hors série

Led MICRO

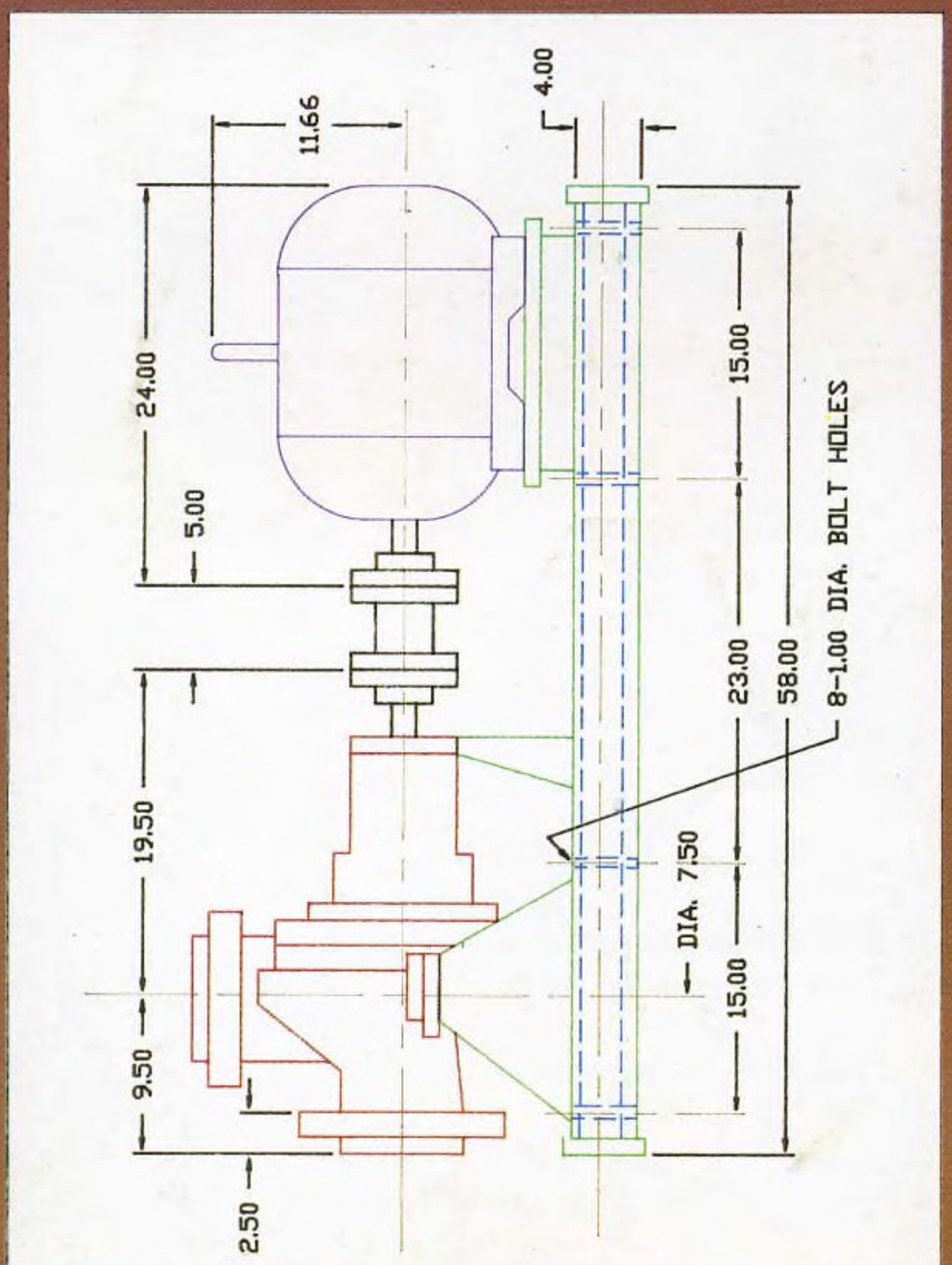
PROGRAMMATION
COURS 2ème CYCLE

COURS
N°37
Suite
2e cycle
N°17

COURS DE PASCAL
les sous-programmes

COURS DE PROGRAMMATION APPROFONDIE :
éditeur de texte

LE TURBO PASCAL
et ses outils de programmation



ISSN 0757-6889

M 1988 - 37 - 18,00 F
3791988018008 00370

MENSUEL FEVRIER 1987 - BELGIQUE 139 FB/CANADA 3,75 $/SUISSE 6,75 FS

VOYAGE AU CŒUR 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.

hors série
Led MICRO
PROGRAMMATION COURS 2e 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 2e 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 0757-6889

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, à la revue
Tél. : 607.01.97
Secrétaire responsable
Annie Perbal

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

Réalisation
Composition-Photogravure
Edi'Systèmes

Impression
Berger-Levrault - Nancy

COURS N°37 Suite 2e cycle N°17

FEVRIER 87

COURS DE PROGRAMMATION APPROFONDIE
de la page 13 à la page 21
– Première partie de l'étude d'un éditeur de texte ..... p. 14
  • Le programme
  • Les variables globales
– Le jeu interactif ..... p. 18
– Déplacement à l'écran ..... p. 18
  • Le tracé des formes
  • L'interaction du mouvement et de l'affichage
– Conclusion ..... p. 21
Dominique Chastagnier
Jean-François Coblentz
Patrick Gueneau

DIALOGUE AVEC NOS LECTEURS
de la page 22 à la page 31
Les problèmes de chameaux .. p. 22
– Les carrés bi-magiques ..... p. 27

C'EST ARRIVÉ DEMAIN
de la page 32 à la page 34

LE TURBO PASCAL
de la page 38 à la page 49
– Présentation ..... p. 39
– Introduction à Turbo Pascal ... p. 39
  • Les éléments de base du langage
  • Les performances
  • Turbo Pascal et MS-DOS
  • Les options de compilation
– Turbo Editor ..... p. 42
– Tool box ..... p. 43
– Graphix Tool box ..... p. 44
Charles-Henry Delaleu

COURS DE PASCAL
de la page 4 à la page 12
– Retour sur les sous-programmes p. 5
  • Gérer les variables
  • Cas des fonctions
– Un exemple déjà rencontré ..... p. 9
– Conclusion ..... p. 12
Dominique Chastagnier
Jean-François Coblentz
Patrick Gueneau

NOTRE COUVERTURE : Dessin réalisé par Autocad. Ce progiciel de D.A.O. est l'un des plus utilisés de nos jours. Il fera l'objet d'une description dans notre prochain numéro.

COURS DE PASCAL

Dominique Chastagnier
Jean-François Coblentz
Patrick Gueneau

Pour vous permettre de souffler un peu, nous terminons gentiment les notions des mois précédents, et nous les mettons en pratique sur un exemple complet, détaillé et commenté. Les bases fondamentales de ce langage puissant doivent être bien comprises pour aborder la suite du cours.

COURS N° 8

PLAN DU COURS

1. Introduction
2. Retour sur les sous-programmes
   2.1. Gérer les variables
   2.2. Cas des fonctions
3. Un exemple déjà rencontré
4. Conclusion

1. INTRODUCTION

Le mois dernier, une étape psychologique a été franchie, celle qui vous permettra enfin de faire du Pascal proprement. Nous avons traité des procédures et fonctions. Ce mois-ci, nous allons finir ce chapitre et traiter en détail un exemple. Pour une fois, nous commenterons abondamment le programme proposé. Vous pouvez être reconnaissant, car il est banal que nous ne fassions pas ce que nous recommandons hautement. En clair, c'est bien la première fois que nous mettrons des commentaires dans un programme, mais il ne faut pas le dire. Surtout, souvenez-vous que les commentaires ne prennent pas de temps en Pascal, puisque le compilateur les ignore, ce qui permet d'obtenir un code (le programme compilé) sans lourdeur inutile. Donc, mettez des commentaires !!!

2. RETOUR SUR LES SOUS-PROGRAMMES

Nous avons décrit la méthode de déclaration des sous-programmes. Nous avons dit qu'une fonction prend une valeur en elle-même. Nous avons indiqué la portée d'une variable dans une sous-structure. Maintenant, nous allons vous donner quelques principes acceptés généralement. Certains sont surtout utiles au début, les autres à tout instant.

2.1. Gérer les variables

Nous avons indiqué que les variables sont déclarées pour la structure qui la déclare (pléonasme ?), mais aussi pour celles qui sont incluses dans cette structure.

```
var
i:integer;

  procedure toto;
  begin
  for i:=1 to n
  do
  ...
  ...
  end;
begin
for i:=1 to 100 do
...
toto;
writeln;
end.
```

Dans l'exemple ci-dessus, la variable i est connue du programme principal et de la procédure toto, mais comment tester proprement sa valeur quand, à chaque appel, i est modifiée localement :

```
program  essai;
  var
    i  :  integer;

  procedure  toto;
  begin
    for i  :=  1  to  100  do
      writeln(i);
  end;
begin
  for i  :=  1  to  2  do
    toto;
end.
```

De telles variables sont dites globales. Il est cependant bon de limiter leur usage au strict minimum, comme par exemple dans le cas où la valeur de la variable dans le programme appelant est reprise dans le sous-programme. Sinon, il vaut mieux créer une nouvelle variable, même si son nom est proche de celui de la première. Par exemple :

```
program  essai-2;
  var
    i  :  integer;

  procedure  toto;
    var
      ii:integer;
  begin
    for ii  :=  1  to  100  do
      writeln(i,ii);
  end;
begin
  for i  :=  1  to  2  do
    toto;
end.
```

Ici, ii est la variable locale qui travaille comme i le faisait avant, mais i est maintenant utilisé localement avec sa valeur globale. Plus de confusion possible. Par contre, une autre configuration peut être source de confusion :

```
program  essai;
 var
  i  :  integer;

 procedure  toto;
  var
   i  :  integer;
 begin
  for i  :=  1  to  100  do
    writeln(i);
 end;
begin
 for i  :=  1  to  2  do
  toto;
end.
```

Ici, dans le sous-programme toto, i est redéfinie, et donc l'instruction d'écriture WRITELN imprime les nombres de 1 à 100, deux fois de suite, la variable i dans toto étant purement locale. Les nuances sont ici très importantes, vous le voyez.
Une bonne habitude est de nommer les variables similaires, au même niveau, de la même façon :

```
var
i:integer;

  procedure toto1;
  begin
  for ii:=1 to n
  do
  . . .
  . . .
  end;
  procedure toto2;
  begin
  for ii:=1 to n
  do
  . . .
  . . .
  end;
begin
for i:=1 to 100 do
. . .
toto1;
toto2;
writeln;
end.
```

Ici, dans toto1 et toto2, les variables ii jouent un rôle équivalent, donc nous reprenons leurs noms de manière identique, car toto1 et toto2 sont au même niveau d'imbrication, appelés par la même structure. Un autre exemple mettant en cause des variables passées par valeur :

```
var
i:integer;

  procedure toto1( i_main:integer);
  begin
  for ii:=1 to n do
  ...
  ...
  end;

  procedure toto2(i_main:integer);
  begin
  for ii:=1 to n do
  ...
  ...
  end;

begin
for i:=1 to 100 do
...
toto1(i);
toto2(i);
writeln;
end.
```

Encore une fois, à niveau d'imbrication égal, le même nom est repris, **i_main.**

2.2. Cas des fonctions

Pour les fonctions, les mêmes critères s'appliquent. Mais il faut faire d'autant plus attention que la valeur de la fonction est utilisée telle quelle dans le programme qui l'appelle. Il sera plus difficile de trouver des erreurs avec des fonctions.

3. UN EXEMPLE DEJA RENCONTRE

Vous vous souvenez peut-être du programme de suivi des notes d'une classe. Nous allons le reprendre, pour l'écrire en Pascal correct, cette fois-ci. Le listing du programme originel est dans le numéro 35 de LED-Micro, l'énoncé dans le numéro 34. Voici le programme remanié. Il est plus clair, et surtout plus agréable d'emploi. Il est fait usage de procédures, de fonctions, de variables locales et globales. Ne pas oublier qu'une variable modifiée au cours d'un sous-programme doit être passée par adresse, pour pouvoir être mise à jour, donc dans l'en-tête de la sous-structure, elle doit être précédée de VAR.

```
program etudiants (input, output);

 const
  nb_etud = 10;
  note_max = 20;
  note_min = 0;
  nb_result = 20;
  long_nom = 30;

 type
  nom = string[30];
   tab_note = array[1..nb_etud, 1..nb_result] of real;
   tab_nom = array[1..nb_etud] of nom;

 var
  fin : boolean;
  etud, note : integer;
  i, j, k : integer;
  t_note : tab_note;
  t_nom : tab_nom;
   moyenne_generale_classe, moy, moy_max : double;



procedure nombre_etud (var n : integer);
begin
  write('nombre d etudiants de la classe : ');
  readln(n);
end;


procedure nombre_note (var n : integer);
begin
  write('nombre de notes par etudiant de la classe : ');
  readln(n);
end;


procedure saisie_nom (var nm : tab_nom);
 var
   i : integer;

(* saisie des noms *)
 begin
  for i := 1 to etud do
   begin
    writeln;
    write(' nom de l etudiant : ');
     readln(nm[i]);
   end;
 end;
```

```
 procedure saisie_note (nm : tab_nom;
                        var nt : tab_note);
  var
   ii, jj : integer;
   ntt : real

(* saisie des notes *)
 begin
  writeln;
  for ii := 1 to etud do
   begin
    writeln(' pour l etudiant   ', nm[ii]);
    jj:= 1;
    repeat
     begin
       write(' saisie de la note', jj, ' : ');
       readln(ntt);
        if ((ntt > note_max) or (ntt<note_min)) then
         writeln(' erreur de frappe, retaper la note')
      else
       begin
         nt[ii,jj]:=ntt;
        jj:=jj+1;
      end;
     end;
   until jj:=note;
  end;
end;

procedure travail (tb_nom : tab_nom;
                   tb_note : tab_note);
 var
  travail_choisi, n_int : integer;
  n_et : string;
  fin : boolean;

 procedure choix (var n : integer);
  const
   tab = '     ';

  begin
   writeln(' choix du travail a effectuer.');
   writeln(' les possibilites sont :   ');
   writeln(tab, '1 : moyenne generale annuelle de la classe');
   writeln(tab, '2 : moyenne generale annuelle pour un etudiant');
   writeln(tab, '3 : moyenne de la classe sur une interrogation');
   writeln(tab, '4 : tri des meilleures moyennes generales annuelle');
   writeln(tab, '5 : fin du travail');
  end;

 function moyenne_generale_annuelle (n : tab_note) : real;
  var
   moy : real;
   ii, jj : integer;

(* moyenne annuelle *)
  begin
   for ii := 1 to etud do
    begin
     for jj := 1 to note do
     begin
      moy := moy + n[ii, jj];
     end;
    end;
    moyenne_generale_annuelle := moy / (etud * note);
  end;
```

```
   function moyenne_etud (nm : tab_nom;
                          nt : tab_note;
                          n_et : string) : real;
    var
      jj, ii : integer;
      moy : real;

(* pour un etudiant *)
   begin
    jj := 0;
    moy := 0;
    for ii := 1 to etud do
      if n_et = nm[ii] then
        jj := ii;
    if jj = 0 then
      writeln(' probleme de nom')
    else
      for ii := 1 to note do
        moy := moy + nt[jj, ii];
     moyenne_etud := moy / note;
   end;

    function moyenne_inter (nt : tab_note;
                            n_i : integer) : real;
    var
      moy : real;
      jj : integer;

(* pour une interrogation *)
   begin
    moy := 0;
    for jj := 1 to etud do
      moy := moy + nt[jj, n_i];
     moyenne_inter := moy / etud;
   end;


   procedure tri_moyennes (nm : tab_nom;
                           nt : tab_note);

    type
      tab_moy = array[1..nb_etud] of real;

    var
      ii, jj, kk : integer;
      moy_max : real;
      t_moy : tab_moy;


(* tri par moyenne des etudiants *)
   begin
    writeln;
    for ii := 1 to etud do
      t_moy[ii] := 0;
    for ii := 1 to etud do
     begin
      for jj := 1 to note do
         t_moy[ii] := t_moy[i] + nt[ii, jj];
       t_moy[ii] := t_moy[ii] / note;
     end;
    for ii := 1 to etud do
     begin
      moy_max := 0;
      for jj := 1 to etud do
       if t_moy[jj] > moy_max then
       begin
```

```
        moy_max := t_moy[jj];
        kk := jj;
      end;
      t_moy[kk] := -1;
      writeln(nm[kk], ' moyenne : ', moy_max);
    end;
  end;


begin
  fin:=true;
  repeat
   begin
    choix(travail_choisi);
   case travail_choisi of
   1 :
       writeln('moyenne : ', moyenne_generale_annuelle(tb_note));

   2 :
     begin
       write('nom de l etudiant : ');
       readln(n_et);
       write('moyenne de ', n_et,' : ');
        writeln( moyenne_etud(tb_nom, tb_nom, n_et));
     end;
   3 :
     begin
       write('numero de l interrogation : ');
       readln(n_int);
       write('moyenne de l interrogation ', n_int,'   :   ' );
        writeln(moyenne_inter(tb_note, n_int));
     end;
   4 :
      4 :
       begin
          writeln(' moyennes par ordre decroissant : ');
          tri_moyennes(tb_nom, tb_note);
       end;

       otherwise
       begin
         fin:=false;
       end;
     end;
    until fin=false;
  end;

begin
  nombre_etud(etud);
  nombre_note(note);
  saisie_nom(t_nom);
  saisie_note(t_nom, t_note);
  travail(t_note, t_nom);
end.
```

C'est plus propre, plus puissant. Chaque module est utilisable dans d'autres programmes, par simple déplacement.

4. CONCLUSION

Nous avons terminé là le cours sur les sous-structures, pour leur présentation générale tout au moins. Le mois prochain, un développement vraiment très important sera proposé, la notion de récursivité. Nous verrons, à l'aide de nombreux exemples les possibilités que cet outil permet, mais aussi les limites raisonnables à ne pas franchir. En attendant, nous vous proposons, comme exercice, de reprendre les énoncés de notre numéro 34 et de les programmer à l'aide de sous-structures, correctement, cette fois, puisque vous disposez de la possibilité de programmer de façon structurée.

COURS DE PROGRAMMATION APPROFONDIE

Dominique Chastagnier
Jean-François Coblentz
Patrick Gueneau

Ce mois-ci, suite de la série «analyse et programmation d'applications» qui servira de support au cours de programmation approfondie pour les mois à venir. Nous appliquons aujourd'hui les deux méthodes complémentaires d'analyse d'un projet telles que nous les avons décrites dans nos précédents cours, à savoir une méthode dite «globale» pour l'éditeur de texte, et une approche par «essai/erreur» (ou maquettage) pour le jeu interactif.

COURS Nº 17

PLAN DU COURS

1. Etude d'un éditeur de texte
   - Le programme
   - Les variables globales
     - le buffer
     - l'écran
     - caractères de contrôle
     - divers
2. Jeu interactif
   - Déplacement à l'écran
     - le tracé
     - l'interaction du mouvement et de l'affichage

Nous voilà en plein dans l'analyse de notre projet d'éditeur pleine page, et si nous ne sommes pas au bout de nos peines, vous vous apercevrez vite que nous arriverons en quelques mois à réaliser sans nous en rendre compte une application qui semblait délicate au premier abord. Il faut prendre l'habitude de ne pas pouvoir tout résoudre en quelques lignes savantes après dix minutes de réflexion. C'est pourquoi nous ne répondrons pas tout de suite à tous les problèmes qu'une telle application soulève, chaque chose en son temps !

