hors série

Led MICRO

PROGRAMMATION
COURS 2ème CYCLE

COURS
N°25
Suite
2e cycle
N°5

COURS DE BASIC : les fichiers

COURS DE PROGRAM-MATION APPROFONDIE : les arbres

COURS DE GENIE LOGICIEL : de la théorie à la pratique

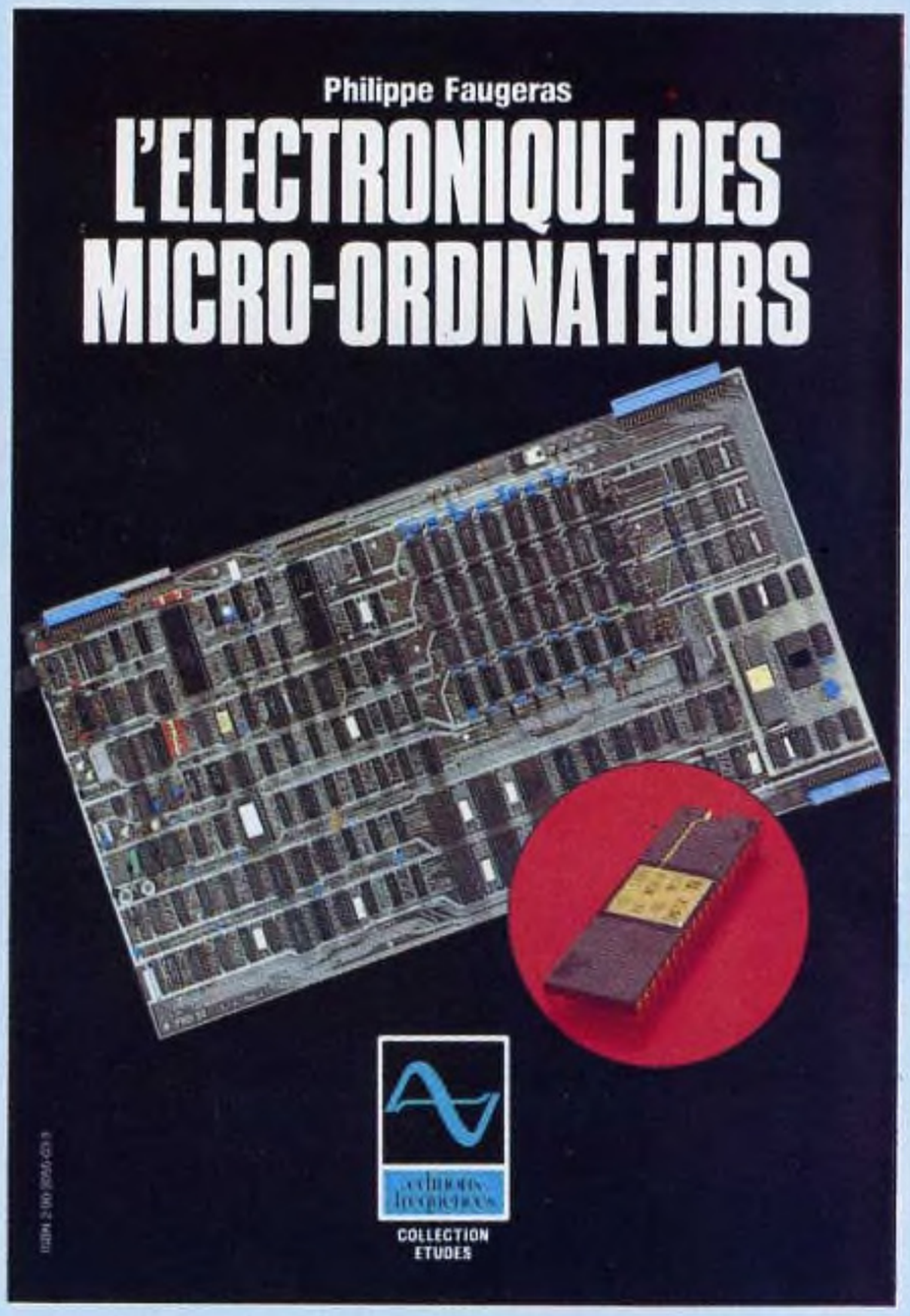
Le MBC 885 de Sanyo

ISSN 0757-6889

M 1988 - N°25 (5) - 18 F MENSUEL DECEMBRE 1985 - 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

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

DECEMBRE 85

COURS N°25 Suite 2e cycle N°5

COURS DE BASIC
Les fichiers
de la page 4 à la page 13
Dominique Chastagnier
Jean-François Coblentz
Patrick Gueneau

COURS DE PROGRAMMATION APPROFONDIE
Les structures de données
(les arbres)
de la page 16 à la page 31

– Présentation de l'arborescence et de la dichotomie ..... p. 17
  • Définitions
  • Les arbres : outils de programmation
  • La dichotomie
– Les arbres binaires simples .... p. 18
  • Elaboration des arbres binaires à tous les niveaux
  • Implantation des arbres binaires en basic
  • Structures parallèles des arbres binaires
– Les arbres binaires équilibrés . . p. 23
  • Description
  • Avantages et manipulation des arbres équilibrés
  • Implantation des procédures complémentaires en basic
– Temps mort ..... p. 27
– Corrections et améliorations à apporter au n° 23 ..... p. 28
– Correction et améliorations à apporter au n° 24 ..... p. 30

Dominique Chastagnier
Jean-François Coblentz
Patrick Gueneau

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

COURS DE GENIE LOGICIEL
De la théorie à la pratique
de la page 35 à la page 49

- les tris ..... p. 35
– Définition 36
  • Exemple 1
  • Exemple 2
– Complexité des algorithmes de tri ..... p. 40
– Tri par extraction
  Tri par ventilation ..... p. 41
– Tri à bulles ..... p. 42
– Tri rapide par arbre binaire 1 ... p. 43
– Tri rapide par arbre binaire 2 ... p. 44
– Tri rapide par arbre binaire 3 ... p. 45
– Tri rapide par arbre binaire ..... p. 46
– Les données logiques et leurs opérateurs ..... p. 47
– Tables de vérité (opérateurs logiques) ..... p. 48
– Comparaison des méthodes de tri ..... p. 49

Charles-Henry Delaleu

NOTRE COUVERTURE : le MBC 885 Sanyo, entièrement compatible PC, clavier Azerty, horloge 8 MHz RAM 256 Ko extensible en option à 640 Ko.

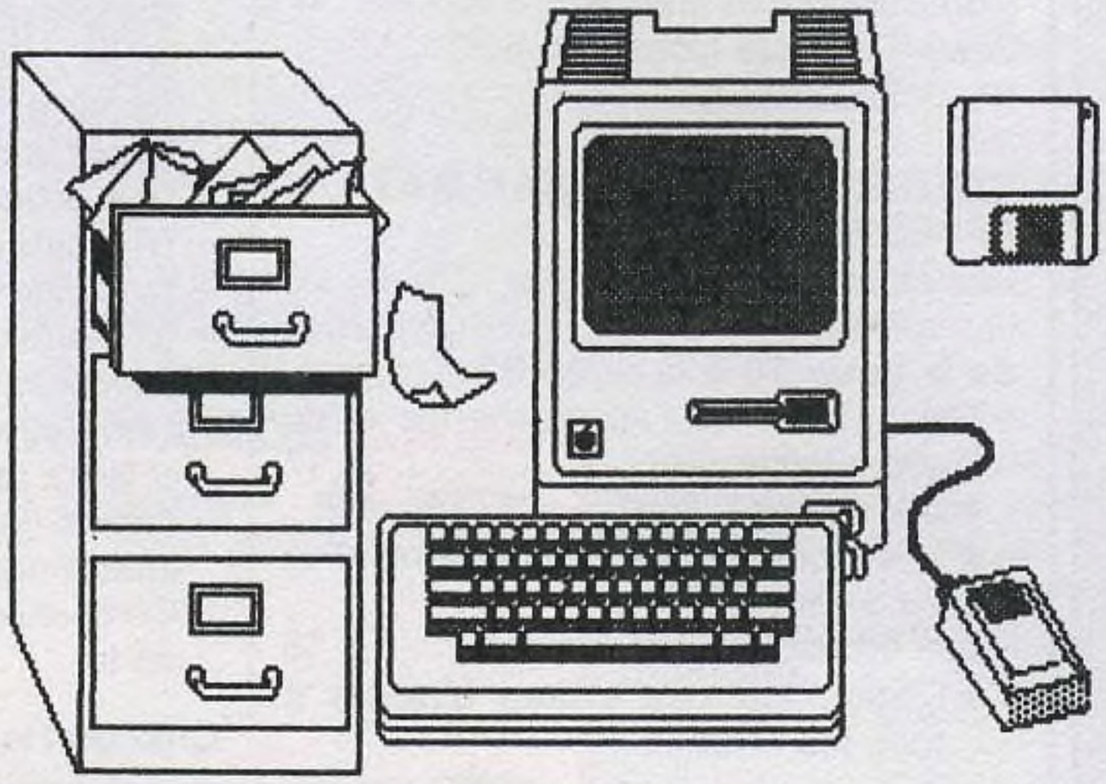
COURS DE BASIC

Dominique Chastagnier
Jean-François Coblentz
Patrick Gueneau

LES FICHIERS

1. Introduction

Les fichiers représentent une part très importante de l'apprentissage de l'informatique. En effet, à ce jour, il sont l'unique méthode de stockage de données, que ce soit pour un petit nombre de données, comme un certain programme de répertoire, ou pour un nombre gigantesque de données, comme pour la Sécurité Sociale, ou tout autre organisme. Pour traiter toute cette information, divers types de programmes sont possibles, mais une seule structure de stockage est utilisée, le fichier. Un fichier peut être manuel ou informatique, mais comme vous pouvez le constater sur notre dessin, mieux vaut demander à un programme de ranger pour vous !!



2. Les définitions de base

2.1. Rafraîchissons-nous la mémoire

De nos jours, la plupart des ordinateurs permettent le stockage de données sur des supports qualifiés de mémoire externe (ou mémoire de masse). Ces supports enregistrent l'information sous forme de fichiers, entités parfaitement déterminées pour le système d'exploitation.
Il est possible d'accéder à un fichier, de le modifier, de lire les données qui se trouvent à l'intérieur, de les sauvegarder à nouveau, de connaître tous les fichiers stockés sur un disque.
Mais cela met en cause deux niveaux de commandes différents. Pour lire le nom de tous les fichiers, y accéder, vous utilisez le système d'exploitation. Pour agir sur le fichier lui-même, vous utilisez les commandes de votre langage préféré, que ce soit BASIC, Pascal ou tout autre.
C'est de la structure des fichiers et des commandes BASIC les concernant que nous allons parler dans cette série de cours.

2.2. La structure d'un fichier

Un fichier est une sorte de longue suite de nombres ou de chaînes de caractères, qui sont appelées données. Chaque donnée est stockée dans une case, structure dont nous allons reparler quand nous détaillerons les types de fichiers. Cette case est repérée par son numéro d'ordre, qui permet de savoir où l'on se trouve dans le fichier. La première case porte donc le numéro 1 et ainsi de suite.

Structure générale d'un fichier

Cela peut vous rappeler la notion de tableau, mais la différence est que vous ne pouvez mélanger nombres et chaînes de caractères dans un tableau, alors qu'ici tout est permis.

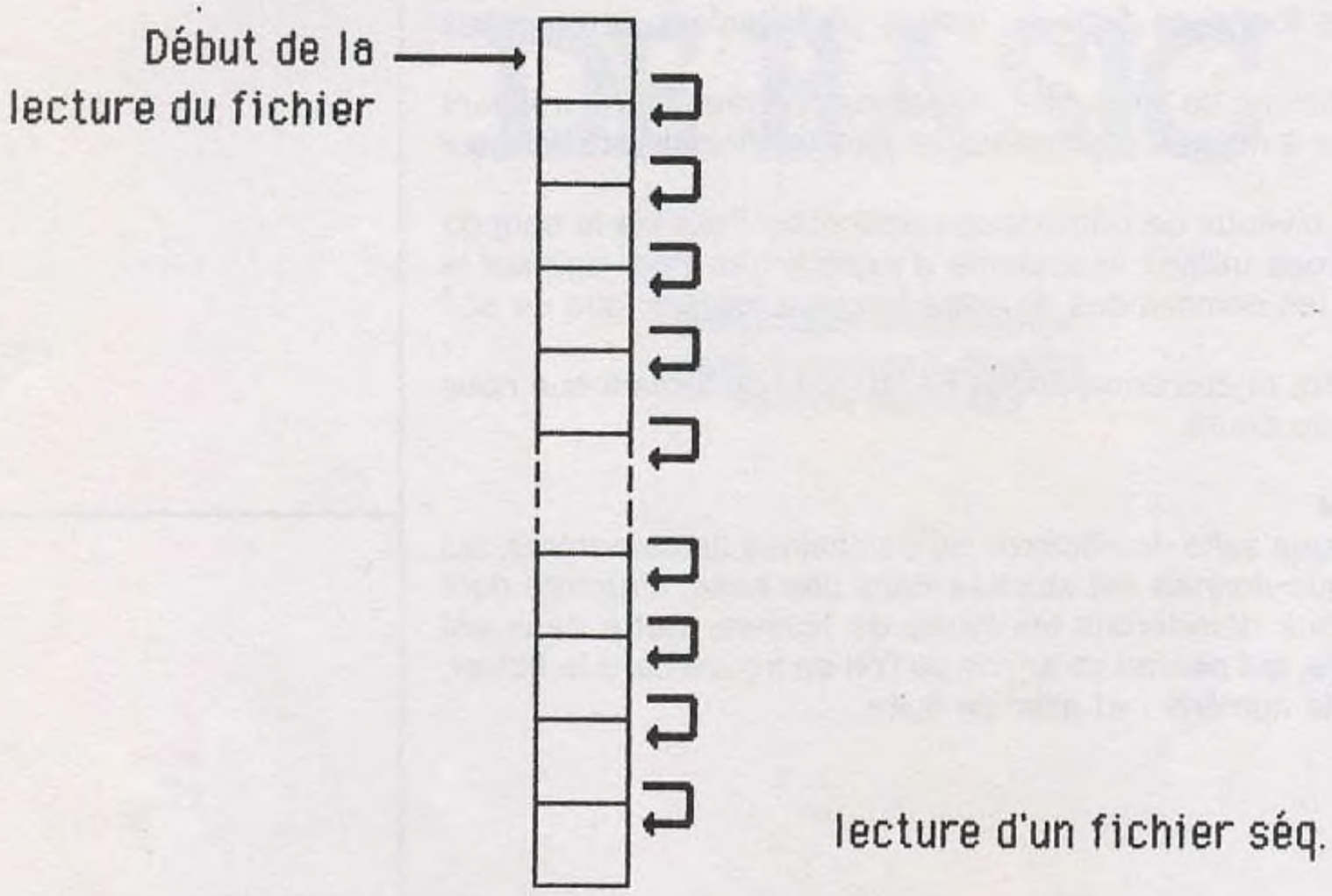
Il existe deux grandes catégories de fichiers :

- Les fichiers séquentiels
- Les fichiers à accès direct

Leur utilisation est fondamentalement différente, comme nous allons le voir immédiatement.

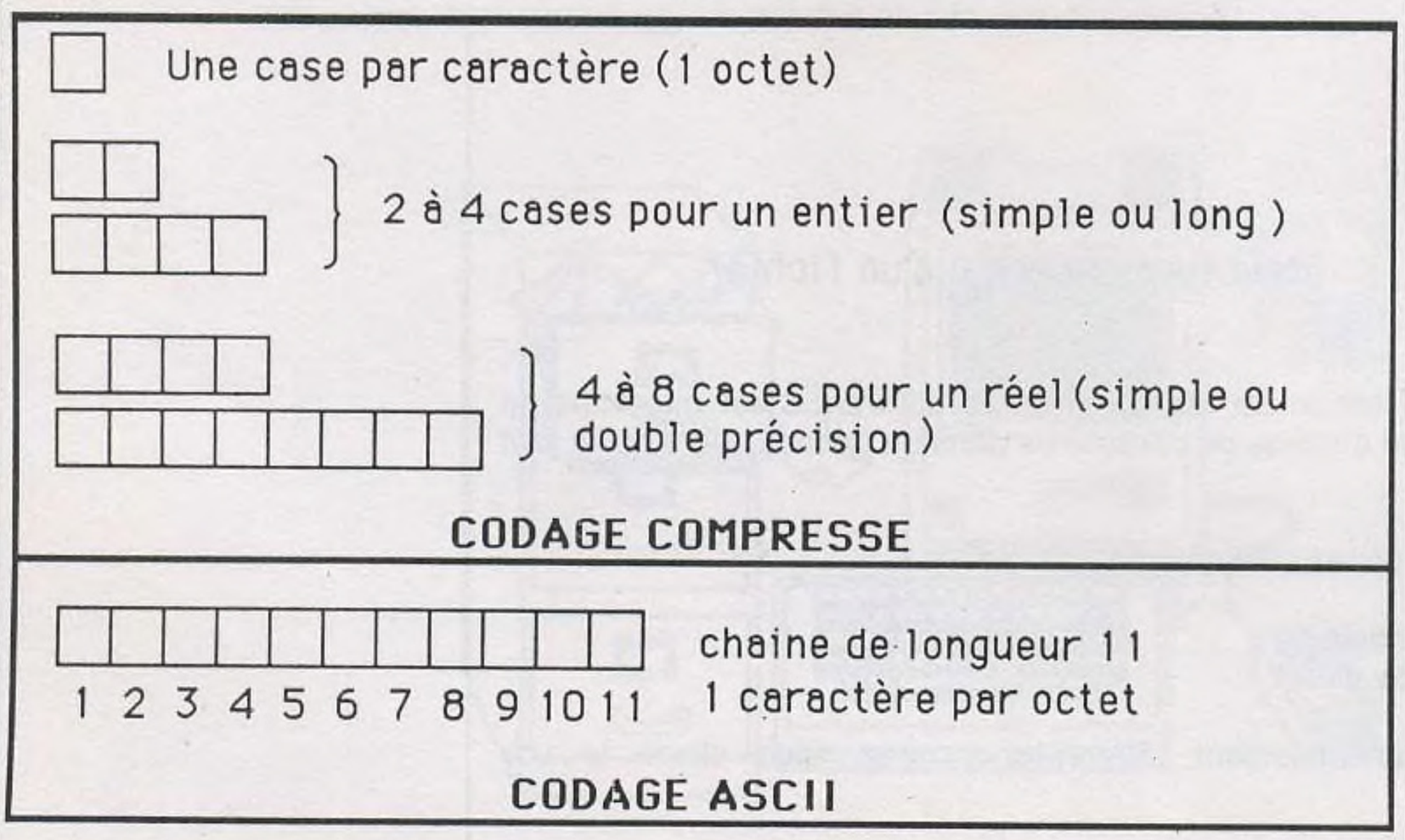
2.3. Les fichiers séquentiels

Ces fichiers ne peuvent être lus que donnée après donnée, en commençant par la première. Ainsi, même si seule la dernière vous intéresse, vous devez lire toutes celles qui se trouvent avant, comme l'indique le schéma.

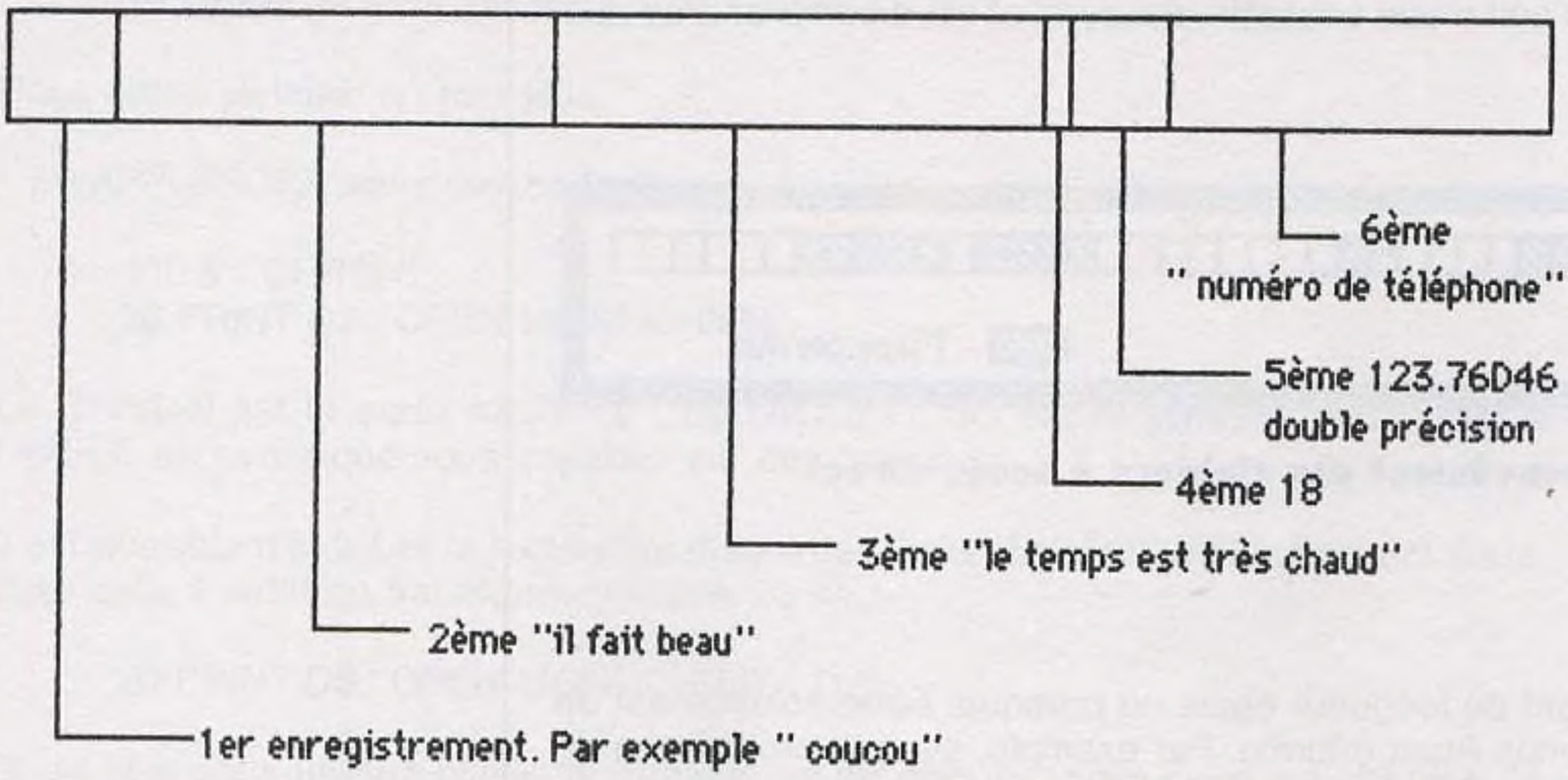


lecture d'un fichier séq.

Ceci peut paraître un handicap énorme, car lire un fichier dans sa totalité demande un temps qui se compte parfois en minutes, mais ces fichiers ont un avantage souvent déterminant. Il est possible de stocker des données de longueur quelconque.
Qu'est-ce que la longueur d'une donnée ? Il s'agit du nombre d'octets nécessaire à son stockage, que ce soit en mémoire vive ou en fichier. Ainsi, par exemple un entier prendra 4 octets, un réel 5, etc. Si vous stockez des chaînes de caractères, la longueur dépend naturellement du nombre de caractères de la chaîne.



