



**DANS  
CE LIVRE :  
LA DISQUETTE**

# **TRUCS ET ASTUCES**

**360 KO  
DE PROGRAMMES  
SUR LA DISQUETTE**

**EDITIONS MICRO APPLICATION**



**LIVRE DATA BECKER**

ATAK! ST

# BIEN DEBUTER

D É C O U V R I R ,  
C O M P R E N D R E E T  
M A Î T R I S E R .

EDITIONS MICRO APPLICATION



LIVRE DATA BECKER

Distribué par : MICRO APPLICATION  
13, Rue Sainte Cécile  
75009 PARIS

et

EDITIONS RADIO  
189, Rue Saint Jacques  
75005 PARIS

(c) Reproduction interdite sans l'autorisation de  
MICRO APPLICATION

'Toute représentation ou reproduction, intégrale ou partielle, faite sans le consentement de MICRO APPLICATION est illicite (Loi du 11 Mars 1957, article 40, 1er alinéa).

Cette représentation ou reproduction illicite, par quelque procédé que ce soit, constituerait une contrefaçon sanctionnée par les articles 425 et suivants du Code Pénal.

La Loi du 11 Mars 1957 n'autorise, aux termes des alinéas 2 et 3 de l'article 41, que les copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à l'utilisation collective d'une part, et d'autre part, que les analyses et les courtes citations dans un but d'exemple et d'illustration'.

ISBN : 2-86899-051-7

Code ER : 255

(c) 1988

DATA BECKER  
Merowingerstrasse, 30  
4000 DUSSELDORF

Traduction Française assurée par Pascal Hausman

(c) 1988

MICRO APPLICATION  
13 Rue Sainte Cécile  
75009 PARIS

Collection dirigée par Mr Philippe OLIVIER  
Edition réalisée par Frédérique BEAUDONNET

## **NOTE AUX LECTEURS**

La Société ATARI a commercialisé en France, depuis septembre 1985, trois modèles différents d'ATARI ST.

Il s'agit du 520 ST, du 520 STF, du 1040 STF et du Mega ST.

En conséquence, nous avons réalisé cet ouvrage pour qu'il réponde à toutes vos questions concernant ces 3 modèles de la famille ST.



# TABLE DES MATIERES

<u>INTRODUCTION</u>	<u>1</u>
<u>1. LES PREMIERS PAS</u>	<u>3</u>
1.1. L'unité centrale	3
1.2. Les systèmes de visualisation	6
1.2.1. Le moniteur	6
1.2.2. Le téléviseur	6
1.3. Le lecteur de disquette	6
1.4. Le clavier	7
1.5. La souris	7
1.6. La mise en marche	8
<u>2. LE TRAVAIL SUR L'ATARI ST</u>	<u>11</u>
2.1. Chargement du système d'exploitation TOS et GEM	11
2.2. Le bureau électronique : GEM DESKTOP	15
2.2.1. Déjà rangé ?	16
2.2.2. Catalogue de la disquette	18
2.2.2.1. Déplacement	20
2.2.2.2. Remplissage	21
2.2.2.3. Agrandissement et rétrécissement	22
2.2.2.4. Faire défiler avec les flèches	23
2.2.2.5. Ça suffit !	25
2.2.3. Les menus déroulants	26
2.2.3.1. Desk	26

2.3. Nous réalisons des disquettes neuves	30
2.3.1. Formatage	30
2.3.2. Copie de disquettes	33
2.3.2.1. Fichier (File)	41
2.3.2.2. Visualisation (View)	47
2.3.2.3. Options	49
2.4. Copie et suppression de programmes et fichiers	55
2.5. Le ST est le roi des fenêtres !	58
2.5.1. De bas en haut	60
2.5.2. Ça marche malgré tout : cinq fenêtres	61
2.5.3. Copie d'une fenêtre dans l'autre	62
2.6. Les icônes	64
2.6.1. Le classeur	64
2.6.2. Le bloc avec souche	65
2.6.3. Pile de papiers avec coin recourbé	65
2.7. Le clavier	69
2.7.1. La machine à écrire	70
2.7.1.1. Alternate	71
2.7.1.2. Control	71
2.7.1.3. Shift	71
2.7.1.4. Caps Lock	71
2.7.1.5. Tab	72
2.7.1.6. Esc	72
2.7.1.7. Backspace	73
2.7.1.8. Delete	73
2.7.1.9. Return	74
2.7.2. Les touches de fonction	76
2.7.3. Les touches d'édition	77
2.7.3.1. Les touches-flèches	77
2.7.3.2. Les touches-flèches avec Shift ou Alternate	77
2.7.3.3. Insert	77
2.7.3.4. Ctr/Home	78
2.7.3.5. Help	78
2.7.3.6. Undo	78
2.7.3.7. Alternate & Help	78

2.7.4.	Le bloc numérique	79
2.7.4.1.	Enter	79
2.7.4.2.	Les chiffres, les opérateurs arithmétiques et le point décimal	79

### **3. LE BASIC DE L'ATARI ST** **81**

3.1.	Comment le ST apprend-il le BASIC ?	81
3.1.1.	Qu'est-ce donc que le BASIC ?	81
3.1.2.	Chargement du BASIC	82
3.2.	Les fenêtres en BASIC	82
3.2.1.	Fonction des fenêtres en BASIC	82
3.2.2.	Réorganisation des fenêtres en BASIC ST	84
3.3.	Les premiers contacts avec le BASIC	85
3.3.1.	Le mode direct	85
3.3.2.	Le mode programme	87
3.4.	Le vocabulaire de base du BASIC	88
3.4.1.	Entrée et sortie : INPUT et PRINT	88
3.4.1.1.	Les numéros de ligne	88
3.4.1.2.	L'instruction PRINT	88
3.4.1.3.	L'instruction INPUT	89
3.4.2.	Saut inconditionnel : GOTO	91
3.4.3.	Si..alors..sinon.. : IF..THEN..ELSE	93
3.4.4.	Les boucles : FOR...NEXT	95
3.4.5.	Les sous-programmes : GOSUB...RETURN	97
3.4.6.	Encore plus de clarté : REM	98
3.4.7.	Vider l'écran	100
3.4.7.1.	Nettoyer les fenêtres : CLEARW	100
3.4.7.2.	Comment s'il ne s'était rien passé : NEW	101
3.5.	Quelques mots sur les variables	101
3.5.1.	Variable numérique simple	101
3.5.2.	Variables numériques entières ou décimales	104
3.5.3.	Variable de texte ou variable alphanumérique	109
3.5.4.	Variable dimensionnée	114

3.6.	Instructions graphiques intéressantes	116
3.6.1.	CIRCLE A,B,C,D,E	116
3.6.2.	ELLIPSE A,B,C,D,E,F	117
3.6.3.	OPENW A	117
3.6.4.	CLOSEW A	117
3.6.5.	FULLW A	117
3.6.6.	GOTOXY A,B	118
3.6.7.	LINEF A,B,C,D	118
3.6.8.	PCIRCLE A,B,C,D,E	118
3.6.9.	PELLIPSE A,B,C,D,E,F	119
3.7.	Aides précieuses	120
3.7.1.	Aides à la programmation	120
3.7.2.	Le tueur de punaises	121
3.7.3.	EDIT aide à débbuger	123
3.8.	Le travail avec le lecteur de disquette	126
3.8.1.	LOAD et SAVE	127
3.8.2.	DELETE FILE, MERGE et QUIT	128
3.9.	Les premiers programmes	128
3.9.1.	Agenda téléphonique	130
3.9.2.	Programme de vocabulaire	130
3.9.3.	Le problème de l'échiquier	134
3.9.4.	Les chiffres du loto	143
3.9.5.	Programme de conversion	147
3.10	Bref aperçu du vocabulaire de base du BASIC	149
3.11	Perspectives	150

#### **4. LE LOGO DE L'ATARI ST** **153**

4.1.	Comparaison entre le LOGO et le BASIC ST	153
4.2.	Chargement du LOGO	154
4.3.	Le vocabulaire de base du LOGO	156
4.3.1.	PRINT	156
4.3.2.	MAKE	156

4.3.3.	CS et CT	160
4.3.4.	FORWARD, BACK, RIGHT et LEFT	161
4.3.5.	REPEAT	164
4.3.6.	TO	166
4.3.7.	PENUP et PENDOWN	168
4.4.	Autres instructions LOGO	170
4.4.1.	BOX	171
4.4.2.	CIRCLE	171
4.4.3.	ELLIPSE	172
4.4.4.	HIDETURTLE & SHOWTURTLE	173
4.4.5.	FILLATR & SETFILL	173
4.4.6.	SHUFFLE & SORT	175
4.4.7.	MOUSE, NODES et TURTLEFACTS	175
4.5.	Bref récapitulatif du vocabulaire de base du LOGO	175
4.6.	Le détecteur d'erreurs de DR Logo	176

## 5. DANS UN MOMENT DE TRANQUILLITE 179

5.1.	Le jargon technique de l'informatique	179
5.1.1.	Comme le temps est passé vite	179
5.1.2.	Différents chips	183
5.1.3.	Au fond, les ordinateurs ne sont pas intelligents	187
5.1.4.	Que veut dire '512 kilo-octets'	192
5.1.5.	La fonction du microprocesseur	195
5.2.	Perspectives d'avenir	197
5.2.1.	Connexions à gogo	197
5.2.1.1.	Deux ports joystick	198
5.2.1.2.	Connexion pour les lecteurs de disquette	203
5.2.1.3.	La connexion Centronics	204
5.2.1.4.	La connexion cartouche	207
5.2.1.5.	La connexion disque dur	208
5.2.1.6.	Les prises MIDI	214
5.2.1.7.	L'interface RS 232	215

5.2.1.8. La prise moniteur	216
5.2.1.9. La prise pour l'alimentation	216
5.2.2. Aspect logiciel	216
5.3. L'importance de l'Atari ST	217

<b><u>ANNEXE</u></b>	<b><u>221</u></b>
----------------------	-------------------

A. Le jeu de caractères de l'ATARI ST	221
B. Programmes de conversion	224
C. Petit lexique de l'informatique	227
D. Messages d'erreur du BASIC ST	238
E. Index	242

## **Chère et cher(e) collègue**

Vous ne voyez tout de même pas d'inconvénient à ce que nous vous appelions ainsi, n'est-ce pas ? Puisque vous êtes à présent sur le meilleur chemin pour devenir un(e) expert(e) en informatique. En douteriez-vous ?

Vous venez de faire un grand pas dans la bonne direction puisque vous êtes en train de lire un livre destiné à vous familiariser avec l'un des micro-ordinateurs les plus puissants du marché. Vous verrez, d'ici peu de temps vous manipulerez votre ATARI ST comme si n'aviez fait que cela pendant toute votre vie.

Cet ouvrage est divisé en cinq chapitres. Les deux premiers permettent d'établir un premier contact avec notre ATARI ST, le troisième aborde le langage de programmation BASIC. Si votre soif d'information n'est pas encore tarie à ce stade de la lecture, le quatrième chapitre apporte un certain nombre d'informations intéressantes ayant pour thème l'ATARI ST.

Après avoir étudié cet ouvrage vous connaîtrez bien votre ATARI ST, et vous serez en mesure de le programmer vous-même en BASIC. N'attendez pas de ce livre une description détaillée et technique de l'ATARI ST, sa vocation est bien plus de vous familiariser le plus agréablement possible avec cette superbe machine qu'est l'ATARI ST. Nous espérons qu'il éveillera en vous l'envie de vous informer plus en détail sur tel ou tel sujet abordé.



## 2 *Bien débiter avec votre ATARI ST*

---

## **1. LES PREMIERS PAS :**

### **INSTALLER ET BRANCHER L'ORDINATEUR**

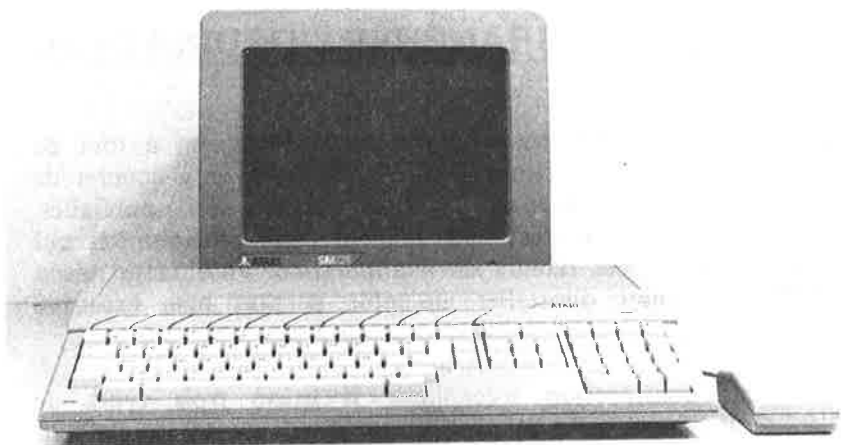
Lorsque le nouvel ordinateur arrive dans le foyer, on a tout de suite envie de le voir fonctionner et d'avoir un aperçu concret de ses extraordinaires capacités vantées dans les publicités. L'expérience montre que c'est bien souvent cette précipitation qui entraîne de regrettables erreurs de manipulation. Pour cette raison nous vous proposons d'installer ensemble et sans hâte excessive notre ATARI ST.

De par leur construction mécanique, il existe trois types de modèles de l'ATARI ST. Le premier type, représenté par le 520 ST/M, mais aussi par les plus anciens 260 ST et 520 ST+, est constitué de l'unité centrale avec clavier intégré, à laquelle on ajoute un boîtier d'alimentation, un ou deux lecteurs de disquettes, la souris et un moniteur (en option sur les modèles ST/M).

La deuxième forme sous laquelle on peut rencontrer l'ATARI ST est le 1040 STF. Dans ce modèle l'alimentation et un lecteur de disquettes ont pu être intégrés dans l'unité centrale. Il ne reste plus alors qu'à brancher un moniteur, et bien sûr la souris, pour pouvoir commencer à travailler.

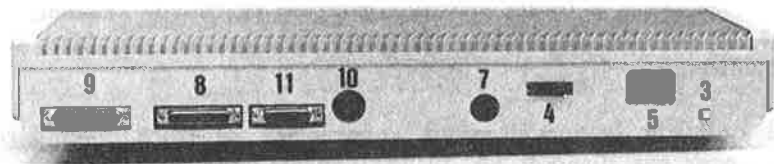
Dernier produit dans la gamme des ATARI ST, le MEGA ST se présente sous une forme tout à fait nouvelle, puisque le clavier a été détaché de l'unité centrale, à la manière des ordinateurs personnels (PC). L'unité centrale renferme également l'alimentation et le lecteur de disquettes, en outre la capacité mémoire a été augmentée.

Quel que soit le modèle dont vous venez de faire l'acquisition, commencez par disposer les divers éléments comme cela est montré sur la photographie.



A présent nous allons relier entre eux les divers éléments. Comme les interconnexions à faire diffèrent légèrement entre les trois types de modèles (ST/M, 1040 STF et MEGA ST), nous préciserons à chaque fois quels sont les modèles concernés. Attention, il est important que dans tous les cas le branchement de la prise secteur se fasse en tout dernier, et après vérification des étapes précédentes. Nous allons commencer par le type 520 ST/M, et relier entre eux l'unité centrale, le lecteur de disquettes et le moniteur (ou le téléviseur).

### 1.1. L'UNITE CENTRALE

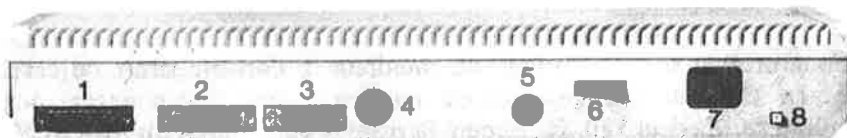


Examinons la face arrière de l'ATARI. Celle-ci présente un certain nombre de connecteurs que nous avons numérotés pour pouvoir nous y retrouver plus facilement.

Les propriétaires d'une machine de type ST/M auront certainement déjà remarqué la présence d'une prise supplémentaire, située entre les connecteurs 6B et 7. Ces modèles sont en effet équipés d'un modulateur (d'où le "M") UHF qui permet de brancher le ST sur l'entrée antenne d'un téléviseur. Veuillez lire au chapitre 1.2.2 comment procéder avec le téléviseur.

Nous allons commencer par relier l'ordinateur et son alimentation. Pour cela insérez le connecteur circulaire de l'alimentation dans la prise 5 de l'unité centrale. Veillez à la bonne orientation du connecteur, il faut que la petite encoche pointe vers le haut.

La face arrière du 1040 STF se présente de manière plus simple :



Le câble secteur servant à alimenter l'ordinateur est inséré dans la prise 7, et c'est presque la seule chose qu'il faille faire ; il en est de même pour le MEGA ST.



## **1.2. LES SYSTEMES DE VISUALISATION**

Il existe deux manières de visualiser les informations video que fournit un ordinateur : la première, la plus économique, consiste à utiliser un téléviseur (couleur ou noir & blanc). L'autre solution consiste à utiliser un écran dédié, appelé moniteur. Le moniteur spécifique à l'ATARI ST, le SM 124, est vraiment d'excellente qualité.

Pour des applications de types professionnels ou semi-professionnels il est préférable d'utiliser un moniteur.

### **1.2.1. Le moniteur**

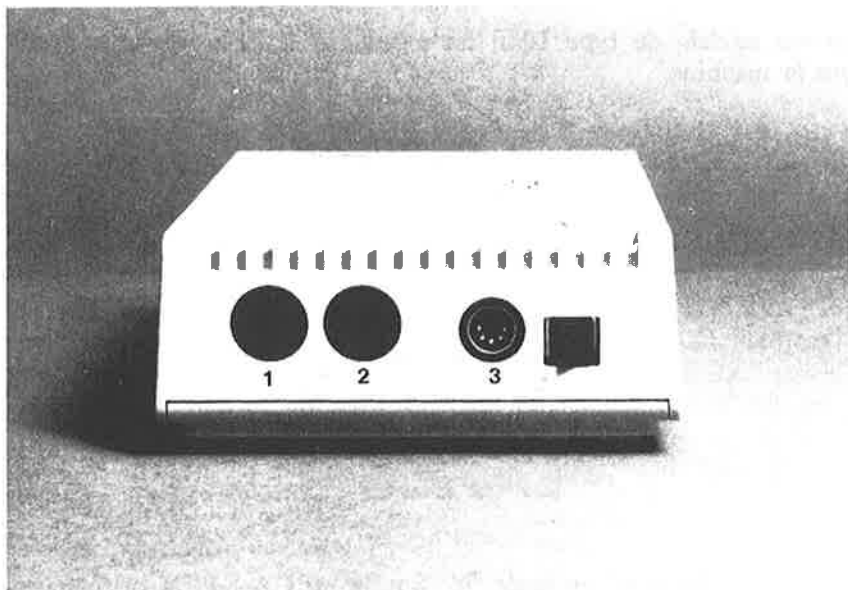
Connectez le cordon secteur du moniteur à l'arrière droit de celui-ci. Le cable video, solidaire du moniteur, doit être connecté à la prise 7 du modèle 520 ST/M ou à la prise 5 du 1040 ou du MEGA ST. Noter à nouveau la petite encoche qui doit pointer vers le haut.

### **1.2.2. Le téléviseur**

Le 520 ST/M peut être relié à un téléviseur via un cable d'antenne, fourni avec l'ordinateur. Ce cable est équipé d'un coté d'une prise antenne que l'on reliera à son homologue coté téléviseur, l'autre extrémité du cable est équipée d'une prise cinch que l'on vient insérer dans la prise se situant entre les prises 6B et 7.

## **1.3. LE LECTEUR DE DISQUETTE**

Le lecteur de disquette doit d'abord être relié à sa propre alimentation. Le cable de l'alimentation sera donc inséré dans la prise 3 du lecteur de disquette, toujours avec l'encoche vers le haut. A présent il faut relier le lecteur à l'unité centrale, ce que l'on réalisera au moyen du cable gris que l'on viendra insérer d'une part dans la prise 1 du lecteur de disquette, et de l'autre coté dans la prise 10 de l'unité centrale.



Si vous possédez un deuxième lecteur, celui-ci sera relié au premier par la prise 2 (prise 1 du second lecteur).

#### **1.4. LE CLAVIER**

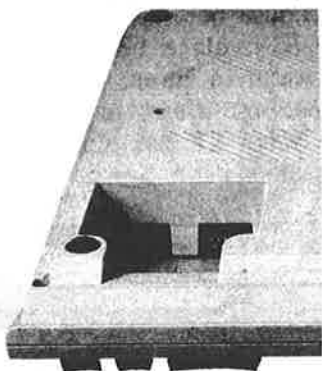
Ce chapitre ne concerne que le MEGA ST puisque sur les autres modèles le clavier est intégré à l'unité centrale. Le cordon en spirale doit être connecté à la prise correspondante de l'unité centrale qui se trouve sur le côté gauche. Au niveau du clavier le connecteur se trouve entre les deux renforcements pour la souris et pour le joystick.

Pour ôter le cable, il faut d'abord appuyer sur la languette de sécurité avant de tirer délicatement sur la prise.

#### **1.5. LA SOURIS**

La souris constitue le dernier périphérique que nous devons connecter avant de commencer à travailler avec la machine. Sur les modèles du type 520 ST/M, deux prises au format convenable se trouvent sur le côté droit de la machine. Connectez la souris sur la prise marquée "0".

Sur les modèle de type 1040 les prises souris se trouvent à droite sous la machine.



On utilisera la prise marquée "0". Sur le MEGA ST, la prise souris est située sous le clavier. Retournez ce dernier, connectez la prise et placez le cable dans la rainure prévue à cet effet. De cette manière le clavier ne sera pas bancal par la suite. La souris étant un élément indispensable à la quasi totalité des applications sur ATARI ST, on laissera cette dernière en permanence sur le coté droit de la machine.

### **1.6. LA MISE EN MARCHE**

Avant de brancher les divers cordons secteur (de l'ATARI, du lecteur de disquette, du moniteur, ...) dans la prise murale, assurez-vous que tous les interrupteurs sont en position OFF (y compris celui du moniteur : la roue du potentiomètre doit être calée dans une position extrême avec un petit "clic"). Il est conseillé d'utiliser une barre de prises multiples sur laquelle on vient enficher tous les cordons secteur. A présent insérez la disquette Language Disk, ou la disquette System Disk si celle-ci est fournie avec la machine, dans le lecteur A (ou dans le lecteur intégré dans le cas du 1040 STF ou du MEGA ST). On procèdera comme sur la photographie.





Maintenant que tout est prêt vous pouvez brancher la prise de la barette multi-prises sur le secteur, et mettre en route les différents éléments dans l'ordre suivant :

1. Le moniteur (ou le téléviseur)
2. Le lecteur de disquettes 2 (s'il existe)
3. Le lecteur de disquettes 1 (s'il existe)
4. L'unité centrale

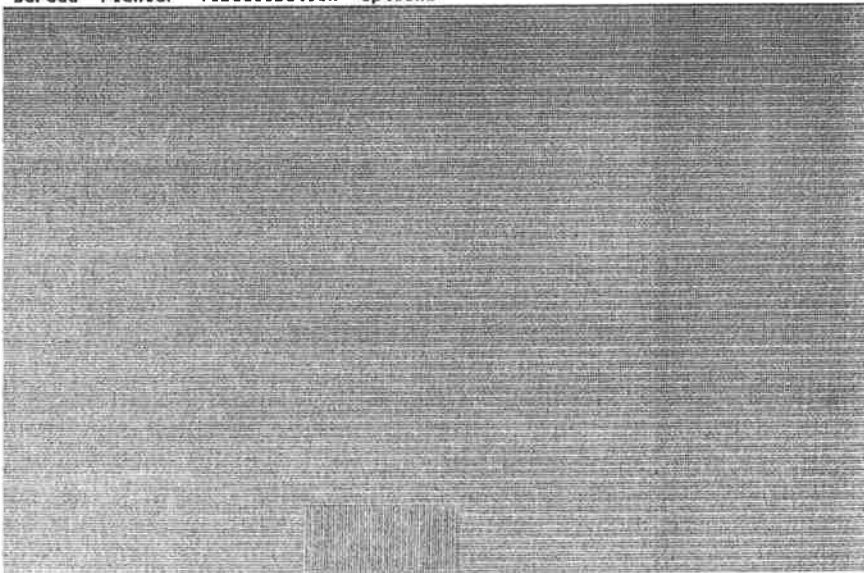


## **2. LE TRAVAIL SUR L'ATARI ST**

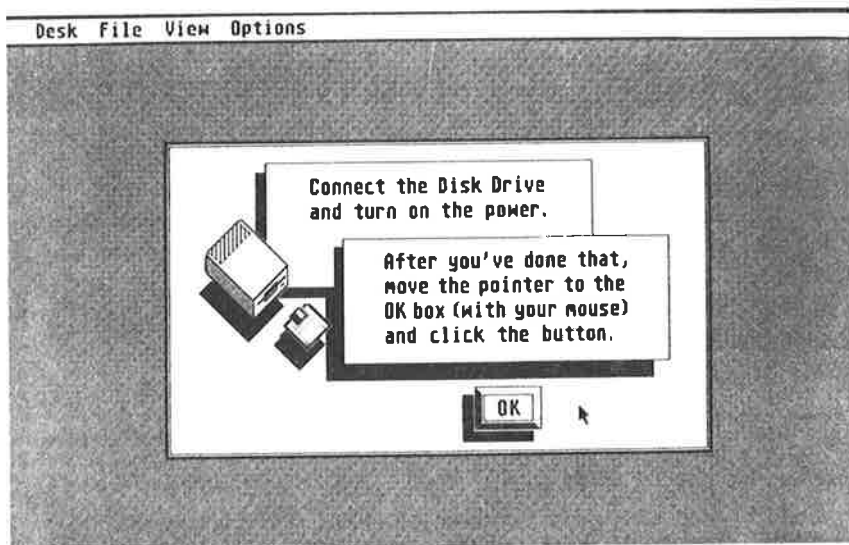
### **2.1. CHARGEMENT DU SYSTEME D'EXPLOITATION TOS ET GEM**

Au bout de quelques secondes, votre moniteur vous montrera l'image suivante. Si cela n'apparaît pas, essayez de tourner les deux boutons à droite du boîtier pour améliorer éventuellement la luminosité ou le contraste. Si cela ne règle pas le problème, vérifiez que vous avez bien exécuté toutes les étapes dans l'ordre décrit précédemment.

Bureau Fichier Visualisation Options



Attendons quelques secondes et déjà la loupiote rouge du lecteur de disquette connecté se met à briller. Quelques secondes encore et l'écran est rempli par une nouvelle image titre.



On vous demande alors de placer la disquette système dans votre lecteur de disquette. Cette disquette système est fournie avec l'ATARI ST. Elle est intitulée SYSTEM.DISK.

Qu'est-ce qu'un système d'exploitation ? Cela pourrait être expliqué brièvement ainsi : avec le système d'exploitation, nous chargeons dans l'ordinateur ce dont il a besoin pour pouvoir simplement travailler. C'est en quelque sorte l'ensemble de nos instructions de travail, comment il doit faire quoi. Tous les messages, tout ce qu'il nous communique, comme nous allons le voir par la suite, tout cela est contenu dans le système d'exploitation. Sans ce système d'exploitation, votre ATARI ST ne sait rien faire du tout.

Nous allons donc faire charger ce système d'exploitation de la disquette dans la mémoire de l'ordinateur. Vous voyez sur la photo comment vous devez placer la disquette dans le lecteur de disquette.



Maintenant que la disquette est placée dans le lecteur de disquette, le message de votre ST vous indique d'amener, avec la souris, la flèche sur le champ OK et d'appuyer sur la touche gauche de la souris.

Nous allons maintenant exécuter cette opération. Déplacez votre souris ST sur le bureau (avec la boule vers le bas et les touches de pression vers le haut), de telle sorte que la flèche se trouvant sur l'écran se déplace vers la petite case OK. Il se peut que vous parveniez à parcourir ce chemin d'un seul mouvement (sans soulever la souris), mais vous pouvez tout aussi bien soulever de temps en temps la souris pour continuer à la déplacer pas à pas, à partir d'une position de départ plus favorable.

Il se peut que le maniement de la souris vous semble un peu difficile au départ mais vous verrez que vous manierez bientôt ce rongeur comme si c'était une troisième main.