PREMIERE PARTIE DE L'ETUDE D'UN EDITEUR DE TEXTE

Avant de commencer la programmation détaillée, nous allons décrire l'agencement du programme, en séparant les modules indépendants. Comme prévu, cette description respectera les règles syntaxiques du pseudo-langage que nous avons étudié au mois de décembre. Nous pourrons ainsi détailler des blocs de programmation, que nous analyserons à leur tour... ainsi de suite jusqu'à ce qu'on obtienne des blocs simples que l'on qualifiera d'élémentaires. Cela ne veut pas dire qu'ils correspondent à une instruction du langage, mais plutôt à une complexité de programmation suffisamment faible pour permettre une programmation aisée.

Le programme

On le découpe en trois parties :

- l'initialisation,
- le corps du programme,
- la fin de l'application.

Jusqu'ici, rien de bien nouveau. Voici l'écriture dans notre pseudo-langage du programme à ce niveau de l'analyse.

```
début
    INITIALISER ;      (appel à la procédure d'initialisation)
tant_que non(FIN)
    CORPS(FIN) ;       (appel à la procédure principale du programme)
                       (si l'utilisateur désire quitter, la variable FIN sera «vrai», sinon
                       elle vaudra «faux»)
fin_tant
    TERMINE ;          (appel à la procédure de fin de programme)
fin
```

Les procédures INITIALISER et TERMINE ne sont pas pour le moment les plus importantes, de plus elles dépendent d'une part du langage de programmation choisi, et d'autre part de l'ordinateur. Nous verrons aussi que, la plupart du temps, la seconde procédure est intégrée au bloc principal CORPS, sous forme d'une option parmi toutes les commandes que ce dernier est chargé d'interpréter.
En fait le plus souvent, CORPS gère un menu d'options visible ou non. Pour notre éditeur, nous avons décidé de passer directement en mode plein écran, mais il serait possible d'afficher au départ un menu général présentant les différents choix disponibles. Comme nos ambitions sont limitées, il y aura peu d'options ne faisant pas partie du mode pleine page ; ce menu est donc inutile.

Entrons maintenant plus dans le détail, avec l'examen de cette procédure CORPS. Nous prenons donc comme hypothèse que l'on dispose des commandes suivantes, codées sous forme de caractères de contrôle :

Insert	: le texte tapé est ajouté sur la ligne du curseur,
Surimp	: le texte tapé remplace le texte de la ligne en cours,
Destruc-Char	: le caractère à gauche du curseur est effacé,
Destruc-Ligne	: la ligne entière est détruite,
Mont	: le curseur monte d'une ligne sans affecter le texte,
Desc	: le curseur descend d'une ligne sans affecter le texte,
Gauche	: le curseur va à gauche d'un caractère,

Droite : le curseur va à droite d'un caractère,
Termi : l'édition du texte se termine.

Les autres commandes seront à ajouter en fonction de vos besoins et des améliorations que vous souhaitez apporter. On peut maintenant écrire schématiquement la procédure CORPS :

```
début procédure
    LITCAR(c) ;          (lecture d'un caractère c)
si «c'est un caractère de contrôle» alors
début_si
suivant la valeur de c (on teste les différentes valeurs de c)
    Insert        : INSERTION ;        (on se met en mode insertion)
    Surimp        : SURIMPRESSION ;    (on se met en mode remplacement)
    Destruc-car   : EFFACE ;           (on efface un caractère)
    Destruc-ligne : EFFACE_LIGNE ;     (on efface la ligne)
    Mont          : Monte ;            (on monte)
    .....
    .....
    Termi         : Fin = "vrai" ;     (on sort de la boucle principale)
    Autre         : ERREUR ;           (il n'y a pas de fonction
                                        correspondant à c)
fin_suivant
                        sinon
    INSERER(c) :     (on ajoute c à l'endroit du curseur)
fin_si
fin_procédure
```

Il faut donc maintenant définir chacun des sous-programmes. On s'aperçoit vite que ces blocs ne sont pas tout à fait indépendants. Ainsi par exemple, le passage du mode insertion au mode surimpression va directement influencer la façon dont la procédure INSERER va modifier la ligne en cours. Par ailleurs, la plupart de ces fonctions modifient le texte à l'écran.
Avant d'entrer plus dans les détails de fonctionnement, il nous faut étudier les variables principales qui vont, d'une part stocker les informations essentielles et, d'autre part permettre la synchronisation entre les procédures. C'est ce qu'on appelle plus communément les variables globales. De même, il est nécessaire de définir quelques constantes importantes qui vont faciliter la portabilité du programme d'un ordinateur à un autre. Nous en profiterons pour étudier des exemples de procédures d'initialisation en BASIC et en PASCAL.

Les variables GLOBALES :

a. Le buffer

C'est un tableau de chaîne de caractères, sa taille est définie par une constante, un élément du tableau égale une ligne de texte. On associe à ce tableau plusieurs variables :

- le nombre de lignes utilisées dans le buffer,
- le numéro de la première ligne du buffer correspondant à la première ligne de l'affichage.

b. L'écran

Nous avons besoin de deux variables de positionnement du curseur et pour définir les caractéristiques de l'écran, de deux constantes déterminant la longueur d'une ligne et

le nombre de lignes affichables.

c. Caractères de contrôle
Afin de rester le plus indépendant possible du contexte, il serait souhaitable de définir des constantes pour chaque type d'opération. Suivant les implantations, elles seront de type caractère (en PASCAL) ou entier (code ASCII en BASIC).
d. Divers
Enfin, on ajoutera le «Beep» sonore, utile en cas d'erreur (caractère de contrôle non affecté, arrivée en fin de ligne, etc.) ; on peut de cette façon paramétrer le curseur affiché, et le message de fin de buffer tel qu'il est présenté à la figure 1.
Voici donc deux exemples d'initialisation du contexte :

A) en PASCAL :

Les variables globales ainsi que la plupart des constantes sont définies dans le corps principal du programme. Cependant, pour les caractères non-affichables, le standard PASCAL ne permet pas d'écrire une constante sous la forme :

sonnerie = chr(7) ;

Il faut donc en passer par des variables globales.

```
program EDITEUR(input,output);

 const

      max_buffer    =2000;
      nb_ligne      =24;
      nb_col        =80;
      fin_de_buffer ='[FIN]';
      curseur       ='_';

 type

      ligne=string[nb_col]; {ou 'packed array[1..nb_col] of char' }

 var
      buffer :array[1..max_buffer] of ligne;
      sonnerie      :char;
      insert,surimp,
      destruc_car,
      ....,termi    : char;     { tous les codes contrôle }
      x_curseur,
      y_curseur     : integer;  { position du curseur }
      nb_occup,                 { nombre de lignes dans le buffer}
      prem_ligne    : integer;  { numéro de la 1ère ligne du  ""    }
      fin,                      { devient 'true' pour sortir}
      mode_ins      : boolean;  { ='true' si insertion,        }
                                {  'false' si remplacement }

 procedure INITIALISER;

  begin
       {définition des codes de contrôle }

      insert        :=chr(9);   { ctrl-I }
      surimp        :=chr(19);  { ctrl-S }
      destruc_char  :=chr(4);   { ctrl-D }
      :::::::::
      termi         :=chr(6);   {ctrl-F }

       {'beep' d'erreur }
```

```
      sonnerie      :=chr(7);   {ctrl-G }

        {initialisation des variables principales }

      x_curseur     :=1;        { le curseur se trouve }
                                { en haut de l'écran }
      y_curseur     :=1;
      nb_occup      :=0;
      prem_ligne    :=1;        { début du buffer }
      fin           :=false;
      mode_ins      :=true;     { par défaut on est en mode
                                  insertion }
      .....

{     il faut ensuite initialiser l'écran:
      effacement + affichage du symbole fin_de_buffer (en haut)
}
      PAGE(output); { correspond souvent à un effacement d'écran,
                      mais dépend de l'implémentation du PASCAL}
      writeln(fin de buffer);

end;{ fin de la procédure INITIALISER }
    ..../....
```

Passons maintenant à la version BASIC :

Puisqu'on ne dispose pas de constantes en BASIC, il faut les remplacer par des variables, comme on l'a fait pour certaines en PASCAL. Pour le reste, rien de bien différent du PASCAL, si ce n'est l'utilisation de variables entières à la place des variables booléennes, et bien sûr la notion fort différente de sous-programme par rapport à celle de procédure. Enfin, bien qu'elle puisse résider dans le sous-programme, les déclarations du buffer et des constantes associées seront plus logiquement effectuées au début du programme principal, donc en tête.

```
10 REM              EDITEUR plein ECRAN
15 REM              ==================
20 MAX=2000         : REM taille du buffer
30 DIM T$(MAX)      : REM déclaration une foix pour toute
40 NL=24:NC=40      : REM constantes dépendant de l'ordinateur
45                    REM (taille de l'écran)
50 GOSUB 1000       : REM appel au sous programme d'initialisation
60 ..........

     ::::::::

1000 REM INITIALISATION
1010 REM définition des constantes code de contrôle
1020 CI=9           :REM code ascii de CTRL-I, Insertion
1030 CS=19          :REM code ascii de CTRL-S,Surimpression
1040 CC=4           :REM code ascii de CTRL-D, Destruction caractère
  ::::::
1200 REM sonnerie ,curseur et message de fin de buffer
1205 CU$="_"
1210 SO$=CHR$(7)    :REM caractère CTRL-G
1215 MF$="[FIN]"
1220 REM initialisation des variables globales:
1230 PR=1           :REM première ligne
1240 MI=1           :REM mode insertion
1250 X=1:Y=1        :REM position du curseur
1260 TL=0           :REM buffer vide donc Total Ligne = 0
1270 F=0            :REM le booléen FIN
1280 REM maitenant on efface l'écran et on affiche le message
1290 CLS            :REM ou HOME suivant votre BASIC
1300 PRINT MF$
1310 RETURN
```

Nous aborderons le mois prochain le cœur du programme avec la définition complète de la plupart des procédures (ou sous-programmes) qui sont appelées dans CORPS. Nous ne détaillerons pas l'ensemble de ces procédures à la fois en BASIC et en PASCAL ; nous donnerons plutôt soit le programme source en pseudo-langage, soit des exemples dans l'un ou l'autre des langages. Si vous avez des problèmes d'adaptation, nous approfondirons alors les points précis en BASIC et en PASCAL.

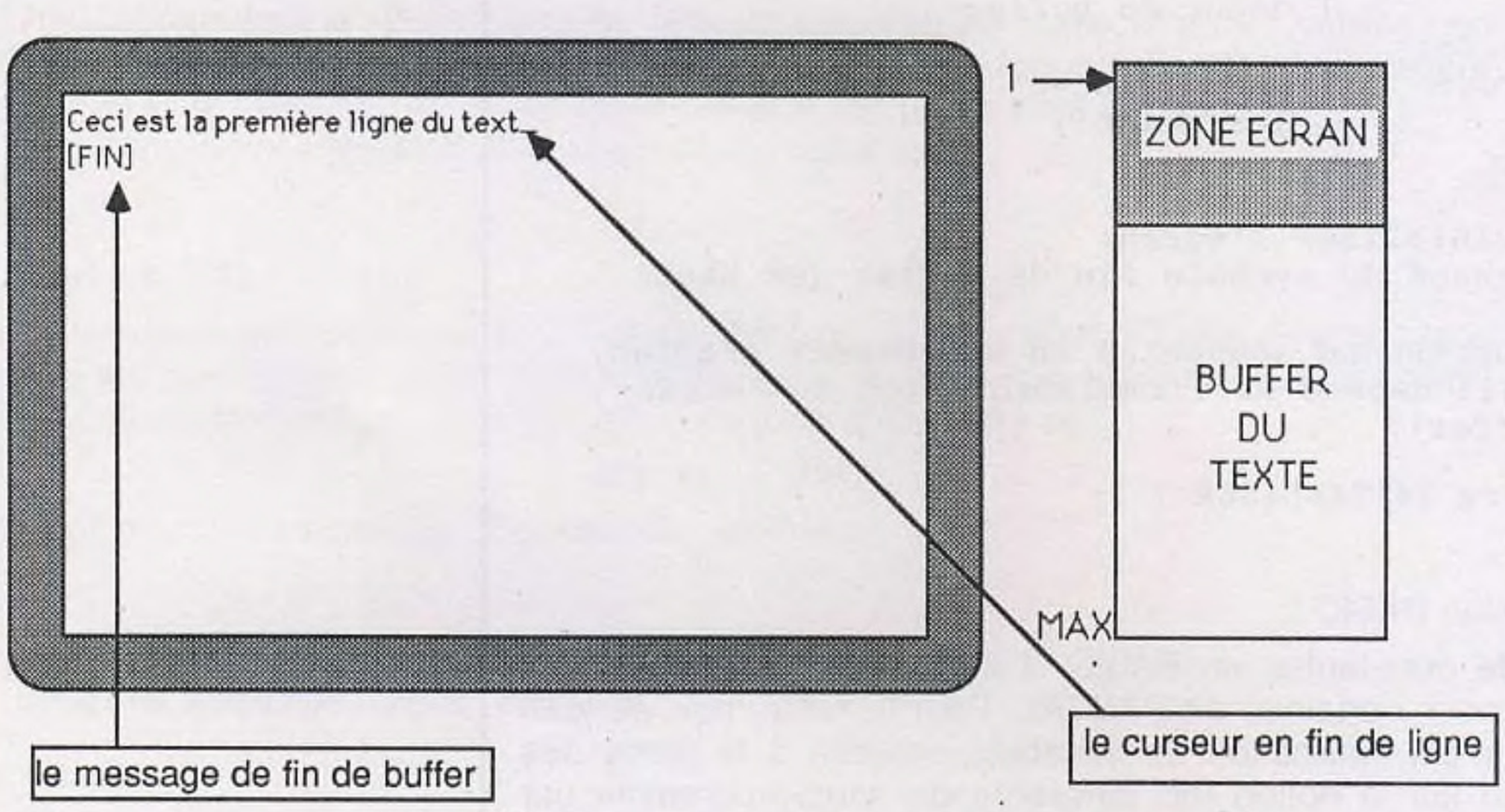


Figure 1 **Définition des VARIABLES GLOBALES et des CONSTANTES**
Ici première_ligne = 1 (on pointe en haut du buffer)
et celui-ci a 1 ligne utilisée

LE JEU INTERACTIF

Contrairement à l'analyse de l'éditeur de texte, nous allons commencer par étudier et essayer de résoudre les parties délicates de la programmation. Il y a plusieurs raisons à cela. Il faut tout d'abord considérer que les impératifs d'un jeu de réflexes imposent un faible temps de réponse. Il faut donc garantir que l'affichage à l'écran sera rapide. Bien évidemment, ces performances sont liées au langage utilisé (interprété ou compilé) mais aussi aux caractéristiques de l'ordinateur dans les différents modes graphiques. D'autre part la recherche d'une bonne méthode de programmation n'est pas simple, aussi vaut-il mieux procéder par essai-erreur, et tester les performances de base des routines qui seront amenées à former le noyau du programme.
Bien que le «cahier des charges», tel que nous l'avions précisé, n'imposait pas de mode graphique (cela essentiellement pour nous permettre de fournir un programme facilement adaptable à toute machine), nous allons traiter dès maintenant des problèmes de rafraîchissement des objets à l'écran aussi bien en mode texte qu'en haute résolution. En effet, la majorité d'entre vous a déjà envisagé la programmation d'un jeu et s'est heurté à ce type de problème.

DEPLACEMENT A L'ECRAN

Il ne faut pas croire que le passage d'un programme interprété à une version compilée va résoudre des problèmes tels que le clignotement de l'affichage ou la vitesse de réaction de la poignée de jeu. Au contraire, on peut sans se tromper affirmer qu'il est possible d'écrire en BASIC interprété de très bons et beaux programmes de jeux. Voici donc quelques exemples d'astuces et solutions suivant les ordinateurs.

A. Le tracé des formes

Le mode texte

Pas ou peu de difficultés, puisqu'une partie du travail est effectuée par l'ordinateur lui-même. Il suffit en effet de se positionner au bon endroit de la page, et d'effectuer un «print» (en BASIC) ou un «write» (en PASCAL), le plus délicat restant la gestion de formes sur plusieurs lignes. Nous détaillerons cet aspect dans les prochains numéros, puisque c'est en mode texte que nous allons réaliser ce programme.

Les lutins

Si votre BASIC dispose de ce que l'on appelle des **«sprites»** en anglais (c'est-à-dire des formes définies à l'avance, que l'on peut disposer n'importe où sur l'écran et manipuler à l'aide de quelques instructions simples, elles existent notamment dans le BASIC des ordinateurs MSX), il ne reste plus qu'à définir les objets comme les missiles, et les météores sous forme de lutins pour ensuite les animer.

Les transferts de zones mémoires

Vous ne disposez pas de lutins, alors, si vous êtes courageux et que vous connaissez votre machine sur le bout des doigts, vous pouvez très bien créer des formes par «BIT» images (cf. figure 2) et effectuer les déplacements par des transferts directs dans la mémoire écran. Ce n'est le moyen le plus simple de mise en œuvre mais il correspond en fait à ce qu'utilisent les programmeurs des jeux que vous achetez dans le commerce. Il vous faudra de plus maîtriser la programmation en langage machine si vous voulez optimiser les transferts de ces zones mémoires.