Comme vous le voyez, ne pas avoir à se préoccuper de la longueur d'un enregistrement (nom du contenu d'une case) est un souci de moins, et un gain de place, car le programme chargé de gérer la place des données optimise la place de stockage, comme le montre la figure ci-après.



Description d'un fichier séquentiel, avec un exemple

En résumé, dès que vos enregistrements auront une longueur par trop variable, il vous faudra passer par un fichier à accès séquentiel. En fait, nous verrons un peu plus loin que cette règle qui semble stricte peut être tournée, dans certains cas.

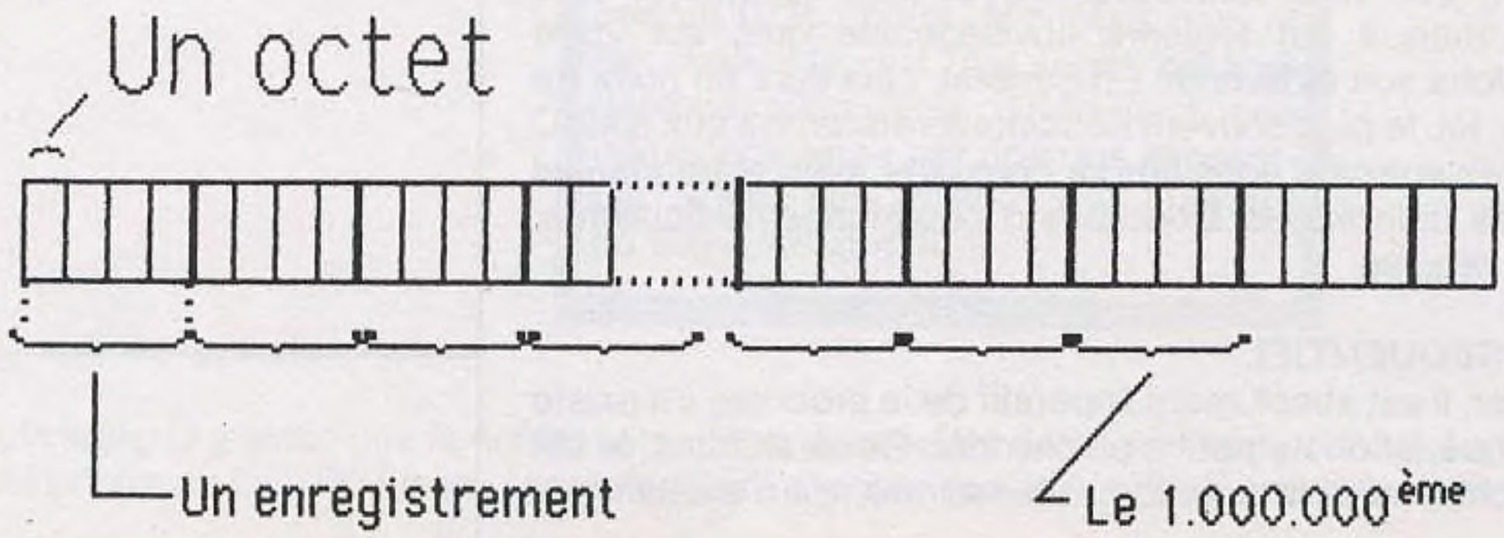
Un fichier à accès séquentiel est donc la tortue dont vous pouvez avoir besoin parfois (il s'avère que ce sont ces fichiers qui sont les plus utilisés en informatique personnelle).

2.4. Les fichiers à accès direct

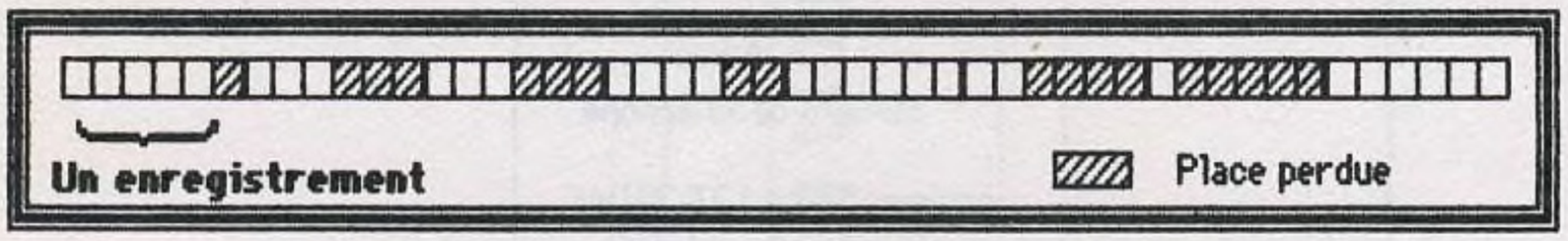
Ces fichiers sont par contre la Ferrari (!!!) des fichiers. En effet, il vous suffit de préciser le numéro d'un enregistrement pour que celui-ci soit lu, et pas un seul autre.

Comment un tel miracle est-il possible ?

Simplement, lorsque vous créez un fichier à accès direct, vous précisez la longueur d'un enregistrement (vous vous souvenez, un enregistrement est une case du fichier), et donc, si vous voulez lire la 1 000 000 donnée, le programme sait qu'il devra passer 999 999 fois la longueur d'un enregistrement. Superbe, non ???



L'inconvénient majeur est naturellement que vos données doivent être de longueur connue, et que la longueur que vous déclarerez sera la plus grande des longueurs de vos données. Ainsi, si certaines données sont plus courtes, beaucoup de place est perdue sur le support magnétique que vous utilisez. Si ce support est un disque, ce n'est pas trop grave, mais s'il s'agit d'une cassette, essayez et vous comprendrez que ceci est un réel problème.



Exemple d'inconvénient des fichiers à accès direct

Par contre, si vos données sont de longueur égale ou presque, cette solution est de loin la meilleure, le gain de temps étant énorme. Par exemple, avec le programme de mot le plus long, si vous décidez de créer une bibliothèque de mots de x lettres, vous pouvez utiliser un fichier à accès direct.

2.5. Conclusion sur les types de fichiers

L'expérience montre que choisir un type de fichiers est toujours un cruel dilemne. Prenons un exemple que vous devez commencer à connaître (!!!). Pour le répertoire, pour choisir un fichier à accès direct qui semble très avantageux en temps d'accès, il vous faudra choisir une longueur pour chaque type de donnée, par exemple 20 caractères pour le nom. Que faire pour les noms de plus de 20 caractères. C'est à vous de choisir si vous coupez le nom, ou si une telle troncature est inacceptable, ce qui est le cas si ce programme vous sert à faire du courrier automatique (du «mailing»). Dans ce cas, si vous voulez un fichier à accès direct, il vous faudra prendre une longueur dont vous serez sûr qu'elle ne sera jamais atteinte. Imaginez la place perdue. Si la longueur moyenne d'un nom est de 8 lettres et que vous en preniez 30, vous perdez 22 caractères par nom, soit 5*22 bytes, plus de 100 par nom. C'est énorme. Les fichiers séquentiels par contre seront beaucoup plus lents.

3. Les commandes

3.1. Un avis très important

Il est nécessaire de savoir que les opérations sur les fichiers sont parmi les moins standardisées. En conséquence, plus que jamais, il est important de connaître les commandes propres au système que vous possédez. Ce qui suit couvrira le plus largement possible les BASIC, mais il est toujours envisageable que, sur votre ordinateur, une partie des opérations soit différente. En général, cela sera un point de détail, mais qui fera tout «planter». Ici, le plus souvent nous nous réfèrerons aux BASIC Microsoft et APPLE SOFT. Pour les autres, il vous faudra comparer avec votre manuel et vous verrez très vite les petites différences. Donc pas d'inquiétudes particulières, mais une vigilance maximale est requise.

3.2. Créer et ouvrir un fichier SEQUENTIEL

Avant toute opération sur un fichier, il est absolument impératif de le créer ou, s'il existe déjà, de l'ouvrir. Cela semble logique, et on ne peut s'en plaindre. Généralement, le fait de demander l'ouverture d'un fichier rend la création automatique, s'il n'existait pas

auparavant. Par contre, sur certains systèmes (Microsoft), le fait de créer un fichier peut effacer un fichier de même nom qui existait déjà. Pour créer un fichier séquentiel, il convient donc d'écrire une séquence d'instructions du type :

```
10 OPEN MONFICHIER.
```

Nous allons détailler un peu plus.

– En APPLESOFT, vous devrez faire :

```
10D$ = CHR$(4)
20 PRINT D$;"OPEN MONFICHIER"
```

Le CHR$(4) est le code ASCII de CONTROLE-D, qui est le symbole permettant à l'APPLE de savoir que vous travaillez sur des fichiers.

Il est possible d'indiquer le lecteur de disquettes, pour les chanceux qui en ont deux. Pour cela, il suffit de transformer la ligne 20 en :

```
20 PRINT D$;"OPEN MONFICHIER , D2".
```

Si de plus vous utilisez plusieurs contrôleurs de disques (pour ceux qui ont plus de deux lecteurs), il faut taper :

```
20 PRINT D$;"OPEN MONFICHIER, Sn, Dm"
```

où n est le slot du contrôleur et m le numéro du lecteur sur ce contrôleur. Une fois toutes ces informations en mémoire, il n'est plus nécessaire de les répéter pour les opérations suivantes. Par contre, pour utiliser un fichier autre sur le lecteur habituel, il vous faudra repréciser le ou les deux paramètre(s).
En Applesoft, vous l'avez donc remarqué, la création est automatique. Par contre, la destruction d'un fichier déjà existant n'est pas automatique. Pour cela, il faut l'ouvrir, puis lui appliquer l'opération DELETE, de la façon suivante :

```
20 PRINT D$;"DELETE MONFICHIER"
```

– En Microsoft, les instructions sont :

```
10 OPEN "O" , N , "MONFICHIER"
```

Attention, ceci est un O, la lettre et non pas un 0, le chiffre. N'oubliez pas, et cet exemple en est une brillante démonstration, que différencier ces deux caractères est presque impossible. Soyez plus malin que le créateur du Basic Microsoft !!!

Ici, la lettre O signifie que le fichier sera utilisé pour écrire dessus et non pour lire. Le O est l'initiale de OUTPUT (vers l'extérieur), qui exprime que le flot de données sortira de

l'unité centrale, pour aller vers un périphérique. Dans ce cas, si le fichier MONFICHIER existait déjà, il est préalablement détruit.
En remplaçant le O par un I, vous indiquez que le fichier servira en lecture de données, donc dans le sens d'une entrée des données (INPUT en anglais).
Le paramètre N est le numéro du fichier. Ce numéro est arbitraire, et sert dans la suite du programme à se référer au fichier plus succinctement que si vous deviez retaper tout le nom. Ceci a été conçu pour permettre l'utilisation de plusieurs fichiers simultanément. C'est très commode.

3.3. Fermeture d'un fichier séquentiel

La fermeture d'un fichier est aussi importante que son ouverture et aussi simple à réaliser. En effet, oublier de fermer un fichier diminue les possibilités du programme qui est en cours d'exécution, car le nombre de fichiers simultanément ouvert est limité par un nombre qui est contenu dans la variable MAXFILES. Si vous le désirez, il est possible de modifier ce paramètre. Négliger la fermeture de fichiers peut donc vous amener à dépasser cette valeur, et le programme plante. En général, les données ne sont pas perdues. Comment dépasser la valeur des MAXFILES ? C'est assez facile. Il suffit de demander l'exécution d'une boucle qui ouvre un fichier et d'oublier de mettre la commande de fermeture dans la même boucle.
Voici comment fermer un fichier :

```
100 CLOSE MONFICHIER
```

Dans le détail, nous aurons les instructions suivantes selon les systèmes :

– En Applesoft :

```
100 PRINT D$ ; "CLOSE MONFICHIER"
```

Eventuellement, cette série d'instructions sera suivie de précisions concernant le lecteur et le contrôleur utilisés. Voir dans le paragraphe précédent pour des détails.

– En Microsoft :

```
100 CLOSE N
```

où, comme précédemment, N désigne le numéro du fichier à fermer.

Vous pouvez alors avoir :

```
100 CLOSE N1, N2, N3...
```

Pour fermer plusieurs fichiers ou encore plus fort vous ferez :

```
100 CLOSE
```

pour les fermer tous en même temps.

3.4. L'écriture de données dans un fichier

Après avoir créé un fichier, vous pouvez écrire à l'intérieur tout ce qui vous passe par la tête. Pour cela, il suffit d'utiliser une commande d'écriture de type PRINT, que vous connaissez. Il est parfois nécessaire de prévenir l'ordinateur que cette écriture se fait dans un fichier et non sur l'écran ou sur l'imprimante. Dans ce but, les ordinateurs utilisent des messages particuliers, qui hélas varient largement suivant les systèmes. Voici les commandes pour nos deux machines de référence.

– En Applesoft, vous aurez :

ouverture du fichier :	10 D$ = CHR$(4)
	20 PRINT D$;"OPEN MONFICHIER"
écriture dans le fichier :	30 PRINT A,A%,A(5), "HELLO", B$
fermeture du fichier :	40 PRINT D$;"CLOSE MONFICHIER"

Comme vous venez de le voir, sur l'Apple II il n'est pas requis de messages particuliers pour l'écriture sur un fichier. En conséquence, l'écriture de messages à l'écran est impossible durant la même période, puisque tout message de ce type ira vers le fichier. Cette limitation est parfois fort gênante.

– En Microsoft :

ouverture du fichier :	10 OPEN "O", 1, "MONFICHIER"
écriture dans le fichier :	20 PRINT *1,A,A%,A(5), "HELLO", B$
fermeture du fichier :	30 CLOSE

Cela semble beaucoup plus simple, mais il y a un détail qui complique tout. En effet, si nous lisions ces données, telles qu'elles ont été écrites, vous auriez «HELLO» et B$ concaténées. Il faut en effet séparer les chaînes entre elles, par une virgule (,) comme suit :

```
20 PRINT *1,A,A%,A(5), "HELLO";",";,B$
```

Avez-vous noté la différence de séparateurs ?
Il y a des virgules entre les variables et des points virgules entre les variables alphanumériques et la virgule ajoutée.

3.5. La lecture de données

Note importante : Il y a au moins une chose évidente à retenir, c'est que pour lire des données dans un fichier, il faut savoir ce que l'on va lire. Ainsi, si vous lisez des entiers, il faut les stocker dans des variables ou des tableaux d'entiers, et non dans des variables ou des tableaux de chaînes de caractères. Donc, comme nous le disait notre prof de physique, ne rien faire sans connaître le résultat (Merci Mr Boussié. Fin du message personnel) à l'avance. Ici cela signifie connaître exactement ce que l'on doit lire.

Pour lire des données, il est nécessaire d'ouvrir un fichier... en lecture (peut-être l'aviez-vous deviné !!!), puis d'utiliser la commande INPUT pour lire des données. Donc la commande INPUT joue le même rôle que pour la lecture de données au clavier, mais avec la nuance que maintenant nous allons tout prendre dans un fichier, sans arrêter le programme.

– En Applesoft, le programme pour le fichier MONFICHIER écrit précédemment est :

ouverture du fichier :	10 D$ = CHR$(4)
	20 PRINT D$;"OPEN MONFICHIER"
lecture dans le fichier :	30 INPUT D,D%,C,K$,Z$
fermeture du fichier :	40 PRINT D$"CLOSE MONFICHIER"

La même remarque doit être faite que pour l'écriture. Lorsque vous êtes en lecture de fichier, toute commande INPUT correspond à une lecture DANS LE FICHIER. Donc, un INPUT pour lire au clavier est impropre. Pensez-y : lorsque vous voulez un contrôle au clavier lorsqu'un fichier est ouvert, il vous faut le fermer.

– En Microsoft :

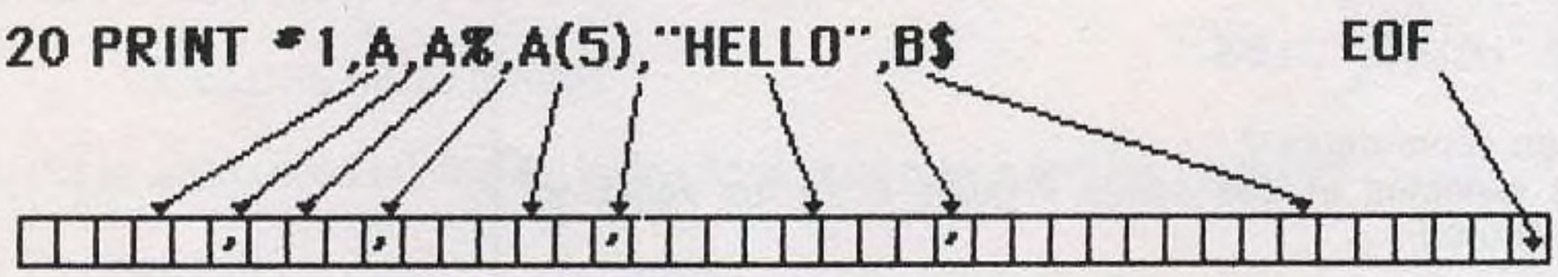
ouverture du fichier :	10 OPEN "O", 1, "MONFICHIER"
lecture dans le fichier :	20 INPUT *1,D,D%,C,K$,Z$
fermeture du fichier :	30 CLOSE

Une remarque assez évidente : les variables stockées dans un fichier ne sont pas obligatoirement lues dans le même nom de variable. Ainsi, dans les exemples, la variable A est lue par la suite sous le nom D. Ceci n'a aucune importance, seule la valeur effective de ces variables est importante.

3.6. Comment fonctionnent l'écriture et la lecture ?

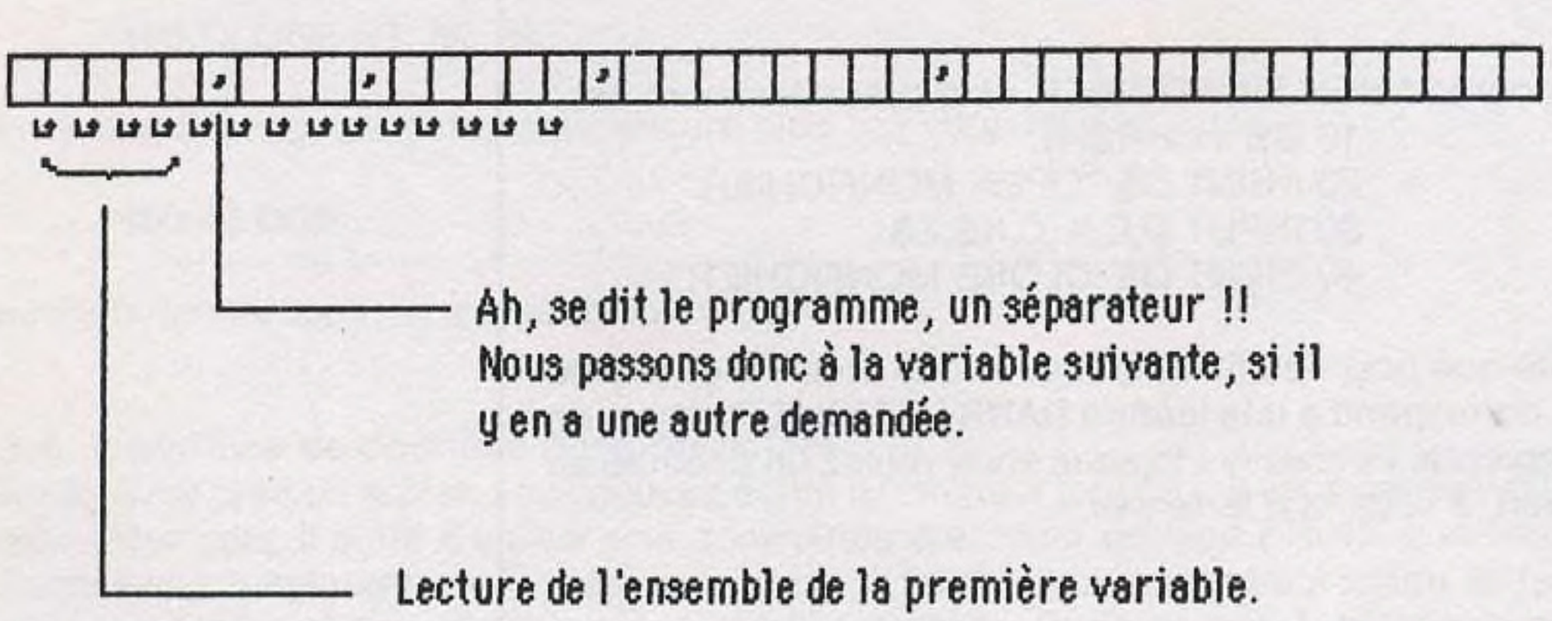
Le fichier est parcouru linéairement et les séparateurs de «cases» sont des retours chariots, des changements de lignes ou des virgules. Regardons notre exemple. Entre le A et le A%, nous avons une virgule. Après le B$, nous avons un retour chariot, même s'il ne vous est pas visible explicitement. Puis, à la lecture, ces mêmes séparateurs permettent au système de passer à la variable suivante, ou d'arrêter de lire le fichier pour une certaine variable.

Voici donc une représentation symbolique de la manière dont un fichier est écrit, variable par variable, y compris les séparateurs. Le symbole EOF est décrit immédiatement après.



Ecriture dans un fichier

De même, la lecture d'un fichier séquentiel et on peut voir une illustration de la méthode tortue que nous avions évoquée au début du cours.



Lecture d'un fichier

3.7. Le symbole EOF de fin de fichier

Le symbole EOF est celui qui est placé par le système à la fin de tout fichier lors de l'écriture. Ce symbole est un opérateur logique comme NOT, < > ,... Il est très commode de s'en servir sur certains systèmes, car ainsi vous ne risquez pas de vouloir lire plus de données qu'il n'y en a de stockées. Il suffit de mettre la lecture dans une boucle de type :

– En Microsoft :

ouverture du fichier : 10 OPEN"O", 1, "MONFICHIER"
lecture dans le fichier : 20 IF NOT EOF(1) THEN I=I+1:INPUT # 1,A(I): GOTO 20
fermeture du fichier : 30 CLOSE

Ou encore plus joli :

– En Microsoft :

```
ouverture du fichier :      10 OPEN "O", 1, "MONFICHIER"
lecture dans le fichier :   20 WHILE NOT EOF(1)
                            30 I=I+1 : INPUT # 1,A(I)
                            40 WEND
fermeture du fichier :      50 CLOSE
```

Ceci évite bien des plantages, les possesseurs d'Apple pourront en témoigner, qui disposent de commandes bien pauvres dans ce domaine.

4. Conclusion

Vous connaissez donc maintenant la plupart des détails sur la structure des fichiers, et des commandes d'utilisations des fichiers séquentiels. Le mois prochain, nous parlerons des fichiers à accès direct, et proposerons des exemples. Ceci sera en particulier la fin d'un certain programme de répertoire téléphonique.