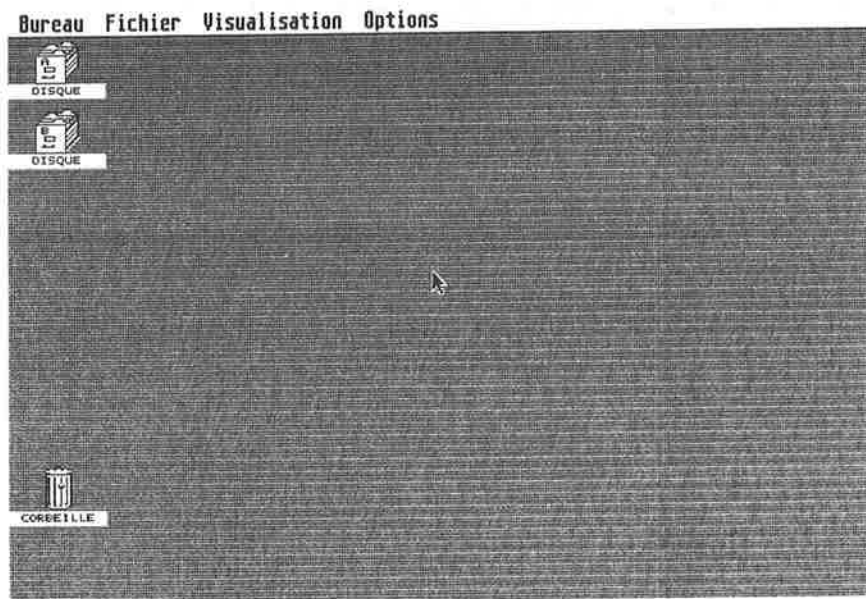
Appuyez maintenant sur la touche gauche de la souris. Dans cet ouvrage, à de rares exceptions près, ce sera toujours la touche gauche de la souris que nous vous demanderons d'actionner. La souris possède en effet une seconde touche de pression mais il y a peu de programmes commerciaux jusqu'ici qui fassent appel à cette seconde touche.

Que s'est-il passé après que vous ayez appuyé sur la touche gauche de la souris ? Un bruit sec, votre lecteur de disquette a écarté la protection métallique et il peut ainsi accéder à la disquette 3 pouces et demi placée dans le boîtier ... l'écran est ensuite vidé (ce qui est provoqué par le programme de système d'exploitation qui a déjà été chargé en partie à partir de la disquette).

Il vous faut maintenant patienter un peu : d'ici environ 30 secondes, la couleur de l'écran va passer au blanc avant que le programme chargé commence enfin à tourner. Il se passe alors quelque chose à l'écran.

## 2.2. LE BUREAU ELECTRONIQUE :

### GEM DESKTOP



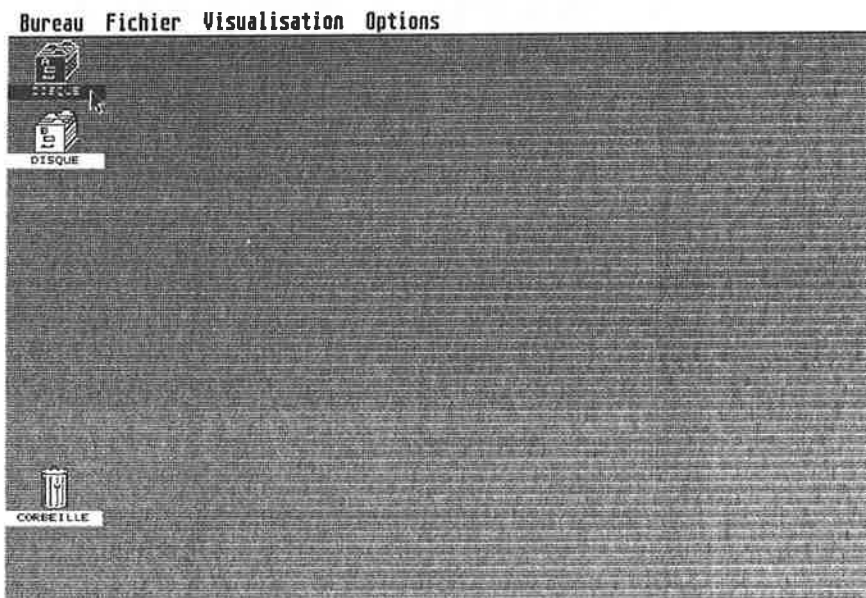
Vous voyez maintenant une ligne blanche dans le bord supérieur de l'écran, qui contient les notions BUREAU (DESK), FICHIER (FILE), VISUALISATION (VIEW) et OPTIONS.

Vous devez vous représenter ces illustrations comme correspondant à votre bureau. Elles montrent deux lecteurs de disquette avec, en bas, la corbeille à papier. Comme nous utiliserons cette comparaison à de nombreuses reprises dans la suite de ce chapitre, nous pouvons nous représenter les lecteurs de disquette, par rapport à notre bureau, comme de grandes boîtes de fiches (les petites images sur le moniteur montrent d'ailleurs également les lecteurs de disquette sous la forme de boîtes de fiches).



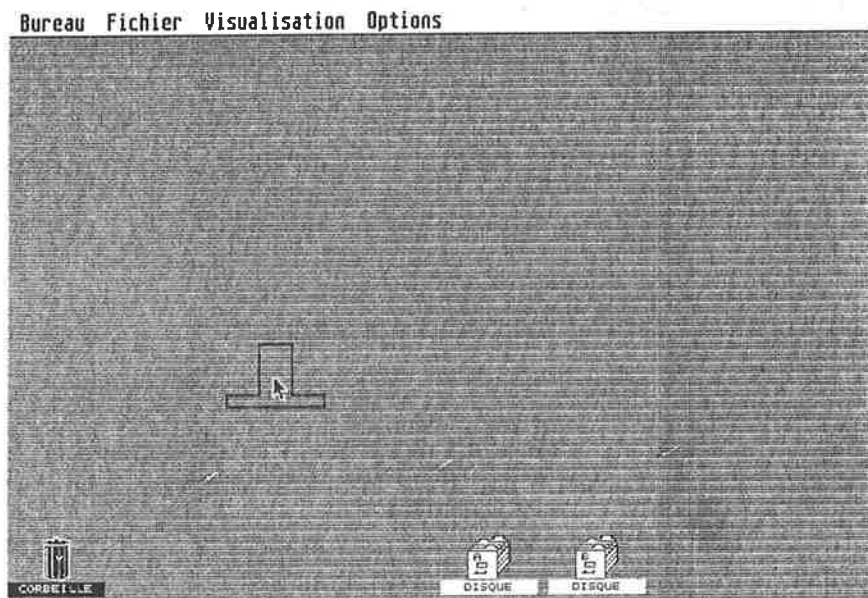
### 2.2.1. Déjà rangé ?

Cette disposition de votre bureau ne vous plaît pas ? Bon, eh bien il nous suffit de tout réorganiser. Amenez donc d'abord la flèche de la souris sur un lecteur de disquette puis appuyez brièvement sur le bouton gauche. Le ST affiche : 'J'ai compris' car le lecteur de disquette ou notre boîte de fiches se trouve maintenant représentée autrement qu'au départ : tout ce qui était blanc auparavant est maintenant noir et inversement.



Appuyez maintenant encore une fois sur la touche de la souris mais tenez-la enfoncée et déplacez alors la souris sur l'écran ... pardon, sur votre bureau, vers la droite. Au lieu de la flèche habituelle, c'est maintenant un morceau du lecteur de disquette que vous déplacez sur l'écran. Si vous relâchez maintenant la touche de la souris, la représentation complète du lecteur de disquette sera affichée exactement là où est apparu pour la dernière fois un morceau du lecteur de disquette. En déplaçant la souris, vous avez entrepris la réorganisation du matériel placé sur votre bureau.

Vous n'avez eu pour cela qu'à déplacer un morceau mais le ST a ensuite transféré les symboles correspondants dans le nouvel emplacement en les faisant disparaître de leur position originelle sur l'écran.

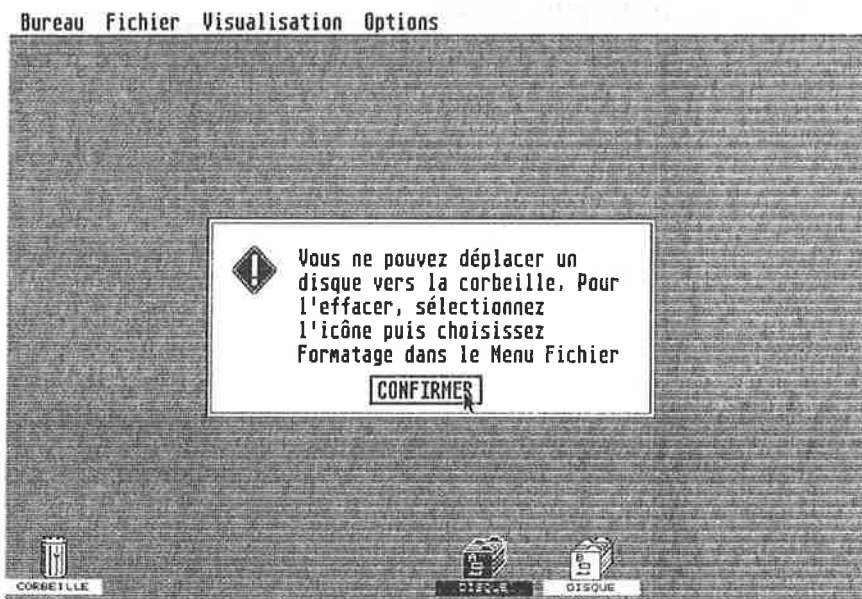


Ne vous gênez surtout pas pour déplacer également la corbeille à papier vers un autre emplacement de l'écran. Un bien beau jouet que ce système d'exploitation, pas vrai ? Et pourtant, le déplacement des images (on parle également "d'icônes") sur l'écran du moniteur ne constitue qu'une petite partie de ce système d'exploitation. A ce sujet : ces "déplacements" ne se déroulent pas, en réalité, sur le moniteur, mais sous le clavier, à l'intérieur de votre ATARI ST, l'écran du moniteur ne sert pour cette fonction qu'à l'affichage.

Quel que soit l'attrait que représente le déplacement sur l'écran, gardez-vous bien cependant de placer l'un sur l'autre les deux symboles des lecteurs de disquette. Le ST penserait en effet aussitôt que vous voulez copier des disquettes et nous n'en sommes pas encore là !

Si vous vous trouvez malgré tout dans ce cas par mégarde, amenez la flèche de la souris sur la possibilité de réponse 'Annuler' et tout rentrera à nouveau dans l'ordre.

Si vous essayez d'amener un lecteur de disquette sur la corbeille à papier ou inversement, votre ST vous avertira que cela n'est pas possible. Il vous suffira alors d'appuyer dans le champ CONFIRMER et tout rentrera dans l'ordre.

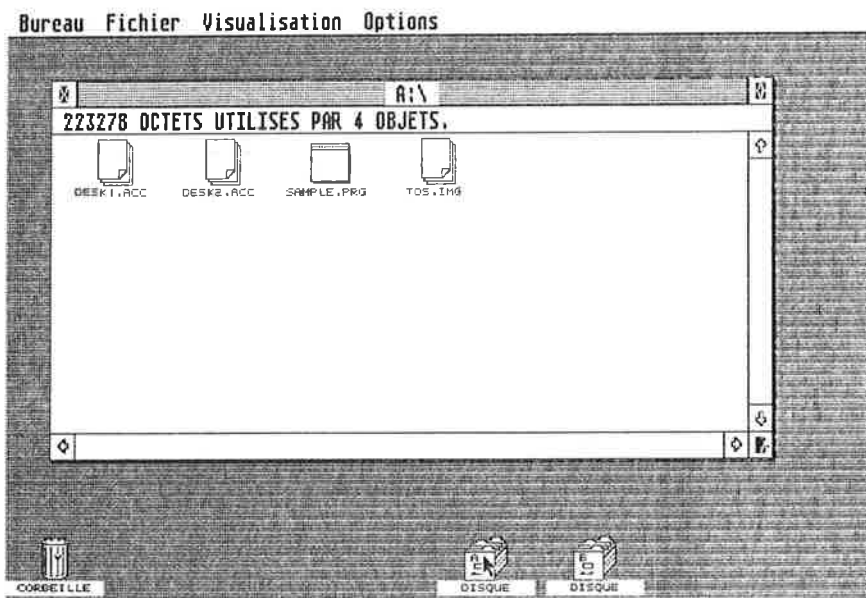


Vous voyez ici combien l'ATARI ST est prévenant à votre égard et toujours prêt à vous aider. Quelles que soient les fonctions que vous exécutiez, il a toujours quelque chose à répondre. C'est ce qu'on appelle être orienté en fonction de l'utilisateur !

### 2.2.2. Le catalogue de la disquette

Nous allons maintenant examiner ce qu'il y a dans notre lecteur de disquette ou dans notre carton de fiches. Amenez pour cela la flèche sur l'image du lecteur de disquette A puis appuyez une fois sur la touche gauche de la souris.

Si vous avez bien fait cela comme il faut, le lecteur de disquette se colore à nouveau en noir, ce qui veut dire que le ST a compris que vous voulez faire quelque chose avec ce lecteur de disquette. Appuyez maintenant encore deux fois, en attendant à peine entre les deux, sur la touche gauche de la souris et une feuille de papier sera expulsée à grande vitesse du lecteur de disquette A. Cette feuille se déplie sous nos yeux.



Comme si vous aviez inséré dans votre boîte de fiches différentes feuilles de séparation pour que vous puissiez vous y retrouver parmi toutes vos fiches, de même la disquette placée actuellement dans le lecteur de disquette contient-elle aussi différentes parties, différents programmes ou enregistrements de données, que vous pouvez maintenant examiner tranquillement à l'écran avant d'effectuer finalement votre choix.

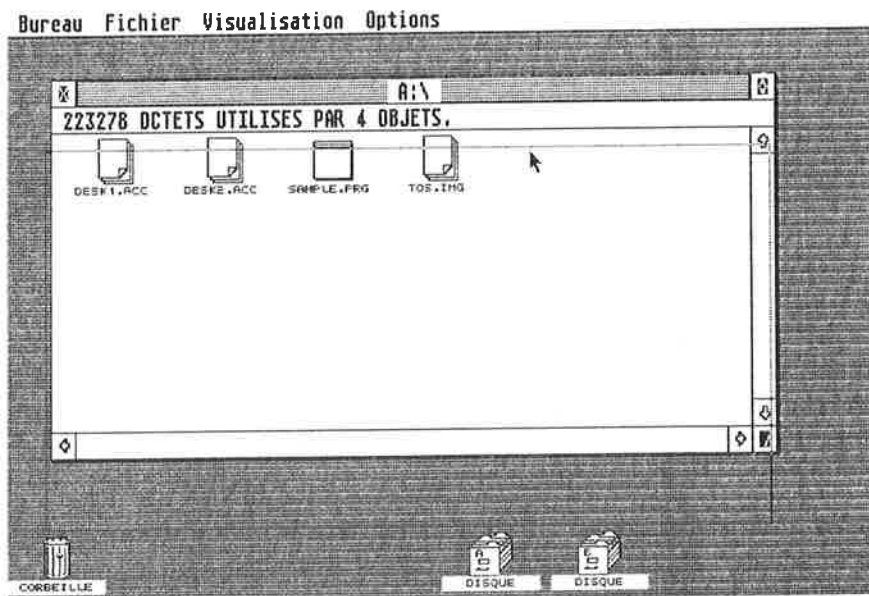
Toutes ces informations constituent en fait ce qu'on peut appeler le catalogue de la disquette qui est actuellement placée dans le lecteur de disquette A.

Examinons le papier de plus près. Vous vous rappelez comment nous avons modifié précédemment l'ordonnancement de notre bureau ? Nous pouvons faire la même chose avec les programmes et les enregistrements de données, sur notre bureau/écran.

### 2.2.2.1. Déplacement

A cet effet, amenez simplement la flèche, à l'aide de la souris, dans la ligne hachurée de la feuille de papier affichée. Remarquez en passant qu'il y a marqué A: au milieu de ce côté. Cela signifie: ceci est le catalogue de la disquette qui se trouve actuellement dans le lecteur A ou 1.

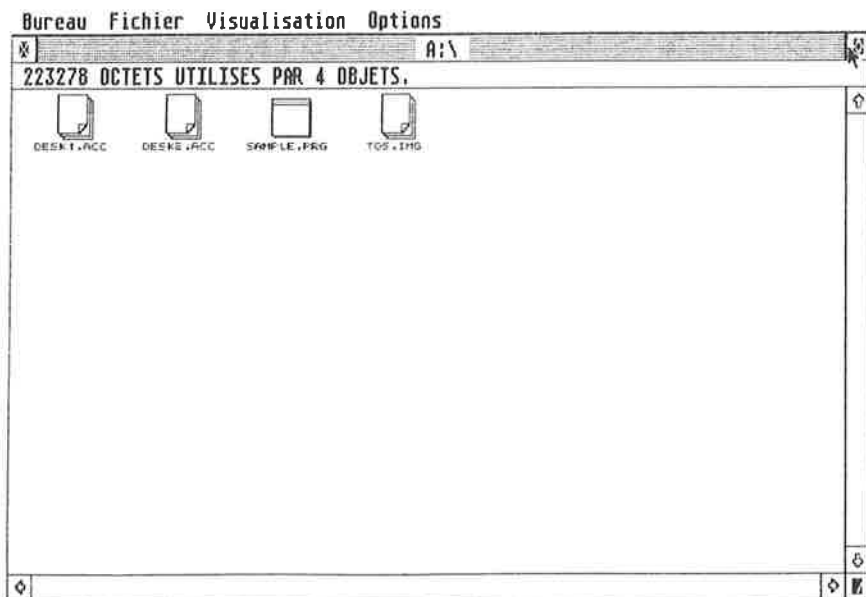
Lorsque la flèche sera donc pointée sur la ligne hachurée, appuyez sur la touche gauche de la souris et tenez-la enfoncée. Lorsque vous déplacerez alors la souris, vous déplacerez en même temps le catalogue tout entier. Cela vous est indiqué par le fait que le cadre du papier se déplace avec la souris, selon un procédé comparable à ce que nous avons vu lorsque nous avons réorganisé notre bureau.



Si vous relâchez le bouton gauche de la souris, on dirait que l'image suit le cadre et, en l'espace d'une seconde, notre feuille de papier se déplace. Il se peut même qu'elle couvre maintenant les représentations symboliques des lecteurs de disquette ou de la corbeille à papier.

#### 2.2.2.2. Remplissage

Que signifient les autres indications portées sur le bord de notre catalogue ? Pour cela, amenez d'abord la flèche, avec la souris, dans l'angle supérieur droit du papier et appuyez sur le bouton gauche de la souris (nous vous prions cette fois de ne pas le tenir enfoncé mais de ne l'actionner qu'une fois). La représentation du papier blanc grandit alors jusqu'à remplir l'écran entier.



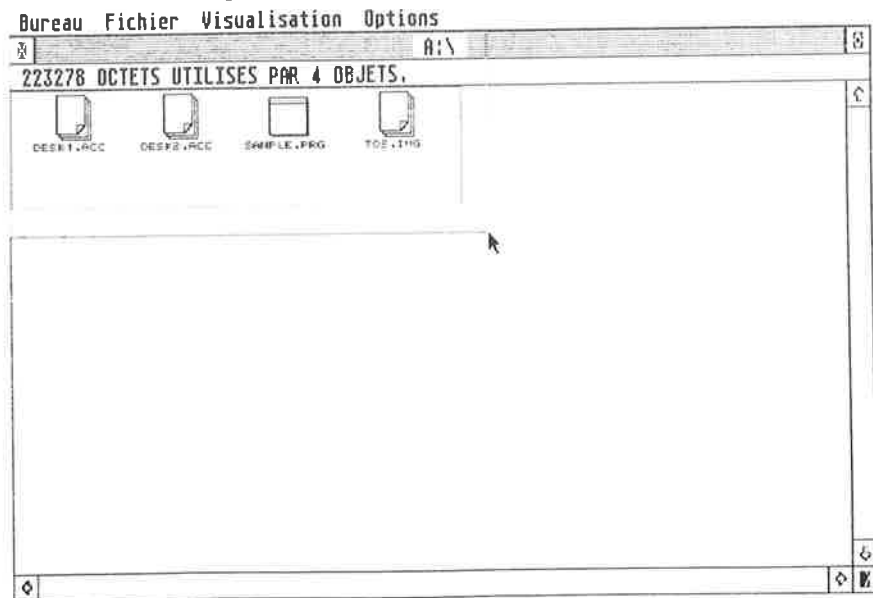
Marquez encore une fois ce petit carreau, en appuyant, et l'opération sera à nouveau annulée ... nous ne verrons plus à nouveau qu'une partie de notre feuille de catalogue.

Donc : le petit carreau dans l'angle supérieur droit est un commutateur pour activer ou désactiver. Il nous permet, la première fois que nous appuyons dessus, d'agrandir le papier pour qu'il remplisse la totalité de l'écran. La seconde fois que nous appuyons, nous revenons à l'image initiale.

### 2.2.2.3. Agrandissement et rétrécissement

Le petit symbole en bas à droite nous permet maintenant de déployer ou de comprimer le catalogue comme s'il était en caoutchouc. Amenez la flèche de la souris vers l'angle en bas à droite de notre feuille de catalogue, appuyez ensuite sur la touche gauche de la souris et tenez-la enfoncée par la suite.

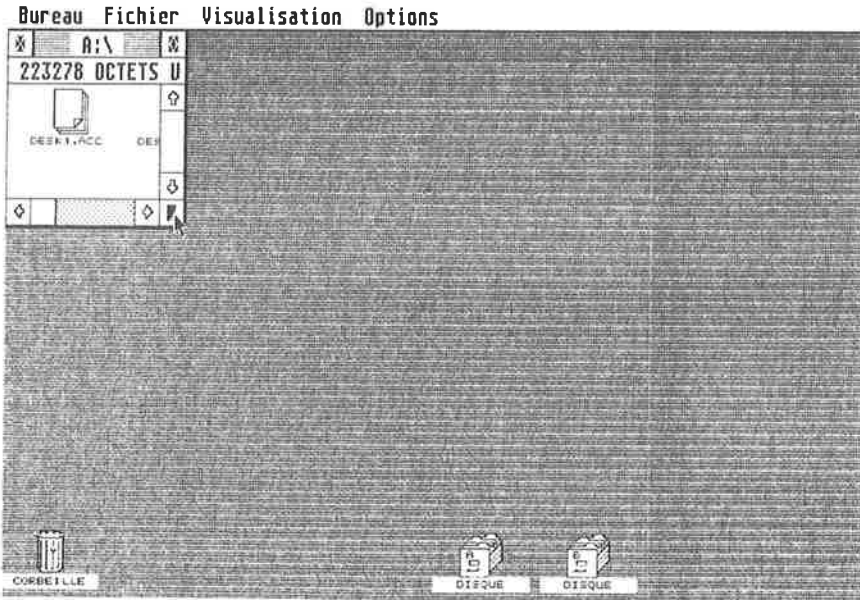
Déplacez maintenant la souris sur l'écran, de ci, de là. Vous pouvez ainsi modifier instantanément la taille de la feuille blanche du catalogue, pour l'agrandir ou la rétrécir.



Mais pour que nous puissions maintenant découvrir aussi les dernières indications portées au bord de notre feuille blanche, compressez le papier, avec l'aide du coin "en caoutchouc", jusqu'à ce que vous ne puissiez plus le réduire davantage.



Relâchez alors le bouton de la souris et vous constaterez que notre catalogue est vraiment devenu minuscule.

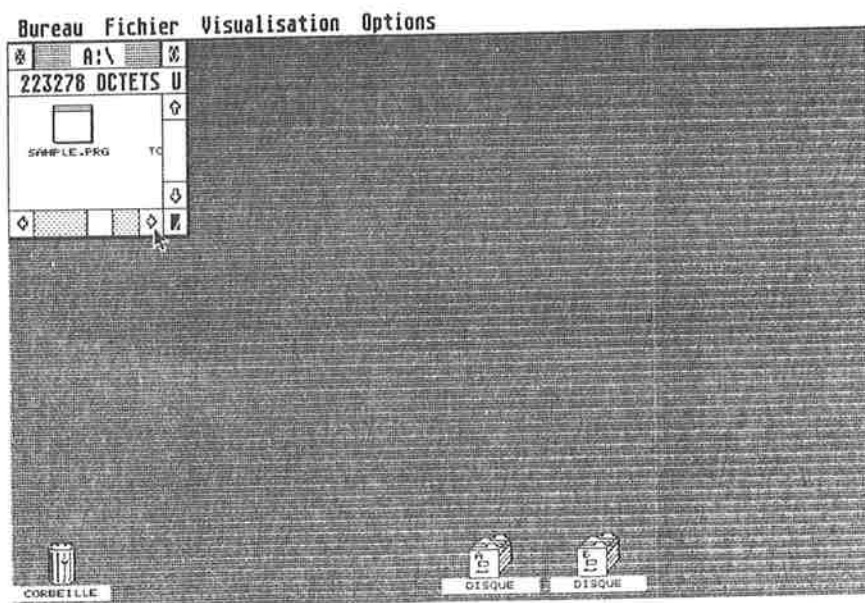


Nous n'en voyons maintenant qu'une petite partie. Si toutefois vous voulez finalement en voir plus, ce n'est pas un problème pour votre ATARI ST : il vous suffit de déployer le catalogue sur tout l'écran, avec le coin supérieur droit (le petit carreau); vous pourrez alors toujours le rapetisser à sa taille de départ en appuyant à nouveau sur la touche de la souris. Vous pouvez également l'étirer vers le bas, c'est-à-dire l'agrandir, au moyen du coin caoutchouc.

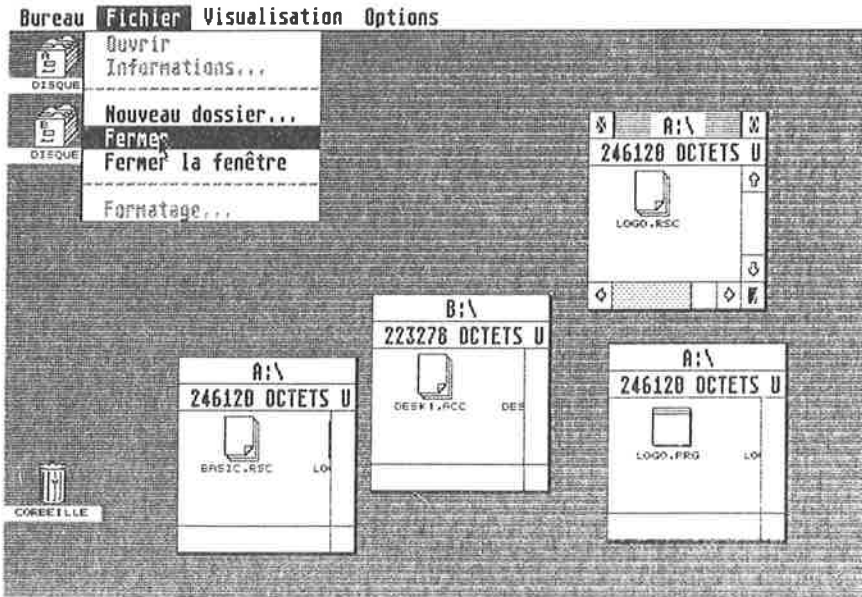
#### 2.2.2.4. Défilement avec les flèches

Tout cela est inutile si vous faites afficher dans l'écran rétréci le contenu, image après image, en vous servant des quatre symboles de flèches qui figurent sur les bords de notre feuille de catalogue.

Vous avez donc comprimé le catalogue autant que possible. Amenez maintenant la flèche de la souris, par exemple, sur le symbole de flèche en bas à droite (à gauche du coin caoutchouc) et appuyez ensuite brièvement sur le bouton de la souris : hop ... vous voyez la prochaine image de notre catalogue. Procédez de même avec les autres symboles de flèches. Par conséquent, bien que le catalogue ne puisse afficher qu'une petite image lorsqu'il est aussi petit, les touches-flèches vous permettent de vous déplacer sur le papier originel aussi bien de bas en haut que de droite à gauche.



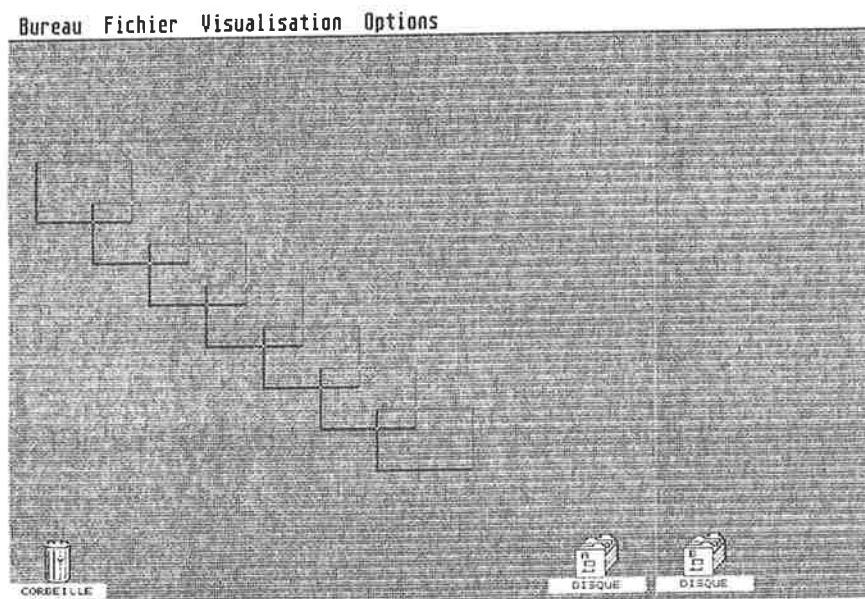
Vous vous demandez peut-être à quoi cela peut servir ? Supposons par exemple que vous ayez simultanément de très nombreux catalogues sur votre écran. Pour qu'il y ait vraiment de la place pour chacun, on rétrécit tout simplement chacun d'entre eux.