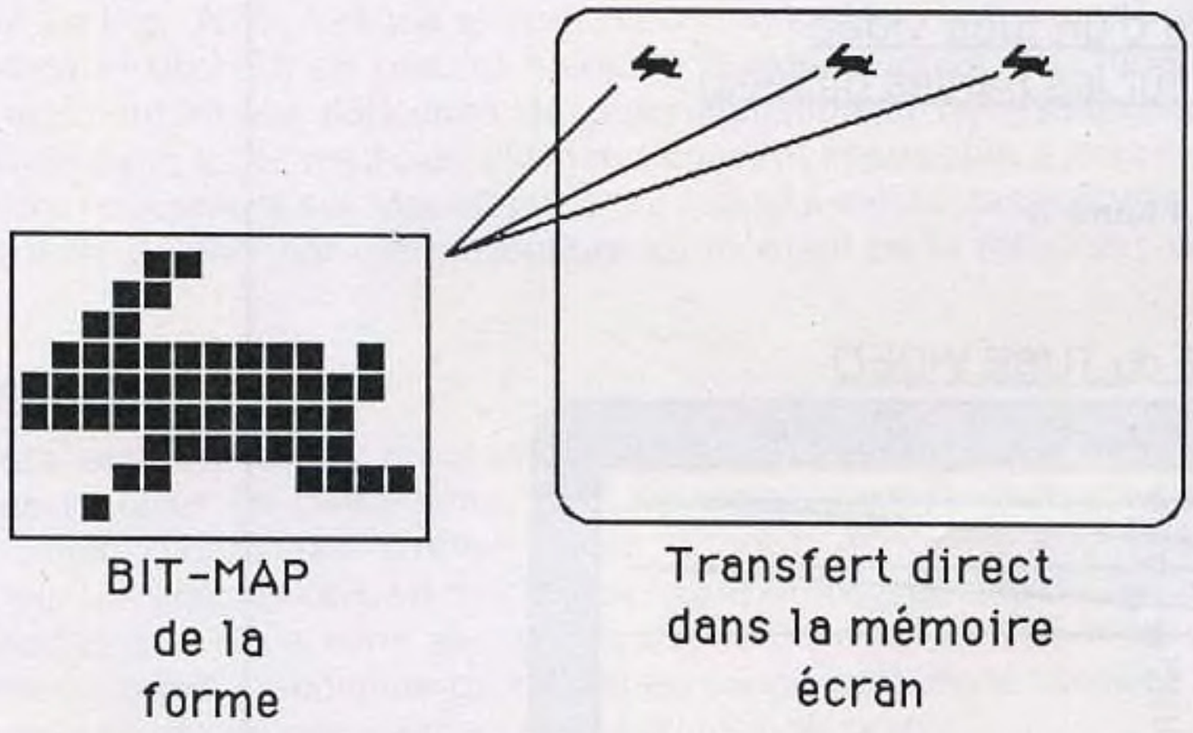
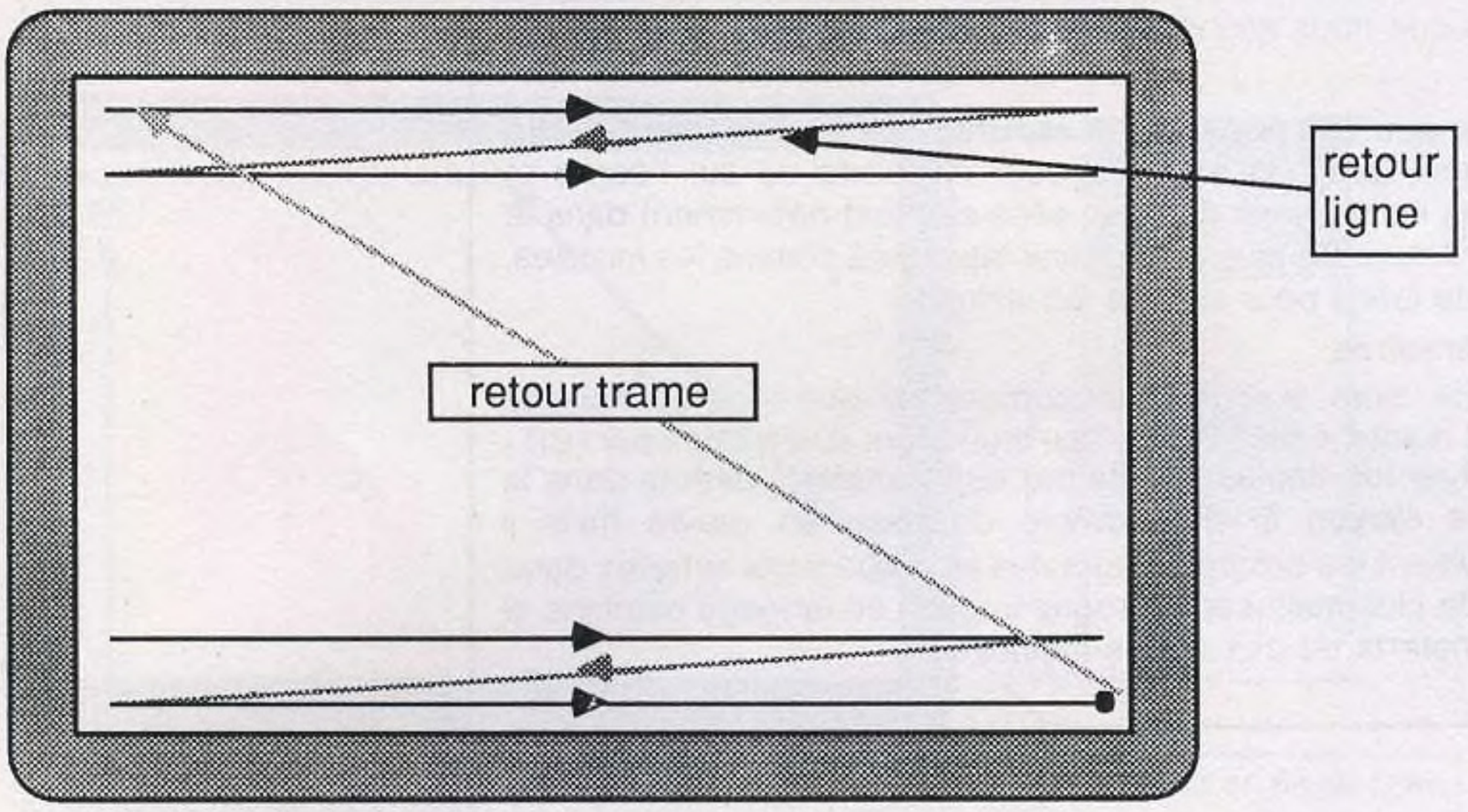


Figure 2

Le tracé par instructions standard (BASIC, PASCAL, etc.)
C'est la solution de sagesse dans une première approche de la programmation en haute résolution graphique. Après une description des ordres nécessaires à la création de l'objet (là encore tout dépend de la richesse du langage dont vous disposez), il suffit de paramétrer cette création de manière à la rendre indépendante de la position à l'écran (attention aux effets sur les bords) et de l'intégrer dans une procédure. Nous fournirons le mois prochain des exemples de programmation en haute résolution.

B. L'interaction du mouvement et de l'affichage

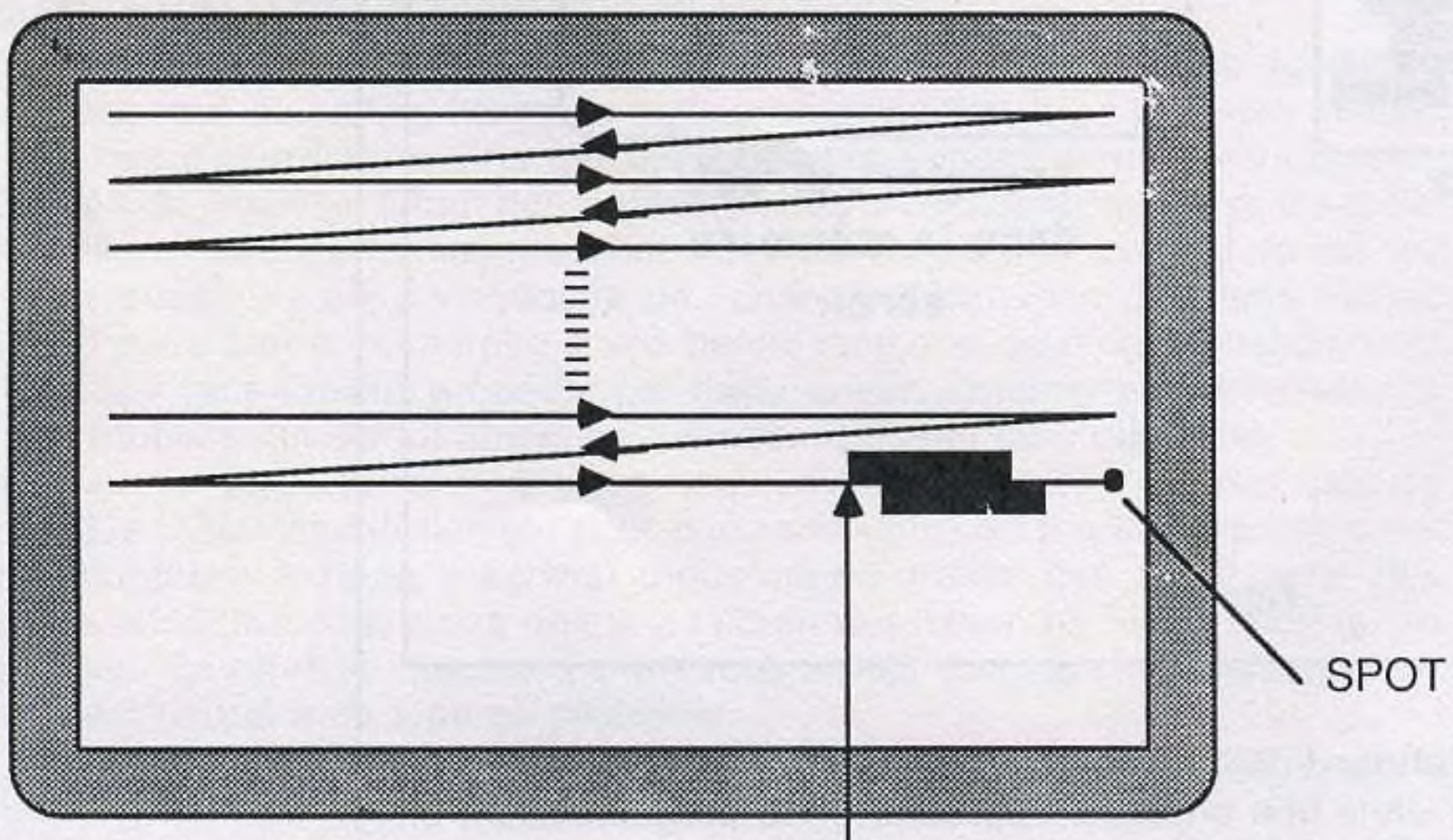
Quelle que soit la méthode de tracé des objets, il faut veiller à la synchronisation des modifications avec le rafraîchissement de l'écran. C'est-à-dire qu'il faut éviter que le spot du téléviseur parcoure la forme pendant qu'on la déplace (figure 3).



Principe de balayage d'un tube vidéo
(le spot n'est pas visible sur les parties grisées)

Figure 3

BALAYAGE du TUBE VIDEO



le rectangle est déplacé vers la droite au moment où le spot l'affiche à l'écran. Comme le balayage est plus rapide que la modification de l'objet il y a distortion de l'affichage.

Le mode multi-pages

Encore une caractéristique liée à l'ordinateur. Les créateurs de jeux s'en servent (et c'est un peu pour cela qu'elle existe) pour séparer la page affichée de celle en cours de modification. On affiche la première page tant que la modification de la deuxième n'est pas terminée et on affiche cette dernière, la première page devenant celle que l'on modifiera, etc. On garantit ainsi qu'il n'y aura aucun affichage en cours de modification.

Cependant, un problème de taille persiste : comment faire porter les modifications que l'on opère sur la page qui n'est pas visualisée ? Il n'y a qu'une alternative : ou bien votre langage le permet, et dans ce cas pas de problème, ou bien il faut trouver des astuces pour tromper l'environnement, lorsque c'est possible.

Par exemple sur Apple en BASIC, le passage en haute résolution implique deux actions principales :

- la sélection d'une page graphique pour l'affichage,
- la mise à jour d'un pointeur du BASIC vers le début de la zone mémoire correspondant à la page affichée.

Il suffit de modifier directement ce pointeur pour changer la page dans laquelle les tracés seront faits, et d'utiliser les chargements des pages affichées par manipulation directe des adresses concernées ! Simple non ?

Les fonctions de synchronisation

Lorsque, pour des raisons diverses, vous n'utilisez qu'une seule page, il reste à accorder les modifications avec le retour du spot du moniteur en haut de l'image (ce que les spécialistes appellent le **«retour trame»**). Pendant cette période, rien n'est affiché à l'écran, on peut donc modifier toute information de la zone mémoire. Malheureusement, on dispose de peu de temps, il faut donc prévoir de mettre à jour la page de haut en bas. Ainsi, lorsque le spot recommence son balayage, il parcourt de gauche à droite et de haut en bas, et n'affiche que les parties déjà modifiées. Ces contraintes augmentent les difficultés de programmation et nécessitent une bonne vitesse de traitement, cette méthode est donc souvent impossible à écrire en BASIC. On la rencontre notamment sur Macintosh, pour lequel il existe des routines systèmes qui permettent de déclencher une procédure au moment de la remontée du spot.

CONCLUSION

Ces différents aspects de la programmation en haute résolution ne seront plus abordés dans la suite de cette série, tout au moins pour la réalisation de ce jeu interactif. Comme d'habitude, n'hésitez pas à nous écrire, si vous désirez des précisions pour les adaptations en mode graphique ou des détails de programmation sur les méthodes que nous vous avons proposées ce mois-ci. Les mois prochains, nous détaillerons la programmation de ce jeu en l'analysant d'une manière similaire à celle que nous venons de commencer pour l'éditeur de texte.

DIALOGUE AVEC NOS LECTEURS

LES PROBLEMES DE CHAMEAUX

Nous allons vous présenter les solutions des problèmes de chameaux dans les deux langages, BASIC et PASCAL. Vous pourrez ainsi comparer vous-même les qualités respectives de ces langages. Mais quel que soit votre choix, une analyse doit toujours précéder la solution programmée.

I. Les chameaux et les bananes

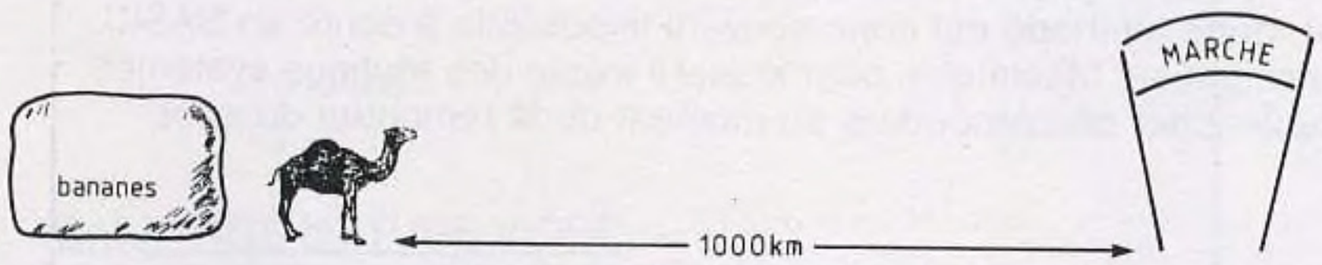


Fig. I

La remarque que l'on doit se faire concerne le nombre de trajets à faire entre un point X et un point Y. Si au point X, on a une charge inférieure à 1 000 bananes, on peut tout prendre d'un coup, puis partir pour Y. Si la charge est comprise entre 1 000 et 2 000 bananes, on a besoin de 2 voyages ; mais alors, attention, il y aura besoin de bananes pour revenir faire le deuxième voyage. Entre 2 000 et 3 000, il faudra 3 voyages et 2 retours et ainsi de suite.
La stratégie est donc de faire n voyages et n – 1 retours pour tout chargement compris entre n × 1 000 et (n – 1) × 1 000, mais dès que le trajet a consommé suffisamment de bananes pour qu'il en reste seulement (n – 1) × 1 000, il n'est plus besoin que de faire (n – 1) voyages et (n – 2) retours.

Dans l'exemple, nous avons 3 000 bananes au départ, 3 allers et 2 retours sont donc nécessaires, ce qui fait 5 fois la même distance d. Lorsque nous aurons parcouru cette distance, s'il ne reste plus que 2 000 bananes, 2 allers et 1 seul retour suffiront. Il faut donc que

$$5 \times d = 1\,000 \Rightarrow d = 200 \text{ km}$$

donc au bout de 200 km, il nous reste 2 000 bananes. Pour d2, ce sera 3 × d2 qui devra valoir 1 000 $\Rightarrow$ d2 = 333 km. La 1 000^{e} banane étant offerte au conducteur (533 km sans manger ?!) et il reste 1 000 bananes pour 467 km à parcourir. On arrivera donc au marché avec 533 bananes.