BIBLIOTHEQUE TECHNIQUE

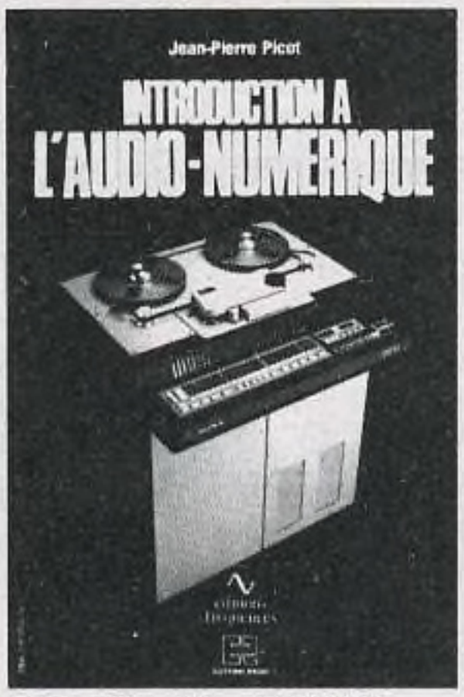
Collection études (format 165 × 240)

E 15. 184 p. Prix : **140 F TTC**

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... à dévorer.

E 01. 320 p. Prix : **165 F TTC**

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 !

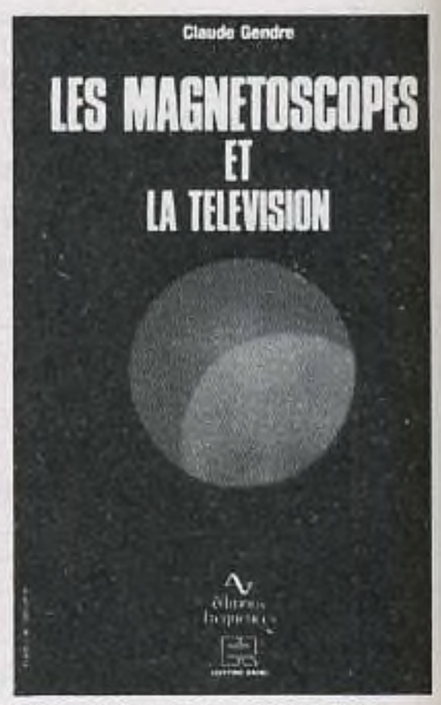


E 05. 160 p. Prix : **155 F TTC**

C'est le premier ouvrage paru en langue française traitant de l'audionumérique ; écrit par un professionnel, avec rigueur, simplicité, il explique brillamment les bases de cette technique : quantification, conversion, formats, codes d'erreurs.

E 04. 240 p. Prix : **154 F TTC**

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 03. 256 p. Prix : **145 F TTC**

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.

A paraître

Faisant suite à la parution de «L'électronique des micro-ordinateurs», cet ouvrage s'adresse aux électroniciens qui désirent s'initier aux montages périphériques des micro-ordinateurs, interfaces en particulier, qui permettent la communication avec le monde extérieur.

E 06. 128 p. Prix : **150 F TTC**

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 : mémoires, clavier, écran, interfaces de toutes sortes.

E 02. 160 p. Prix : **92 F TTC**

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 13. 256 p. Prix : **165 F TTC**

Une sélection des meilleurs articles de la célèbre revue «l'Audiophile» choisis parmi les plus significatifs des quinze premiers numéros, introuvables aujourd'hui. Le tome 1 traite de l'électronique audio, à tubes et à transistors.

E 12. 256 p. Prix : **155 F TTC**

Dans un esprit identique, le tome 2 traite du domaine passionnant que constituent les transducteurs en audio : on y aborde la modélisation théorique des enceintes, la conception géométrique des tables de lecture, le réglage des cellules et des bras.

Collection loisirs (format 135 × 210)

L 07. 160 p. Prix : **68 F TTC**

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 10. 200 p. Prix : **130 F TTC**

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 09. 72 p. Prix : **65 F TTC**

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 11. 160 p. Prix : **85 F TTC**

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 14. 128 p. Prix : **95 F TTC**

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 20. 208 p. Prix : **130 F TTC**

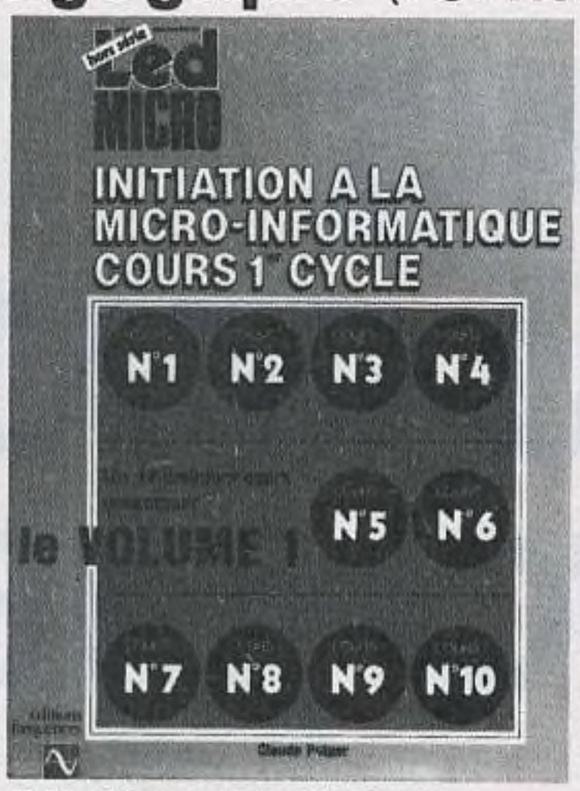
Accessible à tous, «Week-end photo» permet de découvrir de façon simple les différents aspects de la photographie actuelle. Vous y trouverez les bases indispensables pour vous perfectionner, un guide de choix des appareils 24×36 et des illustrations abondamment commentées.

DES EDITIONS FREQUENCES

Collection pédagogique (format 210×270)

P 08. 96 pages. Prix : **115 F TTC**

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 de robots et les fans de cybernétique, enfin réunis !

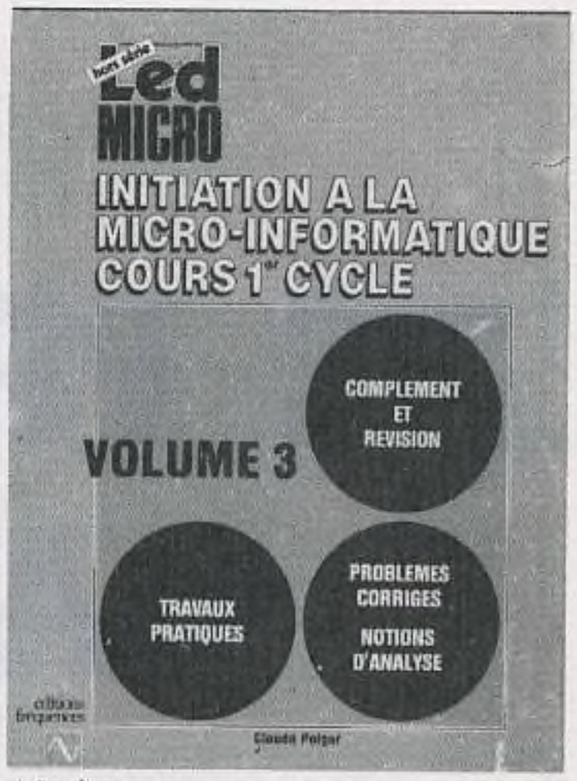


P 16. 272 pages. Prix : **130 F TTC**

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. En voici le premier tome.

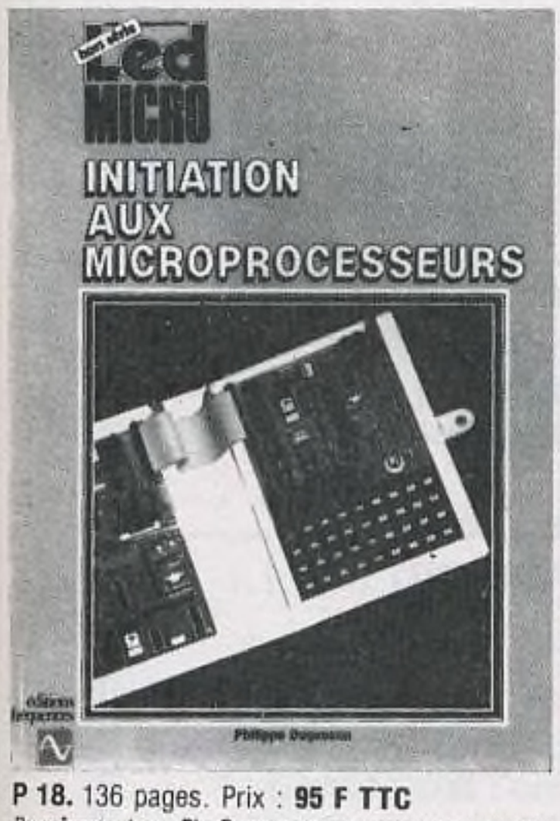
P 17. 208 pages. Prix : **130 F TTC**

Le tome 2 est la suite du tome 1 : l'esprit puissamment didactique de l'auteur s'y retrouve, le contenu du livre permettra d'acquérir un niveau suffisant pour exercer l'analyse, la programmation, la gestion, l'automatisme, la simulation et d'autres choses encore !



A Paraître

Le troisième volume du cours de Programmation, dû à Cl. Polgar, pédagogue apprécié de tous. Il continue dans la lignée d'un réel souci didactique, de haut niveau, maintenant, mais en conservant l'aspect progressif qui fit son succès initial.

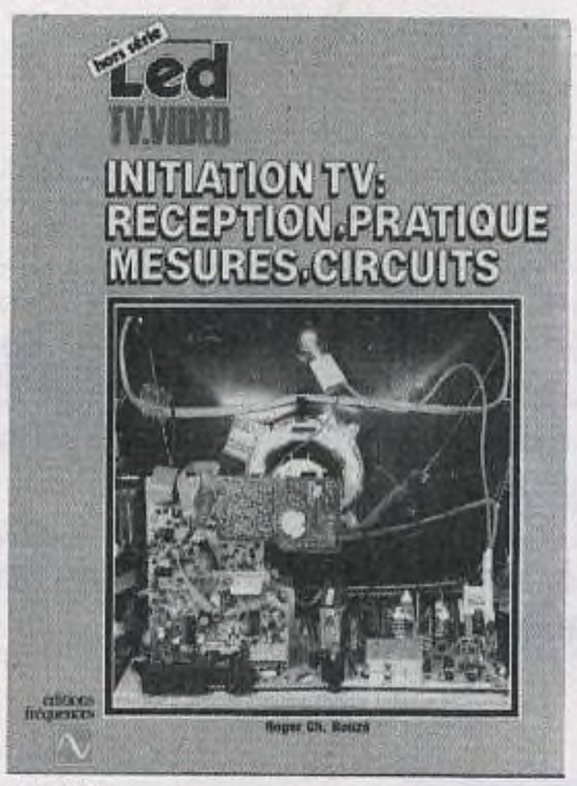


P 18. 136 pages. Prix : **95 F TTC**

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.

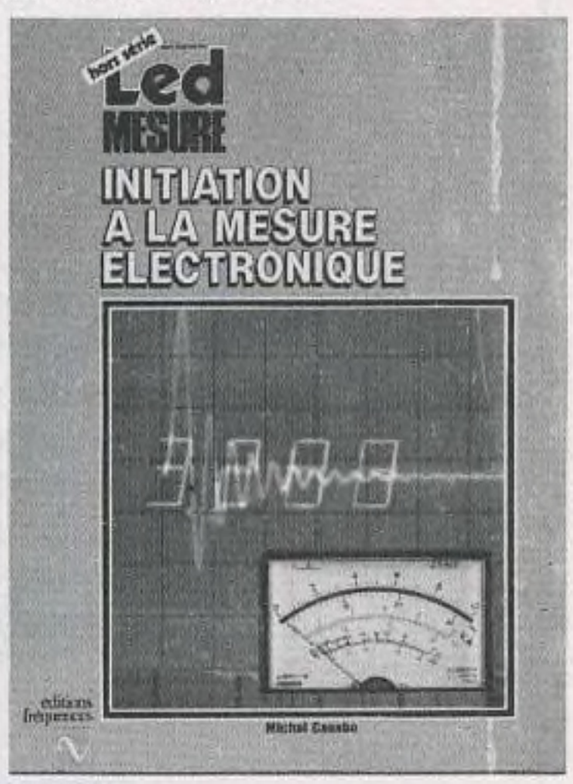
P 19. 104 pages. Prix : **95 F TTC**

Ce cours d'Initiation à l'Electronique Digitale 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.



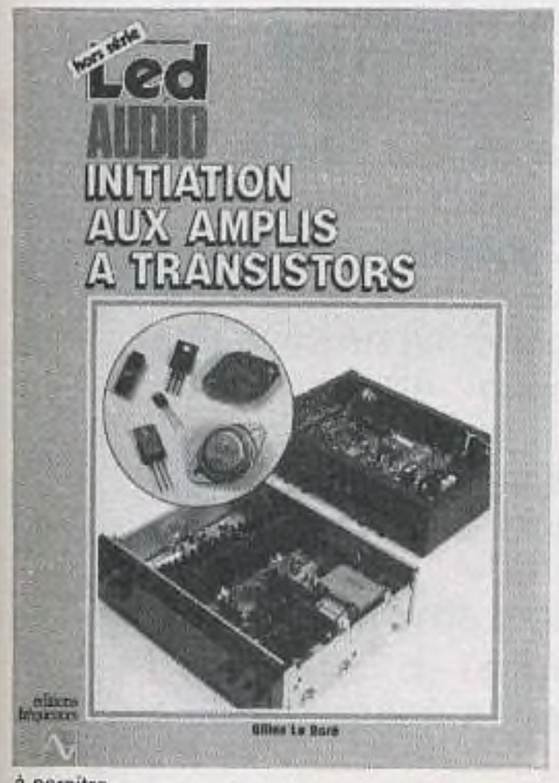
à paraître

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.



à paraître

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.



à paraître

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...

à paraître

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.

En vente chez votre libraire ou aux Editions Fréquences

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

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

E 01 ☐	**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** ☐

Frais de port : + 10 F par livre commandé,
soit la somme totale ci-jointe, de Frs par

CCP ☐ Chèque bancaire ☐ Mandat-lettre ☐

Nom ..... Prénom .....

Adresse .....

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

COURS DE PROGRAMMATION APPROFONDIE

Dominique Chastagnier
Jean-François Coblentz
Patrick Gueneau

Lorsque nous commençâmes les structures de données, nous comptions vous les présenter en deux mois ; mais sur ces arbres aucun d'entre nous ne voulait passer sous silence sa façon de les employer. En conséquence, ce numéro ne présentera que les arbres, la suite (et la fin) des structures de données s'achevant plus tard.
En passant des journées dans les arbres, comme Marguerite Duras, vous accéderez à un des outils les plus usités en informatique mais aussi en mathématiques. Bonne lecture.

COURS N° 5

Les structures de données - Deuxième partie - Les arbres

PLAN DU COURS

1. Présentation de l'arborescence et de la dichotomie
   1.1. Définitions
   1.2. Les arbres : outils de programmation
   1.3. La dichotomie

2. Les arbres binaires simples
   2.1. Elaboration des arbres binaires à tous les niveaux
   2.2. Implantation des arbres binaires en BASIC
   2.3. Structures parallèles des arbres binaires

3. Les arbres binaires équilibrés
   3.1. Description
   3.2. Avantages et manipulation des arbres équilibrés
   3.3. Implantation des procédures complémentaires en BASIC

4. Temps mort

1. PRESENTATION DE L'ARBORESCENCE ET DE LA DICHOTOMIE

1.1. Définitions

La notion d'«arbre» ne se cantonne nullement au seul domaine des structures de données. C'est une idée qui vient à l'esprit du programmeur dès lors qu'on doit envisager une succession de situations dont certaines sont dépendantes des précédentes ; dans l'absolu, c'est un moyen d'envisager tous les cas possibles. Par exemple, pour le loto sportif, on peut schématiser ainsi toutes les combinaisons possibles :

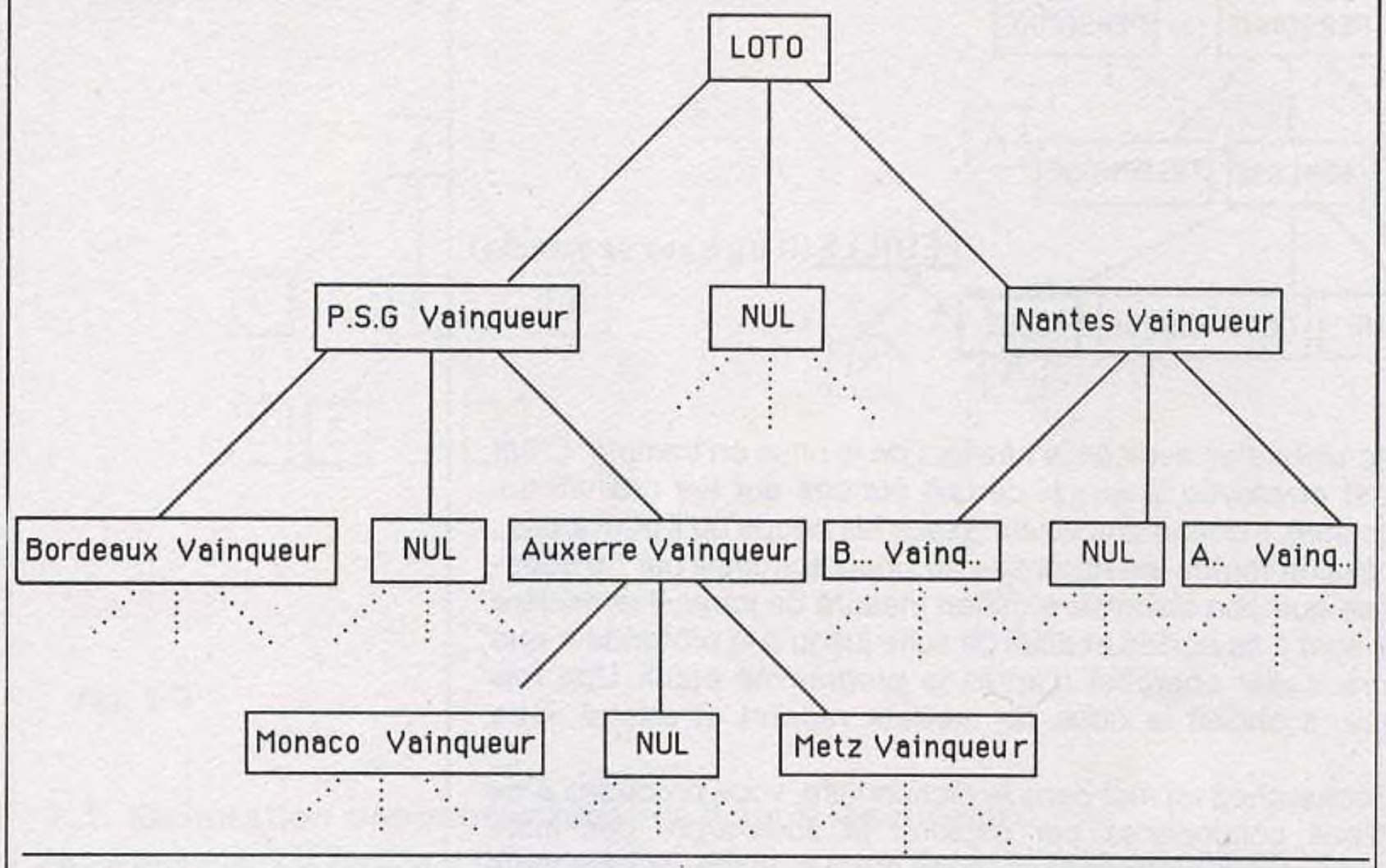


Fig. 1.1.1

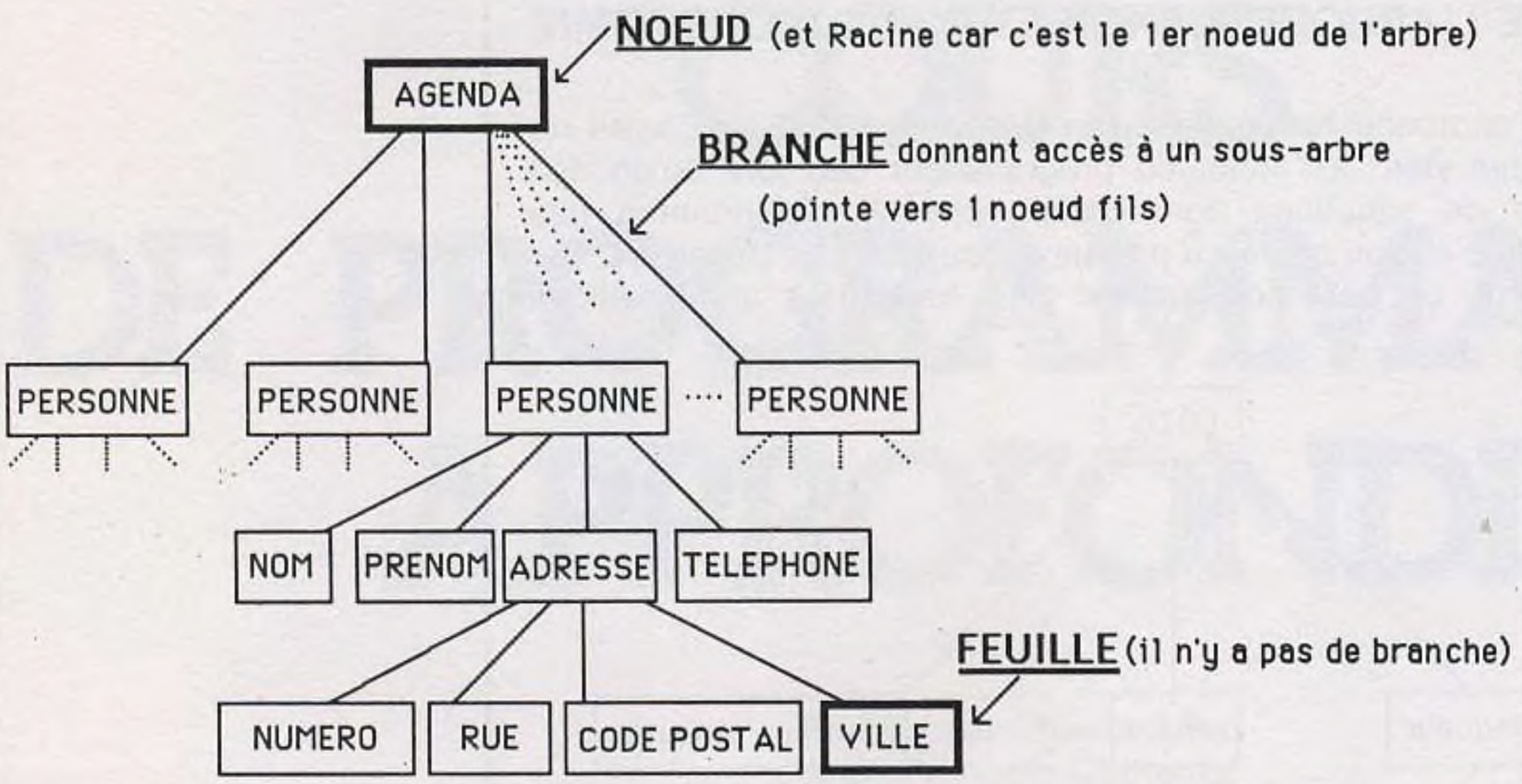
Nous constatons que pour utiliser cette structure, il convient de définir tous les concepts que nous serons à même de devoir manipuler :

- L'**arbre** est l'ensemble de la structure et a pour **racine** l'élément placé le plus haut dans l'édifice.
- A chaque extrémité des **branches** partant de la racine se trouve un **sous-arbre**.
- Chaque sous-arbre a lui-même sa racine qu'on appelle **nœud**. Les informations qu'on souhaite disposer aux nœuds et branches se nomment **étiquettes**.
- Si la racine du sous-arbre n'a pas de branches, on la nomme **feuille**. Si le nœud n'est pas une feuille, il prend le qualificatif d'**interne**.
- Enfin, tout sous-arbre est le **fils** de l'arbre auquel il est attaché. Naturellement, tous les fils d'un même **père** sont **frères**.