Lorsque des sections de l'écran défilent ainsi les unes à la suite des autres (comme c'est le cas ici lorsque nous faisons défiler les différentes parties de notre catalogue), on parle de "scrolling".

#### 2.2.2.5. Cela suffit !

Le dernier coin de notre catalogue, celui qui est en haut à gauche et dans lequel vous voyez un X, est chargé de faire entièrement disparaître de l'écran la feuille du catalogue. C'est ce qu'on appelle fermer la fenêtre. Amenez donc la flèche de la souris dans ce coin puis appuyez sur la touche gauche de votre souris : Le papier blanc qui contenait le catalogue disparaît et notre bureau a l'air à nouveau très propre. Exactement comme au début, vous ne voyez plus sur le bureau que deux boîtes de fiches/lecteurs de disquette et une corbeille à papier. La feuille de catalogue a disparu dans le lecteur de disquette A :



Par conséquent : le coin de fermeture fait à nouveau disparaître le catalogue dans le lecteur de disquette d'où il venait, vidant ainsi l'écran de sorte qu'on ne voit plus que l'image de départ.

### 2.2.3. Les Menus Déroulants

Intéressons-nous maintenant aux indications portées dans la ligne supérieure de notre écran, qui portent les intitulés 'Bureau' (Desk), 'Fichier' (File), 'Visualisation' (View) et 'options'.

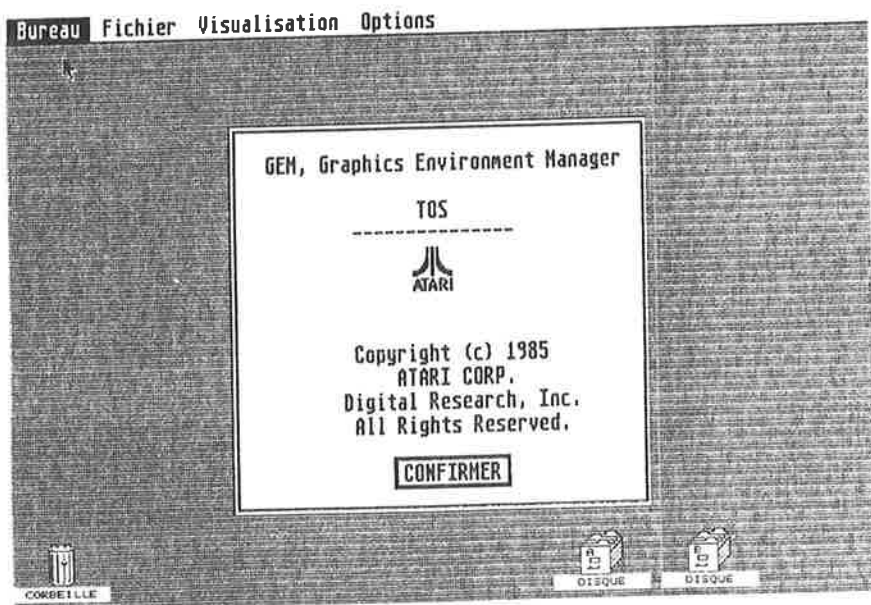
#### 2.2.3.1. Bureau (Desk)

Amenez tout d'abord la flèche, en vous aidant de la souris, sur 'Bureau'. Une liste de fonctions, également appelée menu, défile alors vers le bas, comme un rouleau. Ce type de menu est appelé également 'menu déroulant' (menu tire vers le bas). Vous y voyez les informations suivantes :



Desktop Info (\*)

Plaçons tout d'abord la flèche sur le point 'Desktop Info...' et appuyons sur la touche gauche de la souris. Aussitôt la fenêtre suivante s'affiche à l'écran :



Ce que vous voyez est un message d'information concernant le système d'exploitation TOS et l'environnement GEM, qui ont été soit lus depuis la disquette lors de la mise en route (machines plus anciennes), soit directement lus dans la ROM du ST. Si l'année du copyright ne correspond pas à celle de notre copie d'écran cela signifie probablement que votre 1040 ou MEGA ST est équipé du nouveau TOS "blitter" sur lequel nous reviendrons par la suite.

Vous avez peut-être déjà entendu parler de GEM (Graphics Environment Manager). C'est lui qui gère notamment ces fenêtres que nous avons ouvertes, déplacées...

Pour revenir à l'état de la machine tel que nous l'avons quitté au moment de l'appel de ce message, placez la flèche sur la boîte OK et cliquez une fois (touche gauche de la souris).

Les autres rubriques du menu "Bureau" ne nous intéressent pas pour l'instant, nous y reviendrons par la suite. Avant de poursuivre notre examen des divers menus déroulants, nous allons nous attarder un peu sur un sujet très important.

Nous aimerions que vous fassiez une copie de votre disquette système et/ou langage. Les ST ayant le TOS intégré en ROM ne sont livrés qu'avec la seule disquette langage.

Pourquoi la copie de sécurité est-elle d'une aussi grande importance ?

Imaginez un instant que votre disquette originale soit rendue inutilisable. Cela peut arriver très simplement en renversant du café dessus, elle peut aussi avoir été soumise à un rayonnement magnétique (par exemple si elle a été posé trop près du moniteur ou d'une alimentation). Sans disquette système (ou langage), il devient impossible de travailler avec l'ordinateur. Pour cette raison il est fortement recommandé de travailler avec une copie de l'original, ce dernier étant soigneusement rangé dans un placard, n'intervenant que s'il arrive un malheur à la copie.

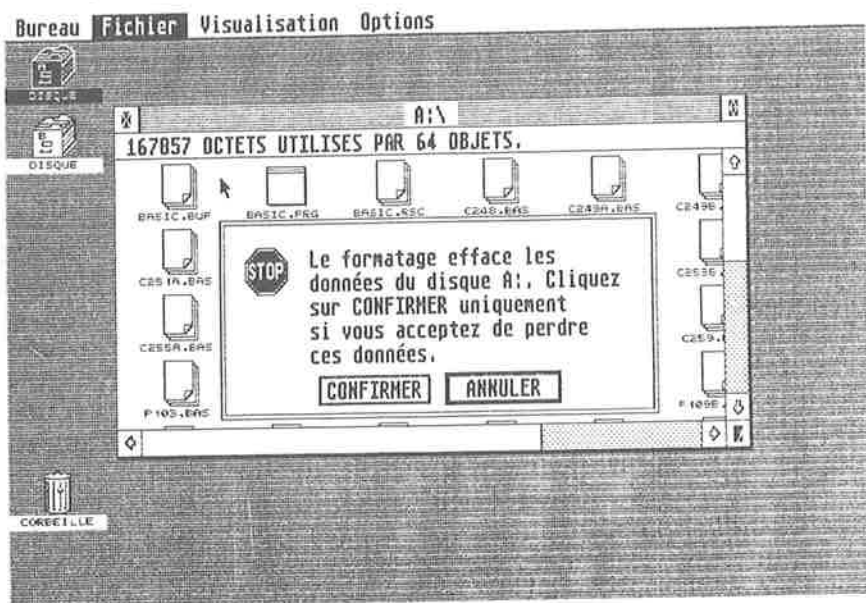
Attention, conformez-vous soigneusement aux descriptions qui vont suivre, c'est la seule manière de garantir le succès des opérations. Si vous disposez de deux lecteurs de disquette, éteignez le second lecteur et lancez à nouveau le système avec un seul lecteur allumé, comme cela est décrit au chapitre 1.5. Dans le cas contraire vous ne pourriez suivre nos instructions, la procédure étant légèrement différente (nous verrons par la suite comment faire) avec deux lecteurs.

## 2.3 NOUS REALISONS DES DISQUETTES NEUVES

Vous avez peut-être remarqué que nous n'avons pas encore parlé de la fonction 'formater' du menu 'Fichier' (File). Pour essayer cette fonction, appelez d'abord, par un clic, le lecteur de disquette A puis la fonction de formatage.

### 2.3.1. Formatage

Un champ d'avertissement attire votre attention sur les terribles effets que la confirmation de cette question pourrait entraîner :



Toutes les données figurant sur la disquette seraient donc détruites si vous laissiez exécuter le formatage.

A quoi peut donc servir la fonction de formatage, si ce n'est pour effacer des disquettes ? Si vous avez déjà essayé de placer des disquettes 3 pouces et demi neuves dans votre lecteur de disquette, vous aurez sans doute remarqué que ces disquettes ne sont pas encore en état d'être utilisées par votre ST. Quoi que vous fassiez, votre ST refuse de travailler avec de telles disquettes. Ces disquettes doivent d'abord avoir été préparées pour leur utilisation sur votre ordinateur ST.



Le formatage consiste justement à effectuer cette préparation indispensable. L'ATARI ST divise pour cela votre disquette en différentes pistes sur lesquelles il pourra plus tard stocker des programmes et des données. Il crée en quelque sorte les tiroirs dans lesquels il placera plus tard les données à sauvegarder.


#### FORMATAGE

FORMATAGE

SORTIR

Nom de l'unité: A:      FORMATER

Nom du disque: \_\_\_\_\_

FORMAT:      

Simple face      Double face

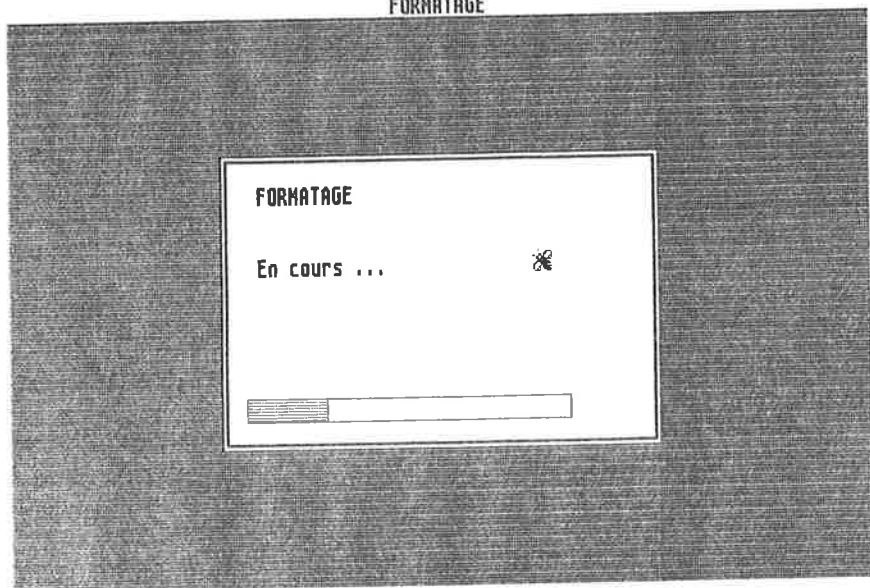
Placez donc maintenant une disquette vierge dans votre lecteur de disquette, donnez un nom à cette disquette et indiquez si vous disposez d'un lecteur de disquette simple face ou double face.

Votre disquette peut en effet être lue sur une ou sur ses deux faces. Sur l'ATARI 520 STF, le lecteur est un simple face alors que sur le 1040 STF, c'est un double face. Pour les lecteurs externes, l'ATARI SF 354 lit et écrit par exemple sur une seule face de disquette alors que le SF 314 lit et écrit sur les deux faces d'une disquette.

Vous devez donc choisir la possibilité conforme à votre lecteur de disquette. Marquez ensuite simplement la case FORMAT par un clic.

Peu de temps après, une barre horizontale apparaît qui vous indique où en est le formatage de votre disquette.

#### FORMATAGE



Une fois le formatage terminé, une fenêtre vous annonce que la disquette a été formatée sans problème, c'est-à-dire que vous disposez, par exemple pour une disquette simple face, de 357376 Bytes ou places mémoire.

Vous pouvez maintenant formater d'autres disquettes ou abandonner à nouveau cette fonction en marquant par un clic le champ 'SORTIE' (EXIT).

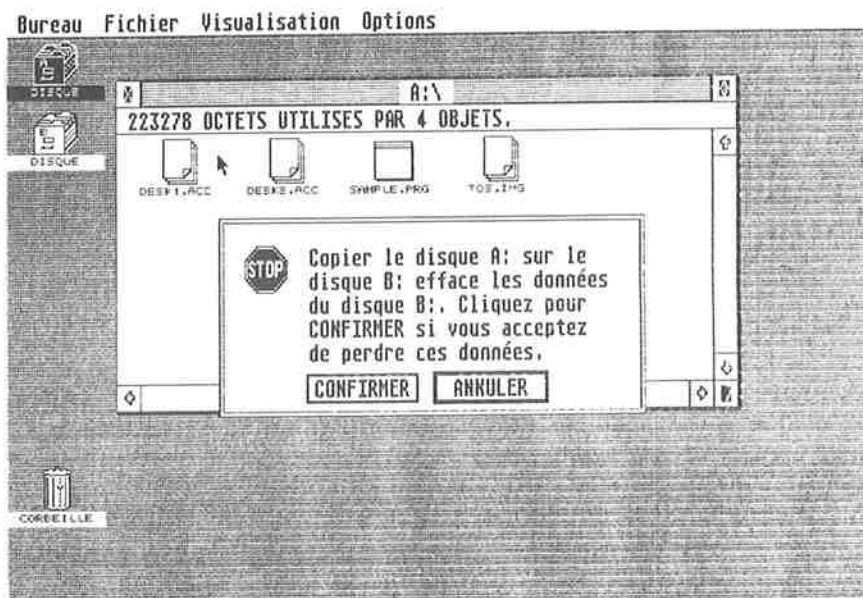
### 2.3.2. Copie de disquettes

Maintenant qu'une disquette formatée se trouve dans votre lecteur de disquette, nous allons enfin pouvoir réaliser la tâche que nous nous étions fixée auparavant, c'est-à-dire copier la précieuse disquette système.

Nota bene : Si vous disposez d'un lecteur de disquette double face (par exemple ATARI SF 314), vous devez malgré tout formater en SIMPLE FACE la disquette que vous utiliserez pour y copier la disquette système. Le lecteur de disquette double face peut en effet lire et écrire dans les deux formats, simple et double face alors que le lecteur de disquette simple face ne peut traiter que les disquettes simple face.

Mais pour que la disquette système puisse être lue aussi bien par les lecteurs de disquette simple face que double face, ATARI fournit la disquette système en format simple face.

Bon, nous pouvons enfin nous attaquer à la copie. Sélectionnez le lecteur de disquette A avec la flèche, par un simple clic, et amenez-le sur le lecteur de disquette B. On vous rappelle à nouveau certaines mesures de sécurité, avant que l'opération de copie (avec une disquette formatée auparavant) puisse commencer.



Si vous disposez de deux lecteurs de disquette, l'opération de copie est très simple : placez la disquette originale (la disquette système en l'occurrence) dans le lecteur de disquette A et la disquette sur laquelle doit se faire la copie dans le lecteur de disquette B. Vous n'avez plus alors qu'à confirmer l'opération par un clic dans le champ 'COPIER'.

Si vous disposez "seulement" d'un lecteur de disquette, on vous demandera à plusieurs reprises d'échanger les disquettes dans le lecteur de disquette A (original = A, copie = B). Votre ordinateur ST chargera en mémoire de grandes parties du contenu de la disquette originale avant de les réécrire sur la disquette encore vide mais déjà formatée.

Emulateur VT 52

Passons maintenant à la description des 4 autres parties que nous trouvons sous le titre Bureau. Si toutefois vous ne comprenez pas tout ce que vous allez lire maintenant, laissez tout simplement de côté ce que vous ne comprenez pas. Si vous voulez plus tard utiliser ces fonctions de votre ST, vous saurez que c'est ici que vous pourrez trouver les informations nécessaires. Entendu ?

Dirigez donc la flèche de la souris sur l'émulateur VT 52 (représentation en vidéo inversée) puis activez-le en appuyant sur la touche de la souris. Notre écran graphique disparaît alors et rien ne se passe apparemment si ce n'est que vous voyez apparaître l'affichage suivant :

```
*****  
| Emulateur Atari de Terminal VT52 |  
| (c) Atari Corp.                  |  
*****  
Appuyez sur:  
1) UNDO pour revenir au Bureau.  
2) HELP pour configurer le terminal.
```

L'émulateur VT 52 que vous venez de connecter ne fait rien d'autre, en fait, que de vous permettre de recevoir des données venant d'autres ordinateurs, à travers l'interface RS 232 ou bien à travers un coupleur acoustique (voir à interface RS 232 dans le chapitre sur les interfaces de l'ATARI ST).

Comme vous n'avez certainement pas de coupleur acoustique et que vous n'avez certainement pas non plus connecté un second ordinateur sur votre ST, nous allons interrompre l'opération en appuyant sur la touche en longueur qui se trouve dans la moitié droite du clavier et qui porte l'inscription 'UNDO'.

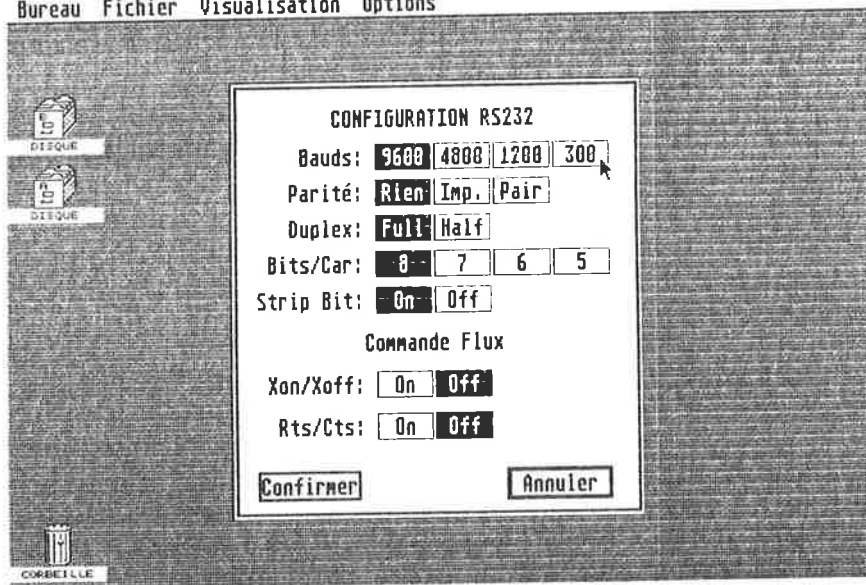
Si toutefois vous possédiez un coupleur acoustique et si vous aviez auparavant adapté l'interface RS 232 correctement à la connexion réalisée, les données pourraient maintenant, après établissement d'une liaison (par exemple à travers un ligne téléphonique), affluer dans votre ST et être affichées à l'écran.

Si vous voulez établir une liaison à travers l'interface RS 232, il ne vous suffit pas d'activer l'émulateur VT 52 avec la souris mais il vous faut encore avoir préparé également l'interface RS 232 pour une réception correcte.

### Réglage de la RS 232

Cela peut être fait directement à partir de l'émulateur VT 52 activé (en actionnant la touche HELP qui se trouve dans la moitié droite du clavier) ou en sélectionnant directement la fonction 'instal. RS 232' dans le premier menu déroulant. Une fois que vous aurez fait cela, un autre menu apparaîtra à l'écran qui se présentera ainsi :

Bureau Fichier Visualisation Options



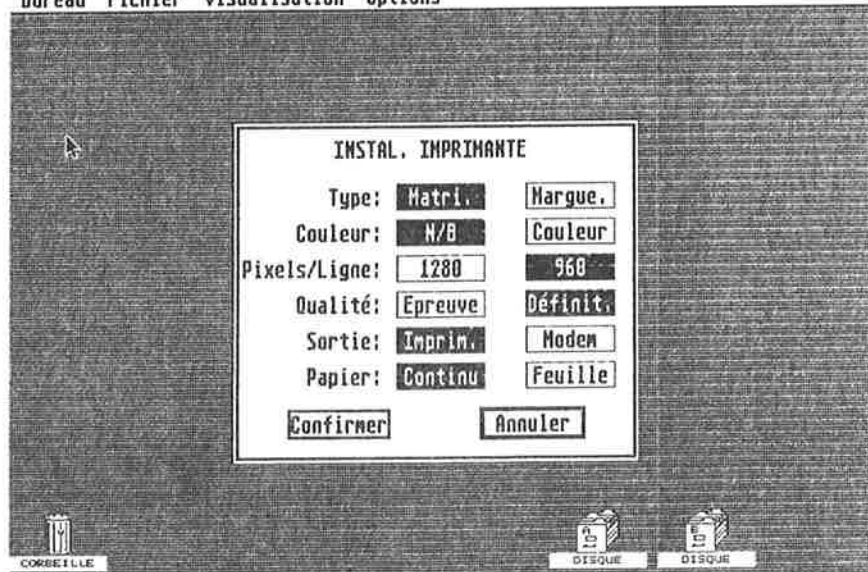
Vous pouvez maintenant choisir parmi les propositions placées les unes au-dessus des autres (par exemple, première ligne : vitesse de transmission en baud de 9600, 4800, 1200 ou 300 bauds) en dirigeant la flèche sur l'option choisie et en actionnant la touche gauche de la souris.

Pour confirmer cette action, le champ de l'écran correspondant apparaîtra en inversion vidéo (blanc sur noir). Si ce réglage a été effectué avec succès, vous pouvez amener la souris sur le champ CONFIRMER et confirmer (en appuyant sur la touche gauche de la souris). Si toutefois le réglage effectué était erroné, appelez la fonction 'ANNULER' sur la droite.

### Réglage de l'imprimante

La structure du réglage de l'imprimante, qui se trouve également dans le premier menu déroulant, est identique.

Bureau Fichier Visualisation Options

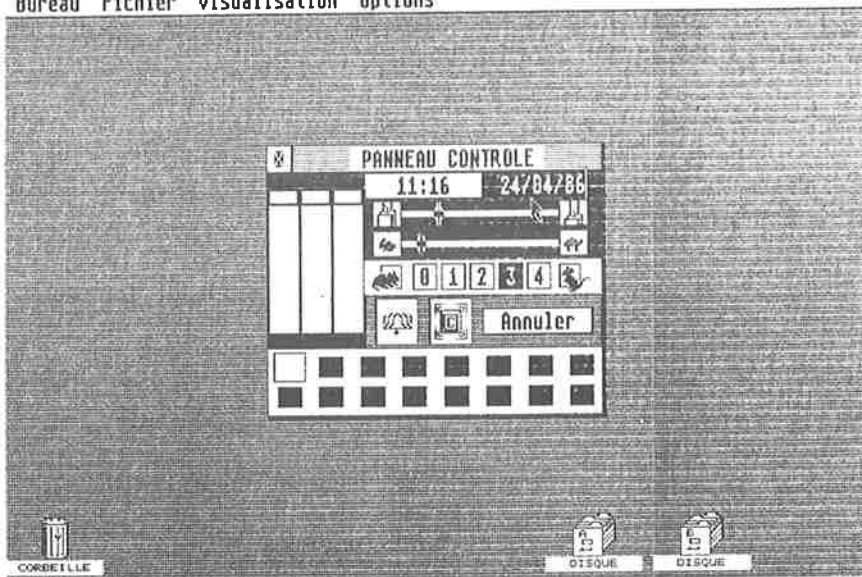


Vous pouvez indiquer ici si votre imprimante est une imprimante matricielle ou à marguerite, une imprimante en noir et blanc ou même en couleurs. De nombreuses autres fonctions peuvent être appelées ici mais nous ne pouvons toutes les évoquer ici en détail.

### Panneau contrôle

Nous allons maintenant découvrir la dernière fonction du premier menu déroulant, le champ de contrôle.

Bureau Fichier Visualisation Options



Suivant le champ sur lequel vous dirigez la flèche et sur lequel vous actionnez la touche de la souris, vous pouvez ici choisir parmi les fonctions suivantes : heure ou date, vitesse de pression d'une touche jusqu'à la première répétition ou vitesse de répétition permanente, frappe d'une touche du clavier, signal d'erreur, couleur et vitesse de la double frappe de la touche de la souris.

Pour régler les couleurs (si vous êtes l'heureux propriétaire d'un moniteur couleur adapté à votre ST), vous pouvez également utiliser les trois 'boutons-poussoirs' de réglage placés sur le côté gauche du champ de contrôle (pour rouge, vert et bleu).



Si vous possédez un moniteur noir et blanc d'un haut niveau de résolution (640 sur 400 points) vous avez toutefois seulement la possibilité de choisir soit une page écran blanche avec écriture noire, soit une page écran noire avec écriture blanche. Il vous suffit pour cela de sélectionner pour les trois boutons-poussoirs de réglage l'écriture opposée.

Poussez les trois curseurs de réglage, avec la souris, dans la direction opposée, jusqu'à la butée; toutes les actions ultérieures auront alors une apparence exactement inverse à celle qu'elles avaient jusqu'ici à l'écran.

Voici encore quelques explications sur les différentes fonctions du champ de contrôle :

*Fonction heure :*

Appuyez sur le champ heure, effacez l'ancien affichage avec la touche Back space (ou en appuyant une fois sur la touche ESC) et entrez alors l'heure actuelle. Confirmez cette action en appuyant une nouvelle fois sur la touche gauche de la souris dans le champ de l'heure.

*Fonction date :*

Procédez comme pour la fonction heure. Veillez toutefois, aussi bien pour l'heure que pour la date, à ne pas effectuer d'entrées incorrectes car votre ST ne les accepte pas. Dans ce cas en effet, il affiche une autre date, possible celle-là.

*Réglage de la première répétition de la frappe d'une touche :*

Déplacez le pointeur sur cette ligne pour l'amener où vous le voulez. Si le commutateur se trouve très à gauche, la répétition de la frappe d'une touche se fera immédiatement après la première frappe de la touche, si par contre le commutateur se trouve très à droite, cela durera plus d'une seconde avant que ne soit répétée la première frappe d'une touche.

*Réglage de la vitesse pour les répétitions ultérieures de la frappe d'une touche :*

Les illustrations parlent ici d'elles-mêmes. Si le pointeur se trouve près du lièvre (côté gauche), la répétition s'effectuera très rapidement; si le pointeur se trouve par contre près de la tortue, cela durera une éternité (plus d'une seconde) avant qu'une touche que vous tenez enfoncée ne soit répétée.

*Réglage de la double frappe de la souris :*

Ici aussi, les symboles parlent d'eux-mêmes : Si vous activez le nombre 0 (souris se reposant sur le côté gauche), vous pourrez exécuter la double frappe très lentement et votre ST vous comprendra. Si vous avez par contre sélectionné la position 4 (souris debout sur le côté droit), il vous faudra vous dépêcher pour pouvoir effectuer une double frappe. Il s'est avéré, dans la pratique, que le réglage idéal est 2 ou 3.

*Cloche et clic de touche :*

Les deux symboles suivants peuvent être activés ou désactivés en actionnant le bouton de la souris, dans l'endroit correspondant, une ou deux fois. La cloche retentit lorsque vous avez commis une erreur grave, le clic du clavier se fait entendre lorsque vous actionnez une touche (ces deux fonctions peuvent être très précieuses comme moyen de contrôle). Plutôt que d'activer ou de désactiver une de ces deux fonctions à travers le champ de contrôle, nous vous conseillons, si nécessaire (si les bips continuels pouvaient gêner quelqu'un d'autre pendant que vous travaillez sur l'ordinateur), de baisser au maximum le bouton de réglage du volume sur le moniteur.

*Les trois curseurs pour RVB ont déjà été évoqués :*

Ils vous permettent de stocker jusqu'à 16 couleurs dans la partie inférieure du champ de contrôle. Ces couleurs pourront ensuite être appelées en cas de besoin, à nouveau avec la souris (pour les moniteurs noir et blanc, ces 16 réglages n'ont aucune fonction).

*Enfin, la touche ANNULER du champ de contrôle :*

Si l'entrée, telle que vous venez de la modifier, ne vous plaît pas, vous pouvez actionner la touche ANNULER. Toutes les valeurs modifiées seront instantanément fixées à nouveau sur leurs valeurs antérieures. Seules la date et l'heure demeurent intactes même lorsque vous appuyez sur ANNULER.