M. Bigo a été le premier à nous envoyer une solution sous forme de disquette avec un affichage remarquable, il a raté de peu la solution optimale en considérant que celle-ci ne nécessiterait qu'une seule étape alors qu'en réalité, deux étaient nécessaires. Par conséquent, son programme ne trouve que la meilleure solution avec une seule étape. Pour les possesseurs de machines autres que l'Apple II, la partie concernant la solution se trouve entre les lignes 350 et 580.

```
50  TEXT : HOME : GOTO 10000
100  REM # CHAMEAU # CH.BIGO LE 18.12.86
105  ONERR  GOTO 230
110  TEXT : HOME
120  VTAB 4: PRINT "========================================="
130  VTAB 6: PRINT "-LA TRAVERSEE DU DESERT AVEC UN CHAMEAU-"
140  PRINT "========================================="
150  VTAB 13: HTAB 3: PRINT " 0-  POUR QUITTER: RETOUR AU BASIC."
160  VTAB 15: HTAB 3: PRINT " 1-  IMPEDIMENTA ET FEUILLE DE ROUTE."
170  PRINT
180  HTAB 3: PRINT " 2-  AVEC  LES FORMULES AFFICHEES."
190  VTAB 22
200  HTAB 20: PRINT "VOTRE CHOIX ";: GET YN$: IF YN$ = "2" THEN 1000
210  IF YN$ = "0" THEN  TEXT : HOME : END
220  IF YN$ > "1" THEN  PRINT : CALL  - 998: GOTO 200
230  TEXT : HOME
240  INVERSE : PRINT "- ENTRER LES DONNEES OU TAPER <RETURN> -": NORMAL : POKE
     34,1: VTAB 2
250  HOME
260  INPUT "DISTANCE @ PARCOURIR..... ";D$: IF D$ = "" THEN D = 1
     000: HTAB 34: CALL  - 998: PRINT D: GOTO 280
270 D =  VAL (D$): HTAB 34: CALL  - 998: CALL  - 868: HTAB 34: PRINT D
280  INPUT "CHARGE MAX..... ";C$: IF C$ = "" THEN C = 1
     000: HTAB 34: CALL  - 998: PRINT C: GOTO 320
290 C =  VAL (C$): HTAB 34: CALL  - 998: CALL  - 868: HTAB 34: PRINT C
300  IF D = 1 AND C = 1 OR D = 1 AND C = 2 OR D = 2 AND C = 2 OR D = 2 AND
     C = 1 THEN 250
310  IF C < D THEN  CALL  - 998: CALL  - 868: GOTO 280
320  IF C > D AND C < D * 3 THEN C = D
330  INPUT "QUANTITE @ TRANSPORTER..... ";Q$: IF Q$ = "" THEN Q = 3
     000: HTAB 34: CALL  - 998: PRINT Q: GOTO 370
340 Q =  VAL (Q$): HTAB 34: CALL  - 998: CALL  - 868: HTAB 34: PRINT Q
350 W = 0: IF Q =  < D * 2 THEN W = 1: GOSUB 610
360  IF W >  = 1 THEN 300
370  PRINT "CONSOMMATION AU KM PARCOURU..... ";: HTAB 34:K = 1: PRINT K
380  PRINT : PRINT "=========================================": PRINT
390  IF C >  = D AND Q < D * 2 THEN  PRINT : PRINT "TRANSPORT EN 1 SEUL PA
     RCOURS.": PRINT : PRINT "STOCK TOTAL A L'ARRIVEE: ";:M = Q - D: PRINT
     M: END
400 A = C / 2:R = C / 2:CH = A + R: REM A)LLER MAX. R)ETOUR.
410 N = Q / C * 2: IF  INT (N) <  > N THEN N =  INT (N) + 1:PS = 1
420 NB = N - 1: IF NB = 0 THEN NB = 1
430  PRINT "NB DE PARCOURS POUR ENTREPOSER.... ";: HTAB 34: PRINT NB
440 P = Q / NB: IF  INT (P) <  > P THEN P =  INT (P) + 1:PS = 1
450 V = P
460  IF D * N =  < Q AND P < C THEN M = Q - (D * N): GOTO 580
470  PRINT
480  IF Q / N >  = D * 2 THEN M = Q - D * NB: GOTO 580
490  PRINT "DERNIER VOYAGE ALLER MAX..... ";: HTAB 34: PRINT V
500 E = D - V
510  PRINT : PRINT "DISTANCE JUSQU'@ L'ESCALE..... ";: HTAB 34: PRINT
     E
520  IF  SGN (E) =  - 1 THEN  GOSUB 610
530  PRINT : PRINT "DISTANCE AVANT NON RETOUR..... ";: HTAB 34:Z = C /
     2 - 1: PRINT Z
```

```
540  IF C < D AND D > Q / (Q / C) THEN  FOR T = 1 TO 8: CALL  - 998: CALL
      - 868: NEXT : INVERSE : PRINT "IMPOSSIBLE- ...NON RETOUR.": NORMAL :
     END
550  PRINT : PRINT "CHARGE DEPOSEE @ CHAQUE ESCALE.... ";: HTAB 34:CE = (C
     H - (E * 2)): PRINT CE
560 CF = CE * N / 2:CG = CF + E:M = CG - V
570  PRINT : PRINT "STOCK TOTAL DANS L'ENTREPOT..... ";: HTAB 34:S = Q -
     E * NB: PRINT S
580  PRINT "QUANTITE @ L'ARRIVEE.....";: HTAB 34: INVERSE : PRINT
     M: NORMAL
590  IF PS = 1 THEN  INVERSE : PRINT "# RESULTATS APPROCHES #";: NORMAL : PRINT
     " ( DECIMALES.)";: GET YN$: REM   POUR GAGNER UNE LIGNE A LA FIN DE L
     'AFFICHAGE.
600  END
610  PRINT "ON VA BOUFFER LA BARAQUE !!": HTAB 34: INVERSE : PRINT "-STOP-
     ": NORMAL : IF W >  = 1 THEN  FOR U = 1 TO 500: NEXT : FOR U = 1 TO 3
     : CALL  - 998: CALL  - 958: NEXT : RETURN
620  END
630 :
1000  TEXT : HOME : CLEAR
1010  INVERSE : PRINT "- ENTRER LES DONNEES OU TAPER <RETURN> -": NORMAL :
      POKE 34,1: VTAB 2
1020  HOME
1030  INPUT "DISTANCE @ PARCOURIR.....D  ";D$: IF D$ = "" THEN D = 1
     000: HTAB 34: CALL  - 998: PRINT D: GOTO 1050
1040 D =  VAL (D$): HTAB 34: CALL  - 998: CALL  - 868: HTAB 34: PRINT D
1050  INPUT "CHARGE MAX.....C  ";C$: IF C$ = "" THEN C = 1
     000: HTAB 34: CALL  - 998: PRINT C: GOTO 1090
1060 C =  VAL (C$): HTAB 34: CALL  - 998: CALL  - 868: HTAB 34: PRINT C
1070  IF D = 1 AND C = 1 OR D = 1 AND C = 2 OR D = 2 AND C = 2 OR D = 2 AND
     C = 1 THEN 1020
1080  IF C < D THEN  CALL  - 998: CALL  - 868: GOTO 1050
1090  IF C > D AND C < D * 3 THEN C = D
1100  INPUT "QUANTITE @ TRANSPORTER.....Q  ";Q$: IF Q$ = "" THEN Q = 3
     000: HTAB 34: CALL  - 998: PRINT Q: GOTO 1120
1110 Q =  VAL (Q$): HTAB 34: CALL  - 998: CALL  - 868: HTAB 34: PRINT Q
1120 W = 0: IF Q =  < D * 2 THEN W = 1: GOSUB 1540
1130  IF W >  = 1 THEN 1100
1140  PRINT "CONSOMMATION AU KM PARCOURU....K  ";: HTAB 34:K = 1: PRINT K
1150  PRINT "=========================================";
1160 A = C / 2:R = C / 2:CH = A + R: REM A)LLER MAX. R)ETOUR.
1170 NX = Q / C:N = NX * 2:PS = 0
1180  IF  INT (N) <  > N THEN N =  INT (N) + 1:PS = 1
1190 H = N - 1: IF H = 0 THEN H = 1
1200  PRINT "Q ";Q;"/C ";C;" *2 #INT<>+1# =";N;" -1 ="
1210  PRINT "NB DE PARCOURS POUR ENTREPOSER.... ";: HTAB 34: PRINT H
1220 P = Q / H: IF  INT (P) <  > P THEN P =  INT (P + 1):PS = 1
1230 V = P
1240  IF D * N =  < Q AND P < C THEN M = Q - (D * N): GOTO 1510
1250  IF Q / N >  = D * 2 THEN  PRINT "Q ";Q;" -D ";D;" *H =":M = Q - D *
     H: GOTO 1510
1260  PRINT "D ";D;"-(Q ";Q;"/H ";H;"#INT<>+1#)= P= V"
1270  PRINT "DERNIER VOYAGE ALLER MAX.....V ";: HTAB 34: PRINT V
1280 E = D - V
1290  PRINT "D ";D;"-V ";V
1300  PRINT "DISTANCE JUSQU'@ L'ESCALE.....E ";: HTAB 34: INVERSE : PRINT
     E: NORMAL
1310  IF  SGN (E) =  - 1 THEN 1540
1320  PRINT "K=";K;"  I = 1 TO D ";D;":K =+ 1:C =C ";C;"-1"
1330  HTAB 14: PRINT "- PATIENCE -"
1340  REM   PETIT GALOP D'ESSAI DAND LE DESERT.# Z= C)HARGE MAX./2 -1 # SE
     RAIT PLUS RAPIDE MAIS ALORS PLUS DE SUSPENS!!
1350  FOR I = 1 TO D:K = K + 1:C = C - 1
1360  PRINT
1370  PRINT "K= ";K;: HTAB 9: PRINT "C= ";C;: HTAB 17: PRINT "NON RETOUR I
     -1=Z ";I - 1
```

```
1380  CALL  - 998: CALL  - 998: CALL  - 868
1390  IF C <  = D / 2 THEN 1410
1400  NEXT
1410  PRINT : PRINT
1420 Z = I - 1: IF Z <  = 0 THEN 1560
1430 CE = (CH - (E * 2))
1440  IF  SGN (CE) =  - 1 THEN 1540
1450  PRINT "CH ";CH;"-E ";E;"*2="
1460  PRINT "PAR ETAPE CHARGE DEPOSEE......CE ";: HTAB 34: PRINT CE
1470 CF = CE * N / 2:CG = CF + E:M = CG - V
1480  PRINT "Q ";Q;" - E ";E;"* H ";H;"="
1490  PRINT "STOCK TOTAL DANS L'ENTREPOT....S ";: HTAB 34:S = Q - E * H: PRINT
     S
1500  PRINT "CE ";CE;"*N ";N;" /2=";CF;"+E ";E;"= ";CG;"-V="
1510  PRINT "QUANTITE @ L'ARRIVEE...........M ";: HTAB 34: INVERSE : PRINT
     M: NORMAL
1520  IF PS = 1 THEN  INVERSE : PRINT "# RESULTATS APPROCHES #";: NORMAL :
      PRINT " ( DECIMALES.)";: GET YN$: REM  POUR GAGNER UNE LIGNE A LA FI
     N DE L'AFFICHAGE.
1530  END
1540  PRINT "ON VA BOUFFER LA BARAQUE !!": HTAB 34: INVERSE : PRINT "-STOP
     -": NORMAL : IF W >  = 1 THEN  FOR U = 1 TO 500: NEXT : FOR U = 1 TO
     3: CALL  - 998: CALL  - 958: NEXT : RETURN
1550  END
1560  FOR T = 1 TO 8: CALL  - 998: CALL  - 868: NEXT : INVERSE : PRINT "IM
     POSSIBLE- ...NON RETOUR.": NORMAL : END
5000  REM PRESENTATION DANS LA FENETRE.
5010  HOME :HA = 0:VA = 0
5020 A$ = "CARAVANSERAIL"
5030 VA = 5:HA = 4
5040  FOR I = 1 TO 13
5050 B$ =  MID$ (A$,I,1)
5060 HA = HA + 2
5070 VA = VA + 1
5080  VTAB VA
5090  HTAB HA
5100  PRINT B$;: HTAB HA
5110  NEXT
5120  FOR TT = 1 TO 1500: NEXT
5130  RETURN
10000  REM # ROULE # LE 12/07/85
10010  TEXT : HOME
10020 RL$ = "!<->->!<"
10030 A = 1
10040 W = W + 1
10050  FOR J = 1 TO 20
10060  FOR I = 1 TO 4
10070  GOSUB 10220
10080  NEXT
10090  NEXT
10100  IF W = 2 THEN 10330
10110  POKE 34,1: VTAB 2
10120 A = 2
10130  FOR N = 1 TO 23
10140  FOR M = 1 TO 4
10150  GOSUB 10280
10160  NEXT
10170  NEXT
10180  IF W = 2 THEN  POKE 35,23: GOTO 10330
10190 A = 23
10200  GOTO 10040
10210  END
10220  VTAB A: HTAB J
10230  PRINT  MID$ (RL$,I,1)
10240  VTAB A: HTAB 40 - J
10250  PRINT  MID$ (RL$,I + 4,1)
```

```
10260  RETURN
10270  END
10280  VTAB N: HTAB 1
10290  PRINT  MID$ (RL$,M,1)
10300  VTAB N: HTAB 39
10310  PRINT  MID$ (RL$,M + 4,1)
10320  RETURN
10330  REM ROULETTES
10340  NORMAL
10350  VTAB 2: POKE 35,22: POKE 33,36: POKE 32,2
10360  POKE 34,2: POKE 35,20
10370  PRINT
10380  GOSUB 5010
10390  GOTO 100
10400  END
```

Pour la solution optimale, nous avons pensé que ce serait une bonne occasion d'utiliser le Pascal. Nous allons donc ensemble analyser ce programme.
Commençons par le programme principal dont les variables sont définies tout en haut, elles sont toutes entières, et le programme principal se borne à les initialiser en questionnant l'utilisateur sur les conditions du trajet. Dans notre cas :

- Changement total : 3000,
- Consommation : 1,
- Charge maximale : 1000,
- Distance à parcourir : 1000.

Une fois ces opérations effectuées, appelons la fonction Voyage dont le résultat est le nombre de bananes restantes.
On part en voyage avec 3 000 bananes pour une distance de 1 000 km, il convient de savoir de combien d'allers-retours on aura besoin pour ces 3 000 bananes, ce qui donnera une consommation kilométrique «instantanée» lorsque tant d'allers-retours sont nécessaires. Il convient alors de déterminer le chargement à partir duquel un aller-retour de moins sera utile, et le kilométrage alors parcouru.
Si le kilométrage est supérieur à la distance qu'il restait à parcourir, on s'arrête avant et on compte ce qui reste à ce moment-là.
Sinon, une étape supplémentaire est nécessaire, le nombre de bananes restantes sera celui obtenu après la nouvelle étape mais celle-ci n'est possible que si le chargement restant n'est pas déjà nul.

```
Program Chameau ;

Var
  Charge_maximale          ,
  Consommation             ,
  Distance_a_parcourir     ,
  Distance_deja_parcourue  ,
  Chargement_total         ,
  Bananes_restantes        : INTEGER ;

Function Voyage(Chargement      ,
                Charge_max      ,
                Consommation    ,
                Dist_parcourue  ,
                Distance_reste : INTEGER ) : INTEGER ;

  Var

    Allers_retours          ,
    Consommat_kilometrique  ,
    Chargement_restant      ,
    Kilometrage             : INTEGER ;
```

```
Begin
  Allers_retours := ( Chargement DIV ( Charge_max * Consommation ) ) ;
  Consommat_kilometrique := ( 2 * Allers_retours - 1 ) * Consommation ;
  Chargement_restant := ( ( Allers_retours - 1 ) * Charge_max ) ;
  Kilometrage :=  ( Chargement - Chargement_restant ) DIV Consommat_kilometrique ;
  Dist_parcourue := Dist_parcourue + Kilometrage ;
  if Kilometrage >= Distance_reste then
      Voyage := ( Kilometrage - Distance_reste ) * Consommat_kilometrique
                + Chargement_restant
  else
      if Chargement_restant <= 0 then
        begin
          writeln (' Distance maximale possible ',Dist_parcourue) ;
          Voyage := 0 ;
        end
      else
        begin
          writeln (' Etape suivante au KM :',Dist_parcourue) ;
          Voyage := Voyage ( Chargement_restant , Charge_max ,
                             Consommation , Dist_parcourue ,
                             Distance_reste - Kilometrage ) ;
        end ;
End ;

{------------------ Programme Principal ----------------------}

Begin
  writeln('donnez le nombre de bananes du chargement :') ;
  readln(Chargement_total) ;
  writeln('donnez le nombre de bananes consommees par kilometre :') ;
  readln(Consommation) ;
  writeln('donnez la charge maximale en bananes acceptee par le chameau :') ;
  readln(Charge_maximale) ;
  writeln('donnez la distance a parcourir :') ;
  readln(Distance_a_parcourir) ;
  Distance_deja_parcourue := 0 ;
  Bananes_restantes := Voyage( Chargement_total , Charge_maximale ,
                               Consommation , Distance_deja_parcourue ,
                               Distance_a_parcourir ) ;
  writeln (' il reste au bout du voyage ',Bananes_restantes,' bananes ');
End
```

LES CARRES BI-MAGIQUES

Nous vous présentons la solution proposée par M. Gueron qui permet d'obtenir 80 carrés magiques 8 × 8 qui ont donc aussi la propriété d'avoir la somme des carrés pour chaque ligne, colonne ou diagonale constante. Le programme tient compte des symétries puisque tous les carrés sont orientés de telle façon que le coin supérieur gauche soit le plus petit des quatre coins.
Comme M. Gueron nous le précise dans sa lettre, il est très facile de passer de cases numérotées de 0 à 63, à des cases numérotées comme d'habitude de 1 à 64. En effet, à l'aide de l'identité remarquable :

$$(x+y)^{\wedge}2 = x^{\wedge}2 + 2xy + y^{\wedge}2,$$

on trouve que la somme d'une ligne ou colonne vaut :

$$252 + 8 = 260$$

et la somme des carrés :

$$10668 + 2 \times 252 + 8 = 11180.$$

Par la même méthode, on peut d'ailleurs montrer que l'on peut commencer à partir d'une valeur quelconque (par exemple de 2 à 65).

```
10 REM programme réalisé par M Jacques GUERON
20 REM BASIC standard+ print using
30 REM création de 80 carres bi-magiques d'ordre 8, tous distincts les uns
40 REM des autres,
50 REM constantes:     - magique=252, bi-magiques=10668
```

```
55 REM classés (coin gauche supérieur toujours inférieur aux autres coins)
60 DEFINT A-Z
65

70 FOR s=0 TO 56 STEP 8
80 FOR t=0 TO 56 STEP 8:IF t=s THEN 640
90 FOR u=0 TO 56 STEP 8:IF u=s OR u=t THEN 630
100 FOR v=0 TO 56 STEP 8:IF v=s OR v=t OR v=u THEN 620
110 FOR w=0 TO 56 STEP 8:IF w=s OR w=t OR w=u OR w=v THEN 610
120 FOR x=0 TO 56 STEP 8:IF x=s OR x=t OR x=u OR x=v OR x=w THEN 600
130 FOR y=0 TO 56 STEP 8
135 IF y=s OR y=t OR y=u OR y=v OR y=w OR y=x THEN 590
140 FOR z=0 TO 56 STEP 8
145 IF z=s OR z=t OR z=u OR z=v OR z=w OR z=x OR z=y THEN 580
150  sy=ABS(s-y):vx=ABS(v-x):tz=ABS(t-z):uw=ABS(u-w)
155  sz=ABS(s-z):ty=ABS(t-y):xu=ABS(x-u):wv=ABS(w-v)
160  st=ABS(s-t):uv=ABS(u-v):wx=ABS(w-x):yz=ABS(y-z)
170 IF sy=vx AND vx=tz AND tz=uw THEN 180 ELSE 580
180 IF sz=ty AND ty=xu AND xu=wv THEN 200 ELSE 190
190 IF st=uv AND uv=wx AND wx=yz THEN 200 ELSE 580


200 FOR A=0 TO 7
210 FOR B=0 TO 7: IF B=A THEN 560
220 FOR C=0 TO 7: IF C=A OR C=B THEN 550
230 FOR D=0 TO 7: IF D=A OR D=B OR D=C THEN 540


240 IF K=0 THEN  s=0:t=8:u=24:v=16:w=56:x=48:y=32:z=40:C=4:B=2:D=7:A=1
250 IF K=4 THEN  s=0:t=8:u=40:v=32:w=56:x=48:y=16:z=24:C=2:B=4:D=7:A=1
260 IF K=12 THEN  s=0:t=16:u=24:v=8:w=56:x=40:y=32:z=48:C=2:B=4:D=7:A=1
270 IF K=20 THEN  s=0:t=16:u=48:v=32:w=56:x=40:y=8:z=24:C=2:B=7:D=4:A=1
280 IF K=28 THEN  s=0:t=32:u=40:v=8:w=56:x=24:y=16:z=48:C=2:B=7:D=4:A=1
290 IF K=36 THEN  s=0:t=32:u=48:v=16:w=56:x=24:y=8:z=40:C=4:B=7:D=2:A=1
300 IF K=40 THEN  s=8:t=0:u=16:v=24:w=48:x=56:y=40:z=32:C=5:B=3:D=6:A=0
310 IF K=44 THEN  s=8:t=0:u=32:v=40:w=48:x=56:y=24:z=16:C=6:B=3:D=5:A=0
320 IF K=52 THEN  s=8:t=24:u=56:v=40:w=48:x=32:y=0:z=16:C=6:B=5:D=3:A=0
330 IF K=60 THEN  s=8:t=40:u=56:v=24:w=48:x=16:y=0:z=32:C=5:B=6:D=3:A=0
340 IF K=64 THEN  s=16:t=0:u=8:v=24:w=40:x=56:y=48:z=32:C=6:B=3:D=5:A=0
350 IF K=72 THEN  s=16:t=48:u=56:v=24:w=40:x=8:y=0:z=32:C=6:B=5:D=3:A=0


360  AA=7-A:BB=7-B:CC=7-C:DD=7-D
370 IF s+C>=z+AA OR s+C>=w+CC OR s+C>=v+A THEN 540
380 IF A^2 + B^2 + C^2 + D^2<>70 THEN 540
390 IF  s*C+t*B+u*DD+v*AA+w*CC+x*BB+y*D+z*A<>784  THEN 540
400 IF  s*CC+t*BB+u*D+v*A+w*C+x*B+y*DD+z*AA<>784  THEN 540
410 IF  s*C+t*CC+u*BB+v*B+w*DD+x*D+y*A+z*AA<>784  THEN 540
420 IF  s*C+t*D+u*B+v*A+w*BB+x*AA+y*CC+z*DD<>784  THEN 540
```

```
430  k=k+1:PRINT "      Bi-Magique **";K;"**"
440 PRINT USING"  ##";s+C;t+CC;u+BB;v+B;w+DD;x+D;y+A;z+AA
450 PRINT USING"  ##";z+DD;y+D;x+A;w+AA;v+C;u+CC;t+BB;s+B
460 PRINT USING"  ##";t+D;s+DD;v+AA;u+A;x+CC;w+C;z+B;y+BB
470 PRINT USING"  ##";y+CC;z+C;w+B;x+BB;u+D;v+DD;s+AA;t+A
480 PRINT USING"  ##";x+AA;w+A;z+D;y+DD;t+B;s+BB;v+CC;u+C
490 PRINT USING"  ##";u+B;v+BB;s+CC;t+C;y+AA;z+A;w+D;x+DD
500 PRINT USING"  ##";w+BB;x+B;y+C;z+CC;s+A;t+AA;u+DD;v+D
510 PRINT USING"  ##";v+A;u+AA;t+DD;s+D;z+BB;y+B;x+C;w+CC

520 GOSUB 660 :REM vérification
530 IF k=80 THEN END
540 NEXT D
550 NEXT C
560 NEXT B
570 NEXT A
580 NEXT z
590 NEXT y
600 NEXT x
610 NEXT w
620 NEXT v
630 NEXT u
640 NEXT t
650 NEXT s


660 PRINT:PRINT "Vérification de la bi-magie"
670 ln1=(s+C)^2+(t+CC)^2+(u+BB)^2+(v+B)^2+(w+DD)^2+(x+D)^2+(y+A)^2+(z+AA)^2
680 ln2=(z+DD)^2+(y+D)^2+(x+A)^2+(w+AA)^2+(v+C)^2+(u+CC)^2+(t+BB)^2+(s+B)^2
690 ln3=(t+D)^2+(s+DD)^2+(v+AA)^2+(u+A)^2+(x+CC)^2+(w+C)^2+(z+B)^2+(y+BB)^2
700 ln4=(y+CC)^2+(z+C)^2+(w+B)^2+(x+BB)^2+(u+D)^2+(v+DD)^2+(s+AA)^2+(t+A)^2
710 ln5=(x+AA)^2+(w+A)^2+(z+D)^2+(y+DD)^2+(t+B)^2+(s+BB)^2+(v+CC)^2+(u+C)^2
720 ln6=(u+B)^2+(v+BB)^2+(s+CC)^2+(t+C)^2+(y+AA)^2+(z+A)^2+(w+D)^2+(x+DD)^2
730 ln7=(w+BB)^2+(x+B)^2+(y+C)^2+(z+CC)^2+(s+A)^2+(t+AA)^2+(u+DD)^2+(v+D)^2
740 ln8=(v+A)^2+(u+AA)^2+(t+DD)^2+(s+D)^2+(z+BB)^2+(y+B)^2+(x+C)^2+(w+CC)^2

750 cl1=(s+C)^2+(z+DD)^2+(t+D)^2+(y+CC)^2+(x+AA)^2+(u+B)^2+(w+BB)^2+(v+A)^2
760 cl2=(t+CC)^2+(y+D)^2+(s+DD)^2+(z+C)^2+(w+A)^2+(v+BB)^2+(x+B)^2+(u+AA)^2
770 cl3=(u+BB)^2+(x+A)^2+(v+AA)^2+(w+B)^2+(z+D)^2+(s+CC)^2+(y+C)^2+(t+DD)^2
780 cl4=(v+B)^2+(w+AA)^2+(u+A)^2+(x+BB)^2+(y+DD)^2+(t+C)^2+(z+CC)^2+(s+D)^2
790 cl5=(w+DD)^2+(v+C)^2+(x+CC)^2+(u+D)^2+(t+B)^2+(y+AA)^2+(s+A)^2+(z+BB)^2
800 cl6=(x+D)^2+(u+CC)^2+(w+C)^2+(v+DD)^2+(s+BB)^2+(z+A)^2+(t+AA)^2+(y+B)^2
810 cl7=(y+A)^2+(t+BB)^2+(z+B)^2+(s+AA)^2+(v+CC)^2+(w+D)^2+(u+DD)^2+(x+C)^2
820 cl8=(z+AA)^2+(s+B)^2+(y+BB)^2+(t+A)^2+(u+C)^2+(x+DD)^2+(v+D)^2+(w+CC)^2

830 di1=(s+C)^2+(y+D)^2+(v+AA)^2+(x+BB)^2+(t+B)^2+(z+A)^2+(u+DD)^2+(w+CC)^2
840 di2=(z+AA)^2+(t+BB)^2+(w+C)^2+(u+D)^2+(y+DD)^2+(s+CC)^2+(x+B)^2+(v+A)^2
850 PRINT cl1;" ";cl2;" ";cl3;" ";cl4;" ";cl5;" ";cl6;" ";cl7;" ";cl8
860 PRINT ln1;" ";ln2;" ";ln3;" ";ln4;" ";ln5;" ";ln6;" ";ln7;" ";ln8
870 PRINT di1;" ";di2:PRINT
880 RETURN
```