A part ces définitions copiées sur les modèles quotidiens, vous noterez quelques détails pratiques. Nos arbres sont antipodistes et poussent la tête en bas, chose déjà employée en généalogie. Tout parcours partant de la racine vers une feuille constitue une «descente» dans l'arbre et symbolise une des hypothèses suggérées par l'arborescence (terme désignant l'usage de l'arbre pour simuler tous les cas de figure). Quant au fait de procéder à toutes les descentes possibles pour envisager toutes les éventualités, nous employons alors le terme d'exploration de l'arbre.

1.2. Les arbres : outils de programmation

L'exploration de l'arbre nous donne par exemple toutes les solutions possibles à un problème donné, et une fois celles-ci fournies, nous pouvons choisir nous-même celle



que nous souhaitons, sans crainte d'en avoir omis une lors de la prise en compte. C'est ce type de méthode qui est employée avec un certain succès sur les ordinateurs d'échecs. A une situation donnée, il donne une valeur à tous les coups qu'il peut jouer ; dans chaque sous-arbre il évalue toutes les réponses au coup accompli par ce sous-arbre puis, suivant la réponse que son advervaire est en mesure de jouer, il considère les nouvelles occasions qui sont à sa portée et ainsi de suite jusqu'à la profondeur que le calculateur est en mesure d'aller chercher d'après le programme établi. Une fois toute l'exploration effectuée, il choisit le coup de meilleur rapport et attend votre réponse.
Vous-même, lorsque vous recherchez un mot dans le dictionnaire, vous procédez à de la recherche par arbre. Vous commencez par explorer le sous-arbre des mots commençant par un I puis, dans ce sous-arbre, la branche N vous donne tous les mots de début IN puis le F, le O, le R et quand vous arrivez au sous-arbre de début INFORMATIQ, il ne reste plus qu'une feuille : le mot INFORMATIQUE.

1.3. La dichotomie

L'exemple précédent ne manque pas de vous montrer la difficulté qu'on rencontrerait si on souhaitait schématiser à l'aide d'un arbre le dictionnaire de la langue française et même, plus modestement, les simples seize matches du loto sportif. Aussi, dans les algorithmes généralement développés, on se restreint à deux cas possibles à partir de chaque nœud ; ainsi à chaque nouveau choix, on élimine la moitié des éventualités. En d'autres termes, on coupe le problème en deux : cela se nomme «dichotomie». Quel est l'intérêt ? Pour rechercher à quel endroit est placé l'élément dont on connaît l'adresse : soit on compare son adresse avec celle qu'on pointe, en augmentant jusqu'à la bonne et cela peut faire beaucoup de tests ; soit on compare directement avec celle se trouvant au milieu. Suivant le résultat, on prend la moitié de l'ensemble dans lequel se trouve notre adresse et dans cette moitié, on compare de nouveau avec l'élément du milieu, et la moitié de la moitié, cela fait déjà beaucoup moins de monde.

2. LES ARBRES BINAIRES SIMPLES

L'adjectif binaire ajouté à côté du terme d'arbre restreint les capacités de la structure en obligeant les nœuds à posséder au maximum deux sous-arbres, l'un portant le nom de sous-arbre gauche et l'autre de sous-arbre droit. Ce qui ne semble, au premier

abord, qu'une simple particularité, se révèle être de manipulation bien plus facile que les arbres généralisés. En effet, dans les arbres «vulgaires», vous ne savez nullement a priori quel sera le nombre maximal de ses fils et par conséquent, il vous est nécessaire de réserver suffisamment de mémoire pour ne pas risquer la surcharge. Au contraire, sur un arbre binaire, il n'est besoin que de deux places mémoire quoi qu'il arrive. Il existe d'ailleurs des méthodes permettant de transformer des arbres quelconques en arbres binaires. Pour cela, il suffit de mettre le premier des fils comme racine du sous-arbre gauche et chacun des cadets comme racine du sous-arbre droit de son frère précédent.

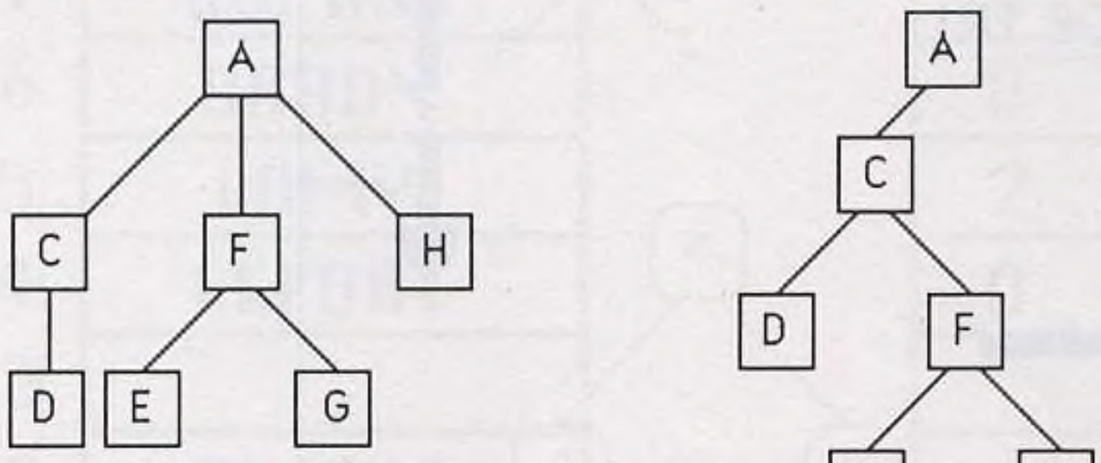


Fig. 2.0

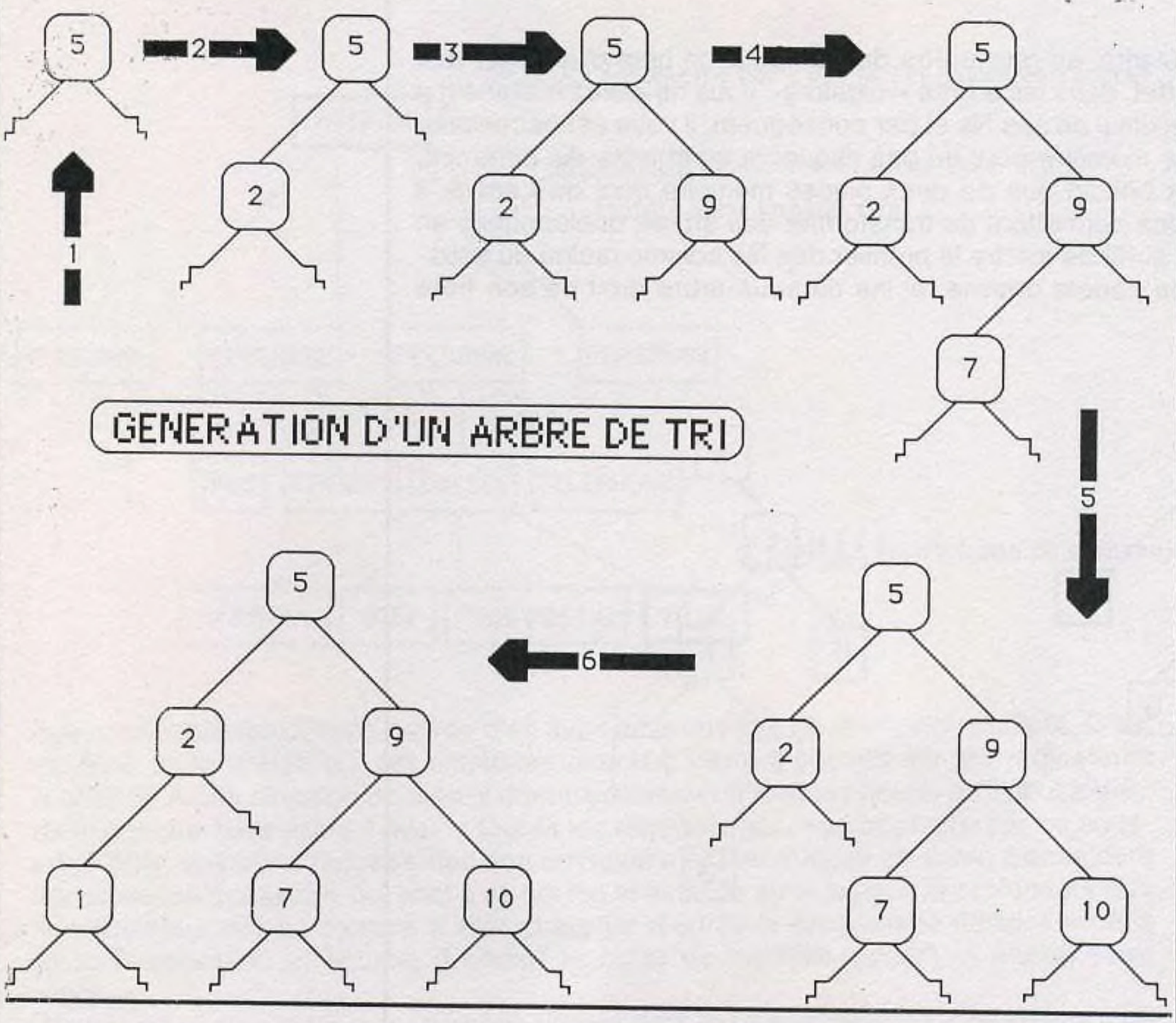
2.1. Elaboration des arbres binaires à tous les niveaux

Nous reprenons la méthode de conception mise au point dans le cours précédent. Le problème que nous aurons avec cette structure proviendra essentiellement du fait qu'elle est adaptée aux langages évolués et se programme aisément dans ces cas-là.

2.1.1. La spécification fonctionnelle reprend les opérations habituelles de gestion de structure plus quelques autres que vous ne manquerez pas d'admettre comme indispensables.

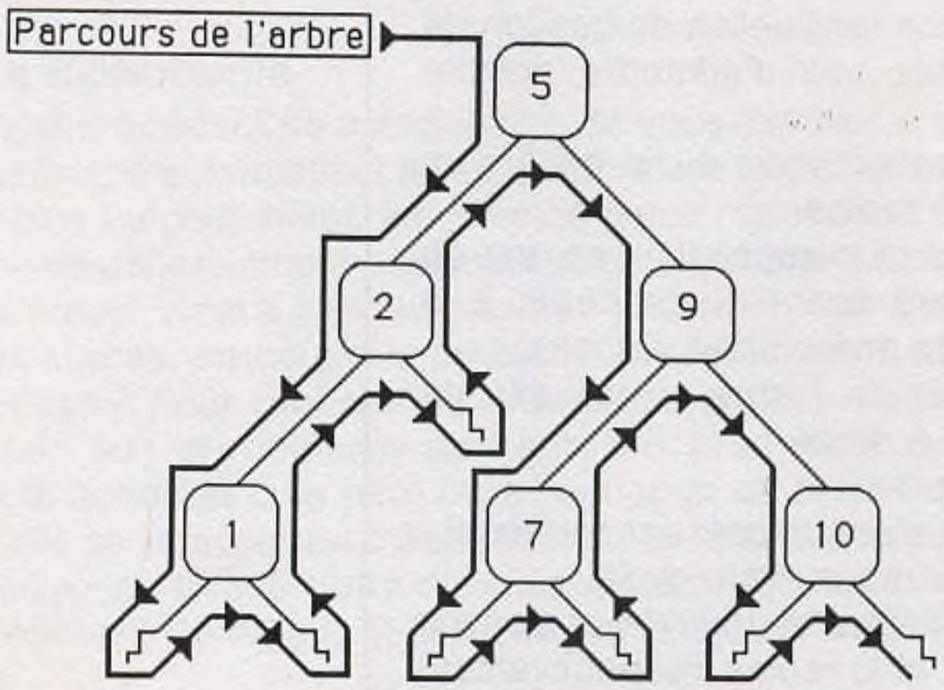
- *En création :* La création de l'arbre lui-même, bien sûr mais aussi celle d'un nouvel arbre à partir de deux sous-arbres et d'une racine.
- *En modification :* L'insertion et la modification sont plus sophistiquées que de coutume car il est ici nécessaire d'explorer l'arbre avant de procéder à la modification souhaitée. Pour cela, on a recours aux procédures d'accès.
- *En accès :* Outre la consultation et le test d'état de l'arbre, il est utile de posséder des fonctions de descente à gauche et à droite.
- *En destruction :* Celle d'un élément sera déjà délicate.

2.1.2. Du point de vue logique, il n'est pas nécessaire de s'encombrer exagérément : un arbre contient une racine, un sous-arbre gauche et un sous-arbre droit.
Par contre, nous avons besoin de parcourir l'ensemble de l'arbre et nous devons savoir l'ordre de préséance entre le sous-arbre gauche, le droit et la racine (respectivement G, D et R). Il y en a six : GRD, RGD (appelée préfixe pour avoir sa racine en tête), GDR (postfixe), DRG, RDG et DGR. A noter que GRD est parfois appelée infixe. De plus, DGR, DRG et RDG, qui placent droite avant gauche, sont exceptionnellement usitées car contraires à nos habitudes de lecture.
Nous emploierons pour notre part uniquement l'algorithme GRD considérant que c'est celui-ci le plus logique et le plus répandu. Voici, vu de l'extérieur, comment s'exécute un rangement par ordre croissant dans un arbre GRD :



GENERATION D'UN ARBRE DE TRI

Fig. 2.1.2.1



TRI PAR ARBRE (Parcours GRD)

Fig. 2.1.2.2

2.1.3. Vous devez nourrir quelques soupçons sur la représentation physique à laquelle nous ferons appel car elle est directement inspirée de la méthode employée

pour les listes linéaires : nous aurons un tableau où seront stockées les informations et deux tableaux parallèles, l'un avec l'adresse du fils gauche et l'autre avec celle du droit.

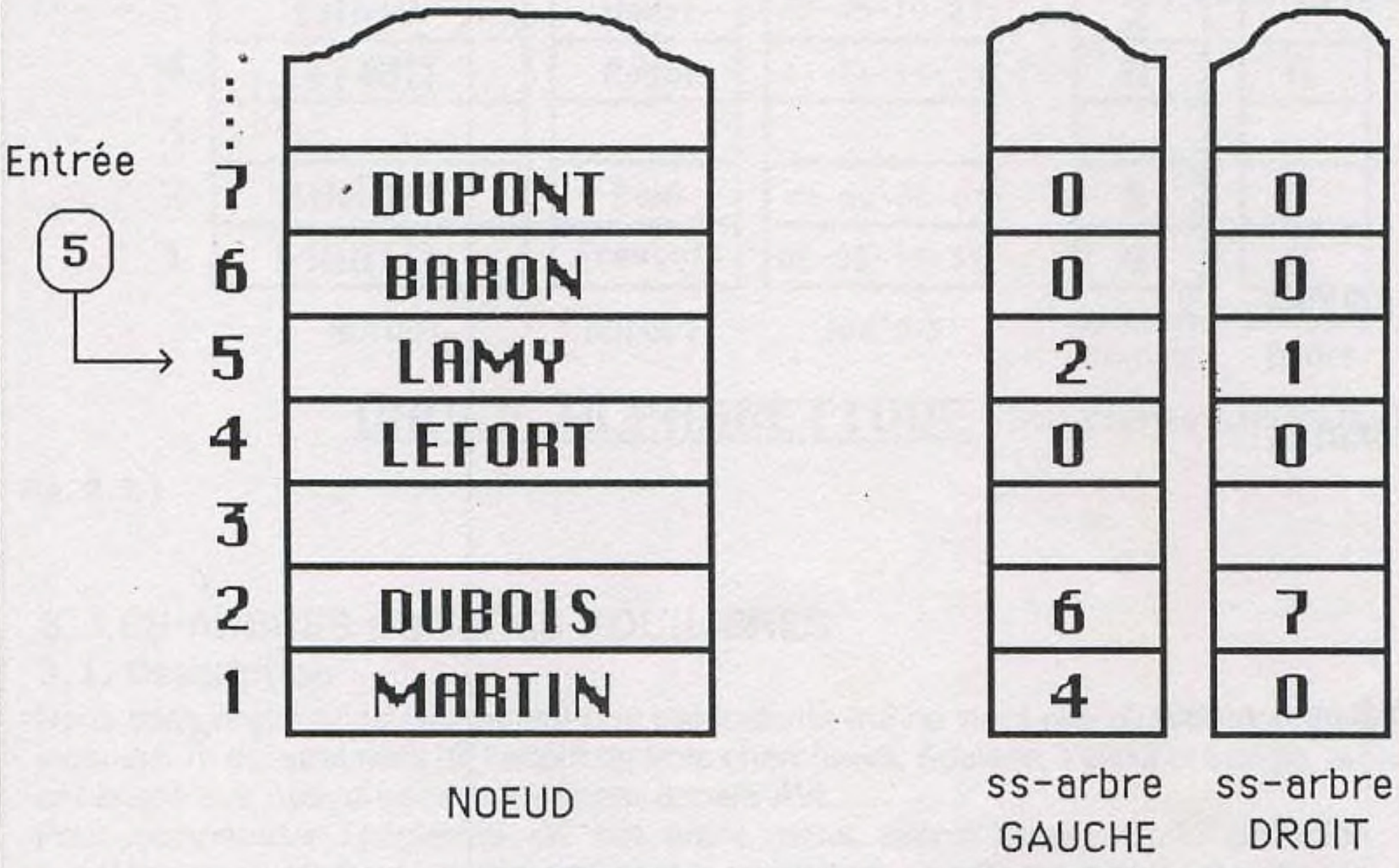


Fig. 2.1.3.1

Pour la manipulation de ce type de tableau, nous vous renvoyons à votre revue du mois dernier où nous avions développé cela en détail, la philosophie est identique.

2.2. Implantation des arbres binaires en BASIC

Soyons honnêtes, le BASIC n'est pas du tout adapté à manipuler des arbres si on le compare à tous les langages acceptant la récursivité. Toutefois, il est possible, comme partout, de le simuler et une fois les procédures écrites, cela devient un jeu d'enfant de les employer.

```
100  REM      INITIALISATION D' AR
     BRE BINAIRE
110 N = 100: REM      DIMENSION DE
      L' ARBRE BINAIRE
120  DIM ARBR$(N),SGAU%(N),SDRO%(
     N)
130 ENTREE = 0:LIBRE = 1
140  FOR J = 1 TO N
150 ARBR$(J) = "VIDE"
160 SGAU%(J) = J + 1
170  NEXT J
180 SGAU%(N) = 0
```

Fig. 2.2.1

```
200  REM    TEST D'ETAT DE L' ARB
     RE BINAIRE
210  IF ENTREE = 0 THEN  PRINT "A
     RBRE BINAIRE VIDE": GOTO 230

220  PRINT "ARBRE BINAIRE REMPLI
     "
230  REM  FIN TEST
```

Fig. 2.2.2.

```
400  REM  DESCENTE A GAUCHE
410 RAC = SGAU%(RAC)
420  RETURN
450  REM  DESCENTE A DROITE
460 RAC = SDRO%(RAC)
470  RETURN
```

Fig. 2.2.3

```
300  REM  CONSULTATION DE LA RACI
     NE
310  PRINT ARBR$(RAC)
```

Fig. 2.2.4

```
500  REM  INSERTION D' UN NOUVEL
     ELEMENT DANS L' ARBRE BINAIR
     E
510 RAC = ENTREE
520 PLACE = SGAU%(RAC) + SDRO%(RA
     C)
530  IF PLACE = 0 THEN  INPUT  INPUT
     ARBR$(RAC):SGAU%(RAC) = 0:SD
     RO%(RAC) = 0: GOTO 600
540  REM  DESCENTE GAUCHE OU DROI
     TE ?
550  IF "GAUCHE" THEN  GOSUB 400:
      GOTO 520
560  GOSUB 450: GOTO 520
600  REM  FIN INSERTION
```

Fig. 2.2.5

2.3. Structures parallèles des arbres binaires

Cela ne pose aucun problème. Nous sommes en présence de la structure logiquement issue des structures parallèles antérieures et de la manière employée pour implanter les arbres binaires en BASIC.

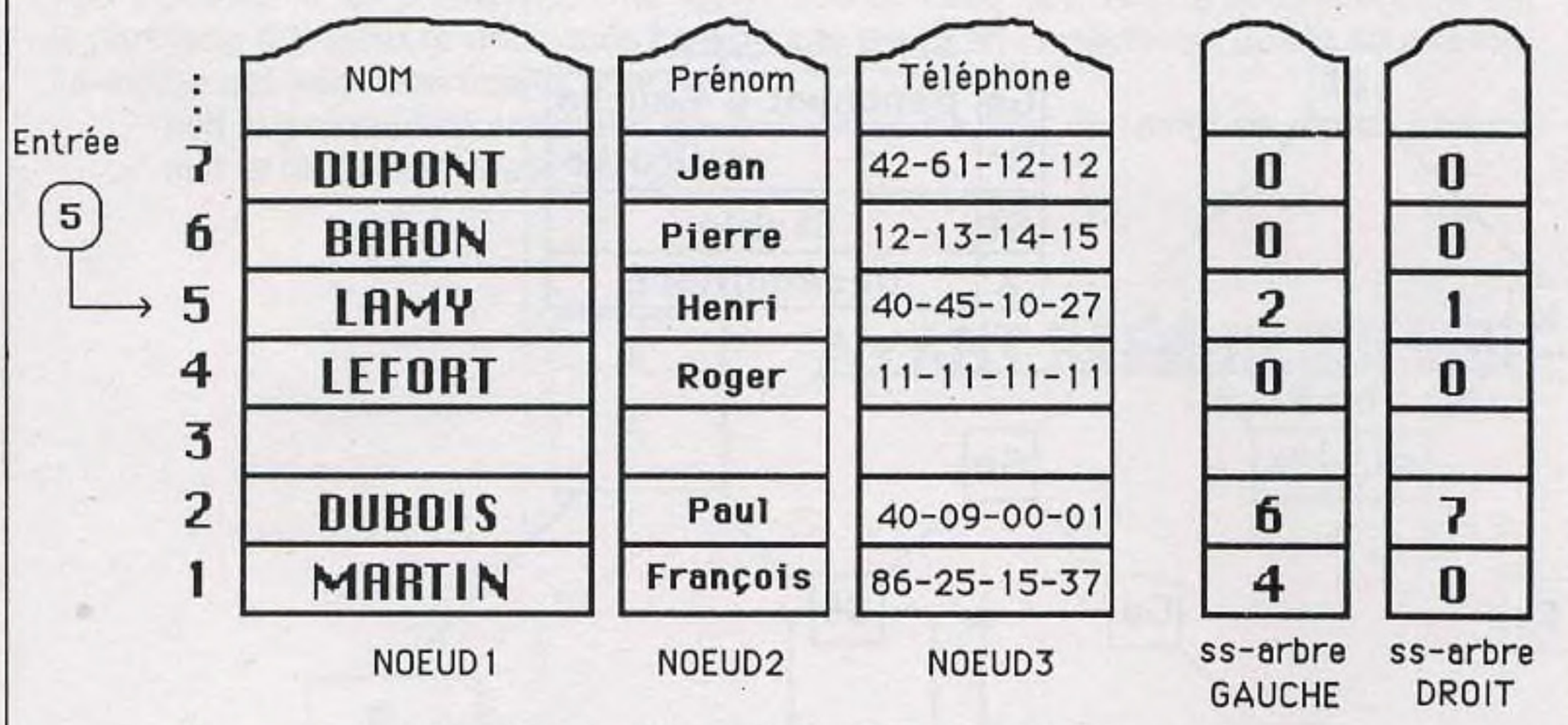


Fig. 2.3.1

3. LES ARBRES BINAIRES EQUILIBRES

3.1. Description

Nous vous présentons maintenant une particularité qui ne vient pas d'un quelconque inconscient collectif mais de l'esprit de trois chercheurs, Adelson, Velskij et Landis, qui ont laissé leur nom à cet arbre encore appelé AVL.
Pour comprendre l'originalité de cet arbre, nous avons besoin d'une définition supplémentaire : la hauteur d'un arbre est le nombre de nœuds sur le plus long chemin entre la racine et une feuille.