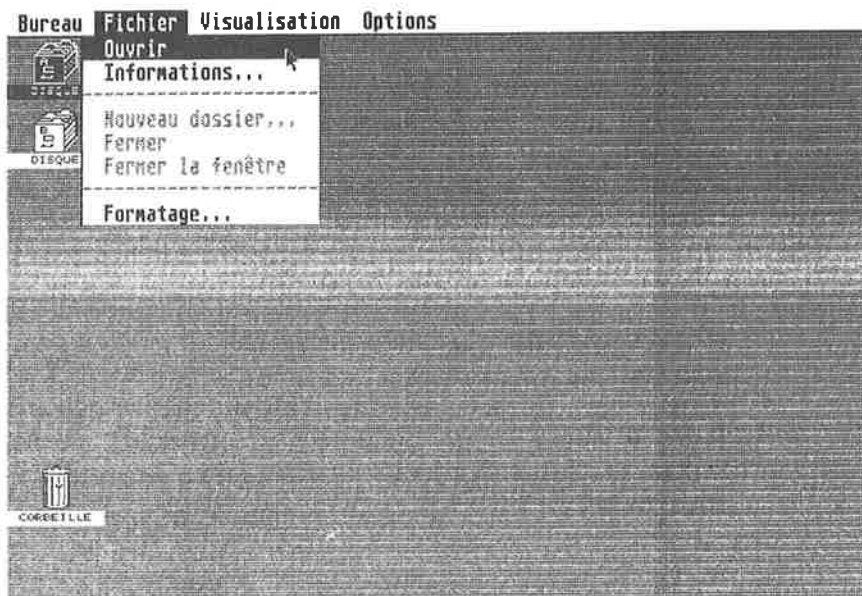
Vous pouvez sortir du 'programme' champ de contrôle en refermant la fenêtre du champ de contrôle ... comment ? Il faut bien sûr fermer en haut à gauche, dans le coin du champ de contrôle, exactement comme nous l'avions fait auparavant avec le catalogue du lecteur de disquette.

Vous pouvez par ailleurs déplacer également le champ de contrôle sur l'écran dans un sens ou un autre en faisant clic sur la ligne hachurée d'en haut, en appuyant sur la touche de la souris, puis en déplaçant la souris sur le bureau.

#### 2.3.2.1. Fichier (File)

Passons maintenant aux fonctions du second menu pull down qui est intitulé 'File' ou 'Fichier'.

Il vous faut, pour cela, appeler d'abord le lecteur de disquette A une fois par un clic puis amener ensuite la flèche, avec la souris, sur le champ 'Fichier'.



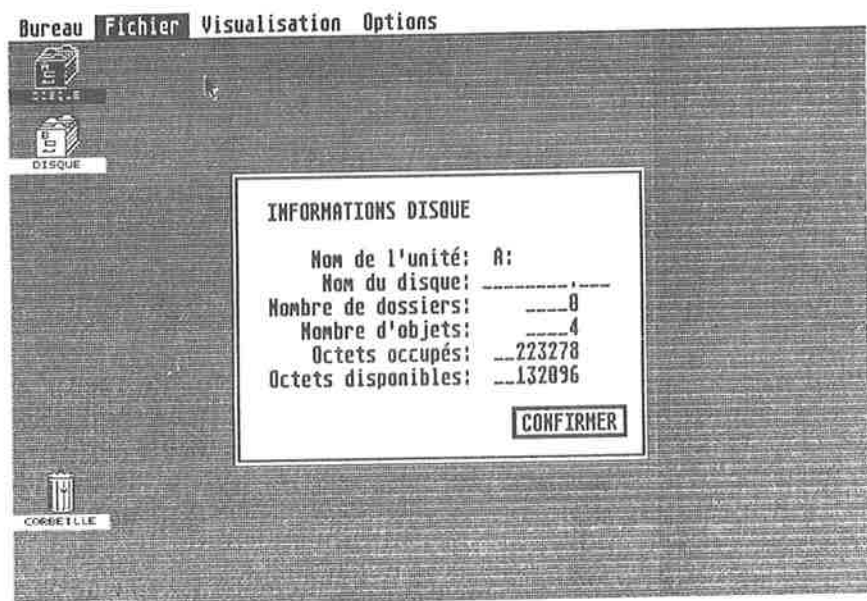
### Ouvrir

On nous propose d'abord 'd'ouvrir' quelque chose. Appelez ce champ avec la flèche et appuyez ensuite sur la touche gauche de la souris ... le contenu du lecteur de disquette A est affiché.

Pourquoi y a-t-il cette fonction alors que la fonction de l'affichage des fichiers du lecteur de disquette A avec un double clic est identique ? C'est une bonne question mais peut-être ATARI a-t-il pensé, pour la fonction 'ouvrir', à tous ceux qui ont du mal à exécuter le double clic ?! En effet, pour faire un double clic, il faut appuyer rapidement deux fois sur la touche supérieure gauche de la souris. En tout cas, il existe donc une seconde possibilité : activer le lecteur de disquette A avec un simple clic puis appeler ensuite la fonction 'ouvrir' dans le second menu déroulant.

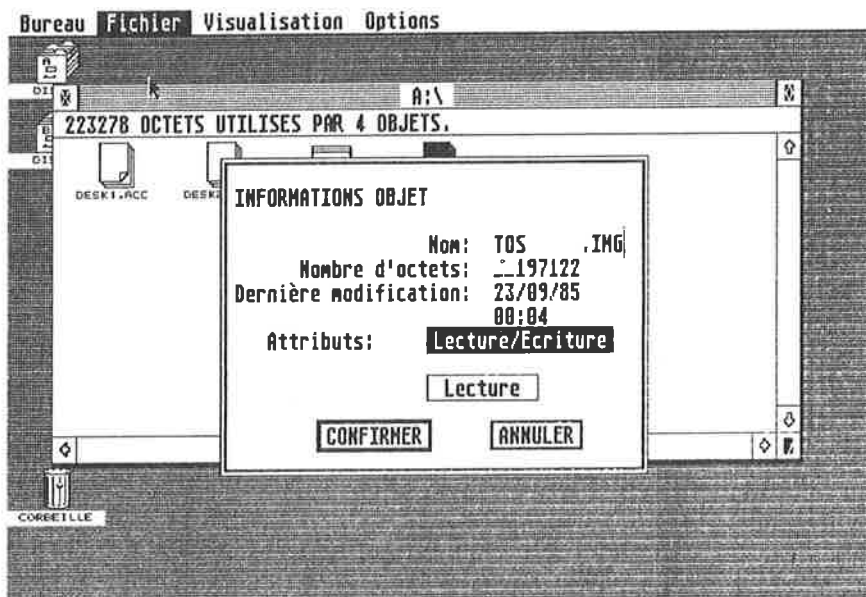
### Visualisation des informations

Activez maintenant le lecteur de disquette A une seconde fois, parcourez le second menu déroulant jusqu'à 'Informations...' (Show Info) et activez cette fonction par un clic, c'est-à-dire en appuyant une fois sur la touche supérieure gauche de la souris. Visualisation infos vous fournit des informations sur le lecteur de disquette (ou sur un fichier si vous avez appelé un fichier).



Vous pouvez apprendre ici comment s'appelle une disquette, combien de classeurs et de fichiers figurent sur la disquette, combien il y a encore de place mémoire libre sur la disquette et combien de place mémoire est déjà occupée par des programmes ou des enregistrements de données.

Vous pouvez uniquement prendre connaissance de ces informations mais vous ne pouvez pas les modifier. Cela se présente déjà autrement si vous activez auparavant un fichier. Faites afficher le catalogue, marquez-y par un clic un symbole de programme ou d'enregistrement de données et parcourez ensuite le second menu déroulant jusqu'à 'Informations...'



Vous pouvez maintenant modifier au clavier le nom de fichier et choisir entre les fonctions 'LECTURE/ECRITURE' (READ/WRITE) et 'LECTURE'(READ ONLY). Confirmez les lignes affichées (comme lors du traitement des informations sur le lecteur de disquette) en vous déplaçant vers le champ CONFIRMER et en "cliquant" cette fonction.

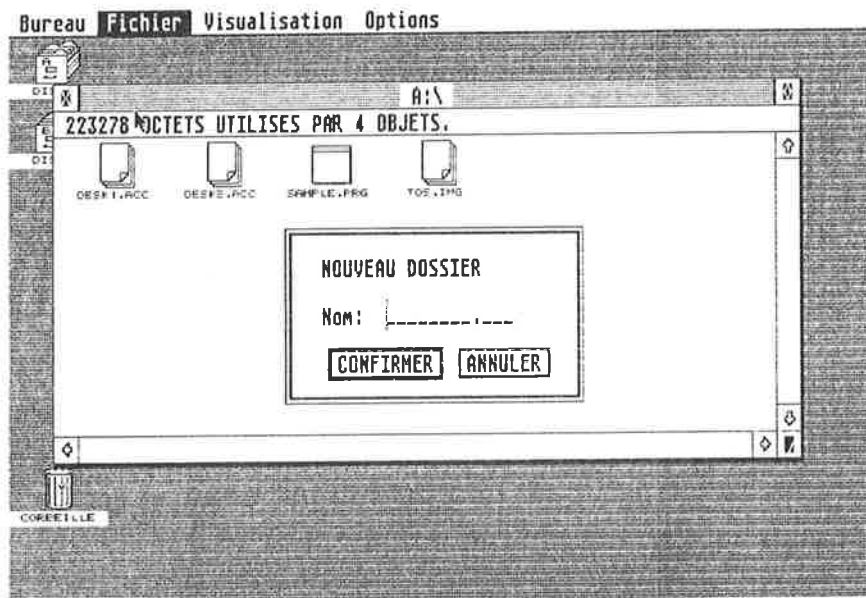
Les 3 fonctions suivantes du second menu déroulant sont 'Nouveau Dossier', 'fermer' et 'fermer fenêtre'.

### Fermer et Fermer fenêtre

Si vous vous souvenez encore de la structure d'une feuille de données (comme précédemment notre catalogue de la disquette avec les fonctions des différents coins), vous pouvez tout à fait passer rapidement sur 'fermer' et 'fermer fenêtre'. Si vous utilisez au lieu de cela l'angle supérieur gauche des écrans d'affichage, les papiers seront rapidement replacés dans le lecteur de disquette/boîte de fiches et ce sera tout.

### Nouveau Dossier

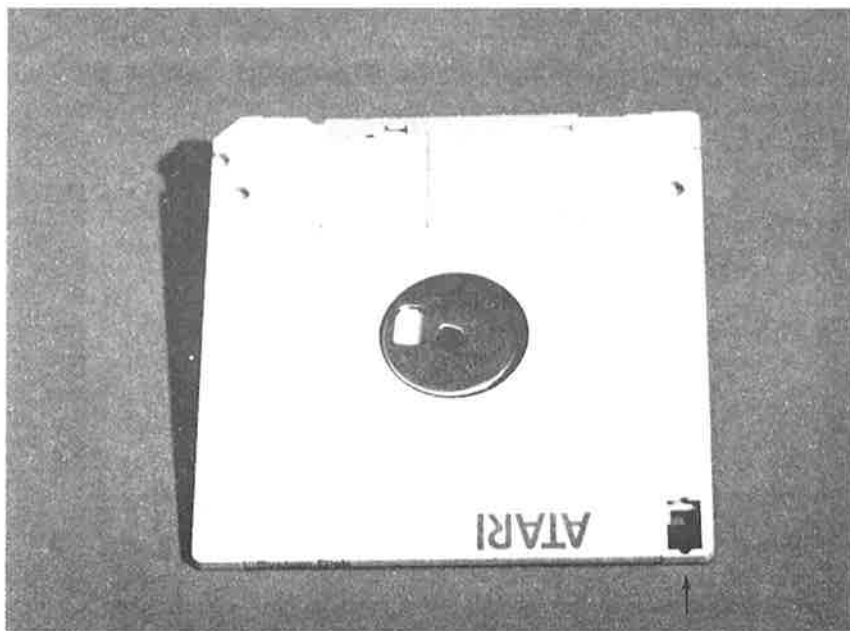
Mais quelle est donc la signification d'un dossier ? Appelez simplement la fonction classeur et attendez de voir ce qui se passe :



On vous demande un nom pour le classeur et vous pouvez ensuite marquer l'affichage CONFIRMER par un clic.

Vous utilisez sans doute encore, pour le moment, l'original de votre disquette système. Cette disquette est protégée contre l'écriture, c'est-à-dire que vous ne pouvez y apporter aucune modification. Après avoir marqué par un clic le champ CONFIRMER, votre ATARI ST se présente à vous, un peu embarrassé, pour vous dire qu'il ne peut pas faire ce que vous exigez de lui. ATARI a prévu cela par mesure de sécurité pour que vous ne rendiez pas inutilisable votre précieuse disquette système, par exemple en effaçant par erreur certaines parties.

La protection contre l'écriture peut être manoeuvrée avec le bouton-poussoir en plastique qui se trouve à l'arrière de la disquette (tout en bas à droite sur la photo).



La disquette est protégée contre l'écriture lorsque le poussoir est tiré vers le bas.

Avant que vous puissiez donc essayer les fonctions de "classeur" suivantes, vous devez faire une copie de votre disquette système ou bien utiliser une disquette vide mais déjà formatée. Un peu déroutant, n'est-ce pas ? Avant d'essayer les fonctions de classeur, lisez donc d'abord la section 2.3 dans laquelle nous vous expliquons comment réaliser une copie de vos disquettes précieuses et comment préparer (=formater) des disquettes vierges pour pouvoir les utiliser.



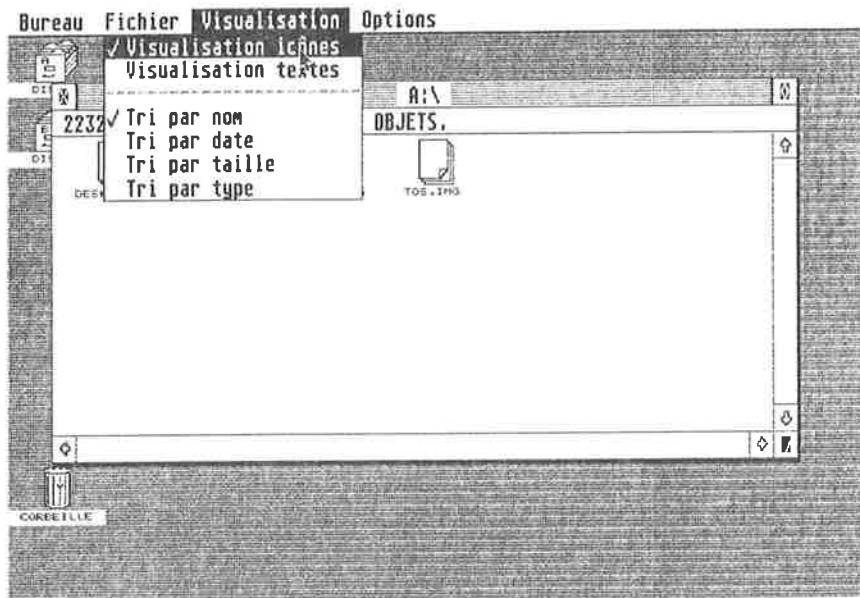
Il faut donc absolument, avant de poursuivre la lecture de cet ouvrage, que vous travailliez désormais avec une copie de votre disquette système ou que vous utilisiez une autre disquette.

Si vous avez maintenant marqué le champ CONFIRMER par un clic et si vous utilisez une disquette non protégée contre l'écriture, vous disposez maintenant d'un classeur et vous pouvez déplacer sur l'écran tous les fichiers de données et les placer, symboliquement, dans ce classeur, comme nous l'avons déjà fait précédemment lorsque nous avons déplacé les objets sur le bureau électronique (lecteur de disquette et corbeille à papier). Amenez donc (en appuyant sur la touche de la souris) quelques programmes dans le classeur, comme vous le souhaitez.

Chaque fois que vous voulez ranger un fichier de données dans un classeur, votre ATARI ST vous interroge auparavant et vous devez cliquer avec un CONFIRMER pour signaler que c'est bien votre intention. Votre ATARI ST veut toujours être sûr. Examinez ensuite, pour une dernière vérification, le contenu du catalogue, après avoir pointé sur lui la flèche et avoir fait un double clic avec la touche gauche de la souris! Si tout est comme il faut, refermez le classeur en 'cliquant' à nouveau dans le coin supérieur gauche. Maintenant, les fichiers de données que vous avez placés dans le classeur figureront cependant deux fois sur votre disquette. Un premier exemplaire dans le classeur et un second là où figuraient ces fichiers auparavant. Il faudrait donc supprimer maintenant ces fichiers originels en les jetant dans la corbeille à papier.

#### 2.3.2.2. Visualisation (View)

Refermons la représentation symbolique et examinons de plus près le menu 3 qui porte le titre 'Visualisation'.



### Index comme icônes ou comme texte

Vous pouvez maintenant sélectionner (et ce sera également affiché ainsi dans le catalogue de la disquette) comment le catalogue doit par exemple se présenter à l'écran ... avec ou sans symboles, trié d'après le nom, la date, la taille ou bien le type de fichier.

Dès que vous avez appelé une fonction, elle est exécutée immédiatement de sorte que vous pouvez voir son effet sur le catalogue affiché peu de temps après avoir marqué d'un clic le point de sélection correspondant.

### Tri d'après le nom, la date, la taille ou les types de fichier

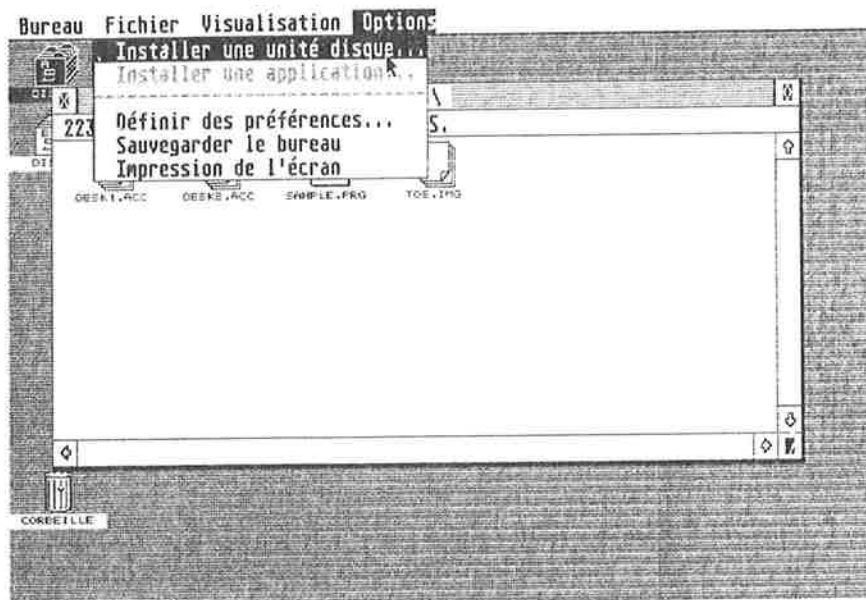
Choisissez vous-même ce qui vous convient, avec texte ou symbole (ce qu'on appelle les icônes) et d'après quel critère doit être fait le tri.

Vous pouvez vous laisser guider dans ce choix par ce qui est réellement affiché dans le menu 'Visualisation' (view). Pour que vous puissiez mieux vous en rendre compte, les développeurs de l'ATARI ST ont prévu que des petits crochets soient placés devant les fonctions déjà exécutées. Bien entendu, vous pouvez supprimer à votre guise ces crochets, par un nouveau clic (la fonction affichée ne s'appliquera plus dans ce cas).

### 2.3.2.3. Options

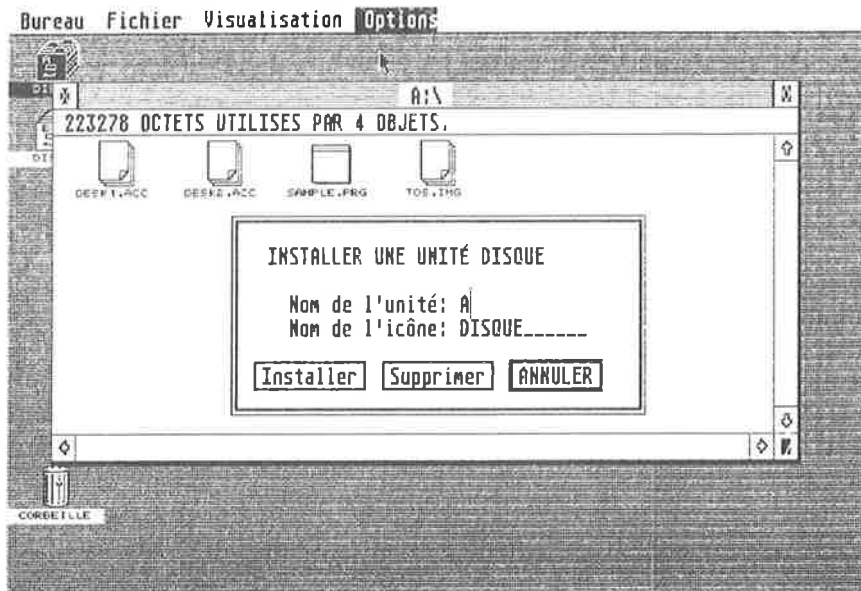
Venons-en maintenant au quatrième menu déroulant, ce qu'on appelle le menu 'options'.

Avant que vous ne le sélectionniez (en dirigeant la flèche tout simplement sur le mot 'options' dans la ligne supérieure de l'écran), il serait préférable que vous appeliez auparavant le lecteur de disquette A (représentation en vidéo inversée que vous obtiendrez simplement par un clic sur le symbole correspondant, avec la touche gauche de la souris). Votre écran devrait maintenant se présenter ainsi :



Installer une unité de disque...

Commençons à nouveau par en haut : 'cliquez' la fonction 1 ('Installer une unité de disque...') :



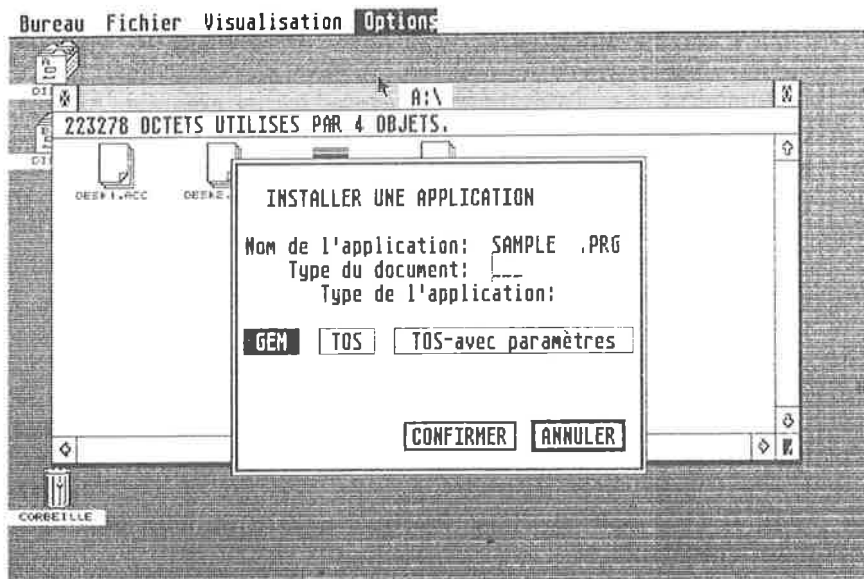
Vous pouvez maintenant modifier l'indication placée sous le symbole graphique (Nom de l'icône - Disque de l'unité pour le moment) ainsi que le nom de l'unité de disquette (A pour le moment).

Il est également possible de 'SUPPRIMER' (ERASE) tout le lecteur de disquette de l'écran (en appelant le champ central); soyez toutefois très prudent avec cette fonction car une fois que vous aurez supprimé les deux lecteurs de disquette, vous ne pourrez plus utiliser votre disquette système d'exploitation (à supposer que vous ayez auparavant 'sauvegardé' votre travail).

Installer une Application...

La fonction 'Installer une application' apparaît très faiblement sur l'écran pour le moment et vous remarquerez que cette fonction ne peut être appelée pour le moment.

Il vous faut pour cela marquer d'abord par un clic un programme du catalogue (chaque programme porte à la fin la marque '.PRG' ou 'TOS' et est représenté par un symbole de bloc) et vous pouvez ensuite déclarer sous 'options' l'utilisation voulue.



Qu'est-ce que cela signifie ?

Notre ATARIST peut non seulement traiter des programmes commandés avec la souris (la partie du système d'exploitation appelée GEM se charge de cela) mais aussi des programmes sans cette interface graphique fabuleuse. GEM (ayez toujours présent à l'esprit qu'il s'agit de la partie du système d'exploitation qui se charge de guider l'utilisateur au moyen de symboles graphiques) peut donc être en quelque sorte 'déconnecté' pour que le programme fonctionne "seulement" avec le système d'exploitation TOS.

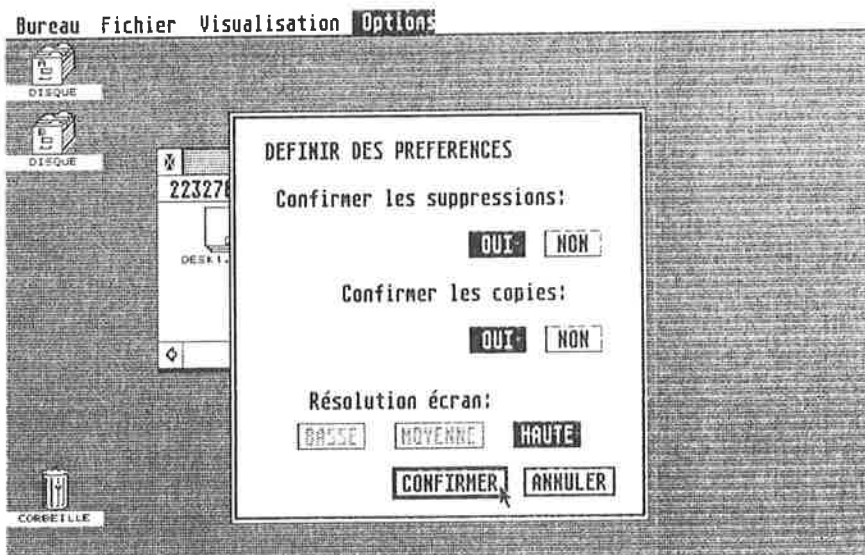
Pour les programmes devant être lancés directement (marqués par un .PRG à la fin ou bien faciles à distinguer des autres images ou icônes parce qu'ils sont représentés par un bloc), nous pouvons sélectionner, avec le point 'Installer une application...' sous quel système d'exploitation notre programme doit être lancé : GEM, TOS ou 'TOS avec paramètres'; c'est ce qu'on appelle sélectionner le mode d'utilisation.

Même si certaines choses restent un peu difficiles à comprendre pour vous, vous comprendrez mieux de quoi nous parlons au fur et à mesure de votre travail sur l'ATARI ST. Nous vous conseillons de relire à ce moment-là les explications de ce chapitre.

Nous pouvons, en outre, entrer également maintenant le 'type de document', c'est-à-dire que vous ne pouvez pas seulement lancer automatiquement les programmes portant la terminaison '.PRG', '.TOS' ou '.TTP' mais aussi des fichiers de données disposant de la terminaison entrée dans le champ 'type de document'. Ces fichiers de données chargeront en effet d'abord, après un double clic, le programme proprement dit (dont le nom se termine par '.PRG', '.TOS' ou '.TTP') avant qu'ils soient eux-mêmes chargés automatiquement par le programme correspondant dans le ST.

Sur une disquette programme LOGO, vous pouvez par exemple entrer 'LOG' dans le champ 'type de document': si vous appelez ensuite un programme LOGO comportant la terminaison '.LOG', 'LOGO.PRГ' sera d'abord chargé et ce n'est qu'ensuite que le fichier de données actuel sera chargé de la disquette dans la mémoire de l'ordinateur. Le menu d'options nous offre encore 3 autres options que nous expliquerons brièvement :

### Définir des Préférences...



Si vous voulez supprimer un fichier, on vous demande normalement, auparavant, de confirmer encore une fois que c'est bien ce que vous voulez faire (interrogation de sécurité). Votre ST procède de même lorsque vous voulez copier un fichier ou une disquette entière.

Si ces interrogations perpétuelles vous tapent sur les nerfs et si vous pensez avoir suffisamment d'assurance dans le maniement des fonctions de suppression ou de copie, 'cliquez' tout simplement le champ 'NON' pour 'confirmer suppression' ou 'confirmer copie'.

Vous économiserez ainsi une étape préalable lors des opérations de copie ou de suppression mais n'oubliez pas que le risque de commettre une erreur fatale sera également beaucoup plus grand !!!

Si vous possédez un moniteur noir et blanc, vous ne pouvez appeler que 'HAUTE' (HIGH) pour le réglage de la résolution de l'écran, les champs 'BASSE' (LOW) et 'MOYENNE' (MIDDLE) sont bloqués.

Lorsque vous connectez un moniteur couleur à votre ST, c'est exactement le contraire, vous ne pouvez choisir qu'entre les résolutions faible ou moyenne; la haute résolution ne peut être appelée sur un moniteur couleur car elle ne serait pas représentée proprement.

Vous confirmez les entrées effectuées en "cliquant" la touche CONFIRMER ou bien vous interrompez cette opération en appelant la touche 'interruption'.

### *Sauvegarder le bureau*

Toutes les modifications que nous avons apportées aux options n'ont qu'une portée limitée à la seule mémoire de l'ordinateur. Si l'on recharge le système (par exemple en déclenchant un RESET), le ST va oublier les modifications et utiliser la configuration telle qu'elle se trouve inscrite sur la disquette.

Si vous voulez que vos modifications soient activées à chaque chargement du système, il faut appeler le point 'Sauvegarder le bureau' du menu 'Options'. Les infos correspondantes sont alors

enregistrées sur la disquette dans un fichier appelé 'DESKTOP.INF', qui sera lu à chaque chargement du système.