LES 6 PREMIERES SOLUTIONS DES CARRéS BI-MAGIQUES 8X8
==========

Bi-Magique ** 1 **

```
 4 11 29 18 56 55 33 46
40 39 49 62 20 27 13  2
15  0 22 25 51 60 42 37
35 44 58 53 31 16  6  9
54 57 47 32 10  5 19 28
26 21  3 12 38 41 63 48
61 50 36 43  1 14 24 23
17 30  8  7 45 34 52 59
```

Vérification de la bi-magie

```
10668   10668   10668   10668   10668   10668   10668   10668
10668   10668   10668   10668   10668   10668   10668   10668
10668   10668
```

Bi-Magique ** 2 **

```
 6  9 31 16 58 53 35 44
42 37 51 60 22 25 15  0
13  2 20 27 49 62 40 39
33 46 56 55 29 18  4 11
52 59 45 34  8  7 17 30
24 23  1 14 36 43 61 50
63 48 38 41  3 12 26 21
19 28 10  5 47 32 54 57
```

Vérification de la bi-magie

```
10668   10668   10668   10668   10668   10668   10668   10668
10668   10668   10668   10668   10668   10668   10668   10668
10668   10668
```

Bi-Magique ** 3 **

```
 1 14 24 23 61 50 36 43
45 34 52 59 17 30  8  7
10  5 19 28 54 57 47 32
38 41 63 48 26 21  3 12
51 60 42 37 15  0 22 25
31 16  6  9 35 44 58 53
56 55 33 46  4 11 29 18
20 27 13  2 40 39 49 62
```

Vérification de la bi-magie

```
10668   10668   10668   10668   10668   10668   10668   10668
10668   10668   10668   10668   10668   10668   10668   10668
10668   10668
```

Bi-Magique ** 4 **

```
 3 12 26 21 63 48 38 41
47 32 54 57 19 28 10  5
 8  7 17 30 52 59 45 34
36 43 61 50 24 23  1 14
49 62 40 39 13  2 20 27
29 18  4 11 33 46 56 55
58 53 35 44  6  9 31 16
22 25 15  0 42 37 51 60
```

Vérification de la bi-magie

```
10668   10668   10668   10668   10668   10668   10668   10668
10668   10668   10668   10668   10668   10668   10668   10668
10668   10668
```

Bi-Magique ** 5 **

```
 2 13 43 36 56 55 17 30
24 23 49 62 34 45 11  4
15  0 38 41 53 58 28 19
21 26 60 51 47 32  6  9
54 57 31 16 12  3 37 42
44 35  5 10 22 25 63 48
59 52 18 29  1 14 40 39
33 46  8  7 27 20 50 61
```

Vérification de la bi-magie

```
10668   10668   10668   10668   10668   10668   10668   10668
10668   10668   10668   10668   10668   10668   10668   10668
10668   10668
```

Bi-Magique ** 6 **

```
 4 11 46 33 56 55 18 29
24 23 50 61 36 43 14  1
15  0 37 42 51 60 25 22
19 28 57 54 47 32  5 10
53 58 31 16  9  6 35 44
41 38  3 12 21 26 63 48
62 49 20 27  2 13 40 39
34 45  8  7 30 17 52 59
```

Vérification de la bi-magie

```
10668   10668   10668   10668   10668   10668   10668   10668
10668   10668   10668   10668   10668   10668   10668   10668
10668   10668
```

. . . . . .
/
. . . . . .

C'EST ARRIVE DEMAIN

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

A partir du 1er janvier 87, la fiscalité américaine changera du tout au tout. Ce changement aura au moins quelques heureux, les sociétés qui proposent des programmes basés sur la gestion de portefeuilles, le calcul des impôts... En effet, les programmes doivent être entièrement refondus pour répondre aux nouveaux critères. Les sociétés qui produisent ces programmes sont donc, depuis l'annonce du changement, au travail à plein temps pour réécrire les nouveaux, ce qui fait tourner des petites maisons de développement qui auraient eu bien des problèmes sans cela. Sans préjuger du contenu du nouveau plan fiscal que je ne connais pas, on peut être sûr que ces sociétés au moins sauront gré au Président Reagan de les faire travailler.

Signe des temps, l'hebdomadaire INFOWORLD, référence aux U.S.A. pour l'information micro, vient de créer une rubrique spéciale MACINTOSH qui paraîtra chaque semaine. Après avoir longtemps critiqué ce «jouet», la revue, comme les autres, doit se plier à la popularité grandissante de cet appareil. Cela doit être un déchirement pour le personnel de ce journal, soutien permanent du standard PC-DOS. Ici au moins, ce sont les meilleurs qui arrivent.

Une grande réunion s'est tenue en France, sur la protection des réseaux informatiques. Le problème actuel est la criminalité informatique qui est, dans plus de 95 % des cas, impossible à détecter et dans les 5 % restant, moins de 20 % des coupables sont punis. Alors des grandes banques s'inquiètent, ainsi que tous ceux qui transfèrent des données sensibles, ou de l'argent. De nombreux cas de détournements de plusieurs millions de dollars ont été prouvés aux U.S.A., mais le plus souvent le coupable est inconnu, et même parfois la fraude n'est retrouvée que par un déficit des comptes mensuels ou annuels. Alors pour

retrouver qui a pu s'en rendre coupable plusieurs semaines auparavant ressemble à la résolution de la quadrature du cercle. Le plus amusant, en France tout au moins, est que les sociétés visées (banques, administrations, compagnies d'assurances) sont réticentes à l'idée de pratiquer des contrôles contraignants sur leurs systèmes, car... cela «signifierait qu'ils ne seraient pas infaillibles». La belle affaire, quand on sait qu'ils ne le sont pas. Mais, devant la montagne «INFORMATIQUE», il semble que le mot d'ordre soit encore mutisme et admiration sans faille. Il me semblait que l'on en était presque au 21e siècle.

On vit une époque merveilleuse !!! Figurez-vous qu'il était déjà possible de faire tourner une machine à base de 68000 (Mac, Amiga, Atari 1024ST) avec une interface qui la transformait en PC compatible. Cela veut dire que pour le prix d'un IBM, vous aviez un compatible IBM (alors que le prix de ces derniers est en général la moitié de celui de l'IBM). Maintenant on fait bien mieux. L'interface est purement logicielle. Cela veut dire que, en plus, l'émulation se traînera à la vitesse de l'escargot essayant de battre le Mirage 2000. Tout cela pour un prix dérisoire, 1 000 $. Pensez, un compatible vaut parfois jusqu'à 500 $. Si quelqu'un comprend, écrivez-moi. Merci d'avance.

Nous parlions de piratage. Dans les universités américaines, cela tourne au sport national. Pour tenter d'y remédier, des logiciels de plus en plus contraignants surveillent toutes les commandes exécutées sur le système, les mémorisent pour permettre, sinon de démasquer le coupable, tout au moins d'empêcher de renouveler les manipulations indélicates par des protections nouvelles. Mais, il y a encore plus fou. Les messageries électroniques sont victimes de programmeurs (à mon sens géniaux bien que très vicieux) qui écrivent des programmes qui dévorent tous les fichiers petit à petit ou qui, une fois utilisés par des clients lambda «croquent» les fichiers de ces derniers, stockés par exemple sur un disque dur. Quand vous voulez relire un de ces fichiers, vous voyez apparaître un message du type «le ver de terre fou a encore frappé». Programmer un tel logiciel est nuisant, mais demande une certaine dose de génie et de connaissance des réseaux et des systèmes. En tout cas, ces problèmes font une contre-publicité importante aux réseaux publics, les clients ayant peur de «verminer» leurs fichiers.

Steve Jobs a annoncé une station de travail qui, dans le cadre des universités, permettrait de simuler des expériences de chimie et de physique. Selon lui, les performances de cette station seraient de loin supérieures à ce qui sortira en 1987 chez les concurrents. Si cela se confirme, ce serait le premier produit de la société qu'il a créé depuis sa sortie de chez Apple. Mais, de plus en plus, les sociétés annoncent des produits bien avant qu'ils ne soient prêts, et très souvent, des problèmes majeurs surviennent et le produit est abandonné. A ce sujet, Popular Computing a réalisé une enquête sur les produits dans ce cas et il apparaît que près de 70 % des logiciels et 50 % des matériels sont dans cette catégorie des produits qui ne sortent jamais, malgré une annonce officielle, souvent fracassante.

C'est Noël et les jouets abondent un peu partout. Il est frappant de voir que l'intelligence artificielle a vraiment fait des progrès, au moins dans le cadre de son application aux jouets grand public. Des poupées qui parlent, répondent aux questions, poussent des cris de joie lorsqu'on les caline, c'est, malgré tout, une nouveauté. Les jeux de questionnaires qui emmagasinent des quantités incroyables de réponses (plus de 10 000 000 pour certains), c'est une réussite de la haute intégration électronique, mais aussi des algorithmes de structuration et de stockage des bases de données. Je n'aime pas la vogue actuelle pour les jouets représentant des robots agressifs et combattants mais, là aussi, l'électronique arrive en force. Il semble que, pour la première fois, le prix des composants et l'imagination aient permis une invasion de la vraie électronique dans le jouet. Depuis dix ans, ce genre de déclaration est à la mode, mais rien ne semblait plus attrayant que le jouet traditionnel. Il semble qu'enfin le pas soit franchi et que l'électronique et l'informatique aient donné des idées aux fabricants. Il faut dire que lorsqu'on peut intégrer des processeurs de type 65C02 dans un jouet à moins de 20 $, cela peut doper la créativité. Mais, encore une fois, jusqu'à cette année, rien de vraiment novateur ne voyait le jour, mais maintenant le pas est franchi. Vous pourrez même vous offrir le robot dont tout le monde parle depuis vingt ans. Celui qui fait le service au bar et peut passer l'aspirateur. Bien sûr, la période d'acquisition de la connaissance dont il a besoin sera longue, mais quel plaisir après.

Le système d'exploitation le plus populaire à l'heure actuelle, nous en parlions le mois dernier, c'est UNIX. Sa vogue est telle qu'il est maintenant possible de trouver en librairie la collection complète des manuels d'origine, édité par la maison mère, Bell. C'est sans doute la première fois que cette société de recherche en communication et en informatique produit un vrai best-seller et elle ne devait sûrement pas y penser en le publiant au début des années 70. A quand le prix Goncourt pour cet ouvrage en sept volumes ? Ou le Nobel pour l'œuvre complète des laboratoires Bell.

Comme les autres, Apple fait marcher la machine à informations diverses, variées, et plus ou moins plausibles. En débrouillant l'écheveau (et certains sont spécialisés dans cette activité qui tient du jeu d'aventure et du puzzle), il ressort que les produits seraient : un Mac Plus doté d'un 68020, avec le reste peu ou pas modifié (on parle d'un écran un petit peu plus grand, mais alors qu'en est-il de la compatibilité avec le reste de la gamme ?). Vient ensuite l'Open Mac, nom de code actuel Milwaukee. Il sera doté d'origine d'une caisse à peu près vide, c'est-à-dire qu'il reprend le principe de l'Apple II. Une caisse avec huit fentes d'extension, la carte logique, un clavier et c'est tout. Tout le reste sera au choix du client, écran (couleur ou non), résolution, mémoire interne (1 Mégaoctet en standard), stockage de masse (éventuel disque dur interne, disquette...), connexions vers le monde extérieur (imprimantes, réseaux, modems). Pour chaque extension, une fente permet de loger l'interface. Tout cela pour un prix approximativement double de celui du Mac Plus actuel (qui baisse d'ailleurs !!!)

La guerre du disque dur, si elle est moins meurtrière que celle de la crevette dans le golfe de Gascogne, n'en fait pas moins des victimes de marques (oui, oui, le «s» est volontaire). Tecmar et Corvus, vous connaissez ? Peut-être pas si vous êtes depuis peu dans le bain. Ce sont les véritables inventeurs du disque dur. On ne parle plus d'eux, simplement parce que, là où ils demandaient, voilà un an, 1 500 $ pour 20 Mégaoctets, on trouve la même chose pour 500 $ aujourd'hui. Cette ruée de petits fabricants vers ce créneau prometteur pour les années à venir a totalement asphyxié les deux pionniers qui faisaient payer fort cher leur savoir-faire. Le temps de s'adapter au niveau du marché, et la demande s'est déplacée vers les petits nouveaux. Même des sociétés récentes (General Computer) ont du mal à suivre la cadence échevelée de la baisse des prix. Le consommateur trouve son compte dans cette évolution, mais lorsque ceux qui veulent étouffer les autres seront parvenus à leurs fins, les prix risquent de remonter sensiblement. D'où l'ampleur de la vague d'achat. Des revues qui, l'an dernier, testaient cinq disques pour un ordinateur, font une revue de plus de cent différents matériels cette année.

Une nouvelle série de langages font leur apparition sur micro-ordinateurs, les langages objets. Il est possible de programmer avec cette excellente structuration des langages comme Pascal, Prolog, Lisp, et même l'obsolète C et le vieux BASIC. Cette technique permet de considérer un ensemble d'informations comme faisant partie d'une même structure, et de définir cette structure plus ou moins souplement, selon qu'il est possible ou non d'introduire des exceptions, par l'intermédiaire de commandes plus générales, ou d'arborescences. Moralité, voilà un nouveau pas vers la haute structuration des programmes, ce qui ne doit plus prendre au dépourvu les lecteurs de LED-Micro. Tous les récents Pascal se sont mis au goût du jour et vont arriver en force sur les plus petits des micros, mais il faudra être un peu plus patient. Il en est de même pour Lisp, Prolog et C. Par contre, des BASIC objets existent maintenant aux U.S.A., pour tous les types de micros. Avec un peu de chance, ils seront en France vers le printemps.

Signe des temps, la revue INFOWORLD, encore elle, consacre dix pages cette semaine à la description des principales stations de travail. Sun, Appolo, VAX, IBM et Hewlett-Packard sont ainsi confrontés dans une revue spécialisée dans la micro à tendance professionnelle. Malgré tout, qui aurait pu imaginer que ce type de performance serait à la portée d'un particulier ou d'une société de trois personnes. C'est pourtant le cas, avec des matériels allant de 60 000 francs à plus de 1 300 000 francs. Pour 100 000 francs, vous aurez maintenant une station Sun, avec un écran couleur 1 024 × 1 024, 2 Mb de mémoire centrale, une souris à trois boutons. Un bon début. Hélas, les prix indiqués sont ceux des U.S.A. et en France ce sera presque le double. Mais, comme je vous l'avais dit l'an dernier, les prix baissent, principalement grâce aux quantités de matériels vendus, qui augmentent sans cesse. Pensez, 7 500 $ pour le moins cher, c'est le prix d'un IBM PC haut de gamme.

LA BIBLIOTHEQUE TECHNIQUE DES EDITIONS FREQUENCES

offre des ouvrages techniques **très actuels** rédigés par des auteurs passionnés et impliqués complètement dans le sujet qu'ils traitent.

Vous trouverez soit des études approfondies sur les techniques ou les technologies de votre métier, soit des initiations théoriques et pratiques de techniques ou technologies que vous désirez approcher ou mieux cerner. Vous découvrirez au verso la description des ouvrages récemment parus ainsi que les commentaires sur les additifs d'éventuelles rééditions.