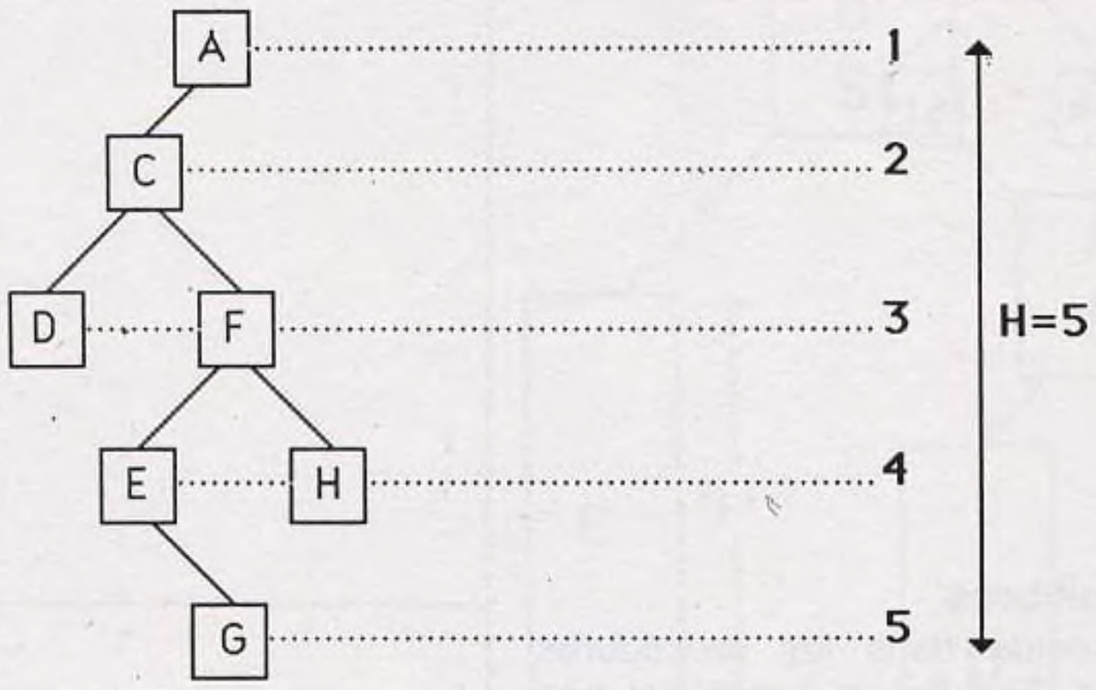
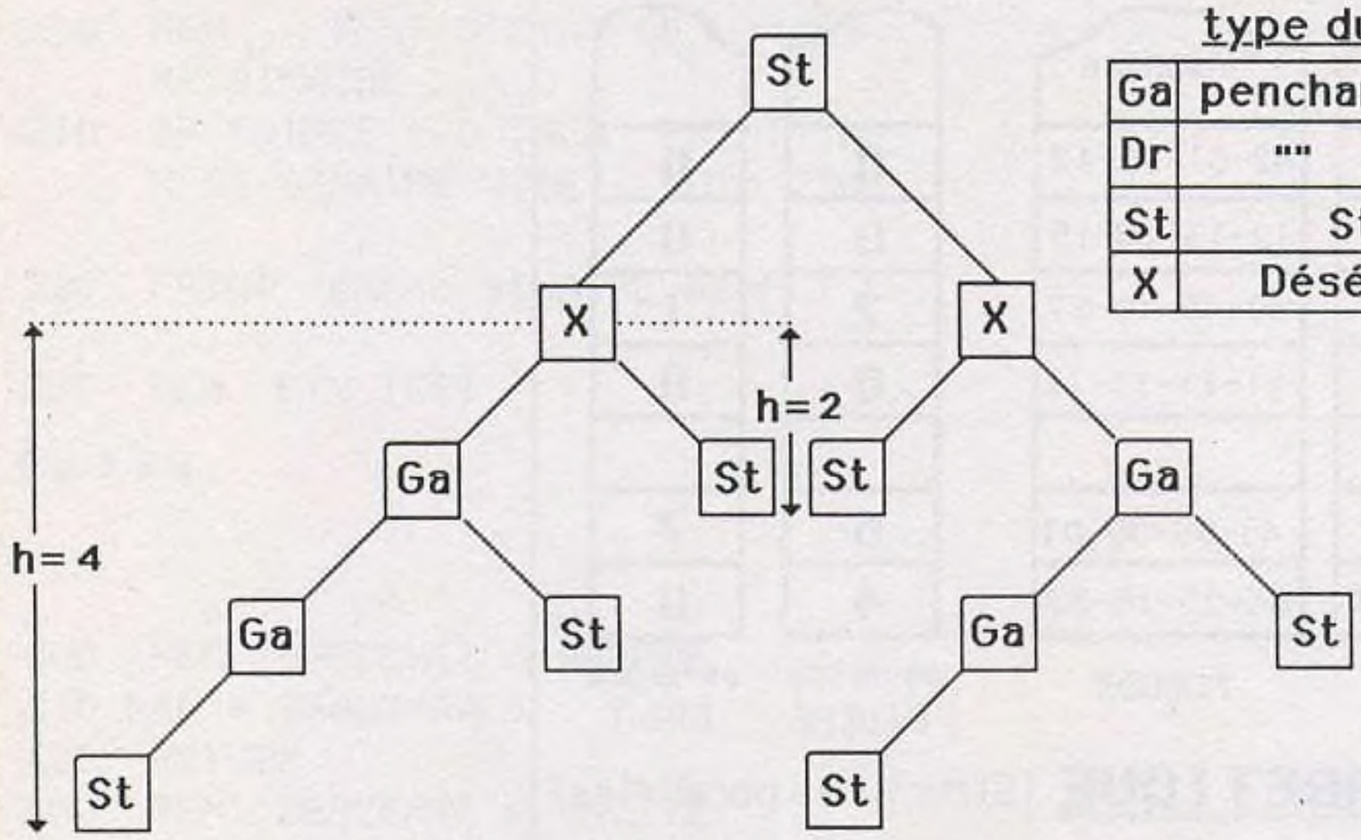


Fig. 3.1.1

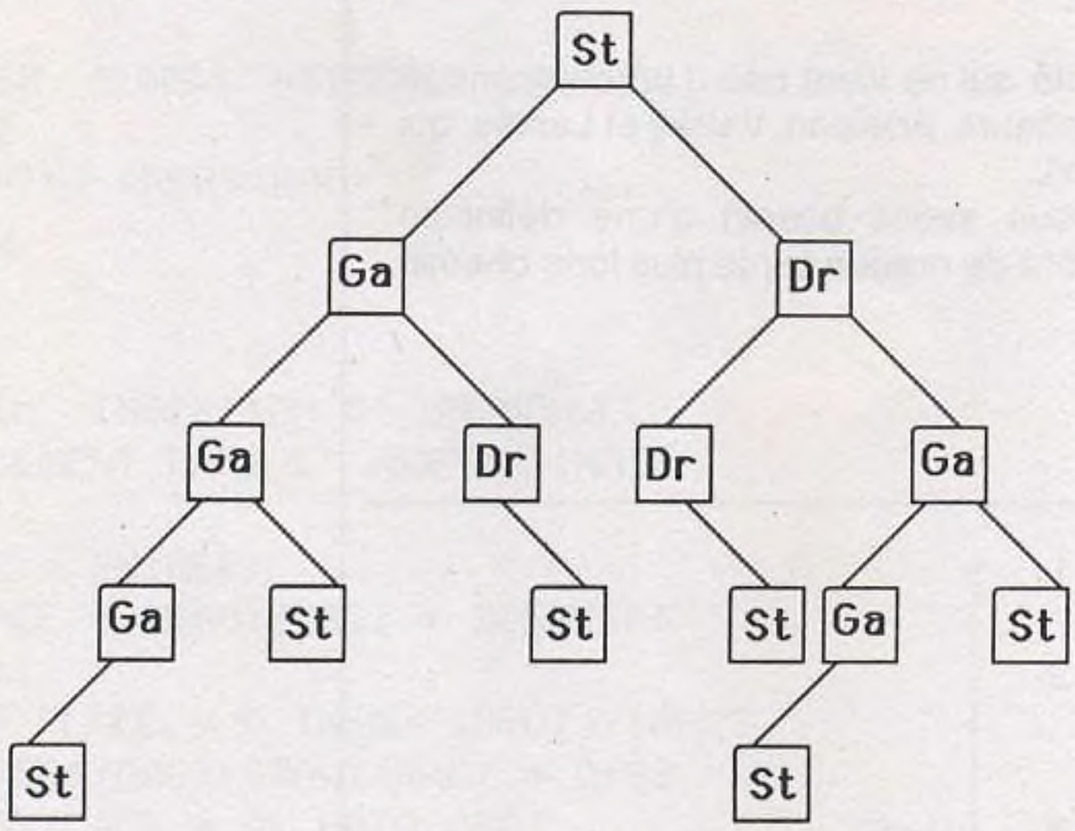
Un arbre est équilibré lorsque, pour tous ses nœuds, la hauteur entre leurs sous-arbres gauche et droit est, au plus, différente d'une unité. Si cette différence est nulle, il est stable ; sinon, il penche à gauche ou à droite.



type du noeud	
Ga	penchant à Gauche
Dr	" " Droite
St	Stable
X	Déséquilibré

ARBRE NON EQUILIBRE

Fig. 3.1.2



ARBRE EQUILIBRE

Fig. 3.1.3

3.2. Avantages et manipulation des arbres équilibrés

3.2.1. Le principal intérêt des arbres équilibrés réside dans les procédures comportant des descentes aux plus profondes feuilles. En effet, si l'arbre est très déséquilibré, il faudra traverser la plupart des nœuds avant d'atteindre l'endroit désiré avec le temps de calcul y inférant. Mais avec un arbre équilibré, on est sûr à chaque nœud que notre choix élimine près de la moitié des éléments, l'arrivée en est d'autant rapprochée.

3.2.2. Cependant ceci risque d'être remis en question lors de l'insertion de tout nouvel élément. En effet, si le dernier arrivé se met au bout d'un sous-arbre de hauteur H alors que son frère est de hauteur H – 1, la différence atteint 2. On procède alors à

un rééquilibrage au niveau de la racine de deux sous-arbres. Il existe deux cas de figure possibles, en prenant comme hypothèse de base que c'est le sous-arbre qui est le plus long des deux (c'est symétrique pour le droit), en considérant que le sous-arbre lui-même est nécessairement penché :

- soit le sous-arbre penche à gauche. Dans ce cas, on remet en cause comme suit le rangement des racines.

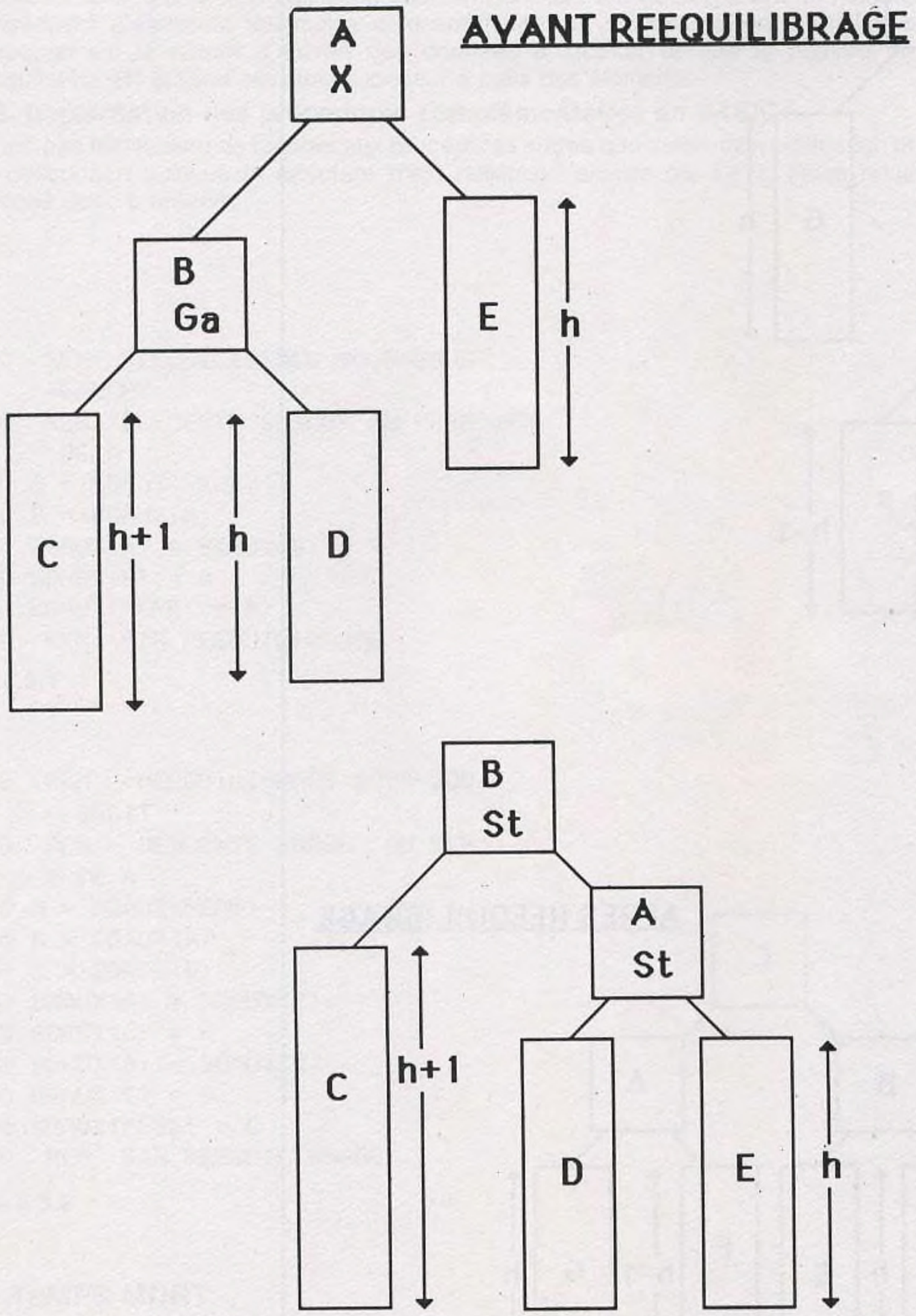


Fig. 3.2.2.1

– soit le sous-arbre penche à droite, et alors, on considère la racine du sous-arbre droit et du sous-arbre gauche et ses deux sous-arbres dont l'un vient de recevoir l'élément supplémentaire et on recale comme ci-dessous :

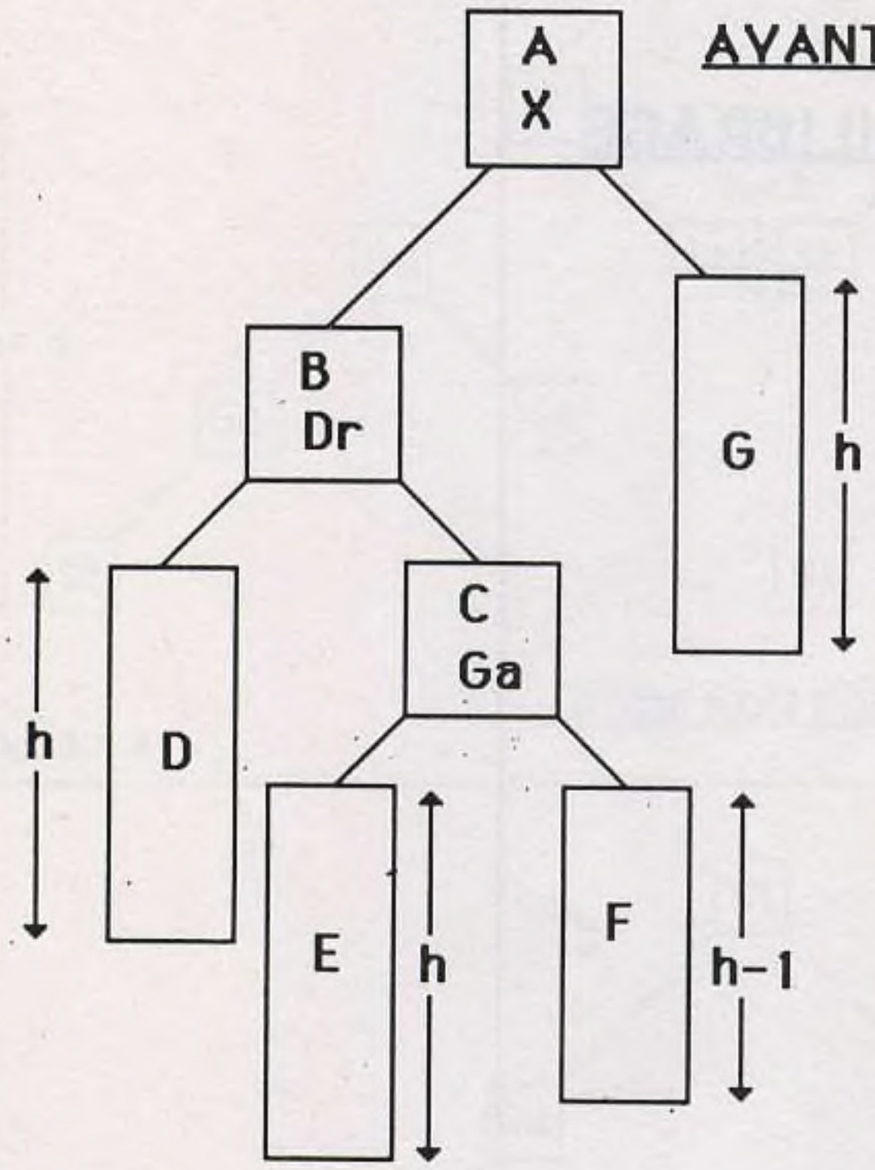


Fig.3.2.2.2

Il est plaisant de remarquer que lorsqu'une insertion modifie la hauteur d'un arbre équilibré, il ne provoque nul rééquilibrage ; inversement, s'il requiert un rééquilibrage, il conserve alors la même hauteur.
De plus, il convient de remonter jusqu'à la racine originelle pour savoir l'impact exact du nouvel élément. La suppression d'un élément entraîne naturellement les procédures inverses de celles proposées.
3.2.3. L'intérêt des arbres équilibrés apparaît pour des arbres contenant un nombre conséquent d'éléments (au moins cinquante) dans la mesure où on a quelques soupçons sur la nature d'arrivée des données à stocker et que le nombre de consultation est environ cinq fois supérieur à celui des éléments.

3.3. Implantation des procédures complémentaires en BASIC

Il n'est pas nécessaire de toucher aux procédures autres que celles de modification et de destruction puisque la structure n'est nullement altérée par l'AVL. Nous nous bornons donc à réécrire :

```
700  REM  REEQUILIBRAGE SOUS-SOUS
     -GAUCHE
710  REM  DESCENTE JUSQU' AU PERE
      DE A
720 A = SGAU%(PERE)
730 B = SGAU%(A)
740 SGAU%(A) = SDRO%(B)
750 SDRO%(B) = A
760 SGAU%(PERE) = B
790  REM  FIN REEQUILIBRAGE
```

Fig. 3.3.1

```
800  REM   REEQUILIBRAGE SOUS-SOU
     --DROIT
810  REM   DESCENTE JUSQU' AU PER
     E DE A
820 A = SGAU%(PERE)
830 B = SGAU%(A)
840 C = SDRO%(B)
850 SGAU%(A) = SDRO%(C)
860 SDRO%(C) = A
870 SDRO%(B) = SGAU%(C)
880 SGAU%(C) = B
890 SGAU%(PERE) = C
900  REM  FIN REEQUILIBRAGE
```

Fig. 3.3.2

4. TEMPS MORT

Nous continuerons le mois prochain en espérant cette fois venir à bout des structures de données, mais déjà ce mois-ci vous avez découvert une des grandes et belles notions de la programmation.

CORRECTIONS ET AMÉLIORATIONS A APPORTER AU Nº 23

1. COURS DE BASIC : L'INSTRUCTION MID$

L'exemple de programme associé à la fonction VAL n'est pas celui qui lui était assigné à l'origine, voici la bonne figure :

```
100 A$="-127"
110 B$="TEXTE 1"
120 C$="1 TEXTE"
130 PRINT VAL(A$),VAL(B$),VAL(C$)
```

et on obtenait

```
-127
   0
   1
```

conversion simple de chaîne en nombre.
car B$ commence par du texte.
car la conversion s'arrête au premier caractère qui n'est pas un chiffre.

Il fallait beaucoup d'imagination pour reconnaître dans cette «révision» de la fonction MID$ la véritable syntaxe. Voici les deux figures telles qu'elles auraient dues apparaître :

```
100 A$="AZERTYUIOP"
110 PRINT MID$(A$,5,3)
```

donne en sortie :

```
TYU
```

```
100 A$="AZERTYUIOPQSDFGHJ"
110 B$=MID$(A$,1,X)
120 C$=MID$(A$,LEN(A$)-X-1,X)
```

où X est la longueur souhaitée.
On obtient l'équivalent pour B$ d'un LEFT$, et C$ d'un RIGHT$ de X caractères.

NOTE : Le premier paramètre d'appel à la fonction MID$ est la chaîne de caractères, **le deuxième, la position du premier caractère retenu** et **le troisième, le nombre de caractères désiré.**

2. COURS DE PROGRAMMATION APPROFONDIE

a. Méthode d'arrondi (partie 1.2)

Comme nous l'a signalé M. Bourdes de Suresnes, il manque un point (qui a disparu à la frappe) avant le 5. Il fallait lire :

```
X=INT(X*100+.5)/100
```

ou, pour éviter tout oubli :

```
X=INT(X*100+0.5)/100
```

b. Partie 3. Calcul sur des entiers longs

La présence de la ligne 128 après la ligne 130 a dû vous sembler pour le moins étrange ! Il fallait en l'occurrence respecter la séquence d'écriture des lignes et non pas la numérotation. Pour tout vous dire, ce programme a été testé (eh oui, nous les testons !) sur Macintosh et le Basic, sur cette machine, n'a plus besoin de numéros de ligne. Il exécute le programme en séquence ligne par ligne. Positionner la ligne «128» avant la ligne «130» fausserait l'exécution du programme.
Il manque une ligne au sous-programme qui commence en 500.
Il faut ajouter :

```
540 PRINT:RETURN
```

Enfin, quelques explications sur ce programme. M. Bourdes nous précise dans sa lettre qu'il ne comprend pas pourquoi éliminer les 0 en trop si on les rajoute juste après :

- Il faut connaître la nouvelle longueur du résultat (ici le tableau T3%). C'est le rôle de la structure WHILE... WEND.
- Ensuite, la partie qui est remise à 0 ne concerne pas celle utilisée pour le calcul, mais au contraire une partie non modifiée. Les lignes 2070 à 2090 servent à annuler toutes les cases non connues. Ceci est très important si on veut intégrer ce module dans un programme plus important et que le tableau T3% est utilisé par d'autres parties de ce programme. Il est toujours préférable de se donner un format de données précis en sortie de sous-programme (cf. figure ci-dessous), afin de maîtriser les informations manipulées.