### Impression de l'écran

Cette fonction parle d'elle-même. Si vous avez connecté une imprimante graphique sur le connecteur Centronics (connecteur 8 du ST, voir photo en chapitre 1.1), et que cette imprimante a été correctement installée dans le menu 'Bureau', alors l'appel de cette fonction lance une impression sur papier d'une reproduction de ce qui se trouve sur l'écran à cet instant.

Le même effet peut être obtenu en pressant simultanément les touches 'Alternate' et 'Help' avec l'avantage supplémentaire de ne pas avoir le menu 'Options' déroulé à l'écran.

Si vous n'avez pas d'imprimante connectée au ST, l'appel de cette fonction vous fera quand même attendre 30 seconde avant que la machine ne vous rende la main.

### Blitter

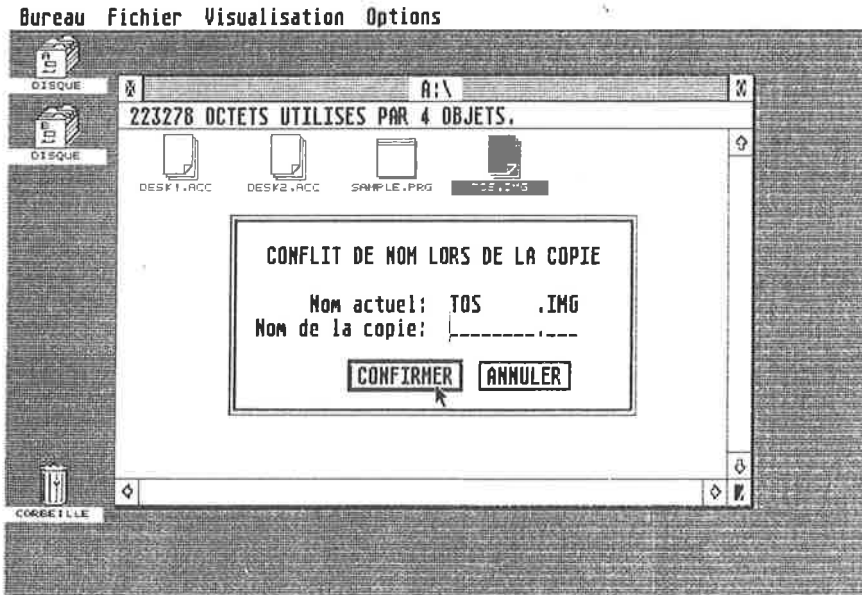
Si votre ST est équipé du TOS Blitter et du Blitter lui-même, alors ce point supplémentaire figure dans le menu 'Options'. Il permet d'activer ou de désactiver le Blitter, l'activation étant visualisée par un petit marqueur. Lorsque le Blitter est activé, de nombreuses opérations se trouvent accélérées, par exemple la mise en place à l'écran d'une fenêtre pour le catalogue. Normalement le Blitter pourra rester activé en permanence ; seuls des jeux s'exécutant trop rapidement, ou quelques applications ne fonctionnant pas du tout pourront demander une désactivation du Blitter.



## 2.4. COPIE ET SUPPRESSION DE PROGRAMMES ET DE FICHIERS

Pour copier isolément des programmes ou fichiers de données, marquez par un clic un symbole du catalogue affiché et placez-le sur le symbole du lecteur de disquette B ou sur son catalogue. Suivant une procédure de messages très proche de celle de la copie de disquettes entières, on vous demande maintenant de placer l'original dans le lecteur de disquette A et la disquette sur laquelle doit être placée la copie dans le lecteur de disquette B.

Vous pouvez, de la même façon, recopier également des programmes ou fichiers de données sur la même disquette. Votre ST vous conseille toutefois auparavant, dans ce cas, de changer le nom du fichier ou programme.



Vous pouvez procéder de la même manière pour la suppression de fichiers, en employant la corbeille à papier : vous faites afficher le catalogue de la disquette et vous amenez alors le programme ou fichier de données devenu inutile 'dans' la corbeille à papier (en réalité, vous placez le symbole du fichier sur la corbeille à papier).

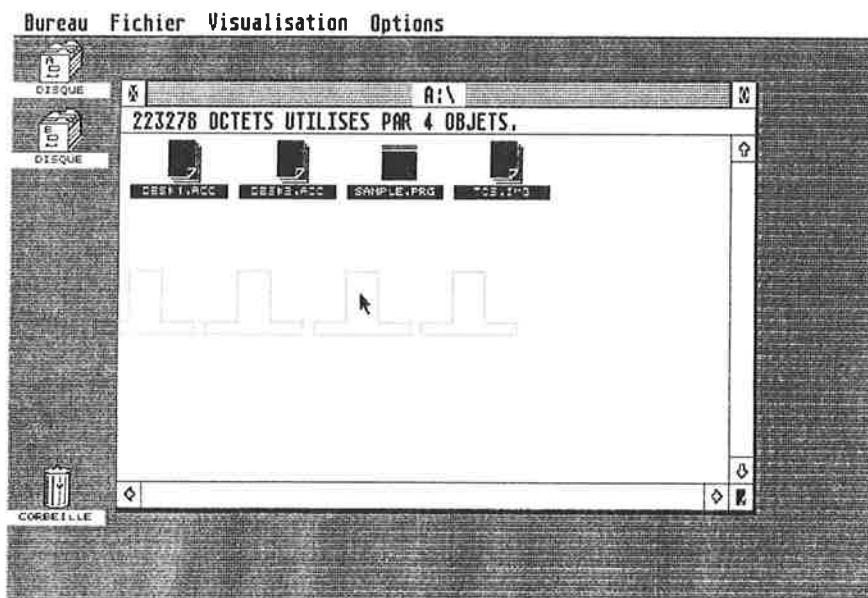
Si vous n'avez pas déconnecté auparavant l'interrogation de sécurité, on vous avertira encore une fois de la puissance de cette fonction, sinon la suppression commencera immédiatement.

Encore un mot sur la corbeille à papier. Lorsque vous jetez chez vous quelque chose dans la corbeille à papier, vous pouvez toujours, si vous vous y prenez à temps, aller fouiller dans cette corbeille pour y repêcher un morceau de papier que vous y auriez jeté à tort. Cela n'est pas possible avec l'ATARI ST. Ce que vous jetez dans la corbeille à papier disparaît pour toujours. Donc prudence lorsque vous utilisez la corbeille à papier !

Ne pouvons-nous supprimer ou copier que des programmes ou fichiers de données isolés ou cela est-il également possible avec un nombre plus important de fichiers ?

Vous pouvez bien sûr supprimer simultanément plusieurs fichiers. Il vous faut, pour cela, tracer un cadre, en appuyant et en tenant enfoncée la touche de la souris (en commençant par en haut), autour des fichiers à copier ou à supprimer (comme avec notre coin en caoutchouc dans l'angle inférieur droit de notre fenêtre d'affichage).

Lorsque vous voudrez ensuite déplacer un de ces fichiers pour le copier ou le supprimer, l'ensemble des fichiers encadrés suivront.



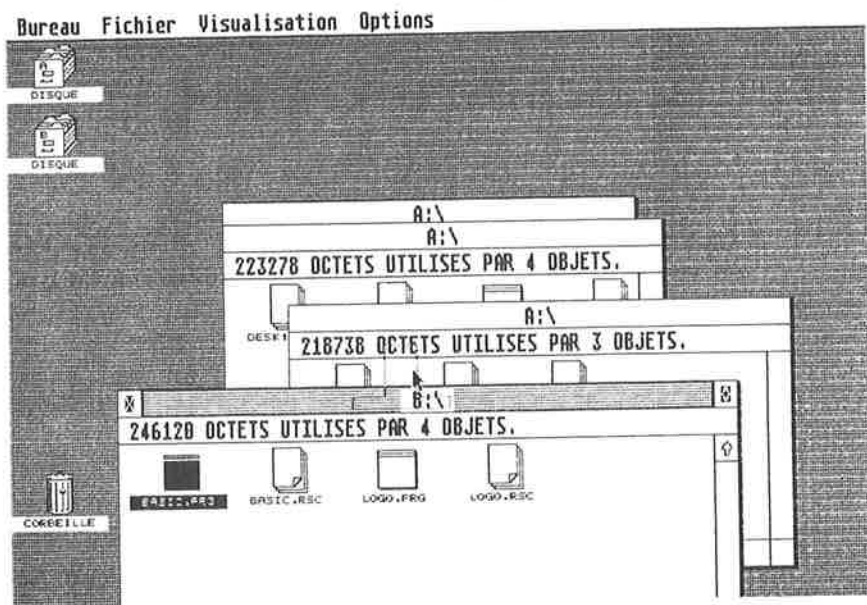
Toutefois, vous pouvez également sélectionner l'un après l'autre les fichiers voulus en tenant simultanément enfoncée la touche SHIFT. Il faut donc tenir enfoncée la touche SHIFT et marquer par un clic différents fichiers de données les uns après les autres. Ici aussi, tous les fichiers suivent lorsque vous voulez en déplacer un. Notez par ailleurs que la fonction de sélection avec le cadre caoutchouc ou en marquant successivement par un clic les fichiers voulus peut également être employée pour la création de classeurs.

Voilà ce que nous avons à dire sur le travail avec le système d'exploitation GEM TOS. Nous espérons que cette section du livre vous aura aidé à dépasser certaines difficultés d'emploi. En effet, si l'interface GEM de l'ATARI ST est essentiellement orientée en fonction des besoins de l'utilisateur, il n'en reste pas moins qu'il faut d'abord apprendre à utiliser correctement ses différentes fonctions avant de pouvoir profiter pleinement de ses avantages.

## 2.5. LE ST EST LE ROI DES FENETRES !

Nous n'avons jusqu'ici ouvert qu'une seule fenêtre sur l'écran, par exemple pour afficher le catalogue de la disquette actuelle sur l'écran. Le système d'exploitation GEM que vous avez auparavant chargé de la disquette dans votre ST peut cependant représenter simultanément jusqu'à quatre fenêtres différentes.

Pour parvenir à cela, nous vous demandons de faire sortir, par quatre fois consécutives, le catalogue du lecteur de disquette A, par un double clic. Notez bien qu'il ne faut pas, bien sûr, que vous refermiez chaque fois la fenêtre affichée, par un clic dans le coin supérieur gauche, avant d'ouvrir la prochaine fenêtre ; l'intérêt est, en effet, de placer par quatre fois un écran sur un autre.



Votre écran/bureau a maintenant l'air de déborder. Si vous regardez bien, vous remarquerez que votre ST a exécuté le fait de feuilleter quatre catalogues exactement comme si vous aviez placé sur votre bureau quatre feuilles se recouvrant l'une l'autre partiellement.

Vous pouvez savoir quelle est pour le moment la feuille actuellement prise en compte par l'ordinateur par le fait que toutes les informations de cadre y sont visibles (par exemple coin de fermeture, etc.) alors que les trois autres feuilles apparaissent 'sans cadre'. Ces trois feuilles sont cependant également délimitées, par rapport aux autres, par des traits mais vous ne pouvez y voir la zone du cadre qui se reconnaît à ses coins de fonction.

L'affichage de catalogue sorti en dernier sur l'écran est donc l'affichage actuel, le seul qui puisse pleinement fonctionner. Vous pouvez utiliser ici la zone du cadre exactement comme vous l'avez fait précédemment avec un seul catalogue représenté à l'écran. Il nous est ainsi possible par exemple d'étirer ou de comprimer le catalogue par un clic dans l'angle inférieur droit (coin caoutchouc). Nous pouvons également amener ce catalogue dans un autre emplacement de l'écran ou le refermer par un clic dans le coin supérieur gauche.

Vous pouvez donc faire avec le catalogue actuel tout ce qui peut vous passer par la tête.

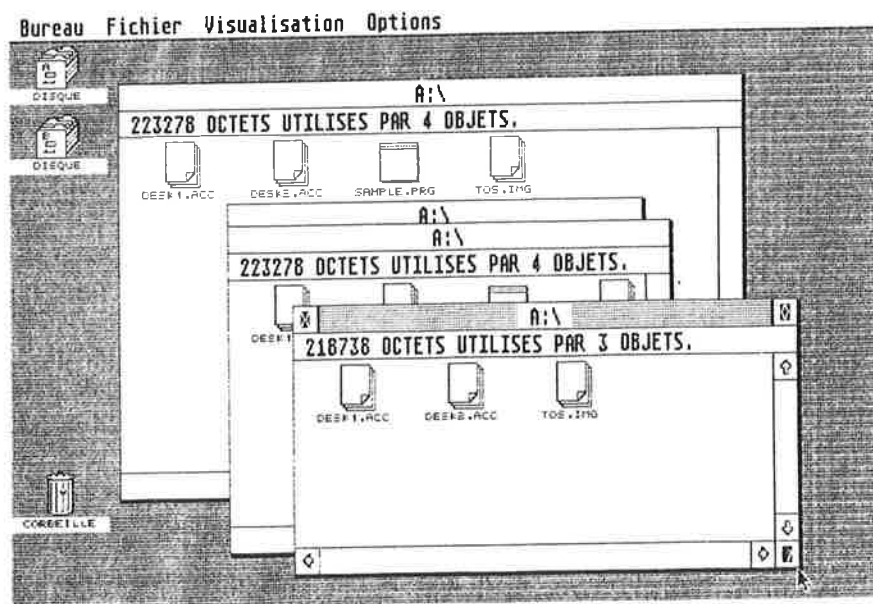
Vous avez certainement remarqué, dans l'emploi des quatre coins ou de la ligne supérieure de déplacement, que votre ST fonctionne de façon très pratique et très proche de la vie quotidienne : il a noté toutes les informations qui se trouvent dessous le catalogue actuellement représenté. C'est donc comme dans la vie réelle : les papiers placés en bas sont d'abord recouverts par celui qu'on place dessus mais si vous retirez le papier de dessus, ceux qui se trouvent dessous redeviennent partiellement ou totalement visibles.

Votre ATARI ST consacre à cette fonction une grande quantité de places mémoire car c'est pour lui la seule façon de retenir en détail l'ensemble des données graphiques. Il ne retient pas seulement comment se présente la feuille de papier qui se trouve directement sous le catalogue actuel mais aussi l'apparence de jusqu'à trois autres catalogues.

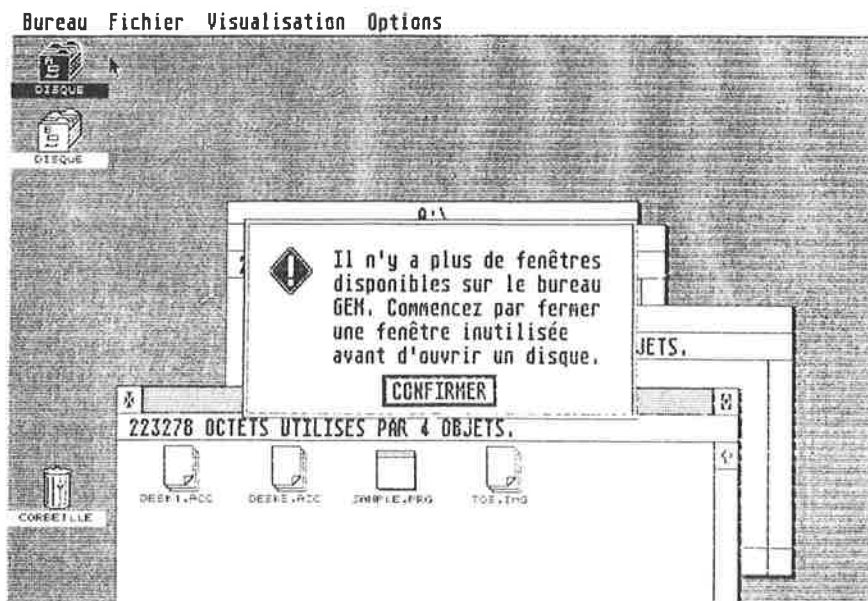
### 2.5.1. Placer dessus celui qui est tout en dessous

Nous allons maintenant essayer de voir s'il est possible de sortir de la pile de papiers un catalogue placé tout en bas. Rien de plus simple : amenez la flèche, avec la souris, directement sur un endroit quelconque d'une autre feuille de catalogue et faites un clic avec la touche de la souris. Le catalogue ainsi marqué sera alors instantanément représenté sur l'écran, complètement déployé, comme ce serait le cas sur un véritable bureau :

Vous trouvez une feuille en bas de la pile de documents que vous avez devant vous, vous prenez cette feuille et vous la posez sur le tas pour mieux la lire.



Notre ST ne travaille pas autrement, quel que soit le contenu actuel de la disquette. Après avoir représenté quatre fenêtres sur l'écran, si vous tentez d'ouvrir une cinquième fenêtre, un message vous indiquera que l'ordinateur ne peut malheureusement accéder à votre désir.

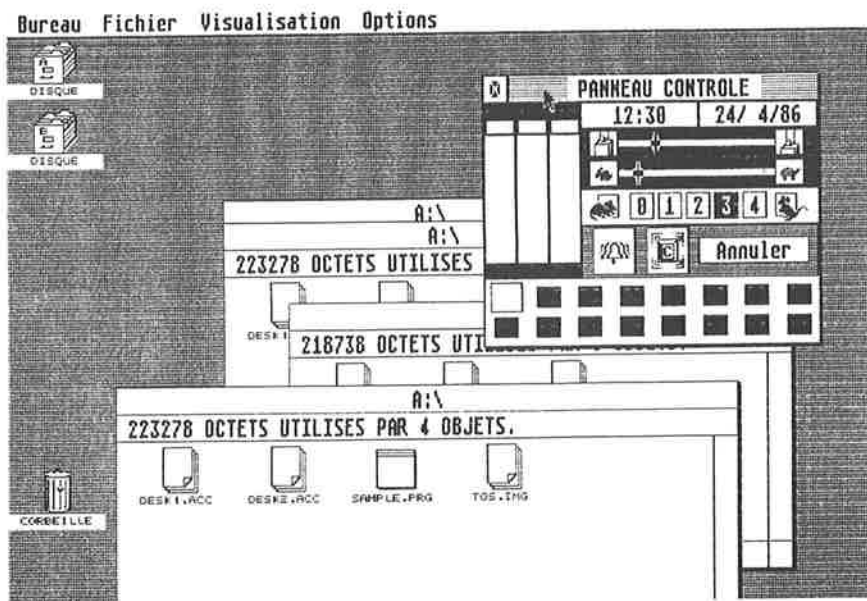


N'en veuillez pas trop à votre ST de se limiter à quatre fenêtres. N'oubliez pas en effet que toutes les images graphiques que votre ST doit retenir consomment énormément de place mémoire.

Vous vous demanderez peut-être maintenant à quoi peut donc nous servir d'avoir simultanément sur notre écran de moniteur plusieurs écrans ou plusieurs catalogues.

### 2.5.2. Cela marche pourtant : cinq fenêtres

Avant que nous ne répondions à cette question, sachez encore qu'il est malgré tout possible de représenter simultanément cinq fenêtres (windows) sur l'écran de votre moniteur. Pour le prouver, ouvrez d'abord 4 fenêtres, essayez, en vain, d'en ouvrir une cinquième et marquez enfin par un clic une des fonctions du 'Bureau' ('Desk'). Le Panneau contrôle vous est alors proposé instantanément. Ce champ de contrôle constitue bien une cinquième fenêtre.



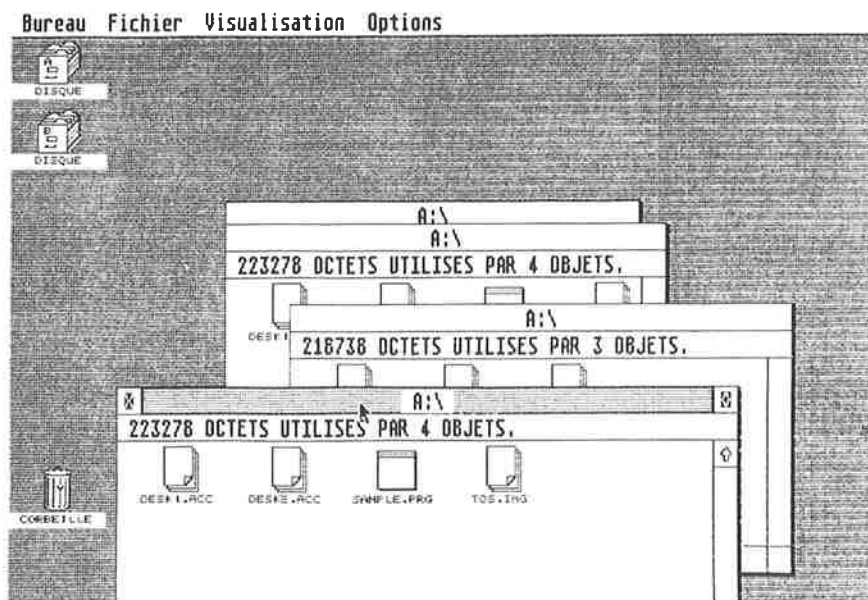
Dans certaines conditions, votre ST peut donc bien afficher encore une cinquième fenêtre qui vous permettra par exemple d'effectuer sans problème les réglages nécessaires pour les interfaces de l'ordinateur.

### 2.5.3. Copie d'une fenêtre dans l'autre

Mais quel est donc l'intérêt d'avoir quatre ou cinq fenêtres représentées simultanément ? Il n'y en a bien sûr aucun si chaque fenêtre représente le même catalogue sur l'écran, comme dans l'essai que nous venons de faire. Mais imaginez donc le cas suivant : vous placez une disquette dans le lecteur de disquette, et vous faites afficher son catalogue par un double clic. Vous placez alors dans le lecteur une autre disquette et vous faites également afficher son catalogue.



Vous avez peut-être sauvegardé sur la seconde disquette un programme ou des données dont vous auriez absolument besoin sur la première disquette. Pas de problème : vous n'avez qu'à amener ce fichier sur le catalogue de votre première disquette, en tenant enfoncé le bouton de la souris.



Cela suppose toutefois que vous ayez auparavant produit le premier catalogue par un double clic sur le lecteur de disquette A et le second par un clic sur le lecteur de disquette B.

C'est ainsi que votre ST saura de quel lecteur de disquette à quel autre doit se faire la copie. Il vous demandera ensuite, si vous n'avez qu'un lecteur de disquette, d'échanger à plusieurs reprises les disquettes dans le lecteur de disquette A. Le ST suppose en effet toujours, pour éviter toute confusion, qu'il y a deux lecteurs de disquette, même quand vous n'en possédez qu'un en réalité. Il fait alors comme s'il copiait d'un lecteur de disquette dans l'autre.

Insérez simplement la disquette contenant le fichier à copier dans le lecteur A, et placez dans le lecteur B la disquette sur laquelle vous voulez copier ce fichier. Le reste est exécuté automatiquement par le ST.

Il existe de nombreux programmes d'application pour le ST utilisant jusqu'à quatre fenêtres simultanément à l'écran, comme les tableurs (par exemple K-SPREAD), les gestionnaires de fichiers (par exemple DATAMAT ST), les traitements de texte (par exemple 1St Word) ou les logiciels de création graphique (par exemple PLUSPAINT ST).

Cette possibilité est entre autre très intéressante pour examiner et comparer des données provenant d'origines diverses.

Au plus tard lorsque nous allons commencer à programmer le ST en BASIC vous verrez comme il est pratique de disposer de plusieurs fenêtres.

## 2.6. LES ICONES

Nous avons déjà évoqué ces symboles graphiques apparaissant à l'écran dans un catalogue, appelés icônes. Il en existe trois types : le classeur (\*), le calepin (\*) et la pile de feuilles cornées (\*).

### 2.6.1. Le classeur

Nous avons déjà expliqué le mode de fonctionnement des classeurs mais il nous reste à décrire les blocs avec souche et les piles de papiers avec coin recourbé.

### **2.6.2. Le bloc avec souche**

La représentation d'un bloc dans notre catalogue comporte, dans son nom, soit l'abréviation 'PRG' (programme), 'TOS' (Tramiel Operating System) ou 'TTP' (TOS Takes Parameters). Cela signifie qu'il s'agit d'un programme qui exécute sa fonction, soit sous le système d'exploitation GEM (avec jusqu'à quatre fenêtres différentes et avec commande par la souris), l'abréviation finale est alors 'PRG', soit sans cette aide, l'abréviation finale est dans ce cas 'TOS' ou 'TTP'.

Vous pouvez lancer des programmes directement en les marquant avec la flèche de la souris et en appuyant ensuite par deux fois consécutives (double clic) sur le bouton gauche de la souris.

Une fois que le symbole de programme marqué par un clic s'est coloré en noir, la flèche de la souris se transformera en une abeille travailleuse (busy bee) et l'écran se videra.

Quelques secondes plus tard, ou peut-être seulement une minute plus tard s'il s'agit d'un long programme, le programme chargé sera lancé. Vous pourrez reconnaître que le programme a été lancé par exemple au fait qu'une nouvelle ligne de menu apparaisse dans la première ligne de l'écran.

### **2.6.3. Pile de papiers avec coin recourbé**

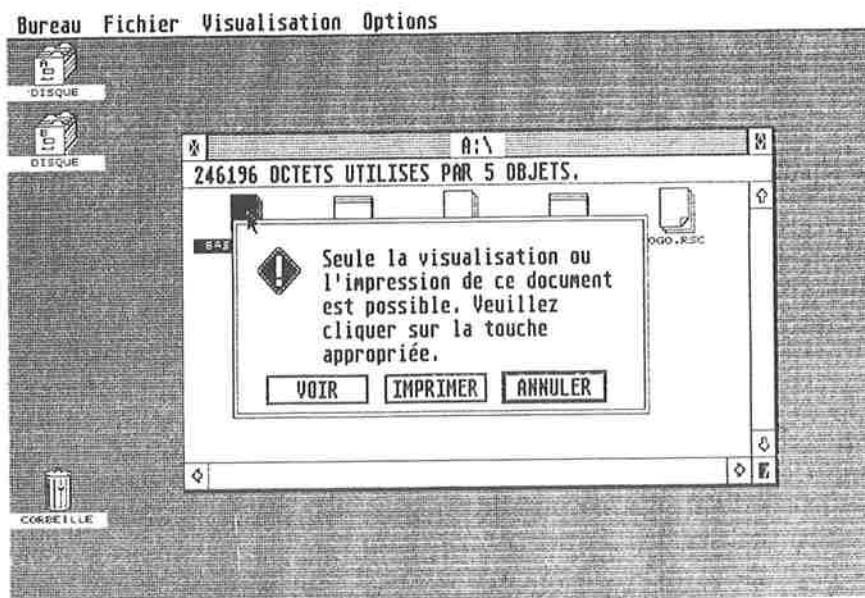
Alors que le bloc représente un programme, le symbole pile de papiers avec coin recourbé représente un fichier.

Qu'est-ce qu'un fichier ? Un fichier peut être aussi bien un programme BASIC ou LOGO (par programme, nous n'entendons pas ici un programme tournant avec le système d'exploitation ATARI mais par exemple un programme pouvant être exécuté en BASIC) qu'un texte sauvegardé par un programme de traitement de texte ou un ensemble d'adresses sauvegardées auparavant par un programme de gestion de fichier.

Il est cependant également possible qu'un fichier soit constitué par des sous-programmes complexes, écrits par exemple en langage machine et devant être chargés automatiquement par un programme principal (représenté par un bloc) lorsque celui-ci sera lancé. Un fichier peut enfin contenir l'image d'un écran, par exemple lorsque vous utilisez un programme de dessin. Ce dernier type de fichiers sera souvent sauvegardé avec l'abréviation 'PIC' (pour picture = image). De tels fichiers ne peuvent d'autre part permettre de reconstituer une image qu'avec le programme de dessin utilisé lors de leur sauvegarde.

Cherchez une pile de papiers avec coin recourbé sur votre disquette BASIC. Si cette pile comporte dans son nom de fichier l'abréviation 'BAS', il y a de très fortes chances pour qu'il s'agisse d'un programme BASIC. Marquez alors ce fichier par un double clic sur la souris.

Votre ST réagit instantanément et affiche maintenant à l'écran les informations suivantes :



Votre ST a donc remarqué de lui-même que le fichier marqué par un double clic n'est pas un fichier pouvant être lancé automatiquement. Il vous offre alors trois possibilités de réaction.

Vous pouvez remettre en cause l'action entreprise (double clic sur un fichier) en marquant d'un clic le champ 'ANNULER'.

Vous pouvez également faire sortir le contenu du fichier sur l'écran du moniteur ou sur l'imprimante connectée à votre ST. Il vous faut, dans ce dernier cas, marquer par un clic le champ 'IMPRIMER' (PRINT).

Nous allons donc examiner ensemble sur l'écran le contenu du fichier. Marquez pour cela par un clic le champ 'VOIR' (VIEW). Différentes choses peuvent alors se produire.