Les titres dont la parution est prochaine sont également mentionnés.

La page suivante comporte la liste complète des titres, leurs codes et leurs prix.

VIENT DE PARAITRE

LES BASES DE L'ELECTRONIQUE

de Raymond Breton

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 simples.

Chapitre I :
Conseils et tours de main.....5
— Le poste de travail
La sécurité
L'équipement
L'organisation
— La réalisation réelle
Travail sur table
Le circuit imprimé
Les méthodes de dépôt de la réserve
Le croquis d'implantation
Savoir souder
— Les préfixes
Chapitre II :
Les composants passifs..... 11
- Courant continu - Résistance
Notions de base
Groupements de résistances
- Application : un adaptateur thermomètre
Définition du problème
Réflexion, manipulations et réalisation
- Le courant alternatif
Représentation imagée
- La self
Le champ magnétique
Variation d'un champ magnétique
Charge et décharge d'une self
Le transformateur
- Le condensateur
Constitution
Charge et décharge
Comportement du condensateur en alternatif
Exemple d'utilisation
- Application : un capacimètre balistique
Cheminement d'une idée
Essais, réalisations
Chapitre III :
Les composants actifs simples.....21
— La diode
La diode en direct et en inverse
Redressement simple alternance
Redressement double alternance
Redressement en pont
Vérification du bon état d'une diode
— Application : une alimentation secteur pour notre thermomètre
Calcul des éléments
Etablissement d'une régulation
— Cas particulier : le condensateur de découplage
— Le transistor
Description
Amplification
Surlignage important
Polarité
Nécessité d'un courant de repos
Les trois montages de base
— Application : des alimentations stabilisées
Alimentation à un seul transistor
Alimentation à deux transistors
Alimentation à trois transistors
— Le transistor à effet de champ
Description
Montage pratique
Les trois montages de base
— Application : détection d'une flamme
Principe de base
Réalisation pratique
Mise au point et évaluations pédagogiques
— Le thyristor
Description, caractéristiques
Montage de base d'un thyristor
Extension en triac
Principe de fonctionnement d'un gradateur
— Application : une serrure électronique
Cahier des charges
Cheminement de la réflexion
Adjonctions et perfectionnements
— Le transistor unijonction
Description et montage pratique
Limites d'emploi
— Application : un générateur en dents de scie
Schéma de base
Charge à courant constant
Reprise du signal à haute impédance
Améliorations du flanc descendant
Domaines d'utilisation
— Tableau calculé des paramètres d'un montage à unijonction
Chapitre IV :
Les composants actifs complexes.....45
- Réflexions
- L'ampli opérationnel
Présentation et schéma
Mécanisme de l'ampli opérationnel
Défauts de l'ampli opérationnel
- Application : un portier vocal
Cahier des charges
- Le circuit RC
Le circuit RC simple
Les réseaux RC complexes
- Tableau calculé des fréquences de coupure d'un réseau RC
- Les circuits spécialisés
Méthodes de description
Les régulateurs de tension
Le temporisateur 555
- Les circuits numériques
La numérotation binaire
Les conventions
Le 4011
Les autres circuits logiques
Chapitre V
Capteurs et transducteurs..... 67
- Les capteurs
Le toucher
La vue
L'ouïe
- Les transducteurs
Le déplacement
Emission d'une onde
Conclusion.....79

PROCHAINEMENT

Collection jaune **Etude autour du 6809 (constructions et logiciels)** de Claude Vicidomini

Collection noire **Les techniques du Son (1er tome)** : collectif d'auteurs dirigé par Denis Mercier
La Synthèse Musicale par Ordinateur de Frédéric Levé

Collection noire (format 165 × 240)

	Réf.	Prix TTC
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	E 15	140 F
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	E 01	165 F
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	E 05	155 F
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	E 04	154 F
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	E 02	92 F
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	E 03	155 F
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	E 06	150 F
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	E 22	150 F
SELECTION DE L'AUDIOPHILE - TOME 1 : L'ELECTRONIQUE 256 p.	E 12	155 F
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.	E 13	165 F
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	E 25	140 F

Collection rouge (format 135 × 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»	L 07	68 F
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	L 10	130 F
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 .	L 09	65 F
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 !	L 11	85 F
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	L 14	95 F
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 × 36 et des illustrations abondamment commentées	L 20	130 F

Collection jaune (format 210 × 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 ! (épuisé)	P 08	115 F
INITIATION A LA MICRO-INFORMATIQUE COURS 1er CYCLE - LE VOLUME 1 de Claude Polgar - 272 p.	P 16	130 F
INITIATION A LA MICRO-INFORMATIQUE COURS 1er CYCLE - LE VOLUME 2 de Claude Polgar - 208 p.	P 17	130 F
INITIATION A LA MICRO-INFORMATIQUE COURS 1er 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.	P 27	190
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	P 19	95 F
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	P 18	95 F
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	P 21	135 F
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	P 23	140 F
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...	P 24	130 F
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	P 26	155 F
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 !	P 28	150 F
INITIATION A LA VIDEO LEGERE - THEORIE ET PRATIQUE de Claude Gendre - 72 p. - Choix d'un standard ? Camescopes 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	P 29	100 F
LES MONTAGES ELECTRONIQUES de Jean-Pierre Lemoine - 276 p. - Véritable encyclopédie. Plus de 1 000 dessins. 25 montages originaux	P 30	250 F
LE TELEPHONE ET LES RADIOTELEPHONES de Roger-Charles Houzé - 96 p., 73 schémas	P 31	130 F
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 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) ouvrages(s) ci-dessous référencé(s) que je coche d'une croix :

							(épuisé)		
E 01 ☐	E 02 ☐	E 03 ☐	E 04 ☐	E 05 ☐	E 06 ☐	L 07 ☐	P 08 ☐	L 09 ☐	L 10 ☐
L 11 ☐	E 12 ☐	E 13 ☐	L 14 ☐	E 15 ☐	P 16 ☐	P 17 ☐	P 18 ☐	P 19 ☐	L 20 ☐
P 21 ☐	E 22 ☐	P 23 ☐	P 24 ☐	E 25 ☐	P 26 ☐	P 27 ☐	P 28 ☐	P 29 ☐	P 30 ☐
P 31 ☐	P 32 ☐								

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 .....

LE TURBO PASCAL

Charles-Henry Delaleu

Contrairement à ce que nous vous avions annoncé dans nos précédents numéros, nous n'allons pas vous proposer un cours sur DBASE III ou TEXTOR mais vous présenter chaque mois un nouveau produit en détail. Nous commençons avec le TURBO PASCAL et ses outils de programmation.

Le PASCAL, conçu à l'origine comme un langage d'initiation à la programmation, est devenu aujourd'hui une référence. Il a dépassé le stade pédagogique pour devenir un langage courant comme le FORTRAN, le COBOL ou PL/I. Dans le monde de la micro-informatique, le PASCAL de référence fut rapidement le PASCAL USCD. Il faut bien admettre que désormais il se vend beaucoup moins et que Borland a réussi un coup de maître avec son TURBO PASCAL.

En effet ce langage possède plusieurs avantages dont celui d'être beaucoup plus économique que les autres PASCAL. Il est rapide et s'adapte facilement au format IBM-PC.

Enfin, grâce à ses additifs (TURBO EDTOR, TOOL BOOK et GRAPHIX TOOL BOOK), il peut être un excellent outil de développement.

LE TURBO PASCAL : PRESENTATION

L'historique : Lorsque M. Wirth développa le PASCAL, il voulait réaliser un langage de haut-niveau, qui obligerait l'utilisateur à réaliser une programmation structurée.
Conçues dans un but universitaire, les performances de ce langage allaient vite dépasser le cadre éducatif pour s'imposer en programmation professionnelle ou personnelle.

Le TURBO : Le TURBO PASCAL fut élaboré par un français. Le langage en version de base suit d'assez près la définition du Pascal standard telle qu'elle fut publiée par **Jensen et Wirth dans PASCAL USER MANUAL AND REPORT.
Il possède
– Un éditeur
– Un compilateur
– Une commande d'exécution
– Des commandes de sauvegarde et de directory
La compilation de TURBO PASCAL est très rapide mais il convient de noter qu'il est impossible de sauvegarder en mémoire de masse un programme compilé. En effet, avant chaque exécution, le programme doit être recompilé.

Les extensions
Afin de rendre plus performant le TURBO PASCAL, Borland a eu l'excellente initiative de mettre sur le marché des outils efficaces.
a) Dans un premier temps, il convient de noter l'existence de TURBO TUTOR qui est un cours de TURBO PASCAL.
b) Dans la version de base, l'éditeur de Turbo Pascal n'est pas une référence en soi. Pour pallier à cet inconvénient, Borland a réalisé TURBO EDITOR qui est un logiciel de traitement de texte spécialisé dans l'écriture de longs programmes en TURBO PASCAL.
c) Lorsque PASCAL apparut sur le marché, les spécialistes en développement de logiciels critiquèrent longuement ses faibles performances dans le domaine de la gestion des fichiers.
Pour remédier à cette faiblesse, et pour ne pas avoir les mêmes critiques que le Pascal de base, Borland a développé TOOL-BOX qui améliore très nettement les performances du Turbo Pascal dans le domaine de la gestion des informations.
d) Un beau dessin est souvent plus parlant qu'un long texte ; il ne manquait que le graphisme à Turbo Pascal. GRAPHIX TOOL-BOX autorise toutes sortes de dessins et animation dans l'environnement du Turbo Pascal.

INTRODUCTION A TURBO PASCAL

Le Turbo Pascal offre de nombreuses extensions en version de base, telles que :
– Variables d'adresses absolues
– Manipulations de bits et d'octets
– Accès direct à la mémoire et aux ports du micro-processeur
– Chaînes dynamiques
– Classement libre des sections dans la partie déclaration
– Support intégral des routines du système d'exploitation
– Génération du code machine en ligne
– Fichiers inclus
– Opérations logiques sur les entiers
– Modules de recouvrement
– Chaînage de programmes avec variable communes
– Fichiers de données à accès direct
– Constantes structurées
– Fonction de conversion de type

Pour les IBM et compatibles :
- Couleurs
- Graphique
- Tortues
- Fenêtres
- Son

Turbo Pascal existe en différentes versions de système d'exploitation. Il est exécutable sous :
- CP/M80
- MS/DOS
- PC/DOS
- CP/M-8)

Une fois les fichiers en ligne sur l'unité de mémoire de masse, le lancement de Turbo Pascal est très simple. Il suffit de taper TURBO puis d'appuyer sur la touche ENTER. Le système demarre et le message suivant apparaît.

```
------------------------------------------
TURBO Pascal system          Version 3.02A
                                   PC-DOS

Copyright (C) 1983,84,85,86 BORLAND Inc.
------------------------------------------

b/w   display 80x25


Include error messages (Y/N)?
```

Fig. I : premier affichage de Turbo Pascal.

Il offre la possibilité de lister les erreurs lors de la compilation. Ensuite, le menu principal de TURBO PASCAL suit.

```
Logged drive: C
Active directory: \PASCAL

Work file:
Main file:

Edit      Compile  Run    Save

Dir       Quit  compiler Options

Text:       0 bytes
Free:  62024 bytes
```

Fig. II : Le menu principal.

Les éléments de base du langage

Comme la plupart des langages, Turbo Pascal utilise un vocabulaire composé de lettres ou de chiffres et de symboles spéciaux. Il existe un certain nombre de mots réservés, ainsi qu'une bibliothèque d'identificateurs spéciaux.

* **absolute**	* **external**	**nil**	* **shl**
and	**file**	**not**	* **shr**
array	**forward**	* **overlay**	* **string**
begin	**for**	**of**	**then**
case	**function**	**or**	**type**
const	**goto**	**packed**	**to**
div	* **inline**	**procedure**	**until**
do	**if**	**program**	**var**
downto	**in**	**record**	**while**
else	**label**	**repeat**	**with**
end	**mod**	**set**	* **xor**

Fig. III : Les mots réservés.

Addr	Delay	Length	Release
ArcTan	Delete	Ln	Rename
Assign	EOF	Lo	Reset
Aux	EOLN	LowVideo	Rewrite
AuxInPtr	Erase	Lst	Round
AuxOutPrt	Execute	LstOutPtr	Seek
BlockRead	Exit	Mark	Sin
BlockWrite	Exp	MaxInt	SizeOf
Boolean	False	Mem	SeekEof
BufLen	FilePos	MemAvail	SeekEoln
Byte	FileSize	Move	Sqr
Chain	FillChar	New	Sqrt
Char	Flush	NormVideo	Str
Chr	Frac	Odd	Succ
Close	GetMem	Ord	Swap
ClrEol	GotoXY	Output	Text
ClrScr	Halt	Pi	Trm
Con	HeapPtr	Port	True
ConInPtr	Hi	Pos	Trunc
ConOuPtr	IOresult	Pre	UpCase
Concat	Input	Ptr	Usr
ConstPtr	InsLine	Random	UsrInPtr
Copy	Insert	Randomize	UsrOutPtr
Cos	Int	Read	Val
CrtExit	Integer	ReadLn	Write
CrtInit	Kbd	Real	WriteLn
DelLine	KeyPressed		

Fig. IV : Identification standard.

Il serait trop long, dans le cadre de cet article, d'expliquer l'ensemble de ces mots.

Les performances
Une récente étude indique que Turbo Pascal serait 26 fois plus rapide que MS-PASCAL en compilation. En mode exécution, la différence serait dans un rapport de 1 à 2 en faveur de Turbo Pascal. Enfin, pour un programme donné, la taille du programme code serait trois fois plus petite sur Turbo Pascal par rapport à MS-PASCAL.
En fait, le compilateur de TB compile en moyenne de 80 à 100 lignes par seconde.
La taille maximum du programme code accepté est de 64 Ko en mode exécution.

Turbo Pascal et MS-DOS
Un des grands avantages de TP est de pouvoir utiliser les procédures se rapportant aux répertoires sous MS-DOS.
Il est possible d'utiliser les commandes :
– Ch Dir
– Mk Dir
– Rm Dir
– G et Dir
De même, lors du listage par la commande Dir en menu principal, l'utilisation de jocker est autorisée :
ex : DIR *.PAS

Les options de compilation
Les options de compilation sont assez nombreuses et bien utiles lors de la mise au point d'un programme.
Elles permettent :
– Définition de la taille minimum du segment code
– Définition de la taille minimum du segment de données
– Définition de la taille minimum et maximum de la mémoire dynamique libre
– Détection des erreurs à l'exécution
Grâce à cette dernière commande, il est possible de trouver dans le code source l'endroit exact où est placée une erreur dans le listing.

TURBO EDITOR

En fait, Turbo Editor est d'abord un programme de traitement de texte spécialisé dans la réalisation d'application tournant sous Turbo Pascal.
Il s'agit d'un éditeur de texte complet qui permet de manipuler des informations texte comme :
– Des caractères simples
– Des groupes de caractères : des mots, des lignes
– Des groupes de lignes : flots de texte
– Des fenêtres de flots de texte
– Des blocs de lignes
– Des fichiers de lignes
– Des affichages écran
Afin de fonctionner correctement, TURBO EDITOR devra obligatoirement être utilisé sur un TURBO PASCAL égal ou supérieur à la version 3.0 et un MS-DOS ou PC.DOS version 2.0 ou plus.
Avec TURBO EDITOR, il est possible de créer son propre éditeur de texte. Deux exemples intervenants sont fournis par le progiciel. Un éditeur classique portant le nom de FIRST-ED présente les fonctions généralement implantées sur un éditeur. La deuxième démonstration est beaucoup plus intéressante. Il s'agit d'un éditeur sophistiqué portant le nom de MICROSTAR. Microstar est un éditeur de qualité. Il est conçu pour présenter toutes les capacités de TURBO EDITOR. Les principales

caractéristiques sont :
– L'annulation des commandes de WORDSTAR
– Une fonction UNDO
– Une impression Multitache rapide en arrière plan
– Un affichage des répertoires comme dans SIDEKICK
– Un système de menu déroulant
Le programme MICROSTAR est fourni avec le code source. Il convient donc de le compiler avant son utilisation. Inversement, cette possibilité autorise l'utilisateur à apporter ses propres idées. De plus, ceci servira d'excellent exemple à toute personne intéressée par le PASCAL.

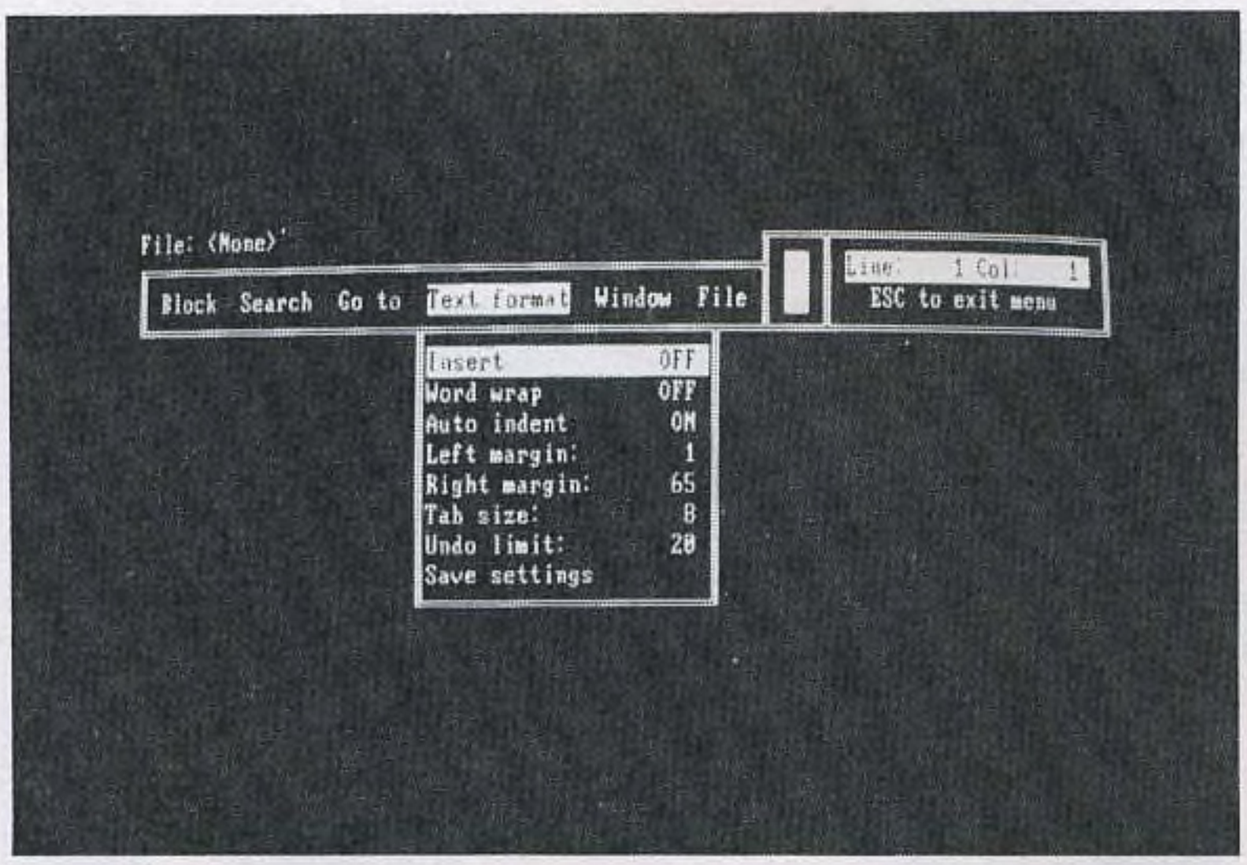


Fig. V : Menu MICROSTAR.