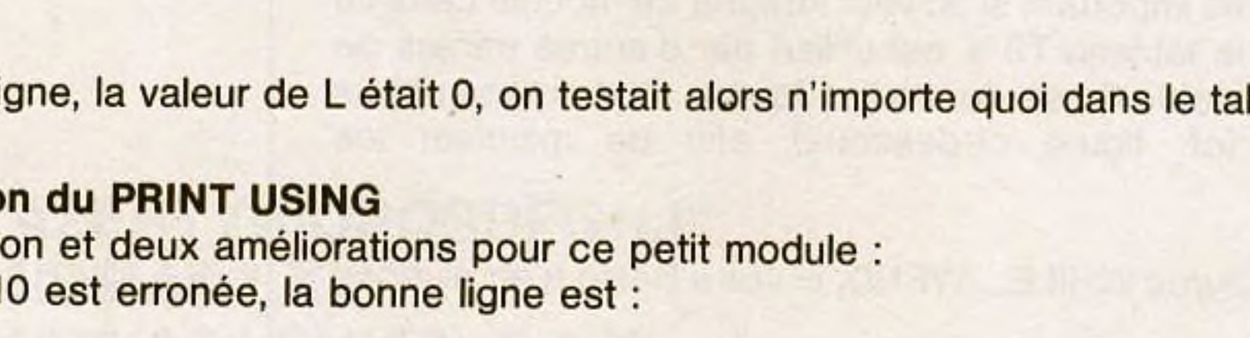
- Voici comment remplacer les structures WHILE... WEND, si votre Basic n'en dispose pas :

```
10 WHILE  <condition>
20 <traitement à effectuer tant que la >
30          <condition est vraie>
40                    ........
 :::
80 WEND
```

sera remplacé par :

```
10 IF NOT(<condition>) THEN GOTO 80
20 <traitement à effectuer tant que la >
30         <condition est vraie>
40                  ........
79 GOTO 10:REM il faut boucler sur le test
80 REM fin de simulation du WHILE
```

CODAGE d'un entier long



CORRECTIONS ET AMELIORATIONS A APPORTER AU N° 24

COURS DE BASIC

a. Programme de répertoire, version complétée :

Le programme de répertoire ne peut pas fonctionner à cause d'une initialisation oubliée : il faut en effet ajouter la ligne suivante au sous-programme de recherche de sous-chaîne commençant en 2000 :

```
2005 L=1
```

Sans cette ligne, la valeur de L était 0, on testait alors n'importe quoi dans le tableau INFO$.

b. Simulation du PRINT USING

Une correction et deux améliorations pour ce petit module :

– La ligne 110 est erronée, la bonne ligne est :

```
110 IF ENT<=10^A-1 THEN NENT=ENT :RETURN
```

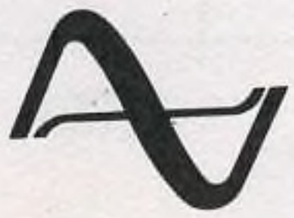
– Pour que ce PRINT USING affiche autant d'astérisques que la taille prévue (ici valeur de A + B), il faut réécrire la ligne 120 qui se charge de recadrer le résultat si la valeur à afficher est trop grande. Dans ce cas, le plus simple consiste à récupérer le résultat directement dans des variables type chaînes de caractères. Ainsi, on aurait :

```
100 REM CALCUL DE LA PARTIE ENTIERE (1ERE PARTIE)
101 REM ************************
105 NENT$=""
110 IF ENT<= 10^A-1 THEN NENT$=STR$(ENT):RETURN
120 REM la partie entière est trop longue
130 FOR i%=1 TO A :NENT$=NENT$+"*":NEXT
140 RETURN


150 REM CALCUL DE BLANCS A AJOUTER
151 REM ************************
152 REM le calcul est beaucoup plus simple
155 ESP=A-LEN(NENT$)
160 RETURN


200 REM PARTIE DECIMALE
210 REM ***************
212 NDEC$=""
215 IF NENT <=10^A-1 THEN GOTO 220
217 FOR I%=1 TO B :NDEC$=NDEC$+"*":NEXT
220 NDEC=INT(DEC*10^B)/10^B
230 NDEC=NDEC*10^B
240 NDEC$=STR$(NDEC)
250 RETURN


300 REM AFFICHAGE
310 PRINT NOMBRE
320 FOR J=1 TO ESP
330 PRINT ESP$;
335 NEXT
340 PRINT NENT$;".";NDEC$
350 RETURN
```

Remarque : Il est intéressant de noter qu'on évite avec cette méthode l'apparition (sur certains ordinateurs) d'espaces dans l'écriture du nombre, qui est due à la place réservée au signe.

– Cette remarque nous amène à formuler une évolution de ce programme pour qu'il soit capable de gérer les nombres négatifs de la même façon que les positifs. A vous de jouer !

C'EST ARRIVE DEMAIN



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

Il va y avoir cet hiver une lutte intéressante entre les deux géants du logiciel intégré Lotus et Ashton-Tate, qui se sont faits connaître grâce à des produits comme 1-2-3 (Lotus) et D-Base II et III (Ashton-Tate), ont sorti simultanément l'année dernière des logiciels intégrés de haut niveau, au moins sur le plan prix (Symphony et Framework). Ces programmes, très semblables pour les caractéristiques techniques, sont destinés à tout faire. Hélas, les acheteurs des deux produits se plaignirent très vite de ces programmes, pour des raisons de détails pratiques ou de limitations incompatibles avec les ambitions annoncées. Ces programmes, fort coûteux, se sont donc vendus, mais connurent un succès somme toute modeste. Désireux de faire oublier ces lacunes, les deux fabricants proposent depuis quelques jours des nouvelles versions, après révisions complètes des anciennes. Il est intéressant de remarquer que ces deux produits ont été créé en même temps, ont connu un semi-échec en même temps, et proposent des modifications simultanément. Ils fonctionnent sur des PC compatibles tous les deux, coûtent le même prix, demandent la même capacité mémoire de base. En bref, tous les ingrédients pour un deuxième round de qualité sont réunis, mais il reste la possibilité d'un deuxième faux départ. Ce serait dommage pour les

acheteurs de produits valant dans les 7 000 francs. Les fabricants également se livrent une lutte acharnée, et dont le moins qu'on puisse dire est qu'elle est de moins en moins «fair-play». Tout se passe à coup de réductions mirobolantes, de promotion sur un ensemble de produits, de «package» à des prix défiants toute concurrence, mais dont la moitié ne sert à rien pour un particulier. En résumé, la guerre des prix prend de plus en plus une allure de tentative de mise à mort. Qui aura les deux oreilles et la queue ?

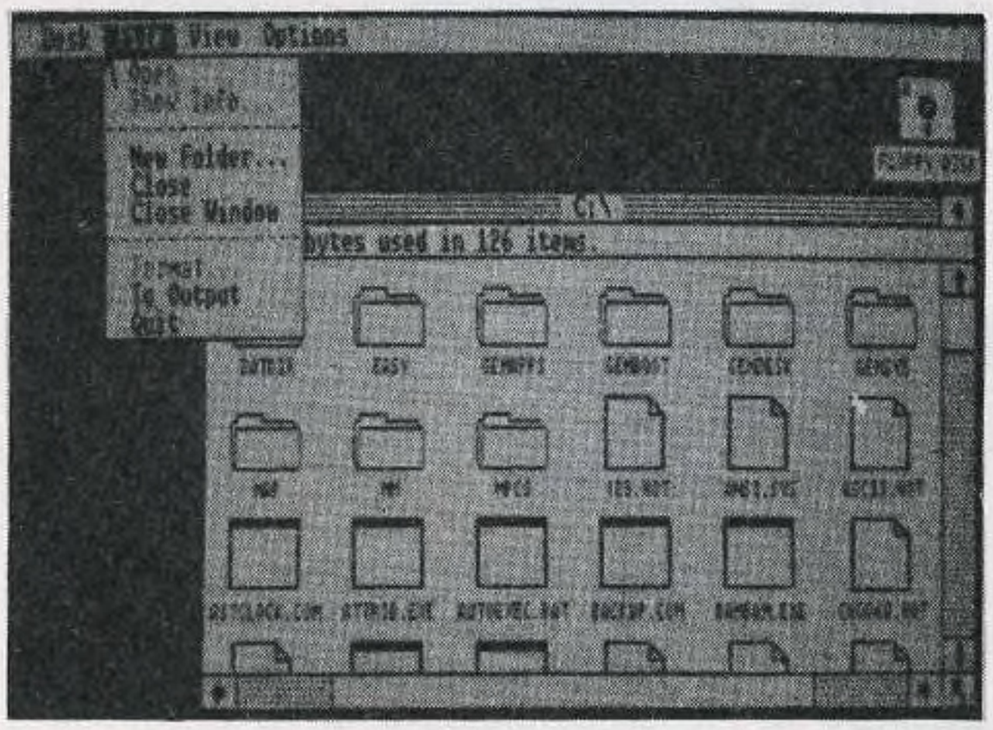
IBM propose ainsi des réductions de l'ordre de 27 % sur la gamme PC, depuis le Junior, jusqu'à l'AT. Pour mémoire, rappelons que le Junior a été retiré de la vente en Europe peu de jours après sa sortie (en 1984) et la raison officieuse était son manque de performance !! Le proposer encore aux USA est un paradoxe à noter, quand on sait que les Américains considèrent les Européens comme des sous-développés. IBM vient de faire encore plus fort. Comme des fabricants indépendants proposaient des compatibles PC-AT, le plus gros de la série, mais avec plus de mémoire sur disque dur, IBM vient d'annoncer la sortie d'un nouveau PC-AT avec plus de mémoire lui aussi. Le problème est que les revendeurs s'étaient largement pourvus du modèle précédent qui venait lui aussi de sortir. Or, ces modèles n'ont plus la moindre chance de se vendre, personne ne voulant s'équiper de matériels «sous-dimensionnés», et d'ores et déjà obsolètes. Il est question d'un refus de la part des revendeurs de vendre du matériel IBM pendant un mois.

Apple, pour sa part continue son numéro à grand spectacle, débuté le jour de l'annonce du LISA, il y a maintenant plus de deux ans. Le régime de la douche froide est un jeu où la firme est passée maître. Un jour on annonce des nouveaux produits, et le lendemain, le même porte-parole annonce la fermeture d'une usine. Un jour on propose des réductions, le lendemain, le même produit proposé à prix réduit est abandonné (c'est le cas du MAC 128, abandonné cette semaine par la firme, alors que des logiciels spécialement étudiés pour lui sont sortis ce mois-ci. Les fabricants indépendants doivent être assez furieux). De même, l'annonce d'un disque dur pour le Macintosh rend furieux les fabricants indépendants, que Apple avait poussé à proposer des matériels semblables il y a quelques mois, et qui risquent de se retrouver avec leurs produits sur la liste des records d'invendus. La valse hésitation n'amuse hélas plus personne ici, et les réductions proposées par Apple, si elles allèchent les particuliers, sont méprisées par les entreprises qui auraient dû être la cible d'Apple pour l'année 1985. Le moins qu'on puisse dire, c'est que c'est raté !!

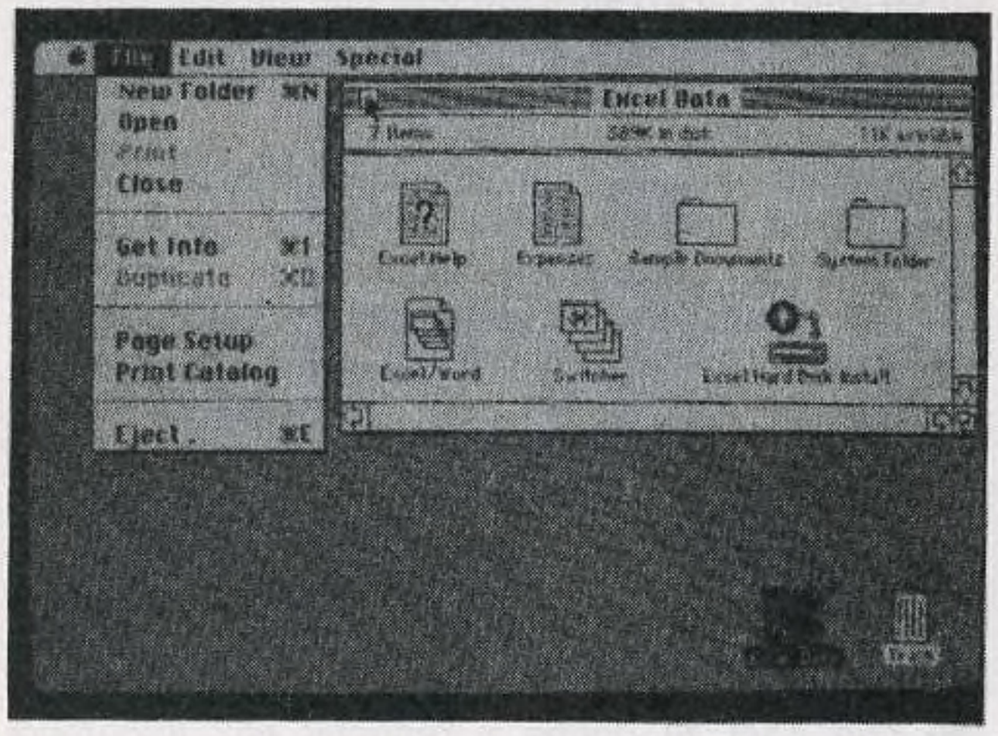
Les compatibles IBM continuent leur petit bonhomme de chemin au milieu de ce capharnaüm, réalisant pour la seconde année fiscale consécutive le meilleur chiffre d'affaires cumulé. En voilà qui pourront embrasser le patron de la division «planification long terme» d'IBM. Ce sont les seuls à pouvoir vendre sans être obligés de réviser leurs prix chaque semaine, car ils sont nettement moins chers que les produits de la grosse firme de Rochester.

Une autre société qui semble avoir quelques difficultés est Atari. En effet, le 520 ST, décrit par toute la presse comme une excellente machine, a du mal à atteindre sa vitesse de croisière, en raison de multiples petits problèmes de jeunesse. Le moins qu'on puisse dire est que l'exemple du Macintosh n'aura pas servi à Jack Tramiel, le p.d.g. d'Atari. En effet, premier exemple, cet appareil devait être livré avec la dernière merveille dans le domaine des systèmes d'exploitation, GEM, de Digital Research. Or, ce logiciel rencontre deux problèmes. Tout d'abord, il a subi un tel retard, qu'il n'aura été prêt que lorsque cet article a été écrit (15 octobre), mais de plus, Apple a réussi à contraindre Digital Research, à l'amiable, de modifier GEM, car il ressemblait par trop au système d'exploitation du Macintosh (pour que Digital ait accepté de modifier GEM sans même un procès, il faut que ce produit soit vraiment une copie du système Apple, voir sur nos photos). Ceci va encore retarder la commercialisation de GEM, au moins de quelques mois. Atari est donc réduit à proposer le 520 ST avec un système d'exploitation maison, dont l'originalité avait peut-être été grande dans les années 1979-1980, mais qui n'intéresse plus grand monde maintenant. En clair, cet appareil peut, pour l'instant, s'identifier à une Ferrari dans laquelle Atari aurait mis un moteur de... disons de Renault 5. Il semble que ceci ait donné à réfléchir aux acheteurs potentiels, ce qui est bien dommage car la machine semble vraiment prometteuse. Avec la concurrence de Commodore (l'Amiga semble tout aussi performant) et les réductions de prix chez Apple (qui bénéficie d'une bibliothèque de produits et de logiciels maintenant très importante), l'Atari 520 ST aura du mal à se faire une place au soleil. D'autant plus que les premiers vendus ont une nette tendance à retourner à l'usine pour cause de plantages intempestifs.

Cela n'arrange en rien l'image de marque vacillante de cette entreprise, en méforme financière depuis deux ans.



Gem.



Le système du Macintosh Finder.

Je vous parle peut-être à chaque fois des nouvelles méthodes de stockage à hautes capacités, mais à chaque exposition des matériels nouveaux sont proposés. A Chicago, des disques optiques ont été dévoilés, qui permettent non seulement de lire des données stockées dessus, mais aussi d'en écrire. C'est là une révolution dans le domaine des supports d'information, même si de tels progrès étaient attendus. En effet, depuis plusieurs mois, des nouvelles filtraient sur ce type de périphériques.

Jusqu'à présent, les disques optiques n'étaient pas autre chose que les disques, compacts ou vidéodisques, que l'on trouve sur le marché pour de toutes autres applications. Ils offraient des capacités de stockage de l'ordre du gigaoctet, 1 000 mégaoctets. Ceci représente vraiment une capacité importante, mais le fait de pouvoir écrire sur de tels supports est vraiment autre chose. Verbatim, connue pour ses disquettes, propose un tel lecteur en démonstration et il sera commercialisé début 1987. Il dispose de 100 mégaoctects par disque, est amovible, ce qui en fait un support de capacité effective quasi-illimitée, pour un prix de l'ordre de 300 $ le lecteur et de 50 $ le disque. Ceci est très compétitif lorsqu'on sait qu'un disque dur de 20 mégabytes (soit cinq fois moins) coûte au minimum le double. Ce type de produit reste malgré tout du domaine de la démonstration, et en un an, les spécifications et les prix sont susceptibles de grands changements. Cela rejoint ce dont je vous ai parlé, les annonces fracassantes, mais fausses, pour éviter qu'un acheteur potentiel choisisse un matériel concurrent avant la sortie réelle d'un produit, parfois deux ans plus tard.

Le fin du fin a été mis au point par Microsoft. Plusieurs mois avant la sortie de EXCELL, leur logiciel à tout faire pour Macintosh, ils en ont proposé des préversions à qui en voulait, afin de leur montrer à quel point ce produit est brillant. La plupart des bénéficiaires ont ainsi attendu parfois six mois la sortie de la version commerciale, sans acheter JAZZ, le produit concurrent, sorti avec près d'un an d'avance.

Pour changer un peu des nouveautés en matériels, une information qui passera peut-être à la postérité. Un sénateur américain vient de proposer une loi sur la propriété privée en matière de fichiers informatiques. Si cette loi est votée, et il y a de fortes chances qu'elle le soit, toute personne violant l'accès à un ordinateur central, à travers quelque voie que ce soit, pourra être condamnée à 10 000 $ d'amende, et à dix ans d'emprisonnement. Ceci, quel que soit le motif de l'entrée sur des fichiers, et même si aucun dommage n'est noté. Une loi équivalente existe déjà, dans le cas d'ordinateurs appartenant à des offices gouvernementaux ou à des compagnies bancaires. La nouvelle loi serait donc une extension au domaine privé. En fait, deux groupes de pressions différents se sont réunis pour la première fois au sujet de cette loi. Les sénateurs qui défendent la propriété privée et ceux dont le seul but est de poursuivre des gens dont ils estiment que leur acte est criminel, en lui-même.

COURS DE GENIE LOGICIEL

De la théorie à la pratique

Charles-Henry Delaleu

LES TRIS

A l'origine, les ordinateurs ont été conçus pour effectuer des calculs. Très vite, l'homme s'est aperçu des possibilités énormes de ces machines et il lui a affecté de nouvelles tâches. Dans un second temps, les calculateurs n'ont plus été un outil privilégié des laboratoires et ils ont fait leur apparition en gestion. Dans cette discipline, le calcul occupe une place de second choix. En effet, la machine sera surtout utilisée pour effectuer des manipulations de chaîne, ainsi que du stockage d'informations. Ici, les notions de tris prennent une signification très importante. Beaucoup d'informaticiens aiment raconter que dans une application de gestion, l'ordinateur occupe 90 % de son temps à réaliser des tris, 8 % en gestion de mémoire de masse, et uniquement 2 % en calcul.

Il est certain que, dans les domaines de gestion de ressources humaines, le calculateur est occupé en majorité par les tris. C'est moins vrai en comptabilité.

De toute manière, les tris sont l'un des chapitres les plus importants des algorithmes. Chaque programmeur se doit de les connaître, de les maîtriser.

Il existe deux grands principes de tri. Le premier, le plus simple, consiste à classer dans un ordre croissant ou décroissant, des nombres, ou des chaînes de caractères. Le second, beaucoup plus subtil, a pour but de réaliser des recherches du style : chercher les personnes de quarante ans qui possèdent une voiture verte et habitent en pavillon.

Le cahier des charges étant différent d'une application à l'autre, plusieurs types d'algorithmes de tris ont donc été créés.

DEFINITION

On appelle tri une opération qui consiste à ordonner un ensemble d'éléments en fonction de clés sur lesquelles est définie une relation d'ordre. En fait, il est possible d'assimiler le terme tri au terme classement. Il existe, de nos jours, quatre grands algorithmes de tri :

- le tri à bulles,
- le tri par arbre,
- le tri rapide,
- le tri par fusion.

Les tris appartiennent à deux grandes classes :

- les tris internes,
- les tris externes.

TRI INTERNE

Le tri interne opère sur des données présentes en mémoire centrale.

TRI EXTERNE

Le tri externe opère sur des données présentes sur des fichiers externes.

NOTA : Les optimisations sont, bien évidemment, très différentes dans les deux cas. En tri interne, il est fondamental de réduire au maximum le nombre de comparaisons et de manipulations. (Dans le cas contraire, la machine serait très vite saturée au niveau de sa mémoire vive). En tri externe, l'efficacité de l'agorithme sera tributaire du nombre d'accès en entrées-sorties sur le ou les disques de mémoire de masse.

EXEMPLE N° 1

Le tri le plus simple est, sans aucun doute, le classement par ordre alphabétique d'une dizaine de noms propres, par exemple (fig. 1). L'algorithme de base serait, dans ce cas :

1. prendre dix noms au hasard,
2. placer ces noms par ordre alphabétique.

Une seconde option serait de classer des nombres par ordre croissant ou décroissant.

Soit dix noms :

DUPONT, DURAND, PETIT, GRAND, LED MICRO, FREQUENCES, EDITIONS, PARIS, AMIENS, CAEN

les dix noms sont pris dans l'ordre ci-dessus. Grâce à un algorithme particulier, nous désirons qu'ils soient rangés par ordre alphabétique. Il nous faut donc, pour cela, un algorithme de tri. Ce dernier effectuera sa tâche en replaçant les noms par passages successifs :

PASSAGE 0 (départ)

DUPONT, DURAND, PETIT, GRAND, LED MICRO, FREQUENCE, EDITIONS, PARIS, AMIENS, CAEN.