Votre écran peut ressembler à ceci :

```
Td$vr  ( Desk File Run Edit Debug About ST Basic -----de
--FIN DE FICHIER--■
```

ou à cela ?

```
10  input "CODE:";a
20  ?a=="chr$(a)
30  ?(a+1)"=="chr$(a+1)
40  ?(a+2)"=="chr$(a+2)
50  ?(a+3)"=="chr$(a+3)
60  ?(a+4)"=="chr$(a+4)
70  ?(a+5)"=="chr$(a+5)
80  ?(a+6)"=="chr$(a+6)
90  ?(a+7)"=="chr$(a+7)
100 ?(a+8)"=="chr$(a+8)
110 ?(a+9)"=="chr$(a+9)
120 goto 10
```

```
--FIN DE FICHIER--■
```

Dans le premier cas, nous avons visiblement affaire à un fichier qui sera chargé automatiquement par un programme principal (représenté par un bloc) lors du chargement de ce dernier, il s'agit donc peut-être d'un programme en langage machine, à moins que nous ayons affaire à une image sauvegardée antérieurement et qui ne peut être affichée sur l'écran du moniteur de façon reconnaissable que par un programme de dessin. C'est pourquoi, lorsque nous tentons de sortir directement le contenu de tels fichiers, comme vous l'avez peut-être remarqué à l'instant, le résultat n'est rien d'autre qu'un jargon incompréhensible que seul l'ordinateur peut décoder et donc utiliser.

Vous avez cependant peut-être obtenu une sortie de texte sur l'écran, comme dans notre second exemple. Vous voyez peut-être même, dans la ligne inférieure de l'écran, la remarque '—SUITE—' (MORE) qui signifie : vous n'avez vu jusqu'ici qu'une partie du fichier et pas plus. Si vous appuyez après cela sur la touche en longueur qui est située sur le bord inférieur du clavier, la touche espace, la page suivante du contenu du fichier sera affichée à l'écran. Si vous appuyez toutefois sur la touche ENTER ou RETURN, seule la ligne suivante du fichier marqué par un clic sera représentée sur l'écran du moniteur. Dans ce cas, l'image "lisible" sur le moniteur vous indique que vous avez vraiment sélectionné dans le catalogue et fait afficher sur l'écran du moniteur un programme BASIC (abréviation finale 'BAS'), un programme LOGO (abréviation finale 'LOG') ou bien encore un texte produit par un programme de traitement de texte (abréviation finale 'DOC' le plus souvent).

Si vous voyez déjà apparaître à l'écran '—FIN DU FICHIER—' (END OF FILE), vous pouvez revenir au bureau, qui nous est maintenant familier et à partir duquel nous avons appelé cette image, en appuyant sur n'importe quelle touche. Si la fin de la représentation du fichier ne s'annonce pas encore, c'est-à-dire si l'affichage —SUITE—, dans la dernière ligne de l'écran, vous demande à nouveau d'actionner la touche espace, ENTER ou RETURN, vous pouvez malgré tout interrompre l'opération en appuyant simultanément sur les touches 'Control' et 'C'. Le catalogue de la disquette placée dans votre lecteur de disquette devrait alors maintenant apparaître à nouveau sur votre écran.

## 2.7. LE CLAVIER

Vous êtes certainement déjà impatient de pouvoir enfin lancer un programme sur votre ordinateur. Nous allons cependant examiner rapidement auparavant comment est construit le clavier du ST. Mais pour que nous puissions voir en même temps ce qui se passe lorsque nous appuyons sur une touche, nous allons lancer un programme.

Vous avez certainement déjà entendu parler du langage de programmation BASIC. Vous savez certainement déjà également que votre ATARI ST "parle", lui aussi, ce langage de programmation. Il ne le parle cependant pas systématiquement. Encore faut-il, en quelque sorte, que nous chargions le vocabulaire nécessaire pour cela dans sa mémoire.

C'est ce que nous allons faire maintenant. Notez toutefois qu'il est possible, au moment où vous lirez ces lignes, que le langage de programmation BASIC soit déjà "intégré" sur votre ST et que vous n'ayez donc plus besoin de le charger tout d'abord à partir de la disquette. Nous vous invitons, dans ce cas, à vous reporter aux explications qui vous seront données dans votre manuel pour le lancement du BASIC.

Placez donc dans votre lecteur de disquette la seconde disquette fournie avec l'ATARI ST, "LANGUAGE DISK" (disquette de langages). Faites-en afficher le catalogue (double clic sur le symbole du lecteur de disquette) et lancez le programme "BASIC.PRG" par un double clic.

Peu de temps après, l'écran se modifie et vous obtenez une image comportant quatre fenêtres. Ne vous occupez pas de cette présentation pour le moment car nous voulons nous contenter, dans un premier temps, d'essayer le clavier de votre ST.

Comme votre ATARI ST peut fonctionner dans de nombreux cas sans le clavier (c'est justement le but de la commande par souris avec les champs de décision et les menus pull down), la plupart des touches de votre clavier ne seront pas utilisées dans tous les programmes.

Même lorsqu'aucun caractère n'est affiché à l'écran, c'est-à-dire lorsqu'une entrée au clavier n'est pas possible, vous pouvez entendre un clic ou même un clic répété chaque fois qu'une touche est frappée mais, dans ce cas, les entrées au clavier sont sans effet du fait qu'elles ne peuvent être représentées.

Mais que se passe-t-il avec les différentes touches lorsqu'une sortie écran est possible, par exemple lorsque vous avez chargé dans votre ST un programme de traitement de texte ?

Cela non plus ne peut être fixé de façon définitive car ATARI a laissé, de différentes manières, au développeurs de programmes la latitude de définir l'affectation des touches en fonction de ce qui est intéressant pour le programme actuellement exécuté.

Nous ne pouvons donc décrire chaque touche de votre ATARIST comme si sa fonction était toujours la même. C'est malgré tout possible dans de nombreux cas.

Nous avons donc chargé le langage de programmation BASIC et nous pouvons maintenant faire des entrées également au clavier. Ce que vous entrez apparaît maintenant dans la fenêtre COMMAND (fenêtre d'instructions), en bas à gauche.

Le clavier de votre ST se compose de quatre blocs :

- 1) machine à écrire
- 2) touches de fonction
- 3) touches d'édition
- 4) bloc numérique (comme sur une calculatrice de poche)

### 2.7.1. La machine à écrire

Vous disposez ici de toutes les lettres de l'alphabet en majuscules et minuscules et de toute une série de caractères spéciaux. Pour écrire sur l'écran les minuscules, il vous suffit d'appuyer sur la touche de la lettre correspondante. Pour les majuscules, vous devez appuyer sur la touche de la lettre correspondante en tenant en même temps enfoncée une des deux touches SHIFT.



#### 2.7.1.1. Alternate

Un certain nombre de touches comportent l'indication de sorties supplémentaires possibles. La fonction marquée en bas peut être sortie sur l'écran en appuyant sur la touche en même temps que sur la touche 'Alternate'. La fonction marquée en haut peut être sortie en appuyant simultanément sur la touche de la lettre, sur SHIFT et sur 'Alternate'.

Essayez-vous à l'emploi de 'Alternate' ou 'Alternate' et 'SHIFT' avec différentes touches; dans un certain nombre de programmes, ces combinaisons de touches vous permettront de sortir à l'écran des caractères spéciaux.

#### 2.7.1.2. Control

Plusieurs touches de votre clavier de machine à écrire disposent en outre d'une cinquième possibilité de sortie qui est obtenue en actionnant ces touches en même temps que la touche 'Control'. Dans un certain nombre de programmes vous pouvez par exemple provoquer une interruption du programme avec la combinaison de touches 'Control-C' ou 'Control-G'. C'est le cas par exemple avec les langages de programmation LOGO ou BASIC.

Nous vous invitons à consulter sur ce sujet l'annexe A.

#### 2.7.1.3. SHIFT

Lorsque vous appuyez, en même temps que sur SHIFT, sur une autre touche que les lettres, c'est la fonction représentée dans la moitié supérieure de la touche correspondante qui sera sortie.

#### 2.7.1.4. Caps Lock

La touche qui se trouve à droite de la touche espace et qui porte la marque 'Caps Lock' est un commutateur : si vous enfoncez cette touche une fois, vous pouvez écrire en majuscules sur l'écran sans avoir à tenir enfoncée la touche SHIFT en même temps que vous appuyez sur les touches de lettres.

Si vous appuyez une seconde fois sur la touche 'Caps Lock', la situation de départ est rétablie: avec SHIFT = majuscules, sans SHIFT = minuscules.

#### 2.7.1.5. Tab

La touche qui se trouve dans la moitié gauche du clavier et qui est marquée 'Tab', ce qui signifie 'Tabulateur', vous permet de faire avancer le curseur actuel sur l'écran de quelques caractères vers la droite. Le curseur marque la position d'écriture à l'écran. Il est le plus souvent représenté par un trait vertical ou par un bloc noir.

Cela peut être très intéressant par exemple lorsque vous réalisez des tableaux et lorsque les nombres sortis doivent être placés directement les uns sous les autres.

#### 2.7.1.6. Esc

La touche Esc (escape = s'échapper) a dans de nombreux programmes une fonction de retour en arrière. Par exemple, dans de nombreux programmes de traitement de texte, vous pouvez revenir en arrière, à travers les différents menus, en appuyant sur la touche Esc.

Dans les programmes de jeux, la touche Esc sert souvent à interrompre le jeu.

Lors de l'utilisation du système d'exploitation GEM, dont nous avons abondamment traité dans ce chapitre, la touche Esc a encore une autre fonction : lorsque vous faites afficher sur l'écran le catalogue d'une disquette, avec un double clic, vous pouvez changer la disquette qui se trouve dans votre lecteur de disquette puis appuyer sur la touche Esc pour faire afficher le catalogue de la nouvelle disquette. Votre ST vous montre donc une nouvelle fois le catalogue de la disquette se trouvant actuellement dans votre lecteur de disquette.

La touche Esc a encore une autre fonction en liaison avec l'utilisation du système d'exploitation GEM: lorsque vous faites par exemple sortir le nom d'un fichier sur l'écran, avec 'View' 'Visualisation', vous pouvez effacer ce nom en appuyant sur la touche Esc si vous voulez entrer un autre nom.

Le clavier de machine à écrire de votre ST dispose enfin de trois autres touches spéciales que nous allons étudier maintenant.

#### 2.7.1.7. Backspace

Backspace (= un espace en arrière) vous permet d'effacer le caractère placé à gauche de votre curseur actuel. Si vous tenez cette touche enfoncée ou si vous appuyez dessus à plusieurs reprises, vous effacez donc votre texte en remontant vers le début du texte.

#### 2.7.1.8. Delete

Delete (= supprimer, effacer) vous permet d'effacer le caractère sur lequel se trouve placé votre curseur actuel. Il s'est cependant avéré dans la pratique que les touches Backspace et Delete ont exactement la même fonction dans certains programmes. Il convient donc de rester prudent, comme nous vous le disions plus haut et de ne pas généraliser hâtivement les explications que nous vous donnons ici. Un programmeur peut donner à chaque touche la valeur qu'il veut. Il pourrait fort bien par exemple affecter à toutes les touches des lettres de l'alphabet grec. Il ne faut donc pas s'étonner qu'un programmeur puisse par conséquent choisir dans certains cas de faire exécuter par les touches Backspace et Delete exactement la même fonction.

#### 2.7.1.9. Return

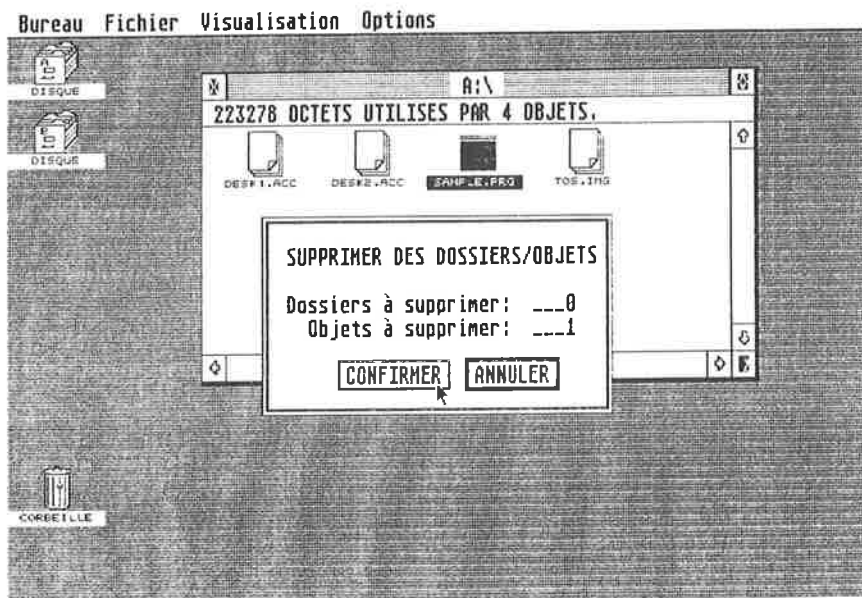
La dernière touche spéciale à évoquer ici est la touche Return (= retour de chariot). La notion de 'Return' vient des machines à écrire où il y a toujours une touche pour faire exécuter un retour de chariot.

Bien sûr, votre ATARI ST ne permet pas d'impression directe sur le papier, sans une imprimante connectée. La sortie des données se fait en effet normalement sur l'écran. Le fait d'appuyer sur la touche Return entraîne donc dans de nombreux programmes que le curseur avance au début de la ligne suivante.

Avec les langages de programmation tels que le BASIC, le fait d'appuyer sur la touche Return a, en outre, pour effet d'entrer dans l'ordinateur, pour traitement, une suite de lettres qui était jusqu'ici simplement affichée sur l'écran.

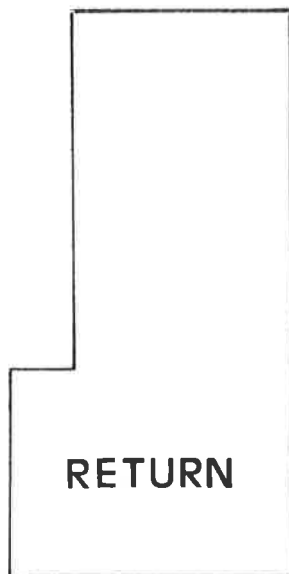
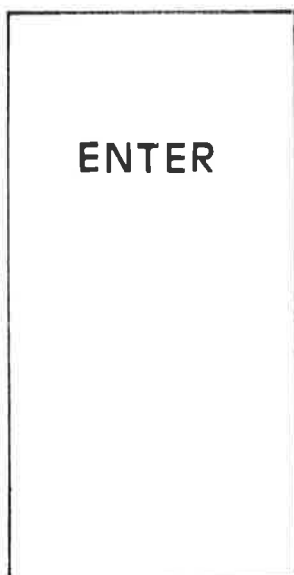
Avec l'emploi du système d'exploitation GEM, la touche Return a une autre fonction : lorsqu'une fenêtre d'information est sortie sur l'écran et que vous pouvez la confirmer en appuyant simplement dans un champ (CONFIRMER le plus souvent) avec la souris, vous pouvez aussi vous contenter d'actionner simplement la touche Return.

Votre ST vous offre en outre, sous GEM, souvent la possibilité de choisir parmi différents champs d'une fenêtre. Si une de ces possibilités de choix est entourée d'un cadre plus épais, cela signifie que vous pouvez sélectionner cette fonction en appuyant sur la touche Return.



### Par ailleurs :

La touche Enter du bloc numérique a en général la même fonction que la touche Return. Dans ce livre, nous désignerons ces deux touches sous le terme de "touche d'entrée".



**DANS CE LIVRE, NOUS APPELONS CES DEUX TOUCHES  
T O U C H E D ' E N T R E E !**

### **2.7.2. Les touches de fonction**

On ne peut malheureusement pas donner d'explication universellement valable sur les touches de fonction F1 à F10 qui se trouvent au dessus du clavier de machine à écrire. Ces touches reçoivent des fonctions qui varient suivant les programmes. Il peut s'agir par exemple de fixer le niveau de difficulté d'un jeu ou bien du chargement ou de la sauvegarde de texte dans un programme de traitement de texte ou encore de la recherche et de l'échange d'enregistrements de données déterminés dans un programme de banque de données. L'affectation de ces touches est donc entièrement laissée à votre imagination.

### 2.7.3. Les touches d'édition

Ce qu'on appelle le bloc d'édition (éditer veut dire, en informatique, traiter -examiner, modifier, compléter-) se trouve directement à côté du clavier de machine à écrire. Ce bloc est constitué de quatre touches fléchées et de quatre autres touches qui portent des noms tels que 'Insert' ou 'Help'.

#### 2.7.3.1. Les touches fléchées

La fonction des touches fléchées se comprend d'elle-même : ces touches vous permettent dans la plupart des programmes de déplacer le curseur dans la direction d'un des quatre points cardinaux.

#### 2.7.3.2. Les touches fléchées avec SHIFT ou Alternate

Si vous voulez déplacer avec les touches fléchées la flèche qui représente la souris, vous devez actionner les touches fléchées en même temps que la touche 'Alternate' ou que les touches 'Alternate' et 'SHIFT'. Dans le premier cas, la flèche de la souris se déplacera par sauts (de 8 points chaque fois) dans la direction voulue. Avec Alternate et SHIFT, la flèche de la souris ne se déplacera que point par point.

#### 2.7.3.3. Insert

La touche Insert sert dans de nombreux programmes à effectuer des insertions dans des textes pré-existants. La touche Insert a souvent, à cet égard, une fonction de commutateur : si vous appuyez une fois sur la touche 'Insert', le texte placé à droite du curseur sera repoussé vers la droite au fur et à mesure que vous écrirez ; si vous appuyez à nouveau sur la touche 'Insert', le mode d'effacement-remplacement sera à nouveau activé, c'est-à-dire qu'en écrivant vous repasserez sur le texte placé à droite du curseur, texte qui sera donc effacé au fur et à mesure.

#### 2.7.3.4. Clr/Home

La touche 'Clr/Home' a dans certains programmes une double fonction : si vous appuyez sur cette touche, le curseur retourne "à la maison" (Home), c'est-à-dire dans le coin supérieur gauche de l'écran; si vous appuyez sur cette touche mais, en même temps, sur SHIFT, il se produit la même chose mais l'écran est en outre effacé auparavant (Clear = effacer, nettoyer).

#### 2.7.3.5. Help

La touche Help (= à l'aide) a dans de nombreux programmes une fonction conforme à son nom : si l'on ne sait plus quoi faire, on peut appeler au secours en appuyant sur la touche Help. On reçoit alors le plus souvent des explications fort utiles.

#### 2.7.3.6. Undo

Nous avons déjà rencontré la touche Undo lorsque nous avons expliqué le programme d'émulateur VT 52. Comme son nom l'indique (undo = annuler, défaire), cette touche permet d'annuler la dernière opération exécutée.

#### 2.7.3.7. Alternate et Help ensemble

Si vous appuyez en même temps sur Alternate et Help, votre ST exécutera une sortie de l'écran sur l'imprimante connectée. Nous vous conseillons toutefois vivement de ne le faire que si une imprimante est réellement connectée à votre ordinateur. Il se pourrait en effet sinon que vous "plantiez" votre ordinateur, c'est-à-dire que votre ST ne sache plus du tout où il en est et se refuse à faire quoi que ce soit. Il vous faudrait alors le relancer en appuyant sur la touche Reset ou en l'éteignant puis en le rallumant. Si vous appuyez une seconde fois sur les deux touches indiquées ci-dessus pendant l'impression, la sortie sur imprimante sera interrompue.



#### 2.7.4. Le bloc numérique

##### 2.7.4.1. Enter

Nous allons, pour terminer, dire quelques mots du bloc numérique. Nous avons déjà parlé de la touche Enter. Elle a la même fonction que Return.

##### 2.7.4.2. Les chiffres, les symboles de calcul et le point décimal

Les 10 chiffres ainsi que les symboles de calcul et les caractères spéciaux se trouvent également sur le clavier de machine à écrire. Tous ces signes sont cependant reproduits une seconde fois dans ce bloc où on peut les utiliser plus commodément parce qu'ils s'y trouvent regroupés.

Encore une remarque : du fait que votre ST, comme tout autre ordinateur, utilise la virgule autrement que nous ne le faisons en France, notez que c'est le point qui sert de séparation, dans les nombres décimaux, entre les parties entière et décimale de ces nombres. Donc, en langage informatique, on dira par exemple "1.50" et non "1,50".



### **3. LE BASIC DE L'ATARI ST**

Nous allons vous donner, dans ce chapitre, une introduction aux possibilités du BASIC. Nous ne pouvons bien sûr que vous donner un aperçu grossier de tout ce qu'on peut faire en BASIC. Nous allons, dès le début, découvrir ensemble quelques instructions, écrire nous-mêmes quelques petits programmes puis approfondir ensuite nos connaissances au vu de programmes tout prêts que vous pourrez taper vous-même. Si nous réussissons ainsi à vous donner le goût de programmer l'ATARI ST dans ce langage de programmation très pratique, nous vous invitons à compléter votre information sur ce sujet au moyen d'ouvrages plus spécialisés sur le BASIC, par exemple avec le livre du **BASIC sur l'ATARI ST de Data Becker et Micro Application**.

#### **3.1. COMMENT LE ST APPREND-IL LE BASIC ?**

##### **3.1.1. Qu'est-ce que le BASIC ?**

Jusqu'à présent, c'était toujours nous qui devions répondre à une question ou une indication du ST. Il serait vraiment temps qu'un véritable dialogue puisse s'établir entre nous et l'ordinateur. Nous voulons pouvoir nous faire comprendre par l'ordinateur et lui faire exécuter pour nous un certain nombre de tâches. Pour y parvenir, il nous faut toutefois apprendre la langue qu'il parle. Heureusement, le ST est tellement intelligent qu'il parle de nombreuses langues. Ces langues ne sont bien sûr pas des langues naturelles telles que le français ou l'espagnol mais bien des langues de programmation.

Malheureusement, lorsqu'il débarque chez nous, le ST n'est pas en mesure de comprendre un langage de programmation. Mais tout ce dont il a besoin pour cela se trouve sur la disquette de langages, **LANGUAGE DISK**, qui est fournie avec l'ATARI ST. Cette disquette contient en quelque sorte le vocabulaire mais aussi la grammaire de chaque langage de programmation.

Lorsque nous avons chargé tout cela dans la mémoire de l'ordinateur, nous pouvons nous faire comprendre de lui à travers le clavier. La disquette LANGUAGE DISK contient deux langages différents avec tout ce qui s'y rapporte. Commençons donc par le BASIC.

### **3.1.2. Chargement du BASIC**

Si la disquette LANGUAGE DISK se trouve déjà dans le lecteur de disquette et si vous avez déjà lancé le BASIC, vous pouvez passer sur les lignes suivantes. Notez toutefois qu'il est possible, au moment où vous lirez ces lignes, que le langage de programmation BASIC soit déjà "intégré" sur votre ST et que vous n'ayez donc plus besoin de le charger tout d'abord à partir de la disquette. Nous vous invitons, dans ce cas, à vous reporter aux explications qui vous seront données dans votre manuel pour le lancement du BASIC.

Si cependant vous venez juste d'allumer votre ordinateur, vous devez d'abord placer la disquette système dans le lecteur de disquette et lancer le système d'exploitation (voir 2.1.). Prenez ensuite la seconde disquette fournie avec l'ATARI ST, LANGUAGE DISK. Faites afficher le catalogue de cette disquette puis lancez le programme BASIC.PRG.

Nous allons enfin pouvoir commencer.

## **3.2. LES FENETRES EN BASIC**

### **3.2.1. Fonction des fenêtres en BASIC**

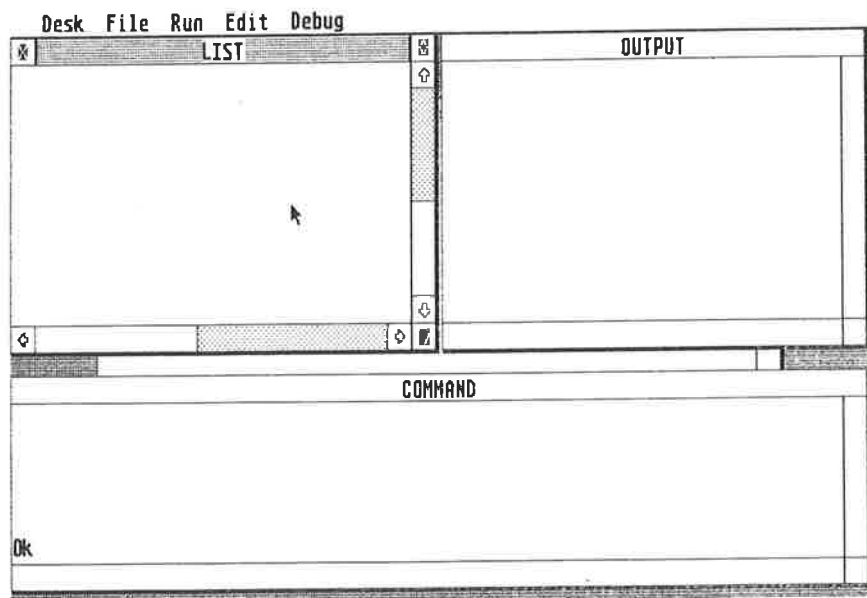
Vous voyez très nettement que le lancement du BASIC a profondément modifié notre écran. La ligne de menu dans la première ligne d'écran comporte d'autres termes et, d'autre part, les quatre fenêtres d'écran possibles sont toutes employées d'emblée :

- 1) L'écran OUTPUT (=sortie) dans lequel s'effectue la sortie du programme.
- 2) L'écran COMMAND (=instruction) dans lequel vous pouvez effectuer les entrées de programme.

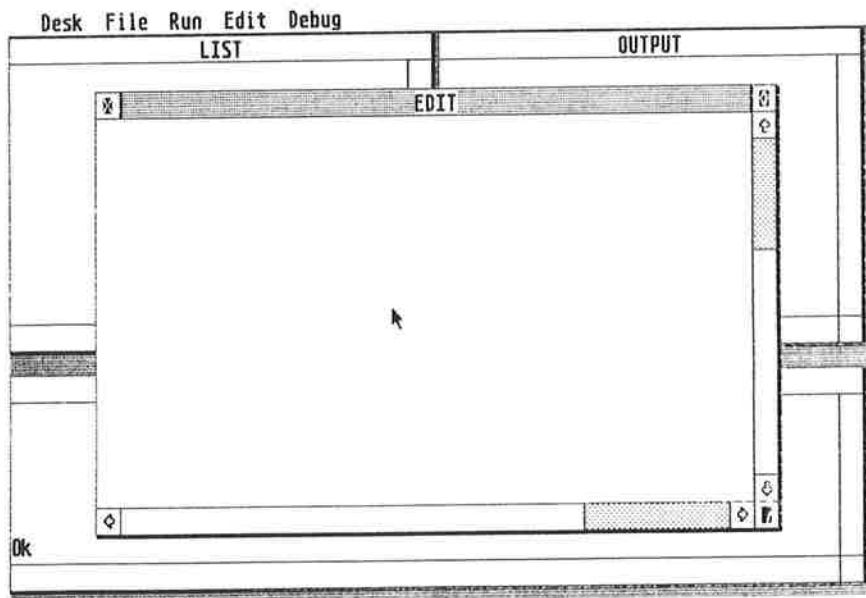
3) L'écran LIST (=afficher le programme) qui est prévu pour la représentation des listings de programmes.

4) L'écran EDIT (=examen, modification) qui n'est que partiellement visible à travers les trois autres écrans. Cet écran vous permettra plus tard d'apporter des améliorations à vos programmes.

L'écran COMMAND est affiché par le ST en début de programme comme fenêtre actuellement appelée. En effet, vous voyez que cet écran est doté des marques de cadre que nous connaissons bien maintenant pour les avoir déjà rencontrées avec la fenêtre de catalogue. Cela ne signifie pas cependant que nous ne pouvons agir que sur cette fenêtre. Si nous voulons faire par exemple de l'écran List la fenêtre "actuelle", pour l'agrandir, la rapetisser, tout simplement travailler avec elle, il nous suffit d'amener la flèche de la souris sur la fenêtre LIST et d'appuyer ensuite sur la touche gauche de la souris. Les marques du cadre de l'écran COMMAND seront immédiatement effacées et l'écran LIST deviendra la fenêtre actuellement traitée.



Nous voyons transparaître l'écran EDIT entre les bords des trois autres écrans. Si vous amenez la flèche de la souris sur ce petit emplacement, l'écran EDIT sera immédiatement ramené vers l'avant pour devenir l'écran actuel de travail.



### **3.2.2. Réorganisation des fenêtres en BASIC ST**

Avant que nous n'en venions tout de suite à la programmation en BASIC vous pouvez "jouer" avec les quatre fenêtres autant que le cœur vous en dit. Vous pouvez disposer les différentes fenêtres à votre guise, comme nous l'avons déjà fait avec les catalogues. Vous pouvez naturellement utiliser également les coins caoutchouc en bas à droite.

