréquences chez votre marchand de journaux

Editions Fréquences 1, boulevard Ney 75018 Paris - Tél. 46.07.01.97