PASSAGE 1

AMIENS, DUPONT, DURAND, PETIT, GRAND, LED MICRO, FREQUENCES, EDITIONS, PARIS, CAEN

PASSAGE 2

AMIENS, CAEN, DUPONT, DURAND, PETIT, GRAND, LED MICRO, FREQUENCES, EDITIONS, PARIS

PASSAGES 3 ET 4

Identique à PASSAGE 2 (les noms sont classés : Dupont Durand)

PASSAGE 5

AMIENS, CAEN, DUPONT, DURAND, EDITIONS, PETIT, GRAND, LED MICRO, FREQUENCES, PARIS

PASSAGE 6

AMIENS, CAEN, DUPONT, EDITIONS, FREQUENCES, PETIT, GRAND, LED MICRO, PARIS

PASSAGE 7

AMIENS, CAEN, DUPONT, DURAND, EDITIONS, FREQUENCES, GRAND, PETIT, LED MICRO, PARIS

PASSAGE 8

AMIENS, CAEN, DUPONT, DURAND, EDITIONS, FREQUENCES, GRAND, LED MICRO, PETIT, PARIS

PASSAGE 9

AMIENS, CAEN, DUPONT, DURAND, EDITIONS, FREQUENCES, GRAND, LED MICRO, PARIS, PETIT.

«Les dix noms sont classés.»

Fig. 1

EXEMPLE N° 2

Dans un service de production d'une grande usine, quatre-vingt personnes fabriquent et testent une pièce déterminée. Afin d'augmenter la qualité, le directeur de la production demande au directeur de l'informatique de lui mettre au point un programme qui conservera la trace du nombre de pièces fabriquées par personne et par semaine. Il désire connaître le taux de produits défectueux par personne. Il veut pouvoir accéder par le nom de l'employé, le nombre d'unités fabriquées, ou le taux de produits défectueux.

La structure des données sera conforme à la figure 2. Cette structure est simple et contient toutes les informations utiles. La nième valeur du tableau «unités fabriquées» indique le nombre d'unités fabriquées par l'employé dont le nom se trouve à la nième place dans le tableau «nom employé». La nième valeur du tableau «taux de produits défectueux» indique le taux de produits défectueux trouvés par le nième employé sur le lot d'unités qu'il a fabriqué.

	Nom des employés	Unités fabriquées	Taux de produits défectueux
1			
2			
3			
4			
5			
	⋮	⋮	⋮
79			
80			

Fig. 2

NOTA :Nom des employés sera un tableau comprenant 80 cases configurées en chaîne de caractères (80 caractères par exemple).
Unités fabriquées sera un tableau comprenant 80 cases configurées en nombre entier.
Taux de produits défectueux sera un tableau comprenant 80 cases configurées en nombre réel. (Il s'agit, dans ce cas, d'un pourcentage).

Le seul problème qui ne soit pas résolu par cette structure est celui de l'ordre. Le directeur de la fabrication veut pouvoir obtenir un compte rendu en appelant un élément d'un des trois tableaux. Une solution à ce problème serait d'inclure au module un sous-programme de tri ; pour déplacer un élément dans le tableau, il faut aussi déplacer les éléments associés dans les deux autres tableaux. Une autre solution consiste à ne pas modifier l'ordre des données mais à construire un tableau de pointeur associé au tableau à partir duquel nous désirons effectuer le tri. La manière la plus simple de construire ce tableau en vue d'impression du tri est de faire un arbre binaire.

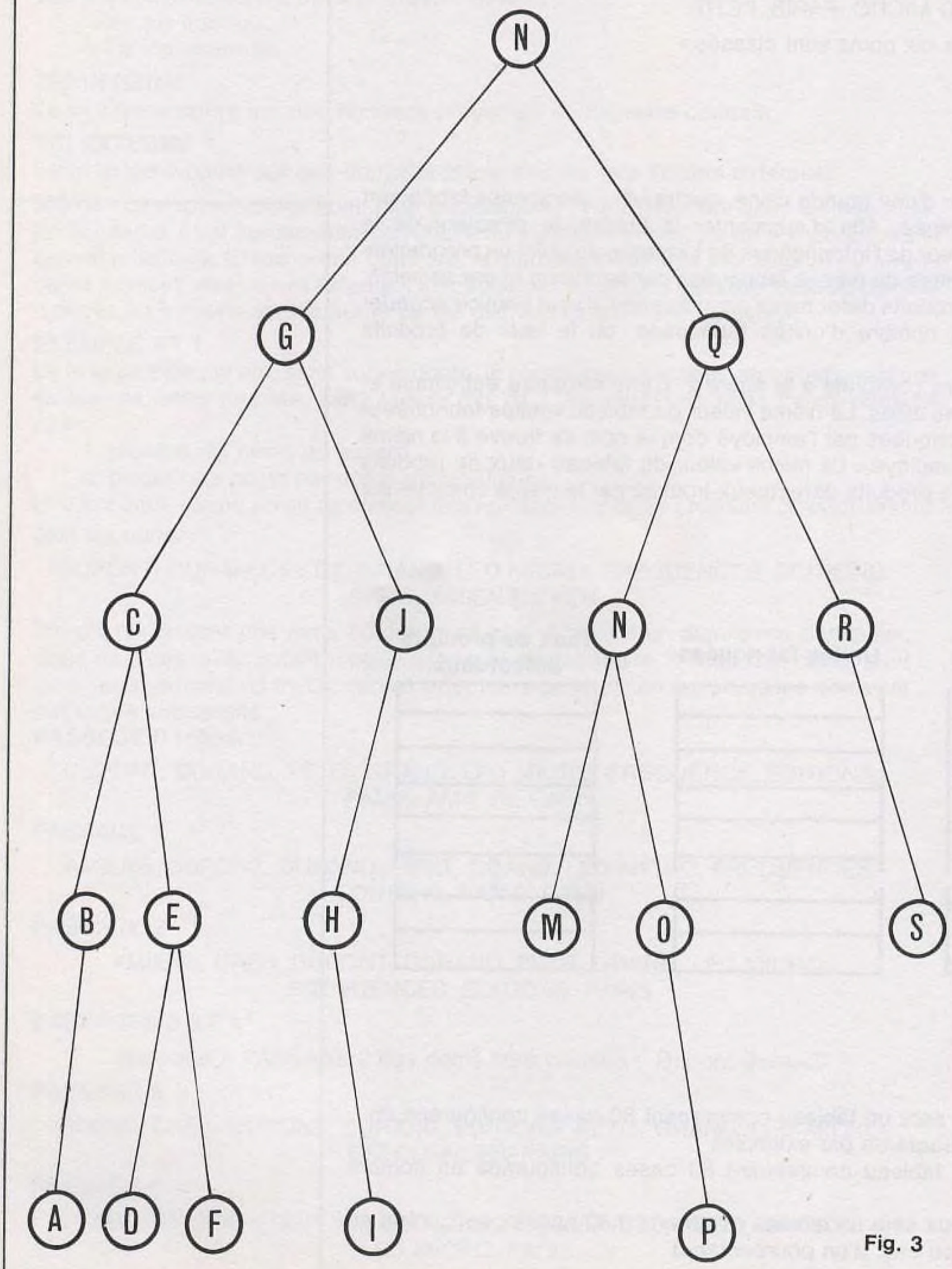
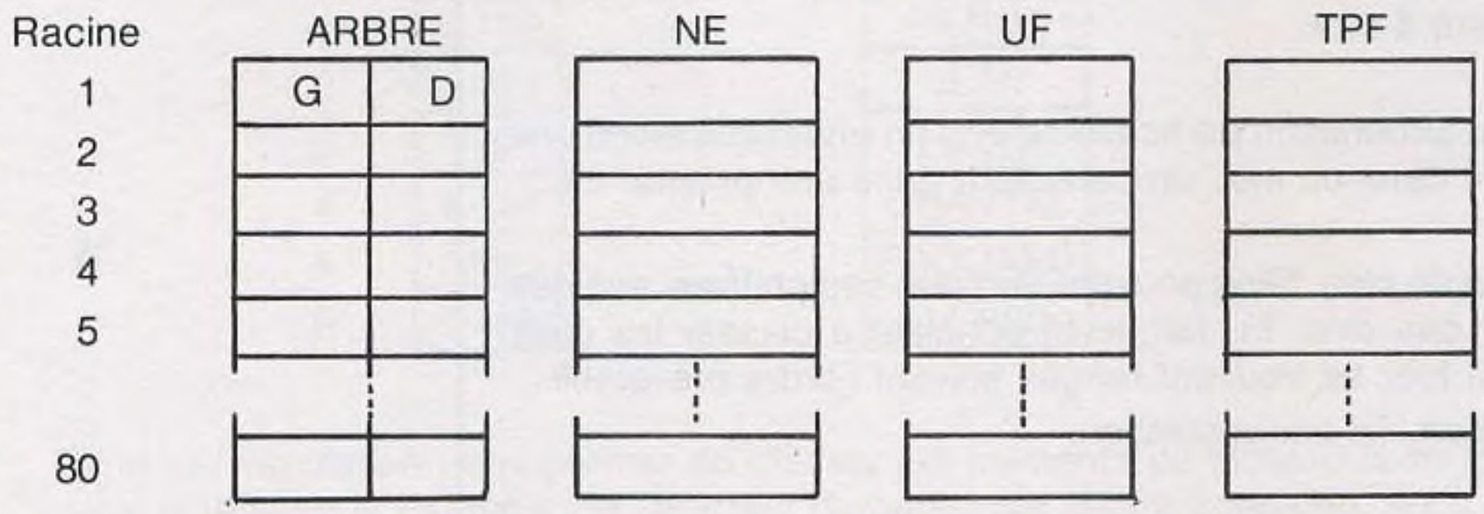


Fig. 3

Un arbre binaire est une structure simple utilisée dans quantité d'applications, de la gestion de données à l'analyse des langages informatiques. Knuth définit l'arbre binaire comme un ensemble fini de nœuds dont certains sont vides et les autres composés d'une racine et de deux arbres binaires distincts appelés sous-arbres à droite et à gauche de la racine. En fait, un arbre binaire est constitué de deux sous-arbres (qui ont à leur tour deux sous-arbres, etc.) ou est vide.
Cette définition est récurrente, elle utilise le terme à définir (arbre binaire) dans sa propre définition. Voir figure 3.

Chaque nœud (représenté par une lettre) possède au plus deux sous-arbres, les sous-arbres sont classés dans un ordre lexicologique. Chaque lettre d'un nœud du sous-arbre de gauche sera d'ordre moins élevé que le nœud lui-même ; à son tour, le nœud d'ordre moins élevé que les lettres des nœuds du sous-arbre de droite. A cause de la définition récurrente de l'arbre binaire, cette relation d'ordre lexicologique est valable à tous les niveaux.
Cet arbre est très simple à comprendre. A partir d'une donnée (ou nœud) partent deux flèches qui vont vers les nœuds suivants. Dans un ordinateur, ces flèches s'appellent «pointeurs» car ils pointent l'emplacement mémoire où se trouve le nœud suivant.

Notre arbre binaire est représenté par un tableau 80 × 2. Chaque élément du tableau A (I, 1) est le pointeur du sous-arbre de gauche, et A (I, 2) est le pointeur du sous-arbre de droite. I indique simplement l'emplacement de la donnée choisie dans les autres tableaux. Le premier élément de chaque tableau est la racine de l'arbre.
La structure des données est donc la suivante :



Le directeur de production peut choisir les éléments du tri pour effectuer l'impression, le tableau sera alors construit.
Le programme sera écrit de la manière suivante :

A. Le programme principal contient les données de base et l'appel à trois sous-programmes.
B. Sous-programme de saisie des données.
C. Sous-programme de stockage des données.
D. Sous-programme rapport.

Grâce à l'arbre binaire, il sera possible de trier les employés l'un après l'autre en fonction de leurs résultats.

COMPLEXITÉ DES ALGORITHMES DE TRI

Il convient d'être vigilant lorsque nous employons un algorithme de tri dans une application. Dans le meilleur des cas, un algorithme de tri effectuera son tri en un nombre de comparaisons égal à n – 1, n étant le nombre d'éléments à trier. Dans le pire des cas, les choses se compliquent notablement et peuvent arriver à un nombre d'opérations très important pouvant se traduire :

- par un temps d'exécution prohibitif,
- par une saturation de l'espace mémoire.

Dans la complexité maximum, le nombre d'opérations atteindra :

n log n opérations.

De ce fait, à chaque opération de comparaison supplémentaire, le temps de traitement se verra augmenter en fonction de l'équation ci-dessus.
Il est donc facilement concevable que le choix et la mise en œuvre de l'algorithme choisi soient d'une importance capitale.

TRI PAR INSERTION

Le tri par insertion procède de la façon suivante. Les éléments du tableau à ordonner sont recopiés dans un deuxième tableau. Chaque fois qu'on prend un élément du premier tableau, on cherche sa place dans le deuxième tableau qui sera, lui, ordonné. La place étant trouvée, on décale les éléments suivant l'élément à insérer, de manière à libérer la position où il sera recopié.

TRI PAR FUSION

Le tri par fusion reprend l'idée du tri par insertion mais ici on découpe le tableau de base en deux tableaux de taille identique qu'on trie séparément. Une fois cette opération réalisée, on fusionne les deux tableaux triés. Ce procédé est exclusivement utilisé pour les fichiers volumineux sur dérouleur de bande.

TRI PAR ECHANGE

Ce tri s'effectue par une série de passages qui permettent de partir de la valeur la plus élevée à la plus faible. En d'autres termes, il consiste à prendre l'élément maximal de l'ensemble de données, puis à l'amener en première position. Cette opération est recommencée sur l'ensemble de données privé de l'élément déjà mis en place. Le nouveau maximum est ainsi mis en deuxième position et ainsi de suite. Ce processus se répète jusqu'au dernier élément à trier.

TRI PAR EXTRACTION

Ce tri consiste à sélectionner une information particulière dans un ensemble selon une clé donnée. Cela peut être un bit dans un mot, un caractère dans une chaîne, etc.

TRI PAR VENTILATION

Le tri par ventilation se fait à l'aide de clés. Elles peuvent être soit des chiffres, soit des caractères. Le tri s'effectue sur ces clés. En fait, le tri consiste à classer les clés. Automatiquement, les éléments à trier se trouvent rangés suivant l'ordre pré-établi.

NOTA : Voir fiche Tri par extraction, Tri par ventilation.

CONCLUSION

Afin de bien comprendre ces notions de tri, les exemples 1 et 2 sont repris dans les fiches suivantes. Les exemples utilisés sont tirés du manuel Hewlett-Packard : «Techniques de programmation en BASIC», réf. 09826-90104. Rappelons que le BASIC-HP est un langage très puissant qui fait une sorte de synthèse des langages suivants :

- BASIC
- FORTRAN
- PL/I
- ALGOL.

La traduction des exemples donnés ne posera aucun problème, ils sont facilement traduisibles en MS-BASIC, FORTRAN, PL/I, PASCAL, etc...

TRI PAR EXTRACTION - TRI PAR VENTILATION

TRI PAR EXTRACTION

Soit les deux tableaux suivants : le tableau A contient les clés, le tableau B les éléments à trier :

	A	B
1	B	PIERRE
2	C	JEAN
3	A	HELENE
4	C	CHANTAL
5	D	EDOUARD
6	E	ALFRED

Objet du tri : Classer les prénoms à l'aide de clés : afficher les prénoms à l'aide de la clé A, puis B, C, D, E.
Tableau avant classement : Pierre, Jean, Hélène, Chantal, Edouard, Alfred.
Résultat du tri : Hélène, Pierre, Jean, Chantal, Edouard, Alfred.

NOTA : Dans notre exemple, il existe un tableau pour les clés associé au tableau des éléments à trier. Ce tableau est facultatif et peut très bien être évité dans le cas d'un classement par ordre alphabétique. Dans ce cas, la clé pourra être la ou les premières lettres des éléments à trier. Dans notre exemple sur le tableau B, nous pourrons prendre comme clé la première lettre de chaque prénom. Les clés seront : P, J, H, C, E, A. Le résultat du tri serait dans ce cas (tri par ordre alphabétique) : Alfred, Chantal, Edouard, Hélène, Jean, Pierre.

TRI PAR VENTILATION

Il est tout à fait possible de prendre comme exemple les deux tableaux A et B utilisés dans le cas précédent, ici les valeurs du tableau A seront les salaires des six personnes, soit :

	A	B
1	6 780	PIERRE
2	8 222	JEAN
3	7 927	HELENE
4	8 424	CHANTAL
5	7 623	EDOUARD
6	5 435	ALFRED

Le tri par ventilation nous permet de classer les éléments du tableau B en fonction des salaires compris dans le tableau A. Exemple : le directeur du personnel désire connaître, sous forme de rapport, l'ordre des salaires des employés de la société. Il suffit de classer par ordre décroissant les données du tableau A, automatiquement les personnes du tableau B seront classées par ordre décroissant de leur salaire. Il est évident que, lors des manipulations, les opérations exécutées sur salaire doivent suivre les éléments du tableau B. Ainsi, le résultat de notre tri nous donnera :

	NOM	SALAIRE
1	CHANTAL	8 424 F
2	JEAN	8 222 F
3	HELENE	7 927 F
4	EDOUARD	7 623 F
5	PIERRE	6 780 F
6	ALFRED	5 435 F

TRI A BULLES

Le tri à bulles est un tri du type tri par échanges. Il permet de trier soit des chaînes de caractères dans un ordre alphabétique, soit des nombres par ordre croissant. Ces processus peuvent, bien entendu, être inversés suivant le sens de présentation désiré.
Soit le programme ci-dessous :

```
10      ! Program: TRI
20      !
30      READ N
40      DATA 10    ! NOMBRE D'ELEMENTS A TRIER
50      ALLOCATE Mot$(N)[6],Temp$[6]
60      READ Mot$(*)    ! LECTURE DU TABLEAU
70      DATA zero,un,deux,trois,quatre,cinq,six,sept,huit,neuf,dix
80      PRINT Mot$(*)
90      PRINT
100 Tri:FOR I=0 TO N-1
110           IF Mot$(I)>Mot$(I+1) THEN
120             Temp$=Mot$(I)
130             Mot$(I)=Mot$(I+1)
140             Mot$(I+1)=Temp$
150             GOTO Tri
160           END IF
170         NEXT I
180     PRINT Mot$(*)
190     END
```

Remarques sur les ordres utilisés

1) ! est équivalent à REM en BASIC standard.
2) ALLOCATE est équivalent à DIM en BASIC standard à la nuance près que l'ordre DEALLOCATE supprime la tableau créé par ALLOCATE.
3) END IF permet de délimiter la fin de l'ordre IF.. THEN. En effet, dans cet exemple l'ordre THEN est étendu de la ligne 120 à la ligne 150.
4) (*) Ce terme correspond à toutes les valeurs d'un tableau, exemple PRINT MOT$(*) à la ligne 180 revient à dire :

```
FOR I = 1 TO 10
PRINT MOT$ (I)
NEXT I
```

NOTA : Les données à trier sont rangées en DATA à la ligne 70. Elles peuvent être soit des chaînes de caractères, soit des nombres. Le programme peut être modifié sans problème afin d'entrer les données au clavier.

Le programme :

- les lignes 30 et 40 déterminent le nombre d'éléments à trier (soit 10 éléments) ;
- la ligne 50 crée deux tableaux de N cases (soit 10 dans notre exemple : N = 10), le terme $ indique qu'il s'agit de chaînes de caractères, et [6] délimite à 6 caractères la longueur de la chaîne ;
- la ligne 80 affiche les 10 éléments avant le tri ;
- la ligne 180 affiche les 10 éléments après le tri (Nota : les éléments sont triés par ordre alphabétique)
- la ligne 100 permet de créer les passages nécessaires au tri (même système que la figure 1) ;
- la ligne 110 compare la valeur (code ASCII) de l'élément du tableau MOT$ à la position I de la boucle et la suivante. Si la condition est vraie, alors il y a classement;
- lignes 120 à 150 : ces quatre lignes sont en réalité la partie du programme où se réalise le classement.

Conclusion :

Tableau avant tri : ZERO, UN, DEUX, TROIS, QUATRE, CINQ, SIX, SEPT, HUIT, NEUF, DIX
Tableau après tri : CINQ, DEUX, DIX, HUIT, NEUF, QUATRE, SEPT, SIX, TROIS, UN, ZERO.

TRI RAPIDE PAR ARBRE BINAIRE - 1

Dans l'exemple n° 2, nous décrirons le cahier des charges du suivi de la qualité dans un atelier de production.
Soit le programme de base suivant :

```
10   OPTION BASE 1
20   DIM Nom$(80)[80],Taux-défect(80)
30   INTEGER Unités-fab(80),Max,Combien
40   Max=80
50   Saisie-donnée(Nom$(*)),Unités-fab(*),Taux-défect(*),Max,Combien)
60   Stoc-donnée(Nom$(*),Unité$-fab(*),Taux-défect(*),Max,Combien)
70   Rapport(Nom$(*),Unité$-fab(*),Taux,défect(*),Max,Combien
80   END
90   SUB Saisie-donnée(Nom$(*),INTEGER Unité$(*),REAL Défectueux(*),INTEGER Max,Combien)
100  SUBEND
110  SUB Stoc-donnée(Nom$(*),INTEGER Unité$(*),REAL Défectueux(*),INTEGER Max,Combien)
120  SUBEND
130  SUB Rapport(Nom$(*),INTEGER Unité$(*),REAL Défectueux(*), INTEGER Max,Combien)
140  SUBEND
```

Il n'y a pas beaucoup de problèmes à la lecture d'un tel programme.
Notons qu'il est composé d'un programme principal de la ligne 10 à la ligne 80, ce programme est suivi par trois sous-programmes qui ne sont pas développés dans cette première fiche.

Remarque

1) OPTION BASE 1 à la ligne 10 signifie que les tableaux sont réalisés à partir du n° 1 et non 0 (dans ce dernier cas, il faudrait écrire OPTION BASE 0) ;
2) ligne 20 : création des tableaux NOM$ en tableau de 80 cases de 80 caractères et Taux-defect en 80 cases (nombre réel) ;
3) ligne 30 : création du tableau Unités-Fab en 80 cases (nombre entier), ainsi que des variables Max., et Combien (nombres entiers) ;
4) lignes 50 à 70 : ces lignes permettent de créer une base commune des données pour le programme principal et les trois sous-programmes ;
5) ligne 80 : fin du programme principal ;
6) lignes 90-100 : ligne 90, début de sous-programme saisie de données non développé sur ce listing ; ligne 100, fin du sous-programme. Nota : cette organisation se retrouve pour les deux sous-programmes suivants (Stoc-donnée et Rapport).

Chaque sous-programme commence par l'ordre SUB suivi du nom du sous-programme. On notera que la partie droite de la première ligne de chaque sous-programme est constituée du passage des données. En effet, dans l'absolu, les traitements effectués dans un sous-programme sont ignorés du reste du programme. Il est donc nécessaire de créer un passage des données nécessaires à plusieurs sous-programmes et au programme principal. C'est la raison d'être de ce système qui autorise ainsi le passage des informations d'une entité à l'autre. Il est ainsi possible de bien structurer son programme et d'appliquer les règles étudiées dans les cours précédents.