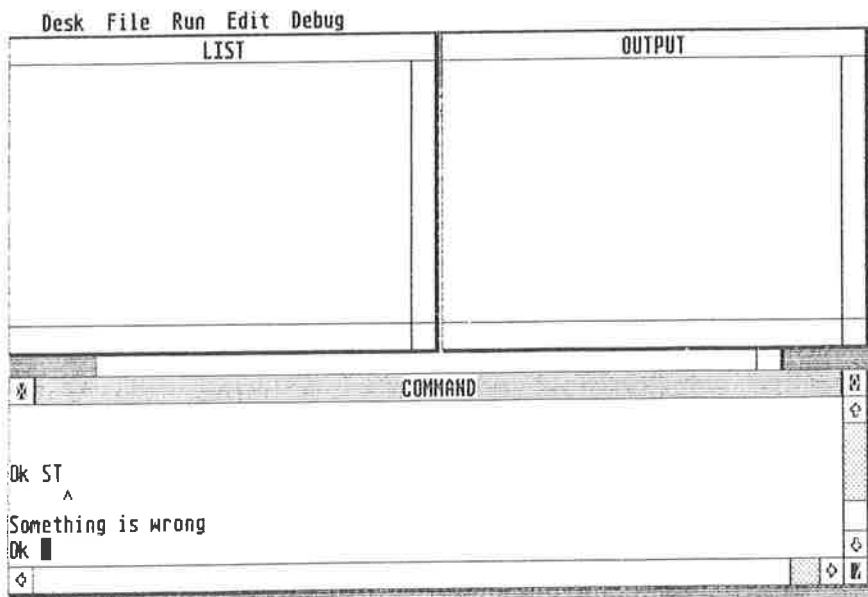
### **3.3. LES PREMIERS CONTACTS AVEC LE BASIC**

#### **3.3.1. Le mode direct**

Maintenant, au travail! Vérifions d'abord si votre ST a bien appris son vocabulaire : entrez les lettres 'st' au clavier. Sur l'écran COMMAND, vous verrez alors les deux lettres ST, à la suite de OK. Il ne s'est rien passé d'autre. Vous pouvez entrer ainsi dans votre ST n'importe quelles autres combinaisons de lettres ou de caractères. La seule chose qui se passe, c'est que ces lettres ou caractères sont représentés simultanément à l'écran, caractère par caractère.

Pourquoi ne se passe-t-il rien ? Le ST attend en fait encore de savoir ce que vous voulez qu'il fasse. Pour lui faire comprendre que nous attendons de sa part une réaction aux lettres "ST" que nous avons si gentiment tapées, nous devons "transmettre" ce que nous avons écrit à sa mémoire. Cela peut être fait avec la touche RETURN ou avec la touche ENTER, c'est-à-dire avec l'une de ces deux touches que nous avons décidé d'appeler "touche d'entrée".

Appuyons donc simplement sur la touche RETURN. Le ST réagit immédiatement : vous pouvez lire 'Something is wrong', ce qui signifie 'quelque chose est incorrect', sous-entendu : dans les termes choisis.



Bien que notre ordinateur s'appelle ST, il n'est pas en mesure de tirer quelque chose de ce mot. ST ne fait pas partie des environ 150 mots qu'il comprend.

Vous pouvez renouveler cette expérience avec quelques mots anglais ou même avec votre nom. Il y a peu de chances que vous tiriez de lui, en appuyant sur la touche d'entrée, autre chose que le message ci-dessus.

Vous allez peut-être penser que le vocabulaire BASIC de votre ordinateur est désespérément pauvre ! Votre ordinateur se permet en outre, après chacune de ses réponses monotones, de sortir sur l'écran, en guise de conclusion, la remarque 'OK'.

En fait, lorsqu'il vous dit "OK", le ST veut simplement vous indiquer qu'il est à nouveau prêt à recevoir vos instructions. Il arrive parfois que vous puissiez faire une entrée au clavier mais que l'entrée correspondante n'apparaisse pas à l'écran. Dans ce cas, il n'y aura pas non plus de réponse OK.



Pour que notre ATARI ST fasse enfin autre chose, entrez donc l'instruction suivante :

**print "atari st"**

Actionnez ensuite la touche d'entrée. Que se passe-t-il ? Si vous avez tapé exactement ce que nous vous avons dit, vous pouvez maintenant voir "atari st" dans l'écran OUTPUT. Pas mal, hein ?

Ce résultat a été produit par le mot "PRINT" qui fait partie du vocabulaire de votre ST. A l'intérieur des guillemets, vous pouvez entrer maintenant, au lieu de "atari st", le nom d'un ami ou tout autre chose. Le ST exécutera toujours fidèlement votre instruction après que vous aurez appuyé sur la touche d'entrée.

On peut donc donner directement au ST l'ordre de faire certaines choses, pourvu qu'on le fasse avec un mot qui fasse partie de son vocabulaire. Ce mode d'entrée est appelé 'mode direct'.

### 3.3.2. Mode programme

Pour faire faire quelque chose par l'ATARI ST en mode direct, il faut entrer chaque fois à nouveau toute la ligne d'instructions. Que faire alors si on veut faire exécuter le même travail de très nombreuses fois ? Il faut écrire un 'programme'.

C'est d'ailleurs pour qu'on puisse écrire des programmes que le langage de programmation BASIC a été inventé. Une fois qu'on a écrit et tapé un programme dans l'ordinateur, on peut le faire exécuter aussi souvent que l'on veut et une instruction suffit pour demander à l'ordinateur de lancer le programme. Ce mode de travail est appelé 'mode programme'.

Nous allons maintenant apprendre à écrire un tel programme. Nous pourrions ainsi amener le ST à travailler pour nous. Nous allons pour cela apprendre quelques-unes des instructions et techniques BASIC les plus importantes. Commençons par vider l'écran OUTPUT, avec 'clearw 2'. Cette instruction, que vous devez taper tout simplement ainsi, efface la seconde fenêtre (CLEAR = effacer; W = abréviation de window = fenêtre; 2 = numéro de la fenêtre). Vidons ensuite l'écran COMMAND avec 'clearw 3'.

### 3.4. LE VOCABULAIRE BASIC DE BASE

#### 3.4.1. Entrée et sortie : INPUT et PRINT

##### 3.4.1.1 Les numéros de lignes

Lorsqu'on programme son ST en BASIC, on doit doter chaque ligne d'un numéro. Après le 'OK', on ne commence donc pas tout de suite par l'instruction mais on écrit en premier un numéro de ligne. Le nombre '10' est habituellement le premier numéro de ligne utilisé. Il peut être alors suivi de l'instruction. Le mieux est alors de continuer, à la seconde ligne, avec '20' etc... Il est préférable d'espacer les numéros de ligne de 10 en 10 car cela permettra plus tard de pouvoir facilement insérer de nouvelles lignes qui pourront s'appeler par exemple '15' ou '17'.

##### 3.4.1.2. L'instruction PRINT

Comme nous l'avons déjà vu, PRINT sert à sortir quelque chose sur l'écran. Avec la ligne

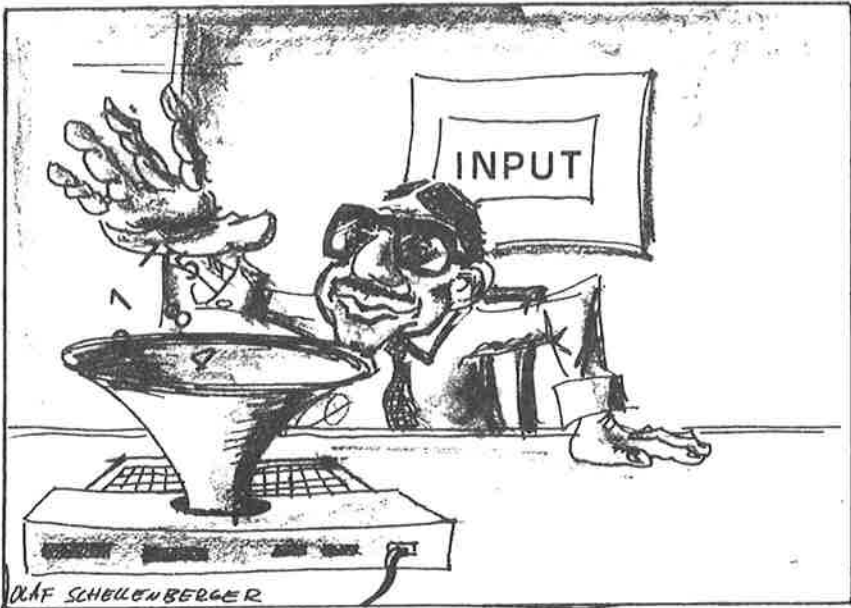
**10 print "atari st"**

nous pouvons donc obtenir la sortie du mot placé entre guillemets. Entrez donc la ligne ci-dessus. Si vous faites une faute de frappe, vous pouvez facilement effacer le texte erroné avec la touche BACKSPACE. Si tout est correct, appuyez sur la touche d'entrée. Il se passe ... rien du tout. Qu'est-ce que cela signifie ?

Rappelez-vous : nous avons dit précédemment qu'on doit d'abord écrire le programme et qu'on peut ensuite le lancer avec une seule instruction. Or nous n'avons pas encore donné cette instruction de lancement du programme. En appuyant sur la touche d'entrée, nous avons simplement transmis notre programme '10 print "atari st"' au ST. Il attend encore que nous lui disions ce qu'il doit en faire. L'instruction de lancement d'un programme est 'RUN' (= allez!). Nous pouvons entrer cette instruction directement au clavier, en mode direct, c'est-à-dire SANS numéro de ligne ou bien nous pouvons parcourir le menu déroulant 'RUN' avec la souris et y marquer par un clic le mot 'RUN'. Sur l'écran COMMAND apparaîtra alors également 'RUN' et vous verrez sur l'écran OUTPUT le résultat de notre programme, c'est-à-dire 'atari st'.

Bien, nous direz-vous, mais nous étions arrivés au même résultat en entrant cette même ligne sans numéro de ligne. Tout juste ! Mais vous allez bientôt voir la différence.

### 3.4.1.3. L'instruction INPUT



Nous allons tout d'abord vider les fenêtres OUTPUT et COMMAND, comme nous l'avons déjà fait plus haut, avec 'clearw 2' et 'clearw 3'.

Notre programme d'une ligne a maintenant disparu. Et pourtant il se trouve toujours quelque part dans la mémoire du ST. C'est pourquoi nous allons maintenant le supprimer réellement de la mémoire en entrant 'new' sans numéro de ligne.

Nous avons jusqu'ici toujours tapé le texte devant être sorti par le ST entre guillemets. Nous allons maintenant écrire un programme tout de même un peu plus pratique. Nous voulons que le ST nous demande quel texte il doit sortir et qu'il exécute notre souhait dès que nous lui aurons répondu. Nous utiliserons pour cela l'instruction INPUT.

Ecrivons donc tout d'abord :

```
10 input "Que dois-je écrire";a$  
20 print a$
```

Marquez à nouveau RUN par un clic. Que se passe-t-il ? Dans la fenêtre OUTPUT apparaît la question de notre ST 'Que dois-je écrire?'. Vous pouvez maintenant entrer n'importe quoi. Appuyez ensuite sur la touche d'entrée et le texte que vous aurez tapé apparaîtra une seconde fois, dans la fenêtre OUTPUT.

Qu'avons-nous fait ? Dans la ligne 10 de notre programme, il se passe la chose suivante :

Après une instruction INPUT, on peut entrer un texte de commentaire entre guillemets. Ce texte sera alors utilisé par l'ordinateur comme texte d'interrogation. Le point d'interrogation est placé par le ST lui-même. Après que vous ayez tapé quelque chose en réponse à sa question et que vous ayez appuyé sur la touche d'entrée, le ST donnera à ce texte le nom 'a\$'. Ce nom est ce qu'on appelle une 'variable'. Le signe dollar indique que cette variable contient un texte. Si vous voulez entrer des nombres, avec lesquels des calculs puissent être effectués dans la suite du programme, il ne faut pas mettre le signe dollar.

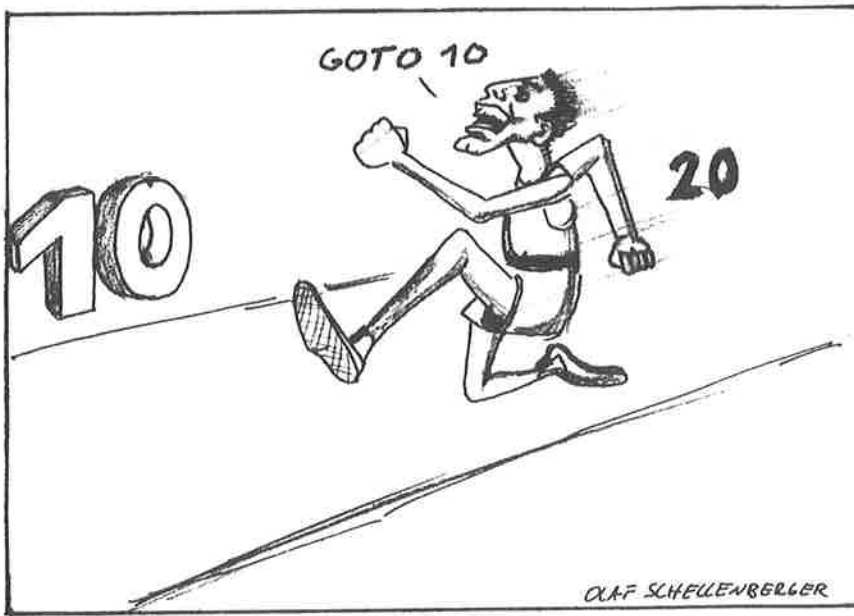
En ligne 20, le ST sort alors le texte appelé 'a\$'.

L'ordinateur a alors terminé l'exécution du programme et OK apparaît à nouveau dans la fenêtre COMMAND.

Donc :

- L'instruction INPUT affecte à une variable un texte ou un nombre.
- L'instruction PRINT sort un texte ou le contenu d'une variable.

### 3.4.2. Saut inconditionnel : GOTO



On peut écrire en deux lignes le programme le plus simple et le plus dénué d'intérêt que les fanas d'informatique entrent régulièrement, avec beaucoup de fierté, sur les ordinateurs qui leur tombent sous la main, par exemple dans les expositions.

Pour que vous puissiez, vous aussi, répondre à ces fanas, entrez aussi dans votre ST le programme de deux lignes suivant, non sans avoir auparavant vidé la mémoire avec NEW et les 3 écrans avec CLEARW 1, CLEARW 2 et CLEARW 3 :

```
10 print "atari st"
20 goto 10
```

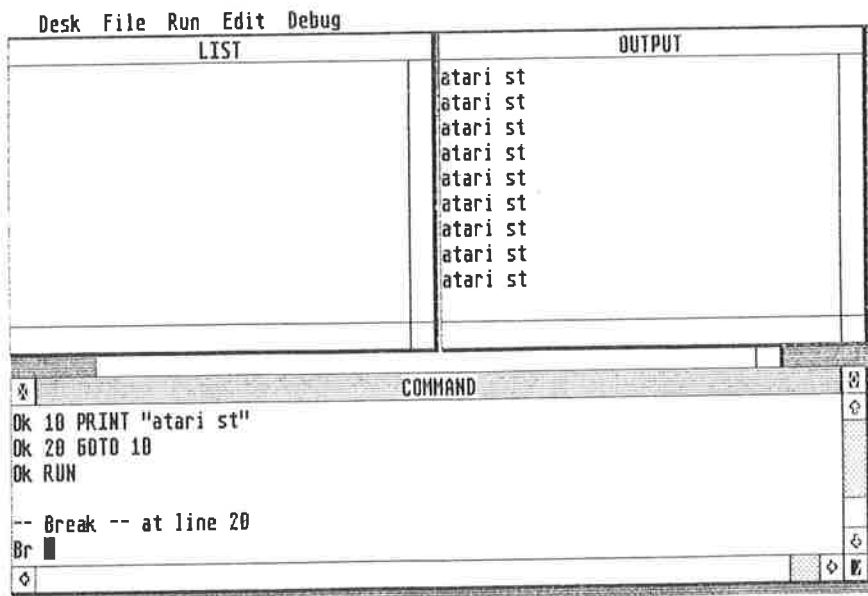
Lancez à nouveau ce programme avec 'RUN'. Que se passe-t-il ? Dans la fenêtre OUTPUT apparaît sans cesse 'atari st'. Si vous interrompiez la lecture de ce livre, il ne vous resterait, dans l'ignorance de nos explications, qu'à actionner la touche Reset ou à éteindre votre ordinateur puis à le rallumer pour pouvoir sortir de ce programme.

Que signifient en effet ces deux lignes ?

Ligne 10 : PRINT fait sortir le mot magique 'atari st' sur l'écran.

Ligne 20 : 'goto 10' entraîne un saut à la ligne 10. 'atari st' sera donc sorti une nouvelle fois et ce indéfiniment, jusqu'à ce que vous actionniez la touche Reset ou que vous éteigniez l'ordinateur.

Il y a cependant une manière plus élégante pour interrompre un programme. C'est en effet beaucoup plus facile si vous appuyez simultanément sur les touches Control et G. Suivant le point où en est l'exécution du programme à ce moment, l'écran affichera '— Break — at line 10' ou '— Break — at line 20' (Interruption du déroulement du programme en ligne 10 ou en ligne 20).



Nous pouvons faire repartir ensuite le programme du même point, en entrant l'instruction CONT, qui est l'abréviation de continuer.

Nous pensons en tout cas avoir atteint, avec notre mini-programme, notre but dans cette section qui était de vous faire découvrir l'instruction GOTO.

Donc :

*L'instruction GOTO entraîne un saut au numéro de ligne placé à la suite de GOTO.*

3.4.3. Si ... alors ... sinon :  
IF ... THEN ... ELSE

Alors que l'instruction GOTO est toujours exécutée, quelles que soient les circonstances, IF ... THEN permet d'obtenir un saut qui ne sera exécuté que si une condition déterminée est remplie.

Voici un petit programme illustrant l'emploi de cette instruction. Videz d'abord la mémoire et les trois fenêtres.

Entrez ensuite le programme suivant. (N'oubliez pas la touche d'entrée après chaque ligne !)

```
10 input "1er nombre";n1
20 input "2d nombre";n2
30 if n1=n2 then 60 else 40
40 print "Les nombres sont différents."
50 goto 70
60 print "Les nombres sont égaux."
70 end
```

Lancez maintenant ce programme avec RUN. On vous demande alors le premier nombre. Entrez n'importe quel nombre et appuyez sur la touche d'entrée. On vous demande ensuite un second nombre. Après avoir entré le second nombre et avoir appuyé sur la touche d'entrée, vous voyez soudain surgir une autre phrase. Votre ST vous a ainsi indiqué si les deux nombres sont ou non égaux.

### Comment cela fonctionne-t-il ?

Les lignes 10 et 20 ne posent plus de problème. Le premier nombre entré s'appellera 'n1' (sans signe dollar car il s'agit d'un nombre) et le second s'appellera 'n2'.

La ligne 30 examine ensuite si n1 est égal à n2.

### Il y a alors deux possibilités :

#### 1. Les nombres sont égaux

Votre ST va alors à la ligne 60 pour sortir le message indiquant que les nombres sont égaux. Le ST continue ensuite l'exécution du programme, en passant à la ligne 70 où le programme se termine.

#### 2. Les nombres sont différents

Le ST va alors en ligne 40. Pourquoi ?

L'instruction IF ... THEN ... ELSE fait que SI une condition est remplie, ALORS on saute à la ligne indiquée, mais SINON, c'est-à-dire si la condition n'est pas remplie, l'ordinateur passe au numéro de ligne placé après ELSE. Dans notre cas, par conséquent : SI  $n1=n2$ , ALORS va en ligne 60, SINON va en ligne 40.

Le ST est donc arrivé en ligne 40 où il a sorti le message indiquant que les nombres sont différents. Il passe ensuite à la ligne suivante où on lui demande d'aller en ligne 70 où le programme se termine.

#### Donc :

Avec IF ... THEN ... ELSE, on peut provoquer des sauts dépendant de conditions déterminées.



#### 3.4.4. Les boucles avec FOR ... NEXT

Il serait intéressant de pouvoir fixer combien de fois le programme doit être exécuté, c'est-à-dire combien de fois il doit revenir au début du programme après en avoir atteint la fin. On utilise pour cela ce qu'on appelle une 'boucle'. Conservons le programme en mémoire. Ne tapez donc pas 'NEW'. Mais vidons les trois fenêtres.

Entrons maintenant encore les quatre lignes suivantes :

```
5 input "Combien de parcours";p
6 for i=1 to p
70 next i
80 end
```

Voulez-vous savoir comment notre programme complet se présente maintenant ? Rien de plus facile ! Vous n'avez pas oublié d'appuyer sur la touche d'entrée à la fin de chaque ligne. Parfait, nous pouvons maintenant faire afficher le programme complet. Allez pour cela dans le menu déroulant EDIT et marquez le terme LIST par un clic.

Instantanément apparaît dans la fenêtre LIST le programme entier :

```
5 input "Combien de parcours";p
6 for i=1 to p
10 input "1er nombre";n1
20 input "2d nombre";n2
30 if n1=n2 then 60 else 40
40 print "Les nombres sont différents."
50 goto 70
60 print "Les nombres sont égaux."
70 next i
80 end
```

Qu'avons-nous fait ? Nous avons ajouté au programme les lignes 5 et 6. La ligne 70 se présentait auparavant ainsi : 70 end

En entrant une nouvelle ligne, à laquelle nous avons également attribué le numéro 70, nous avons donc fait effacer par le ST l'ancienne ligne 70 et celui-ci a placé dans sa mémoire la nouvelle ligne 70. La ligne 70 se présente maintenant ainsi : **70 next i**

La ligne 80 a également été ajoutée. Vous voyez comme il est facile de compléter un programme ? Il suffit d'entrer les nouvelles instructions, il n'est pas nécessaire de taper à nouveau les anciennes instructions car elles sont encore en mémoire.

Qu'avons-nous donc modifié ? La ligne 5 ne pose certainement pas de problème. La variable *p* reçoit la valeur correspondant au nombre de parcours du programme que vous souhaitez. Vient ensuite la ligne 6 où est situé ce qu'on appelle le début de la boucle. La fin de la boucle se trouve en ligne 70. Tout ce qui est compris entre ces deux lignes est donc à l'intérieur de la boucle.

Essayons de suivre le déroulement du programme.

Ligne 5 :

Le nombre de parcours est placé dans le tiroir *p*.

Ligne 6 :

Début de la boucle. La variable *i*, qu'on appelle aussi le 'compteur' de la boucle, se voit attribuer, la première fois, la valeur 1. Les fois suivantes, elle aura la valeur résultant de la ligne 70 où cette valeur est augmentée de 1. Quoi qu'il en soit, la ligne 6 testera chaque fois si la valeur de *i* est déjà supérieure à *p*. Tant que ce ne sera pas le cas, la boucle suivante sera exécutée.

Lignes 10 à 60 :

Programme comme plus haut.

Ligne 70 :

*i* est chaque fois augmenté de 1 puis on saute au début de la boucle, en ligne 6.

Lorsqu'en ligne 6 la valeur de *i* est supérieure à *p*, on saute par dessus la boucle, directement en ligne 80 (fin du programme).

Vous voyez que FOR ... NEXT est une instruction très puissante qui fait beaucoup de choses sans qu'on s'en rende compte.

### **3.4.5. Les sous-programmes : GOSUB ... RETURN**

Les programmeurs en veulent souvent au BASIC de posséder une instruction GOTO qui ne permet malheureusement que trop facilement de faire ce qu'on appelle des programmes 'spaghetti', c'est-à-dire des programmes où tout va dans tous les sens comme dans un labyrinthe.

Lorsqu'on veut écrire un programme, on s'assoit normalement à une table et on réfléchit, ligne par ligne, à la façon dont le programme devra plus tard fonctionner dans tous ses éléments. Une fois ce projet sur le papier terminé, on peut l'entrer ligne par ligne dans le ST, avant de lancer le programme avec RUN.

Dans une situation idéale, non seulement le programme fonctionne exactement comme on l'a voulu mais tous les souhaits possibles ont été prévus lors de la programmation. Mais il ne s'agit là bien sûr que d'une situation idéale.

La plupart des programmeurs amateurs commencent en fait par s'asseoir devant leur ordinateur et se mettent à programmer, sans grandes réflexions préalables. Peu de temps après, ils s'aperçoivent qu'il manque encore quelque chose ici (GOTO) et aussi là (GOTO) et enfin encore là-bas (GOTO).

Le jour où vous écrivez votre programme, vous n'avez pas trop de mal à vous y retrouver dans le désordre que vous avez vous-même construit mais vous pouvez redouter le pire si vous essayez de reprendre ce programme deux semaines plus tard pour l'améliorer. Ou encore plus catastrophique : si vous aviez l'intention de faire terminer votre programme par quelqu'un d'autre, il risque de mettre beaucoup plus de temps à le comprendre qu'il ne lui en faudrait pour réécrire correctement tout ce que vous avez fait. Toute idée de structure a été oubliée et le programmeur ne sait plus comment se tirer de ce mauvais pas.

Il vaut donc mieux avoir développé le programme tranquillement sur sa table de travail. Vous pouvez ainsi ajouter les différents éléments les uns aux autres jusqu'à ce que le produit final paraisse suffisamment achevé.

Par rapport à notre programme de deux lignes, cela signifierait : la sortie sur l'écran de la suite de lettres 'atari st' est un programme en soi, ce qu'on appelle un sous-programme, qui peut être séparé du programme principal.

```
10 gosub 100
20 goto 10
100 print "ST";
110 return
```

Notre programme de deux lignes est maintenant devenu un programme de quatre lignes mais les lignes 100 à 110 nous montrent clairement qu'il y a là un sous-programme.

L'instruction RETURN de la ligne 110 entraîne qu'on revient à la ligne suivant l'instruction GOSUB et que le programme se poursuit à partir de là (ligne 20).

Le gain en clarté n'est peut-être pas considérable avec notre petit sous-programme. Mais examinez un peu les programmes plus longs que nous avons imprimés, pour différentes applications, dans la neuvième section de ce chapitre, et vous reconnaîtrez avec nous que les sous-programmes facilitent considérablement le contrôle du déroulement d'un programme !

#### **3.4.6. Encore plus de clarté : REM**

Si vous écrivez REM (abréviation de remarque) au début d'une ligne, mais bien sûr après le numéro de ligne, votre ST ignorera cette ligne lors de l'exécution du programme et il passera directement à la ligne suivante.

Pour le programmeur, les lignes REM sont très importantes : elles nous permettent de faire suivre l'expression REM de 'commentaires' qui rendront nos programmes beaucoup plus faciles à comprendre, donc aussi à modifier ultérieurement.

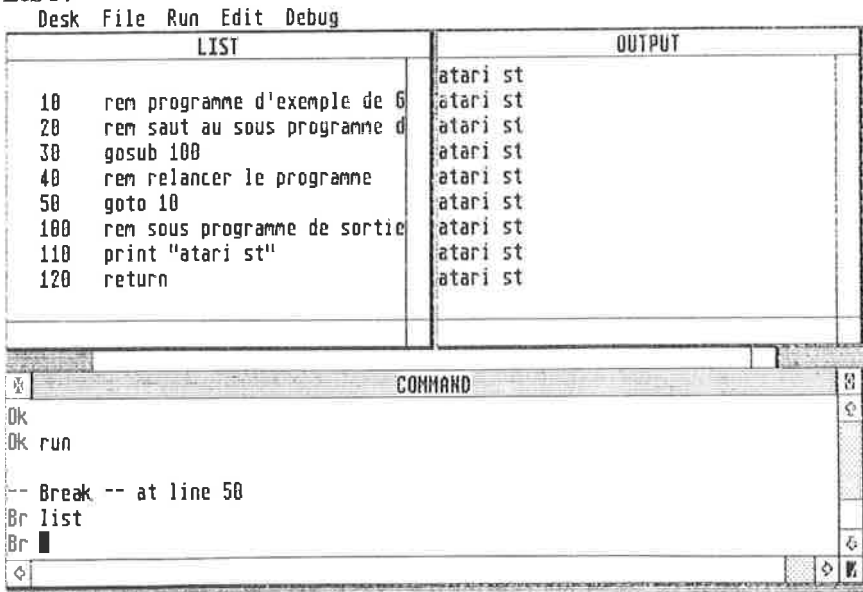
Modifions notre programme d'exemple précédent en y incluant des lignes REM :

```

10 rem Programme d'exemple de GOSUB
20 rem Saut au sous-programme de sortie à l'écran
30 gosub 100
40 rem relancer le programme
50 goto 10
100 rem sous-programme de sortie à l'écran
110 print "atari st"
120 return

```

Avant de lancer le programme avec RUN, examinez son listing avec LIST.



Même si vous ne relisez plus ce programme avant des années, vous pourrez comprendre d'emblée ce qui s'y passe.

Donc :

Les lignes REM nous aident à rendre nos programmes plus clairs et donc plus compréhensibles.

### **3.4.7. Effacer en un clin d'oeil**

#### **3.4.7.1. Nettoyer une fenêtre : CLEARW**



Vous vous rappelez certainement comment nous avons effacé les fenêtres chaque fois avant de lancer un programme. Nous utilisons pour cela l'instruction

**clearw n**

La lettre 'n' représente les valeurs 0 à 3, c'est-à-dire quatre valeurs différentes, ce qui correspond exactement au nombre de fenêtres dont dispose le BASIC du ST.