TOOL BOX
Nous l'avons déjà écrit, mais à l'origine, PASCAL fut un langage d'initiation à la programmation structurée. De ce fait, l'une de ses plus grandes faiblesses concerne l'utilisation de fichiers de données.
Afin de pallier ces défauts, certains fabricants de compilateurs PASCAL ont ajouté des extensions, tels les fichiers indexés.
TOOL BOX est un ensemble de programmes outils qui augmente les performances du Turbo Pascal en gestion de fichiers.
Il contient trois possibilités importantes :
– Un système d'accès
– Un système de tri
– Un système d'installation

Le système d'accès TURBO
Il permet d'accéder à des enregistrements dans un fichier, en utilisant des clés. De plus, vous pouvez également accéder aux enregistrements dans un ordre défini. Le système d'accès turbo est architecturé en arbre binaire. Il est donc surtout utilisé pour des accès rapides à des informations.

Le système de tri TURBO
Il utilise un algorithme de tri rapide afin d'assurer un tri performant. Avec cet outil, vous pouvez trier n'importe quel type de données, que ce soit un article simple ou sur des clés multiples, ainsi que différents types de données dans le même programme.

GINST : le système d'installation général
Cet utilitaire permet de configurer TOOL BOX en fonction du type de terminaux utilisé.

Nota : Des cas bien fournis servent d'exemples aux différents chapitres de TOOL BOX.
Les possibilités en TOOL BOX sont :
1) Accès standard aux données
   1-1 Accès séquentiel
   1-2 Accès aléatoire
2) Structure des arbres
   2-1 Clés
   2-2 Articles
   2-3 Pages
3) Gestion des arbres
   3-1 Insertion d'une clé
   3-2 Suppression d'une clé

Résumé des fonctions
1- Initialisation et mise à jour des fichiers de données
2- Initialisation des fichiers d'index
3- Mise à jour des fichiers d'index
4- Recherche dans les fichiers d'index

Les pages 41 à 48 du manuel sont très intéressantes car elles décrivent les principales routines utilisées en gestion de fichiers :
1- Déclaration
2- Création de fichiers
3- Ajout d'enregistrement
4- Recherche d'enregistrement
5- Suppression d'enregistrement
6- Recherche séquentielle
7- Champ status (fichiers sans index)
8- Restauration d'index

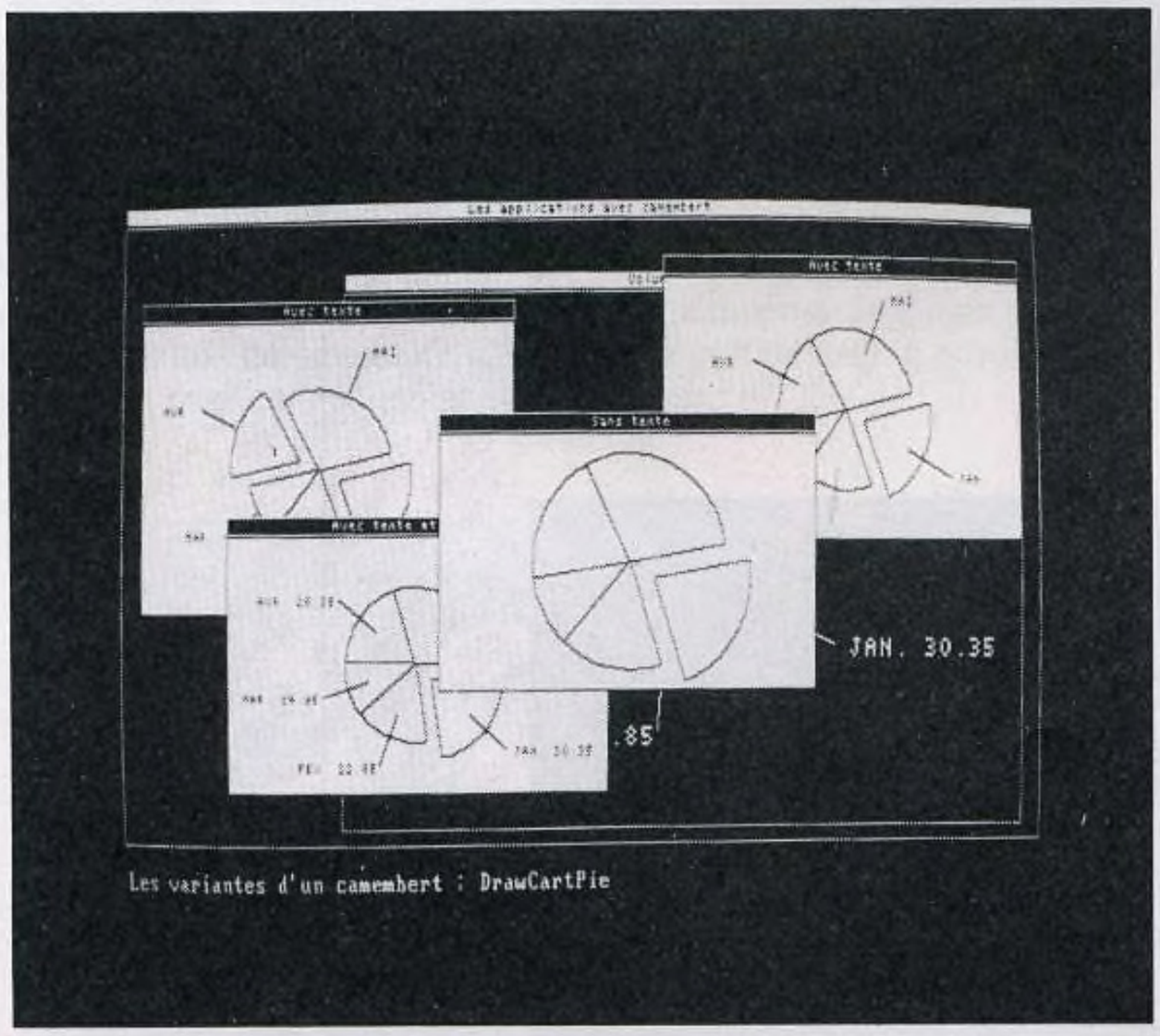
GRAPHIX TOOL BOX
Grâce à GRAPHIX TOOL BOX vous pouvez développer des graphiques monochromes haute résolution afin de donner plus d'impact à vos programmes. GRAPHIX TOOL BOX est architecturé pour surpasser les cartes graphiques IBM ou Hercules.
Turbo Graphix TOOL BOX constitue un package aux possibilités diverses. Les procédures élémentaires permettent de dessiner :
– des points
– des lignes
– des rectangles avec un hachurage facultatif
– des elipses
– des cercles
Pour des applications de gestion, ou pour des applications scientifiques, le graphisme est un outil très important. Aussi, avec Graphix Tool Box, rien n'est plus facile que de créer :
– des camemberts avec en-têtes
– des histogrammes avec hachurage programmable
– des traces différentes
– des lissages de courbes

- des ajustements de courbes
- des modélisations de lignes et de solides
- des axes de coordonnées identifiés
- des graphes de toutès formes
- des rotations
- des translations

Turbo Pascal Graphix Tool Box comprend dans sa disquette programme :

- un code source pour chaque exemple
- des outils pour dessiner des graphiques simples
- des outils pour dessiner des graphiques complexes
- des routines pour enregistrer et récupérer les graphiques sur disque
- des outils pour envoyer les images d'écran vers les imprimantes compatibles IBM
- une gestion complexe des fenêtres graphiques
- deux différentes polices de caractères pour les légendes
- le choix du style du tracé de ligne
- des routines de dessins rapides

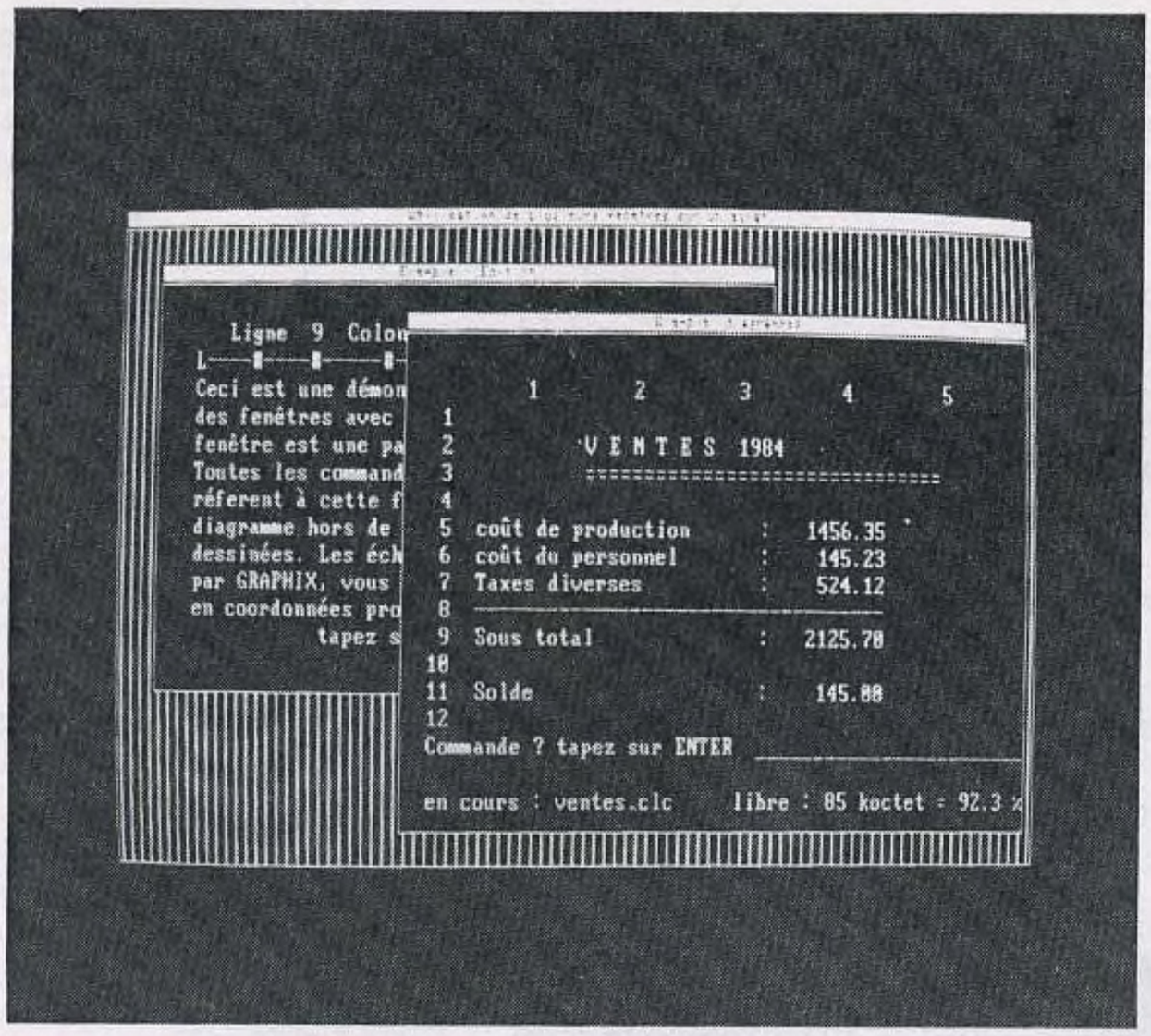
Les différentes figures suivantes donneront une indication sur les possibilités de GRAPHIX TOOL BOX.
Sur la figure 6, nous avons une représentation multi-fenêtres d'un camembert.



Les variantes d'un camembert : DrawCartPie

Fig. VI : Les fenêtres. a) Les camemberts.

Sur la figure 7 nous avons en fond une courbe de base en second plan associée à une courbe de lissage qui donne une meilleure présentation. Sur la fenêtre, en avant, une autre courbe est représentée.
Sur la figure 8, différents exemples de présentation sont fournis. La sinusoïde est dessinée de plusieurs manières.
Sur la fenêtre inférieure gauche le tracé est effectué à l'aide de petits ronds. Sur la fenêtre inférieure droite, il s'agit d'un trait plein avec présentation de petites croix.
Enfin, sur la fenêtre centrale, nous avons un autre choix de présentation.



b) Les fenêtres texte.

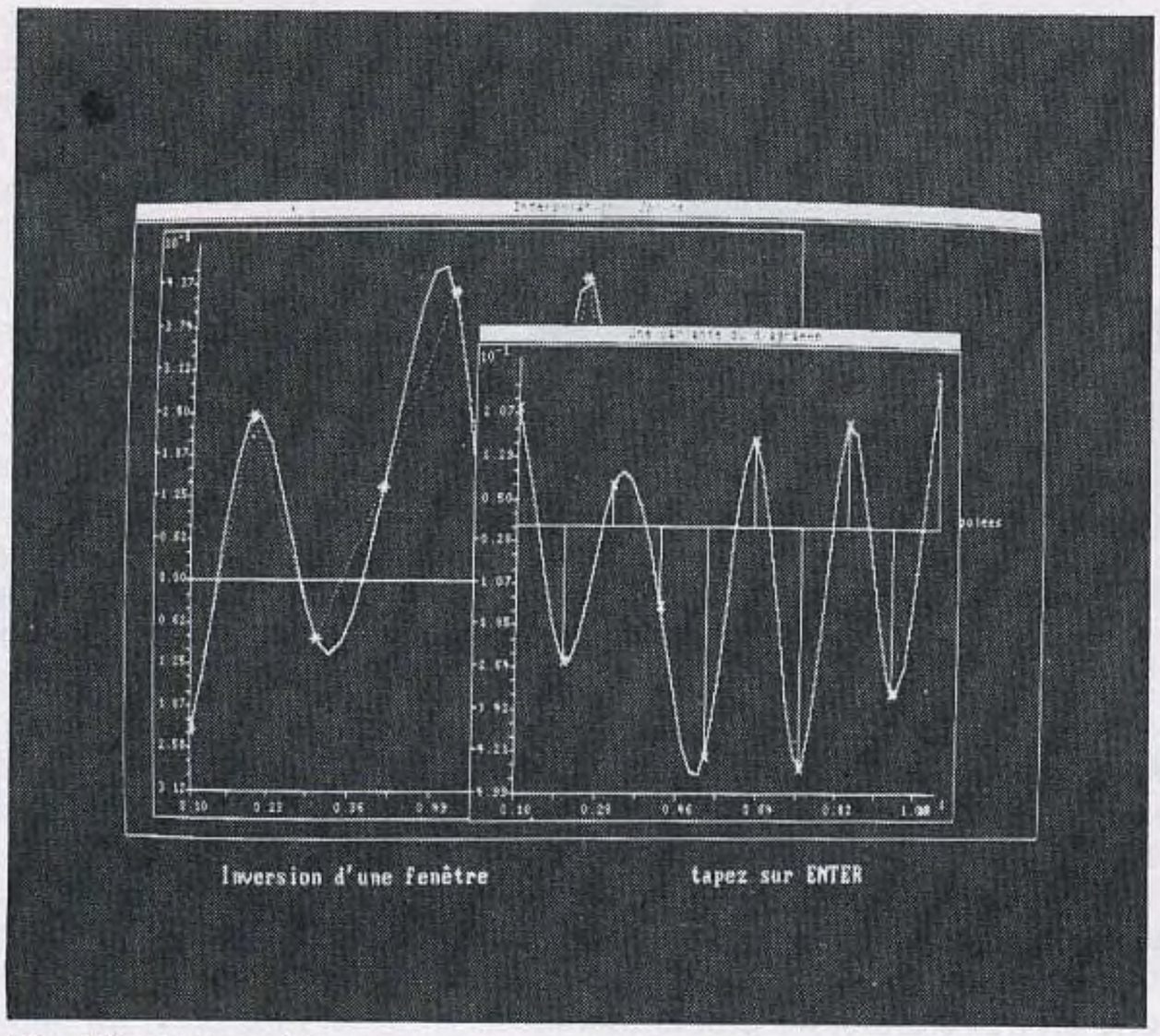


Fig. VII : Les courbes.

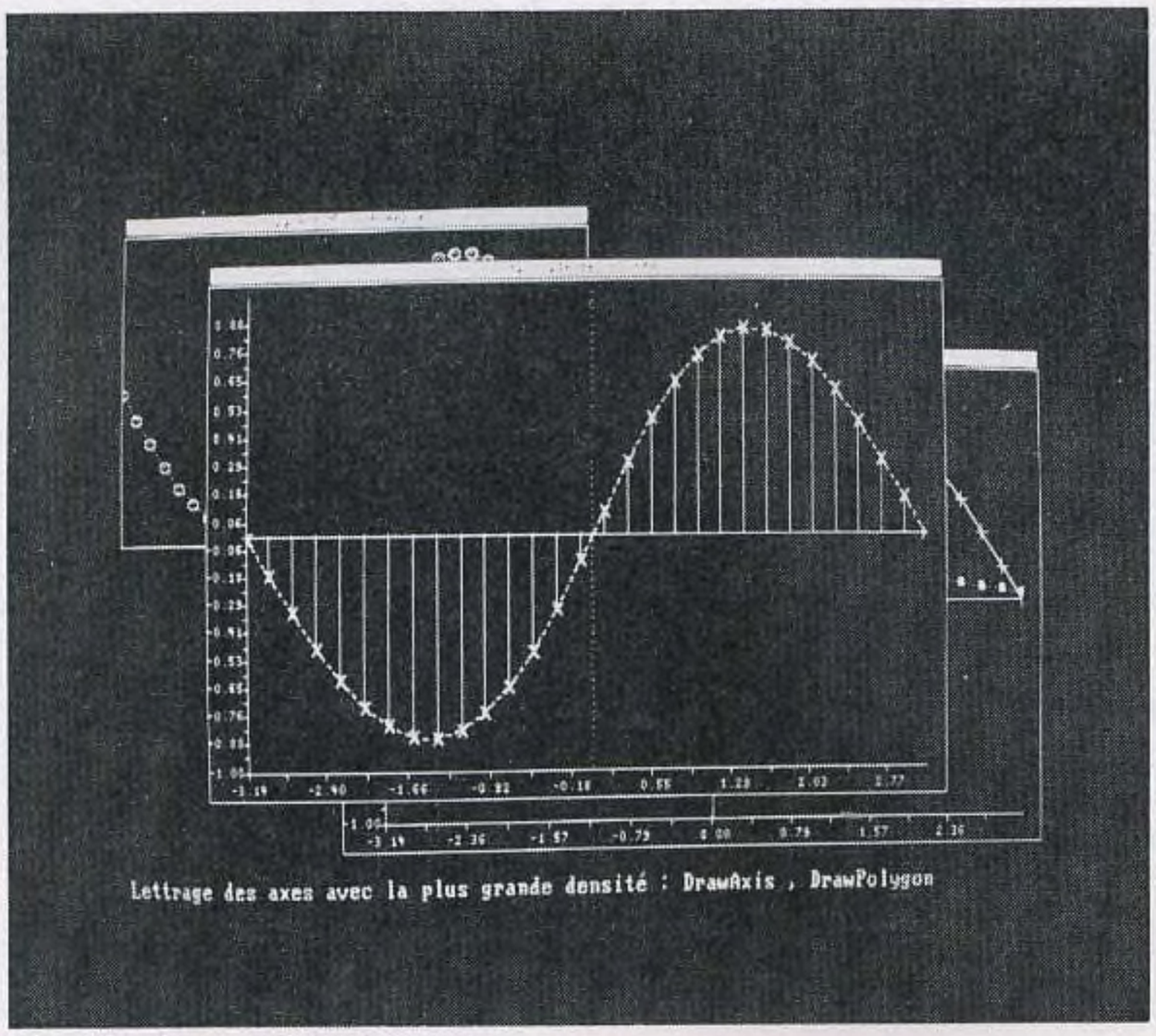


Fig. VIII : Les présentations.