TRI RAPIDE PAR ARBRE BINAIRE - 2

Après avoir étudié le programme principal, voyons plus en détail les sous-programmes suivants :

```
 90   SUB Saisie-donnée(Nom$(*),INTEGER Unités(*),REAL Défectueux(*),INTEGER Max,Combien
 91   DIM Lequel$[8]
 92   INPUT "Nouvelle ou ancienne donnée?",Lequel$
 93   IF Lequel$="Nouvelle" THEN
 94   Saisie_ nouv.(Nom$(*),Unités(*),Défectueux(*),Max,Combien)
 95   ELSE
 96      Edit-anc(Nom$(*),Unités(*),Défectueux(*),Max,Combien)
 97   END IF
100   SUBEND
101   !
110   SUB Stoc-donnée(Nom$(*),INTEGER Unités(*),REAL Défectueux(*),INTEGER Max,Combien)
111   Const-fichier( Fichier)
112   OUTPUT  Fichier;Nom$(*),Unités(*),Défectueux(*)
113   ASSIGN  Fichier TO
120   SUBEND
121   !
130   SUB Rapport(Nom$(*),INTEGER Unités(*),REAL Défectueux(*),INTEGER Max,Combien)
132   OPTION BASE 1
133   INTEGER Racine,I,Quellezone
134   ALLOCATE INTEGER Arbre(Combien,2)
135   Init-arbre(Racine,arbre(*))
136 Question: INPUT "Quelle zone (1=Nom,2=Unités,3=Défectueux)?",Quelle zone
137   IF Quellezone<1 OR Quellezone>3 THEN Question
138   FOR I=2 TO Combien
139      SELECT Quelle-zone
140      CASE 1
141         Chaîne(Racine,Arbre(*),I,Nom$(*))
142      CASE 2
143      .  Construction(Racine,Arbre(*),I,Unités(*))
144-     CASE 3
145         Construction(Racine,Arbre(*),I,Défectueux(*))
146      END SELECT
148   NEXT I
149   Inorder(Racine,Arbre(*),Nom$(*),Unités(*),Défectueux(*))
150   SUBEND
```

A la lecture du listing, trois remarques s'imposent :

– à la ligne 94, la saisie de la nouvelle information n'est pas développée, elle pouvait se faire de la manière suivante :

```
INPUT Nom$ (I)
INPUT Unités (I)
INPUT Défectueux (I) ;
```

– à la ligne 96, on rencontre le même problème pour l'édition d'une ancienne donnée (ex. Print Nom$(I)) ;

– la ligne 111 suppose la création d'un fichier sur support de mémoire de masse.

Le sous-programme «Rapport» comprend l'initialisation, la construction et le cheminement à travers l'arbre binaire. Le sous-programme Init-arbre qui est appelé à la ligne 135 sert à initialiser les nœuds racines (les premiers éléments) de chaque sous-arbre afin qu'il soit vide. Le sous-programme chaîne appelé à la ligne 140 sert uniquement à entrer la I^e variable dans le tableau Nom$(*) qui se trouve dans la structure d'Arbre(*), si l'utilisateur désire que les données soient triées à l'aide de Nom$(*). De même, si Unités(*) ou Défectueux(*) servaient pour le tri, le sous-programme Construction (appelé aux lignes 143 et 145) aurait été utilisé pour construire l'arbre.

Nota : Les lignes 136 à 146 permettent de déterminer sur quelle zone se fera le tri :

- soit sur les noms,
- soit sur les quantités fabriquées,
- soit sur le taux de rejet.

L'ordre SELECT CASE est issu du langage PL/1.

TRI RAPIDE PAR ARBRE BINAIRE - 3

Soit les trois sous-programmes suivants :

- Init-arbre
- Chaîne
- Inorder.

```
200  SUB Init_arbre(INTEGER Racine,Arbre(*))
210  COM /Arbre/ INTEGER Non,Gauche,Droite
220  Non=0
230  Gauche=1
240  Droite=2
250  Racine=1
260  Arbre(Racine,Gauche)=Non
270  Arbre(Racine,Droite)=Non
280  SUBEND
281  !
290  SUB Chaîne(INTEGER Racine,Arbre(*),Indice,As(*))
300  COM /Arbre/ INTEGER Non,Gauche,Droite
310  IF As(Indice)<=As(Racine) THEN      ! Recherche du sous-arbre de gauche
320     IF Arbre(Racine,Gauche)=Non THEN ! Quand la branche est trouvee
330        Arbre(Racine,Gauche)=Indice   ! le programme pointe le nœud suivant
340        Arbre(Indice,Gauche)=Non      ! avec le pointeur gauche de la branche
350        Arbre(Indice,Droite)=Non      ! puis considere le nœud suivant
351                                      ! comme une branche.
360     ELSE
370        Chaîne(A(Racine,Gauche),Arbre(*),Indice,As(*))
380     END IF

390  ELSE                                ! Recherche du sous-arbre de droite
400     If Arbre(Racine,Droite)=Non THEN !
410        Arbre(Racine,Droite)=Indice
420        Arbre(Indice,Gauche)=Non
430        Arbre(Indice,Droite)=Non
440     ELSE
450        Chaîne(A(Racine,Droite),Arbre(*),Indice,As(*))
460     END IF
470  END IF
480  SUBEND
481  !
490  SUB Inorder(INTEGER Racine,Arbre(*),Noms(*),INTEGER Unites(*),REAL Defectueux(*))
500  COM /Arbre/ INTEGER Non,Gauche,Droite
510  IF Racine<>Non THEN
520     Inorder(A(Arbre,Droite),Arbre(*),Noms(*),Unites(*),Defectueux*
530     PRINT Noms(Raine),Unites(Racine),Defectueux(Racine)
540     Inorder(A(Racine,Droite),Arbre(*),Noms(*),Unites(*),Defectueux(*))
550  END IF
560  SUBEND
```

Le sous-programme Inorder traverse la structure quand l'arbre est construit. «Inorder» signifie que chaque nœud est imprimé entre chacun des sous-arbres associés. La construction de ces trois sous-programmes est assez proche.

NOTA : Les lignes 210, 300 et 500 commencent par l'ordre COM.
Un bloc COM est un bloc dont les données sont communes à plusieurs sous-programmes. Dans cet exemple, le nom de ce bloc commun est Arbre.
Un bloc COM est accompagné de variables qu'il représente, ces dernières sont, bien entendu, nommées et déclarées (en entier dans notre exemple).

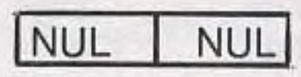
TRI RAPIDE PAR ARBRE BINAIRE

Soit dix noms compris dans notre fichier d'employés :

- 1 GANDHER
- 2 BLANCHARD
- 3 ULIVI
- 4 DELASNERIE
- 5 LEBON
- 6 GUILLOT
- 7 HARANG
- 8 GRIPERAY
- 9 RIOU
- 10 VALERA

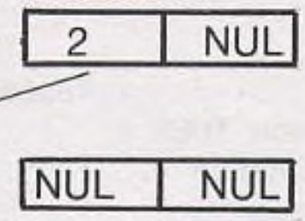
Constitution de l'arbre par l'utilisation du sous-programme chaîne : exemple d'un tri.

1) Après l'exécution du programme Init-arbre

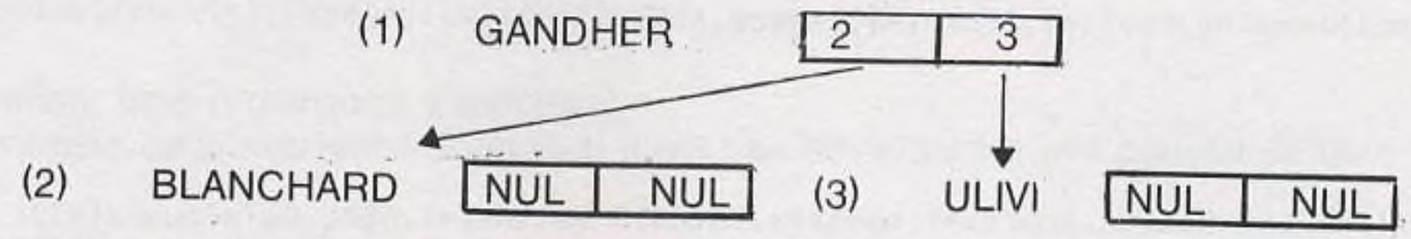


2) Après insertion dans l'arbre à l'aide du sous-programme chaîne :

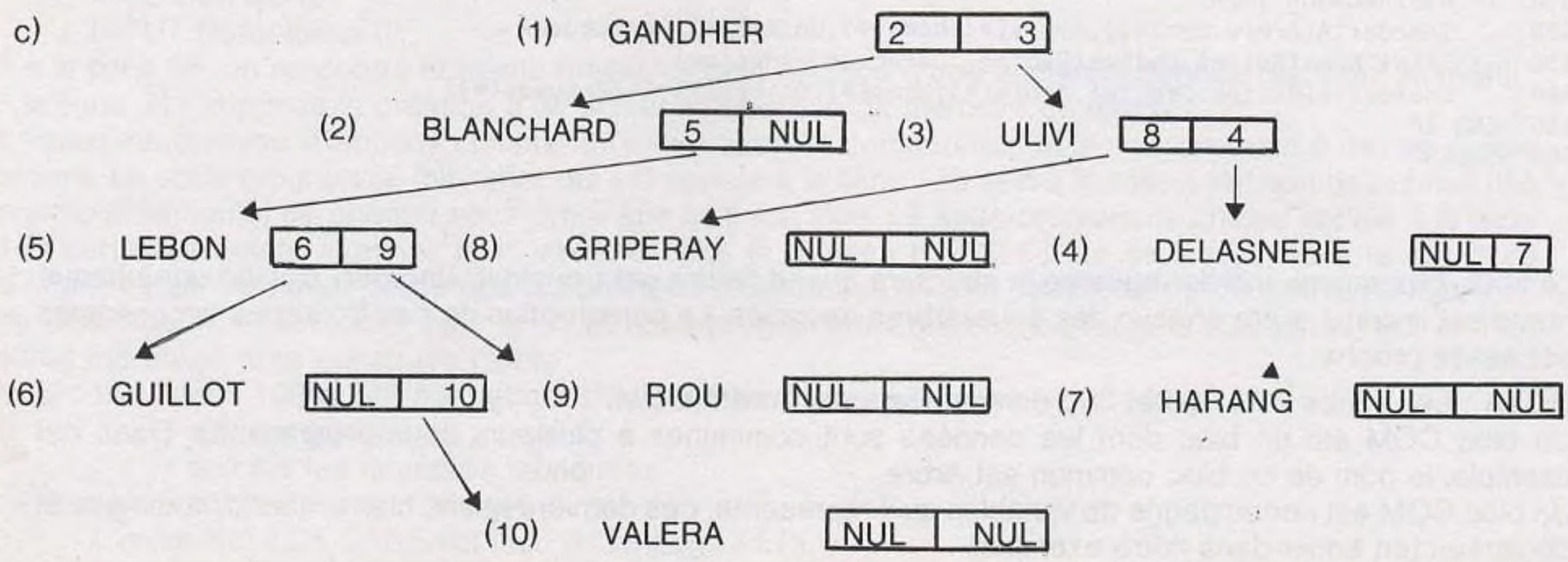
a)



b)



c)



LES DONNÉES LOGIQUES ET LEURS OPÉRATEURS

Dans les algorithmes de tri, les données logiques et leurs opérateurs ont une signification très importante. Ils permettent le bon déroulement de l'algorithme.

DONNÉES LOGIQUES

L'ensemble des données logiques est l'ensemble [faux, vrai], ou ensemble des valeurs logiques. Il convient de préciser que les données logiques sont des données qui sont internes au programme, à la différence des valeurs numériques et autres chaînes de caractères.

LES CONSTANTES LOGIQUES

Il existe deux valeurs logiques :

- **VRAI**
- **FAUX.**

LES VARIABLES LOGIQUES

Il s'agit de variables qui, au cours du déroulement d'un programme peuvent prendre les valeurs VRAI ou FAUX.

LES OPÉRATEURS DE COMPARAISON

Les opérations de comparaison sont effectuées sur deux éléments qui appartiennent obligatoirement à un même ensemble (données numériques, chaînes de caractères, etc.)

Soit :

Signification	*Notation générale*
plus grand que	>
supérieur ou égal à (⩾)	⩾ =
égal à	=
différent de (#)	< >
inférieur ou égal à (⩽)	< =
inférieur à	<

LES OPÉRATIONS LOGIQUES

On appelle opération logique, les opérations internes à l'ensemble logique.
Soit :

Signification	*Notation générale*
NON	**NOT**
ET	**AND**
OU (inclusif)	**OR**

TABLES DE VERITE (OPERATEURS LOGIQUES)

1) Soit deux propositions A et B, la table de vérité correspondant aux opérateurs logiques est :

A	B	NOT B	A AND B	A OR B
F	F	V	F	F
F	V	F	F	V
V	F	V	F	V
V	V	F	F	V

NOTA : F = FAUX, V = VRAI.

2) Les opérateurs logiques peuvent être étendus grâce aux comparateurs. Ceci autorise des champs d'application plus larges.
Soit nos deux propositions A et B, les tables de vérité correspondantes sont :

A	B	A < B
F	F	F
F	V	V
V	F	F
V	V	F

A	B	A < = B
F	F	V
F	V	V
V	F	F
V	V	V

A	B	A = B
F	F	V
F	V	F
V	F	F
V	V	V

A	B	A < > B
F	F	F
F	V	V
V	F	V
V	V	F

(dans le cas du OU exclusif)

A	B	A > = B
F	F	V
F	V	F
V	F	V
V	V	V

A	B	A > B
F	F	F
F	V	F
V	F	V
V	V	F

COMPARAISON DES MÉTHODES DE TRI

Nous avons vu qu'il existait plusieurs sortes de tri. En fait, chaque tri a ses avantages et ses inconvénients. Il est évident que le tri le plus simple à réaliser est le tri à bulles. Toutefois, son choix ne devra être effectué que sur des opérations limitées. Plusieurs facteurs entrent en ligne de compte. Nous avons remarqué, dans les critères de qualité d'un tri, que plusieurs observations devraient être réalisées devant le choix final.

LE TRI A BULLES

Il sera utilisé dans les petites applications manipulant un nombre d'éléments restreints.

LE TRI RAPIDE

Il s'agit d'une méthode qui s'applique à la grande majorité des cas. Il est fiable et s'adapte facilement. Il est certainement le meilleur algorithme général.

LE TRI PAR ARBRE

Plus long que le tri rapide simple, le tri par arbre a l'avantage d'être de qualité constante.

NOTA : Il est possible grâce à une analyse très fine de modifier de façon notable la vitesse d'exécution d'un tri. La manière dont aura été conçu l'algorithme interviendra d'une façon très importante dans le temps de traitement.

Voici un tableau récapitulatif des temps de traitement de différents algorithmes de tri.
Soit n le nombre d'éléments à trier :

n	10	100	1 000	10 000
Tri à bulles	1	150	20 000	
Tri par extraction	0,8	50	5 000	
Tri par insertion	0,8	45	4 500	
Tri arbre	0,2	20	400	8 000
Tri rapide	0,2	15	200	4 000

Ces données sont indiquées par rapport à une base de temps de 1. Nous aurons donc sur une machine effectuant le tri de 10 éléments en une unité de temps suivant l'algorithme de tri à bulles, une comparaison facile à réaliser sur les vitesses de traitement en fonction du nombre d'éléments du type de tri.
L'unité de temps peut être très variable suivant le type de machine utilisée. De plus, le langage choisi accompagné de son compilateur influera sur la vitesse de traitement.

Remarques :

A) Les tris à bulles, par extraction et par insertion, sont limités à un nombre restreints d'éléments à trier.
soit 1 000 éléments en programme compilé
100 éléments en programme interprêté.

B) Les tri par arbre et rapides sont de loin les plus rapides, toutefois il convient de noter que le nombre d'éléments à trier doit toujours rester raisonnable : attention à la saturation et aux temps prohibitifs.

PETITES ANNONCES

Sté vd Advance 86 compatible 3e niveau, 256 K, MS/DOS, 2 drives × 360 K + moniteur Zenith monochrome + imprimante matricielle 132 C, 160 C/S + deux logiciels gestion. fichiers : 20 000 F. Tél. 16 (1) 39.58.84.84.

Vds Atari 800 XL + lecteur de disk 1050 + nombreux logiciels sur K7, cartouches et disquettes (jeux et éducatifs) + 4 livres pour 4 000 F. Espinasse 196, rue Beauregard 73000 Chambéry. Tél. : 79.75.04.43.

Vends Micro-Professor MPF1-B avec Basic, RAM 4 k, PIO, 1 100 F. Collection complète Micro-Système : 700 F + port. Nefussy 145 G, ch. de Choulans 69005 Lyon.

Vds CBM 64 Secam + lect. K7 + Vic 1541 (disq.) + manettes + livres + jeux : 4 500 F. M. Moren H091 Cheverny 77100 Meaux.

Vends TRS mod. III 48 ko + 2 drives + progs utilitaires (compta., stock, etc.) + jeux : 8 000 FF + en prime imprimante Logabax 132 colonnes, 120 cps à réparer. Vends TRS mod. 1 16 ko avec clavier minuscules accentuées + progs : 2 000 FF. Durr Michel 18, rue Laperouse 31120 Portet/Garonne. Tél. : 16-61.72.23.18.

Vds T-Adler Xtronic-PC, Z80 A, 64 K + drive 320 K, CP/M, Pascal/M, macro 80 + docs et livres tech. 5 800 F à déb., assembleur 8080/8085 (ER) 50 F, Z80 (Eyro) 20 F, CP/M (ER) 20 F Fis J-L 38.76.94.20.

Vds Apple IIC + moniteur IIC + souris + logiciels. Matériel sous garantie. Tél. : 45.92.86.46 le soir.

A vendre Hector HRX 64 k (Forth + Basic) + disc 2 2 × 800 k (juin 85) + moniteur couleur + imprim. Seikosha GP 100 A + 50 K7 jeux et prog. (Pyrentexte, etc.) + disquettes prog. (Visicalc, etc.) + 15 diskettes + nombreuses doc. Prix : 12 000 F. Tél. : 56.21.86.98 ap. 20 h.

Vds ZX81 + 16 K + alim. + manuel + livres 70 progs + cass. stock-car + cass. vierge 900 F. Bannier Patrice 3, rue des Ecoles 28330 Authon-du-Perche.

Vends doubleur de densité pour TRS ou Prof 80 avec 1791 neuf et testé : 500 F. Tél. 40.95.08.24 (province).

Vends micro-ordinateur Tandy TRS 80, color 2 Basic étendu 64 k + K7 Tandy CCR 82 + fichier color file. L'ensemble 2 000 F. Tél. : (1) 60.16.72.98.

BON DE COMMANDE

Pour compléter votre collection de Led-Micro

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

Je désire le n° ☐ (cocher le ou les nos désirés)

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24

au prix de 18 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 .....

Bulletin d'Abonnement

Je désire m'abonner à Led Micro (10 numéros). France : 160 F - Etranger : 240 F, à partir du n° .....

Nom ..... Prénom .....

N° ..... Rue .....

Ville ..... Code Postal .....

Envoyez ce bon accompagné du règlement à l'ordre des Editions Fréquences à :
EDITIONS FREQUENCES 1, boulevard Ney, 75018 PARIS
MODE DE PAIEMENT : CCP ☐ - Chèque bancaire ☐ - Mandat ☐

TOUT SUR LES PÉRIPHÉRIQUES

NOUVEAU

dans la COLLECTION «ETUDES» aux éditions fréquences

- 85 schémas
- 20 tableaux
- 136 pages

Prix : 150 F

Les périphériques font partie intégrante d'un système informatique. En parallèle de l'unité centrale, qui gère et synchronise l'ensemble, ils sont reponsables de différentes fonctions comme :

- la mémoire de masse : unités de disques souples et de disques durs, lecteur de cassettes ;
- le dialogue avec l'utilisateur : clavier, écran vidéo, imprimante ;
- les télécommunications : modem.

Tous ces périphériques sont décrits dans cet ouvrage avec, pour chacun d'eux, une partie technologie (principe de fonctionnement, caractéristiques techniques) et une partie interface (coupleurs d'entrées-sorties, connecteurs de liaison).

Dans chaque grande catégorie (mémoire, imprimante), une analyse comparative des différents produits existants est effectuée.

Philippe Faugeras, docteur ingénieur en électronique, est responsable matériel dans une entreprise d'informatique traitant des réseaux de P.C. Au préalable, il a acquis son expérience en travaillant sur des sujets comme les automatismes et les télécommunications dans deux grandes sociétés françaises (Bull, CGE). Philippe Faugeras est l'auteur d'un premier ouvrage «L'électronique des micro-ordinateurs» paru aux Editions Fréquences.

En vente chez votre libraire et aux Editions Fréquences.

BON DE COMMANDE

Je désire recevoir l'ouvrage **«Périphériques interfaces et technologie»** au prix de **160 F** (150 F + 10 F de port).

Nom .....

Adresse .....

.....

A adresser aux EDITIONS FRÉQUENCES 1 boulevard Ney, 75018 Paris

Règlement ci-joint :

Par chèque bancaire ☐ par chèque postal ☐ par mandat ☐

Alice.

Le Victor PC ne coûte que 24.900 F n'en déplaise à [redacted].

Le Victor PC 15 ne coûte que 24.900 F*.

Certains d'entre vous penseront peut-être – *et nous en connaissons qui aimeraient bien que ce soit vrai* – qu'à 24.900 F*, il ne peut s'agir que d'un PC "bradé". Une telle réaction est d'ailleurs compréhensible quand on songe aux prix pratiqués sur le marché, en matière de PC. Prenons par exemple [redacted]. Son PC coûte 50 % plus cher que le Victor PC 15.

Et pourtant, les performances du Victor PC 15 sont équivalentes, *voire supérieures*, à celles de l'[redacted] PC. La preuve, la voici :

Alors que la plupart des micro-ordinateurs propose une capacité de stockage de 10 Mo, le Victor PC 15, lui, offre une capacité de 15 Mo ! De plus, l'utilisateur du Victor PC 15 bénéficie, grâce à un moniteur de 14 pouces, de 30 % de surface écran supplémentaires (la quasi-totalité du matériel concurrent étant équipée d'un moniteur 12 pouces).

Et ce n'est pas tout ! Le Victor VU – l'interface utilisateur – permet un gain de temps appréciable en guidant dans son travail l'usager, par de simples messages organisés comme des menus. Finie, désormais, la consultation fastidieuse et peu pratique du manuel du système d'exploitation !

Et l'on pourrait parler des 5 emplacements d'extensions disponibles pour accroître les possibilités du PC...

Non décidément, [redacted] devra se faire une raison et s'accommoder de la présence sur le marché du Victor PC 15 ! Un PC compatible avec les standards du marché, aussi performant que celui que fabrique [redacted] et à un prix bien plus séduisant que celui affiché par [redacted].

Car au risque de le répéter et de déplaire à [redacted], ces 50 % sont difficilement justifiables. D'ailleurs les vendeurs d'[redacted] doivent déjà en savoir quelque chose...

Lesquels vendeurs d'[redacted] ne vont sans doute guère apprécier que nous vous donnions nos coordonnées - et que vous puissiez nous contacter à Victor Technologies - Tour Horizon, 52, quai de Dion-Bouton, 92800 Puteaux (tél. : 778.14.50) ; ou encore à Lyon : (7) 234.12.45 ; Montpellier : (67) 64.71.72 ; Nantes : (40) 89.24.28. Mais l'on ne peut contenter tout le monde et [redacted] !

** Configuration complète avec clavier et écran monochrome. Prix H.T. au 1/9/85. (Possibilité de location financière : 700 F par mois sur 48 mois - CEGEDATA.).*

VICTOR®

Comme [redacted] moins cher qu'[redacted]