CONCLUSION

Il ne fait aucun doute, pour personne, que les prix des machines en micro-informatique ont subi de très nettes baisses. BORLAND, avec ses différents progiciels, suit le même chemin. Il semble que cela réussisse parfaitement.

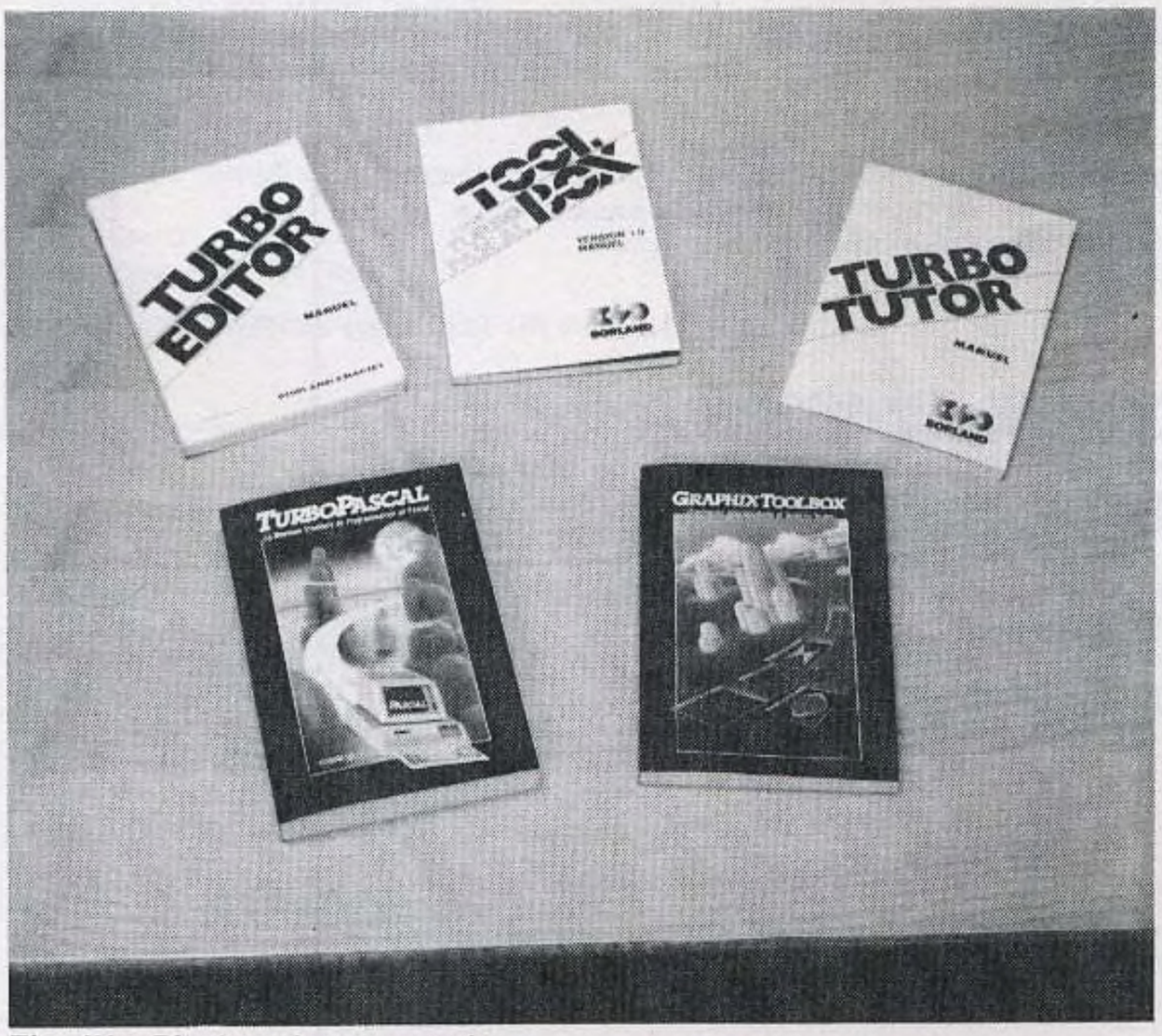


Fig. IX : Photos des manuels.

Chaque disquette est accompagnée d'un manuel bien réalisé et bien documenté. Nous regrettons seulement la qualité des reliures, assez médiocre. Après quelques recherches, certaines pages sautent très vite.
Nous avons bien aimé les performances des produits TURBO PASCAL. Pour être parfait, il aurait fallu pouvoir enregistrer sur disque les programmes codes et simplifier les commandes d'utilisation. L'emploi systématique de la touche de contrôle est plutôt pénible.
En conclusion, nous dirons que TURBO PASCAL est sans aucun doute un des meilleurs choix pour qui désire programmer en PASCAL sur micro-ordinateur.

ADDITIF

Pour cloturer cet articule, il nous a semblé intéressant de présenter un programme en TURBO PASCAL.
Il s'agit d'un programme réalisant un nombre aléatoire ; pour en comprendre le déroulement, il suffira de lire les parties remarque du programme.

```
        random.pas

program RandomTest;
{
        This program tests out the Randomize procedure.  It also
        calculates a chi-square value as a test of Random itself.
        Chi-square values between 3 and 16 are desirable, with
        values close to 8.3 being optimum.


}
var
  S1,S2,Indx,Jndx,Count   : Integer;
  Sum,T,NP                : Real;
  Tally                   : array[0..9] of Integer;
{
   Randomize Procedure For MS-DOS & PC-DOS Turbo Pascal


   This new Randomize has two Integer parameters.  If they are both 0, then
the random number seed is set randomly.  If either of the parameters is
nonzero, then they are both stored directly into the 32 bit seed.


   To set the seed randomly (Randomize(0,0)), the procedure calls MS-DOS
to get the current time.  This is a 32 bit value, which is also stored
directly into the seed.  On some systems, (i.e. the NCR Decision Mate V),
the clock does not tick, so the time never changes.  Randomize checks this,
and if the clock hasn't changed after a Delay(100), it asks the user to hit
a key.  While waiting for the key, it continuously increments two counters.
These are then stored into the seed.


{ Please note:  This routine is for MS-Dos/PC-Dos Turbo ONLY! }

procedure Randomize(I,J: Integer);

var
  RSet     : record
               AX,BX,CX,DX,BP,SI,DI,DS,ES,Flags: Integer;
             end;
  Ch       : Char;
```

```
begin
  if (I=0) and (J=0) then begin      { Generate a random random number seed }
    RSet.AX:=$2C00;                                    { DOS time of day function }
    MSDos(RSet);
    I:=RSet.CX;                                  { Set I and J to the system time }
    J:=RSet.DX;
    Delay(100);    { This delay may have to be increased for faster systems }
    MSDos(RSet);
    if (I=RSet.CX) and (J=RSet.DX) then begin          { Clock isn't ticking }
      I := 0;
      J := 0;
      while KeyPressed do
        Read(Kbd,Ch);                                  { Clear keyboard buffer }
      Write('Hit any key to set the random number generator: ');
      repeat
        I := I+13;
        J := J+17
      until Keypressed;
      Read(Kbd,Ch);                                    { Absorb the character }
      WriteLn
    end
  end;
  MemW[DSeg:$129]:=I;  { This is the core of the routine: store a 32 bit }
  MemW[DSeg:$12B]:=J;  {  seed at locations DSeg:$0129...DSeg:$012C      }
end; { of procedure Randomize }


begin  { main body of program RandomTest }
  Writeln('Enter count <= 0 to end program');
  repeat
    Write('Enter count:           ');          { get # of samples }
    ReadLn(Count);
    if Count > 0 then begin                    { do random number test }
      Write('Enter seeds (S1 S2):  ');         { get 2 integers for seed }
      ReadLn(S1,S2);
      Randomize(S1,S2);                        { set random number seed }
      FillChar(Tally,SizeOf(Tally),0);         { clear tally array }
      for Indx := 1 to Count do begin          { generate Count numbers }
        Jndx := Random(10);                    { range is 0..9 }
        Tally[Jndx] := Tally[Jndx] + 1         { count how many of each }
      end;
      Sum := 0.0;                              { clear sum for X^2 }
      NP := Count/10.0;                        { theoretical number for each }
      for Indx := 0 to 9 do begin              { for each possible result do }
        Write(Tally[Indx]:5);                  { write total for that value }
        Sum := Sum + Sqr(Tally[Indx]-NP)/NP    { and calculate X^2 }
      end;
      WriteLn;
      WriteLn('Chi-Square (9 degrees of freedom) = ',Sum:8:3)
    end
  until Count <= 0
end. { of program RandomTest }
```

Fig. X : Listing : programme générant un nombre aléatoire.

BON DE COMMANDE

Pour compléter votre collection de Led-Micro ·

A retourner aux EDITIONS FRÉQUENCES 1, boulevard Ney - 75018 Paris

Je désire le n° ☐

1 2 3 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 27 28 29 30 31 32 33 34 35 36 37

(cocher le ou les n^{os} désirés).

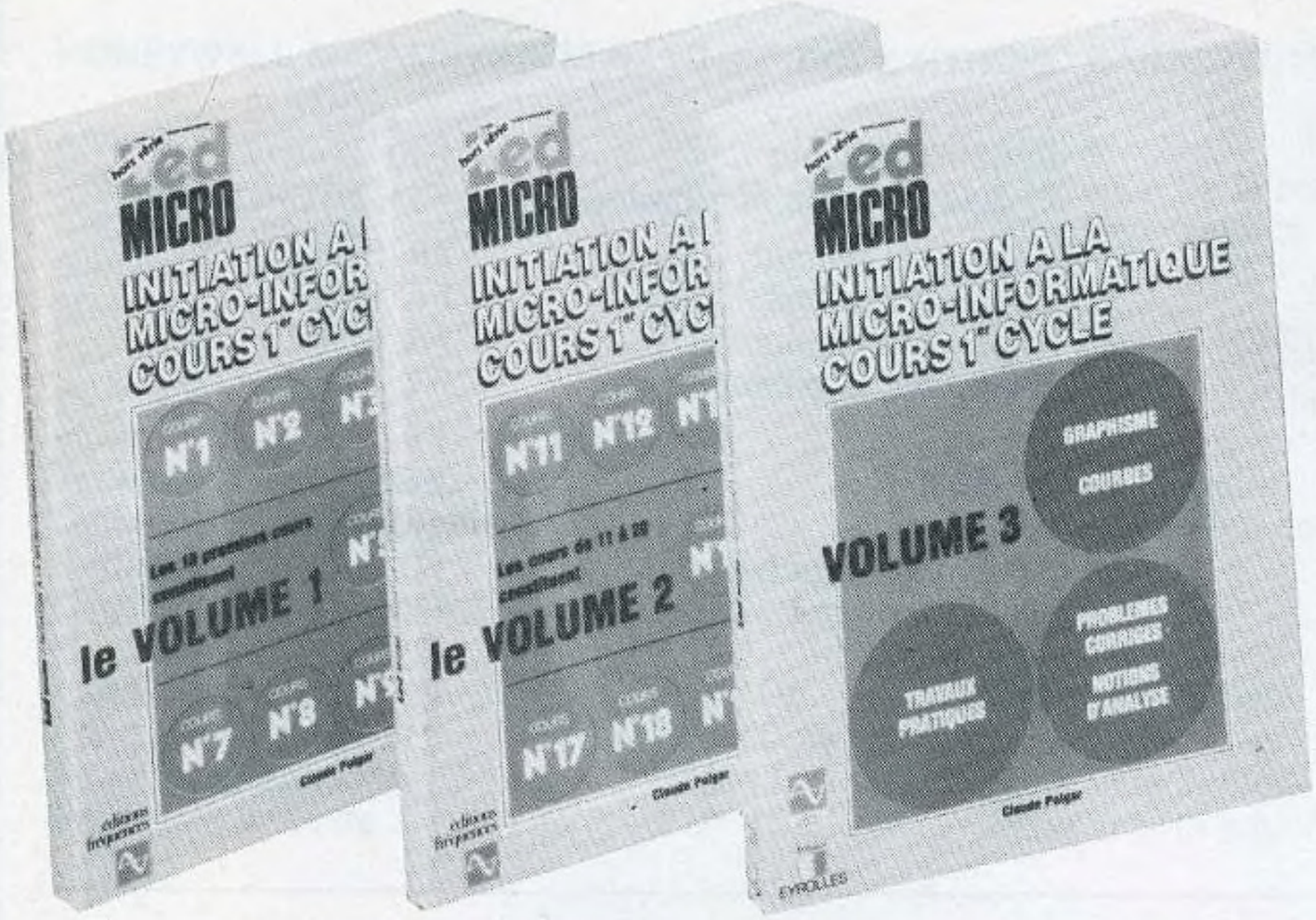
AU PRIX DE 22 F par numéro (port compris).

Je joins à la présente commande le montant de ..... F par CCP ☐ ch. bancaire ☐ mandat ☐

NOM ..... Prénom .....

Adresse .....

Ville ..... Code postal .....



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 × 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 :

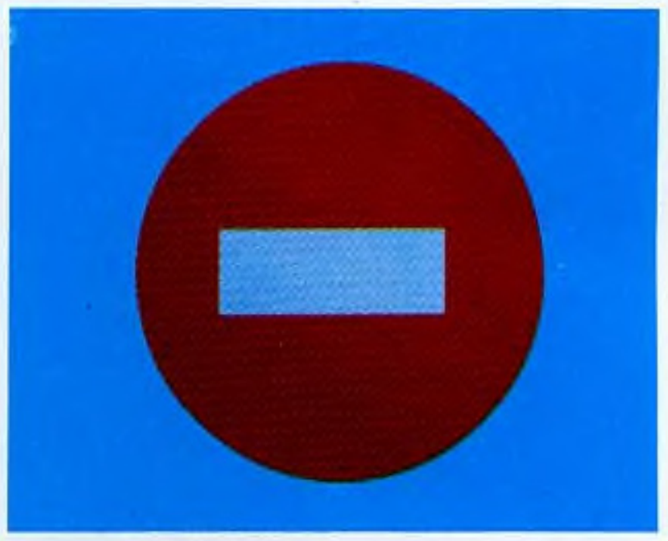
J'enseigne les mathématiques dans une Université de Sciences Humaines et j'ai été amenée, alors que n'avais moi-même reçu aucune formation à la micro-informatique, à initier des étudiants de 1re année de Mathématiques et Sciences Sociales (MASS) à la programmation en S-BASIC (sur Goupil-3), dans le but 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 nos 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 1er 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 ENTIERE.
★ Beaucoup de programmes utilisent des assemblages de ces instructions et opérateurs... dont la combinaison n'est pas toujours facile.

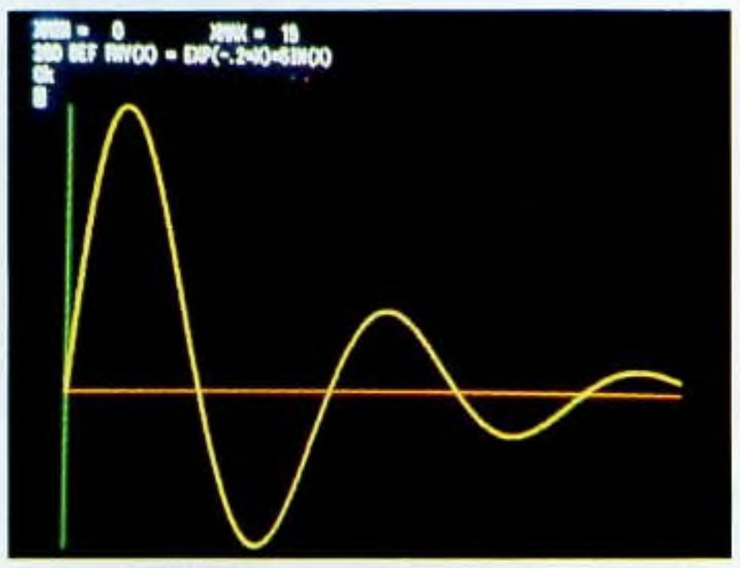


3.17 Compléments

★ 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 éléves : sur la précision et les erreurs dues à l'arrondi, sur la sélection, les boucles.

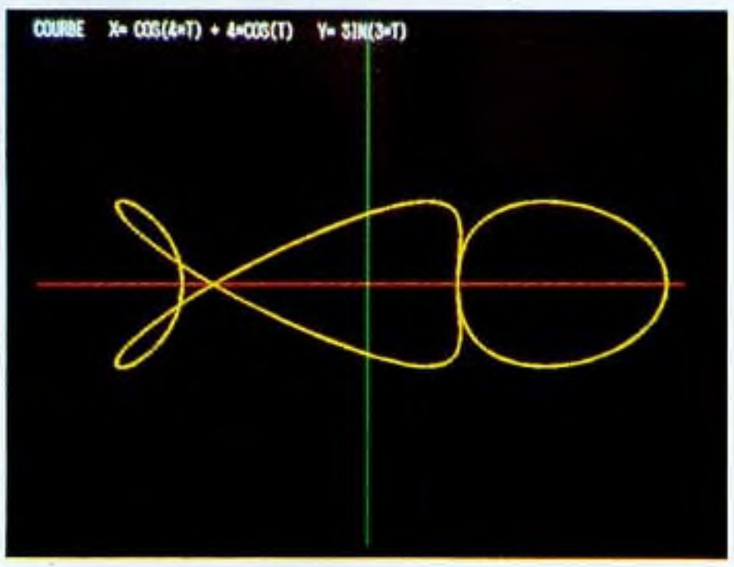
3.18 Graphisme

★ 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).



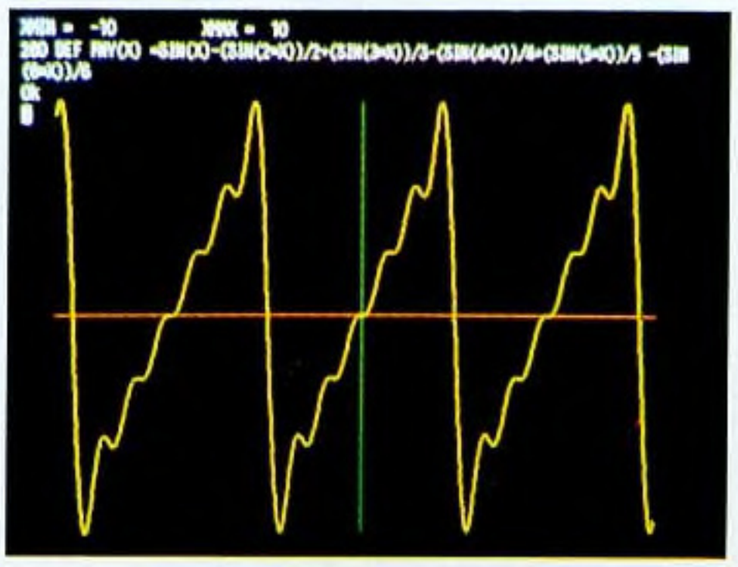
3.19. Dessin des courbes

★ 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 21×27, 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