Ces quatre valeurs correspondent aux fenêtres suivantes :

- n=0 : effacer la fenêtre EDIT
- n=1 : effacer la fenêtre LIST
- n=2 : effacer la fenêtre OUTPUT
- n=3 : effacer la fenêtre COMMAND

### 3.4.7.2. Comme s'il ne s'était rien passé : NEW

Il s'agit encore d'une vieille connaissance. Lorsque vous voulez effacer tout ce qu'il y a dans la mémoire, c'est-à-dire aussi bien les variables que le programme, vous devez employer l'instruction NEW. Mais attention! Il faut bien réfléchir avant d'employer cette instruction !

## 3.5. QUELQUES MOTS SUR LES VARIABLES

Vous l'avez certainement déjà remarqué : sans les variables, ces lettres qui peuvent porter 'avec elles' des phrases entières ou aussi des nombres, rien n'est possible en informatique. Commençons par les nombres.

### 3.5.1. Variable numérique simple

Nous allons d'abord demander à votre ATARI ST de retenir des nombres. A cet effet, nous allons constituer un tiroir dans sa mémoire et nous donnerons un nom à ce tiroir. Nous avons utilisé différents noms jusqu'ici (a, A\$, n1, etc...). Cette fois, nous allons simplement choisir le nom 'st'. Notre tiroir s'appellera donc 'st'.

Nous avons toujours utilisé, jusqu'ici, l'instruction INPUT pour affecter une valeur (un nombre ou un texte) à une variable. Il y a cependant une autre possibilité qui est encore plus simple. Comme en mathématiques, il nous faut pour cela écrire une équation.

Si nous voulons par exemple indiquer à notre ST que le tiroir appelé ST doit contenir la valeur 6, nous écrirons simplement :

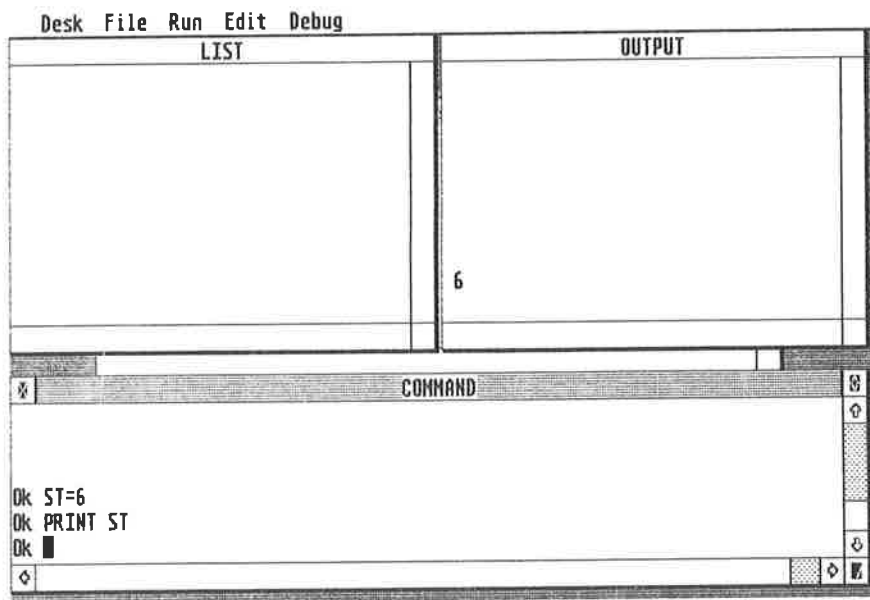
**st=6**

N'oubliez pas d'actionner à nouveau, pour terminer, la touche d'entrée. Parfait. Aucun message d'erreur n'a été sorti sur l'écran COMMAND et nous n'y voyons que le très rassurant OK. Notre ST a maintenant noté que le tiroir ST contient la valeur 6 !

En douteriez-vous ? Pour le vérifier, il vous suffit de faire sortir sur l'écran OUTPUT le contenu du tiroir ST. Vous souvenez-vous de l'instruction de sortie sur l'écran ? Bravo ! La formule magique est en effet PRINT. Pour sortir le contenu du tiroir appelé 'st', nous taperons donc, sans désespérer :

**print st**

Si vous avez suivi toutes les étapes, le nombre 6 devrait maintenant apparaître sur votre écran OUTPUT.



Comme vous avez cependant écrit 'st=6' à la ligne précédente et que cette ligne est encore visible sur l'écran, vous soupçonnez peut-être une vague tricherie. Est-ce que l'ordinateur ne serait pas aller chercher le nombre 6 dans l'écran COMMAND ? Pour que nous puissions vous prouver qu'il n'y a pas de triche et que le tiroir ST ne se trouve pas seulement à l'écran mais aussi beaucoup plus profond dans la mémoire de votre ordinateur ST, videz tout simplement l'écran en entrant CLEARW 2 pour l'écran OUTPUT et CLEARW 3 pour l'écran COMMAND.



Entrez maintenant à nouveau l'instruction d'interrogation de la mémoire (ou des tiroirs) :

**print st**

La valeur 6 est à nouveau sortie. Le doute, qui est d'ailleurs une très saine réaction, n'est donc plus permis. Vous pouvez maintenant créer d'autres tiroirs en leur donnant un nom et une valeur numérique :

Agnes = 4  
Brigitte = 5  
Claudia = 7  
Doris = 6

Si vous faites maintenant sortir sur l'écran le contenu de ces tiroirs, avec la formule magique PRINT, vous ne serez pas déçu.

La place mémoire disponible sur votre ordinateur ST n'a pas été considérablement affectée par ce travail de stockage en mémoire. Vous pourriez toutefois remplir cette place mémoire si vous affectiez des nombres à de nouvelles variables (c'est-à-dire aux noms que nous donnons aux tiroirs) pendant 24 heures. En fait, votre ordinateur ST ne perdrait pas son calme même si vous lui demandiez de mémoriser plusieurs milliers d'entrées. Il est vraisemblable que c'est vous qui vous lasseriez le premier.

Un nom de variable peut comprendre en tout 31 caractères. Cela signifie que l'ATARI ST ne tient compte que des 31 premiers caractères du nom d'une variable. Par conséquent, pour lui, la variable appelée :

societenationaledescheminsdeferfrancais

sera la même variable que celle appelée :

societenationaledescheminsdeferbelges

En effet seuls les 31 premiers caractères, 'societenationaledes cheminsdefer' sont pris en compte. Avouez que 31 caractères pour un nom de variable, cela devrait tout de même suffir en règle générale.

Il faut encore tenir compte de deux autres points dans le choix d'un nom de variable :

1. Le nom de variable ne doit pas être un mot BASIC réservé. Il ne serait donc pas possible d'appeler une variable 'print'. Essayez un peu : `print=5`

Résultat : Something is wrong

car il s'agit du mot PRINT que nous connaissons déjà bien et qui fait partie du vocabulaire BASIC de votre ST.

2. Le premier caractère d'un nom de variable doit être une lettre. Peu importe d'ailleurs qu'il s'agisse d'une minuscule ou d'une majuscule car votre ST ne fait pas la différence lorsqu'il s'agit d'un nom de variable.

Vous pourriez donc parfaitement donner à vos variables des noms tels que A3, Ac, B, ou encore X, Y ou C.

Vous ne pourriez pas utiliser par contre les noms suivants : 4A ou 1e.

### 3.5.2 Variables numériques entières et décimales

Nous allons placer encore d'autres nombres dans notre ST mais en essayant maintenant de voir si la haute précision de notre ordinateur pour les exercices mathématiques ou pour les nombres est susceptible d'être réduite.

Qu'est-ce que cela signifie ? Tapez donc simplement

`print 2/7` (ce trait, dit trait de fraction, signifie 'divisé par')

Résultat : 0.285714

Notre ST a donc résolu cet exercice de division avec six chiffres après la virgule.

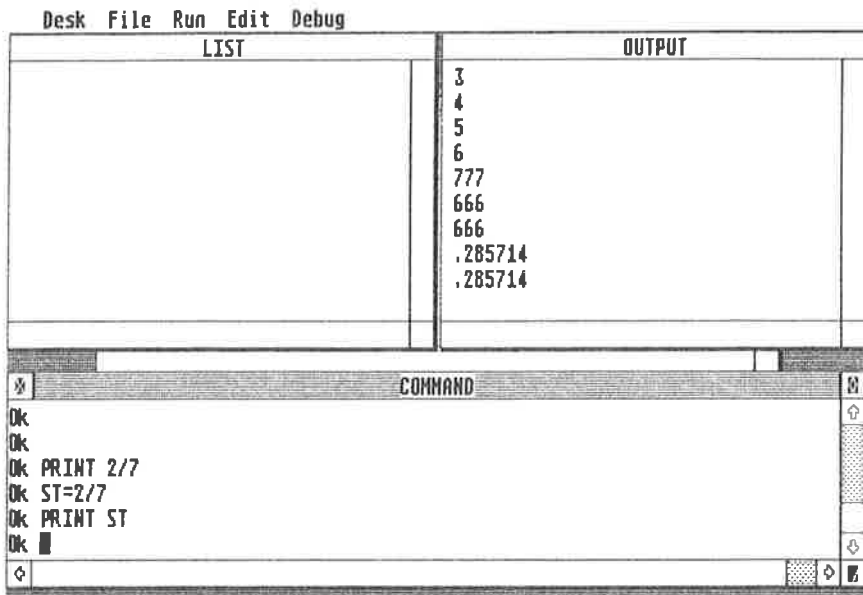
Comment pourrions-nous maintenant placer ce calcul dans une variable ? Il nous faut à nouveau trouver un nom de variable. Pourquoi ne pas garder tout simplement ST ? Nous affecterons maintenant à ce nom le résultat de l'opération précédente, chiffre pour chiffre,  $st=0.285714$  en l'occurrence. Nous pouvons aussi faire calculer le résultat directement lors de l'opération d'entrée :

$st=2/7$

Si vous entrez maintenant

`print st`

le résultat apparaîtra sur l'écran OUTPUT :



Vous vous doutez certainement que le stockage d'un nombre comportant autant de chiffres après la virgule doit occuper plus d'une place mémoire.

Mais si vous n'êtes pas mathématicien ou si vous ne voulez calculer avec votre ordinateur que la table de multiplication, il serait dommage de gaspiller inutilement autant de places mémoire. D'autre part, votre ordinateur calcule naturellement beaucoup plus vite lorsqu'il n'a pas à représenter des chiffres après la virgule. Si vous n'avez donc pas besoin d'une telle précision en décimales, vous pouvez marquer le nom de vos variables numériques de façon à ce que l'ordinateur sache qu'il peut travailler, avec ces variables, en faible précision. Il vous suffit pour cela de placer un signe de pourcentage à la fin du nom de variable si vous n'avez besoin d'aucun chiffre après la virgule.

Par exemple :     `st%=2/7`  
                      `print st%`

donnera comme résultat : 0.

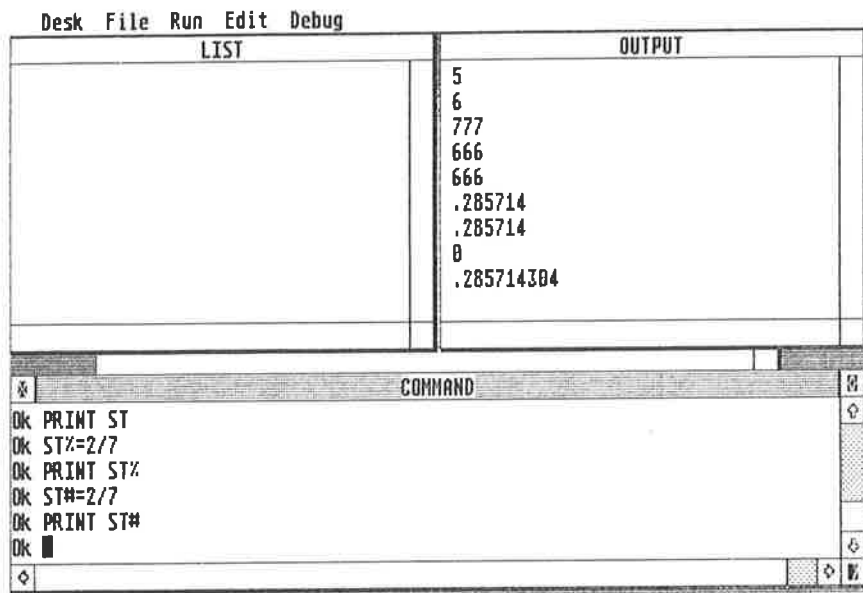
Deux indications en passant :

1. L'origine américaine du BASIC fait que la 'virgule' est représentée en informatique par un 'point décimal' séparant la partie entière d'un nombre de ses décimales.
2. Pour les nombres inférieurs à 1, comme sur les calculatrices de poche, votre ordinateur ne marque pas de 0 avant le point décimal. Il affichera donc toujours .56 pour 0.56 par exemple. Vous pouvez, pour vos entrées, utiliser toutefois les deux façons d'écrire.

Si toutefois vous aviez besoin dans vos calculs d'une plus grande précision que les 6 chiffres après la virgule que nous avons obtenus jusqu'ici, vous pouvez réclamer un plus grand nombre de décimales en faisant terminer le nom de variable par un dièse.

Par exemple:     `st#=2/7`  
                      `print st#`

Résultat : 0.285714304



Avec ce mode de calcul, le résultat n'est cependant qu'approximatif à partir du sixième chiffre après la virgule.

Il convient donc de bien réfléchir, lorsque vous voulez faire effectuer des divisions, à ce qui est le plus important pour vous : l'économie de place mémoire et la vitesse de calcul ou la précision de calcul ! L'intéressant est que, grâce aux différentes indications de précision, nous pouvons utiliser trois fois le même nom de variable.

Pour vous en assurer, entrez :

print st	Résultat : 0.285714
print st%	Résultat : 0
print st#	Résultat : 0.285714304

Pour le traitement de variables représentant des nombres entiers, il faut toutefois encore tenir compte du point suivant : avec ce mode de stockage de variables, les nombres utilisés ne doivent pas être supérieurs à 32767 ni inférieurs à -32767. Dans le cas contraire en effet, votre ordinateur ST retiendra des nombres négatifs à la place des nombres positifs et inversement des nombres positifs à la place des nombres négatifs.

Nous pouvons encore ajouter un dernier caractère à la fin d'un nom de variable : le point d'exclamation. Ce caractère indique qu'il faut calculer en simple précision, c'est-à-dire avec six chiffres après la virgule.

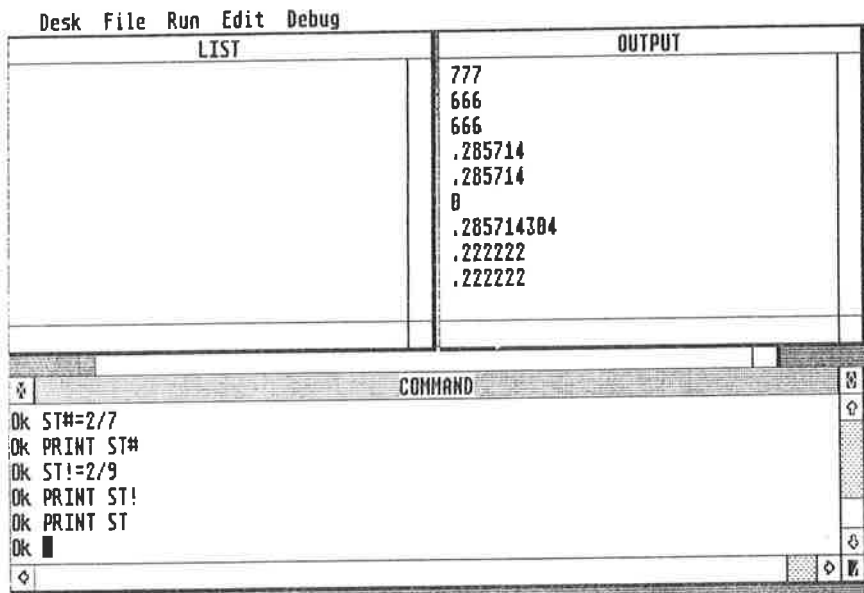
Vous vous demandez certainement quelle différence cela fait par rapport aux noms de variable sans additif, comme ST par exemple. C'est une bonne question. Les deux noms de variable ST et ST! se réfèrent au même mode de stockage des nombres. Il s'agit en fait de la même variable qui peut donc s'écrire de deux façons différentes. La seconde façon, ST !, vous permet de marquer nettement que la variable correspondante est utilisée en simple précision.

```
st!=2/9  
print st!
```

Résultat : 0.222222

```
print st
```

Résultat : 0.222222



Comme la variable ST ! est identique à la variable ST, le contenu de la variable ST est également modifié puisqu'il prend la même valeur que la variable ST !.

Nous en avons assez dit pour le moment sur le stockage des variables numériques. Voyons maintenant comment stocker également des textes dans la mémoire du ST.

### 3.5.3. Variable de texte, variable chaîne de caractères ou variable alphanumérique

Vous vous demandez peut-être pourquoi nous consacrons une partie spéciale au stockage de textes dans des variables. Pourquoi ne pourrait-on pas stocker des textes également dans les types de variable que nous avons étudiés jusqu'ici ?

Essayons. Voyons si nous pourrions stocker le mot 'ce' dans la variable ST que nous connaissons déjà :      st=ce

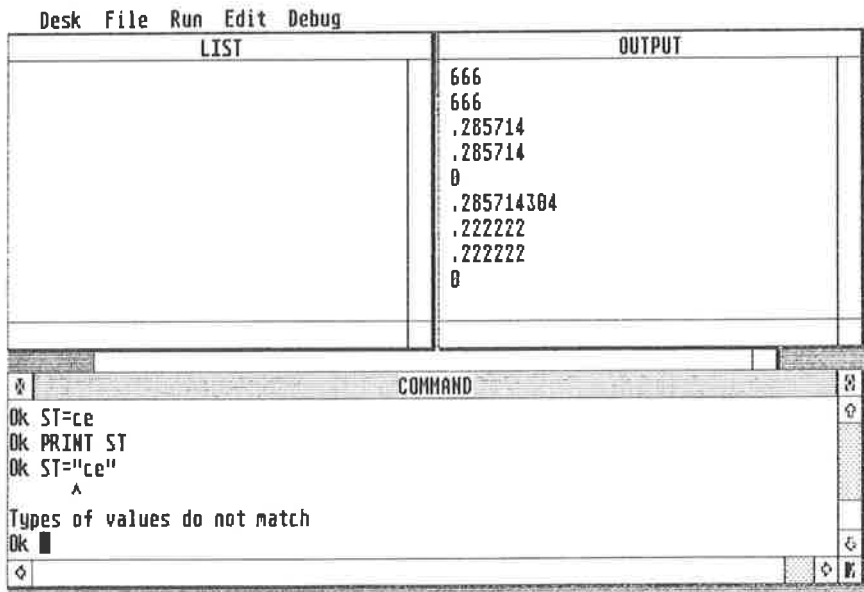
Voyons ce que contient maintenant la variable ST :

print st                      Résultat : 0

Ca ne marche pas. En fait, lors de cette entrée, votre ST a compris le mot 'ce' comme étant le nom d'une autre variable. Comme aucune valeur n'avait encore été, et pour cause, affectée à cette variable, l'ordinateur a considéré qu'elle contenait 0. Il a donc copié le contenu de cette variable 'ce', soit 0, dans notre variable ST. C'est comme cela que la valeur de ST est passée à 0.

Souvenez-vous de la section sur l'instruction PRINT : lorsque nous voulons sortir des textes sur l'écran, il nous faut les encadrer de guillemets. Essayons de faire la même chose pour affecter des textes à des variables :

st="ce"



Résultat :

Un nouveau message d'erreur, 'Types of value do not match', ce qui veut dire 'les types de valeur ne se correspondent pas'.



L'ordinateur veut vous dire avec ce message que vous n'avez pas utilisé le bon type de variable car la variable numérique ST ne convient pas au stockage de texte !

C'est pourquoi nous devons utiliser ici un nouvel additif pour marquer les noms de variables : le signe dollar, \$.

Si vous placez ce caractère à la fin d'un nom de variable, celle-ci sera considérée comme une variable chaîne de caractères. On dit aussi variable alphanumérique. Une telle variable peut stocker des textes pouvant comporter jusqu'à 255 caractères. Essayons donc à nouveau l'exemple précédent, cette fois avec le type de variable qui convient :

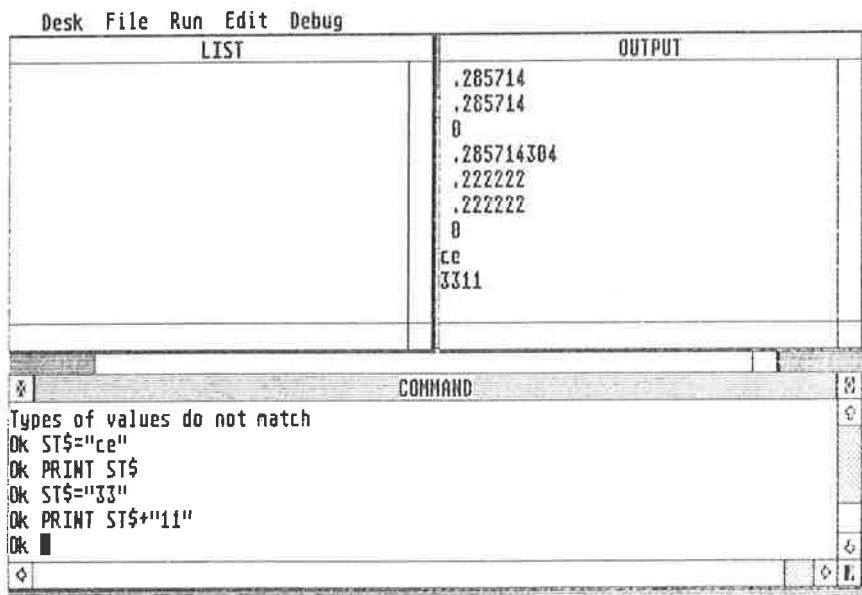
```
st$="ce"  
print st$
```

Résultat : ce

Vous pouvez naturellement également stocker des nombres dans une variable alphanumérique, à condition bien sûr que ces nombres soient également placés entre guillemets. Sinon vous obtiendrez à nouveau à l'écran le message d'erreur: 'Types of value do not match'. D'autre part, il n'est pas possible de calculer directement avec une variable alphanumérique. Essayez par exemple :

```
st$="33"  
print st$+"11"
```

Résultat : 3311



On n'a donc pas additionné ici les nombres arithmétiquement mais simplement ajouté un texte à la suite de l'autre : la variable `st$` avec le texte "33" a fusionné avec le texte entré, "11", pour donner 3311.

Nous avons indiqué plus haut que les variables de texte ne peuvent comprendre 'que' 255 caractères. 'ce' en comporte par exemple 2.

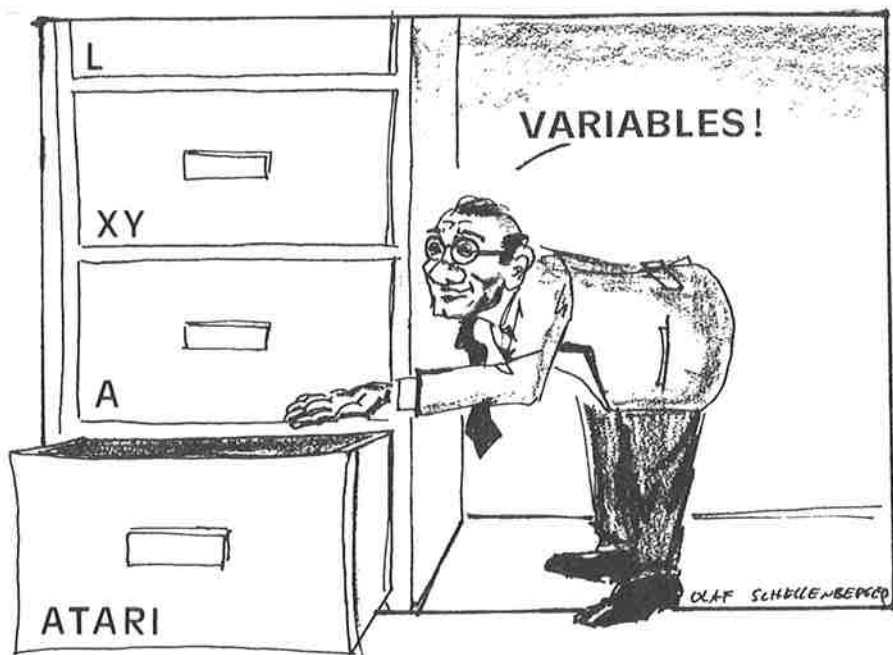
Vous comprenez donc tout de suite qu'une page de format A4, comprenant environ 2000 caractères, ne rentrera pas dans une variable de texte unique mais qu'il faudra au moins 8 variables de texte pour stocker une telle page.

#### Encore deux indications :

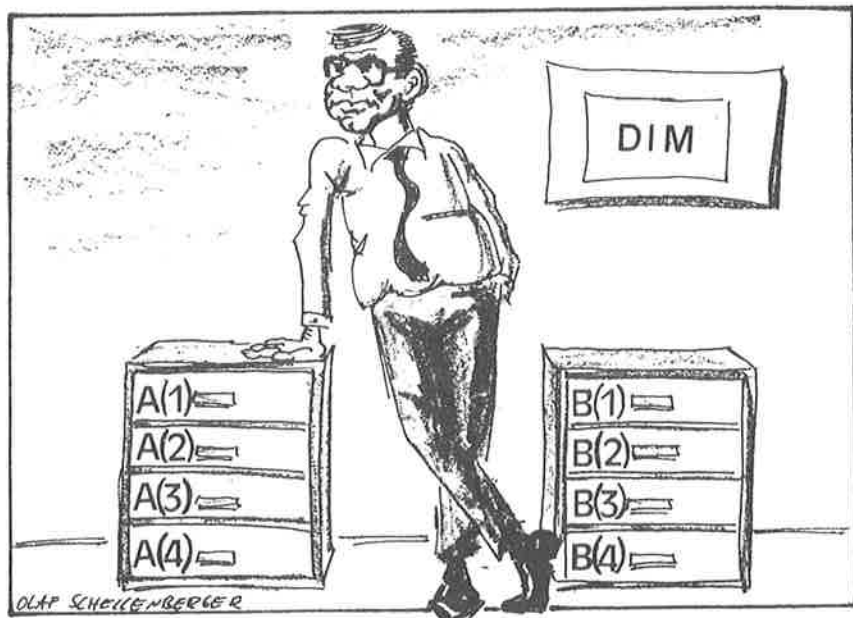
1. Lorsque vous entrez l'instruction `CLEAR`, toutes les valeurs de variables numériques et de texte placées dans votre ordinateur sont effacées. Il est donc recommandé de toujours entrer l'instruction `CLEAR` avant de commencer un nouveau travail sur l'ATARI ST.

Notez par ailleurs que le simple fait d'entrer une nouvelle ligne de programme ou de lancer un programme efface également toutes les valeurs de variables pouvant être stockées dans la mémoire.

2. Nous vous avons indiqué que vous pouvez stocker trois différentes valeurs numériques pour un même nom de variable, si ce nom de variable se termine par un caractère final chaque fois différent: % = sans décimales, ! = avec 6 décimales, c'est-à-dire simple précision, # = avec 9 décimales, c'est-à-dire double précision. Toujours avec le même nom de variable, terminé par \$, vous pouvez encore stocker un texte. Mais pour un nom de variable donné avec un caractère final donné, le ST ne retient jamais qu'une seule valeur, c'est-à-dire le nombre ou le texte que vous avez affecté en dernier à la variable correspondante. Chaque fois que vous affectez une nouvelle valeur à une variable, l'ancienne valeur de cette variable est instantanément annulée.



### 3.5.4. Variable dimensionnée



Nous venons donc d'indiquer que nous pouvons utiliser quatre variables différentes avec le même nom et pas plus : chaîne de caractères avec \$ pour les textes, double précision avec #, simple précision avec ! ou sans rien et nombres entiers avec %.

L'instruction DIM nous permet cependant de fixer pour une variable ce qu'on appelle une dimension. Dans ce cas, il nous sera possible, à condition d'utiliser le nom de cette variable avec ce qu'on appelle un indice, d'utiliser 100, voire 1000 valeurs différentes avec le même nom de variable.

Cet indice doit être ajouté, entre parenthèses, à la suite du nom de variable, après le dimensionnement. DIM ST(1000), entré en début de programme, signifierait donc qu'il y aura 1000 variables différentes sous le même nom ST. Chacune sera donc dotée d'un indice différent et pourra donc également recevoir une valeur numérique différente : ST(1), ST(2) ... jusqu'à ST(1000).

Cette méthode vous permet de ne pas avoir à chercher constamment de nouveaux noms de variable pour chaque situation de programme. Vous pouvez ainsi conserver le nom de base de la variable. Cette méthode vous permet par ailleurs également d'appeler différentes valeurs d'une variable sans devoir définir chaque fois un nouveau nom de variable. Vous pourriez donc fort bien utiliser à nouveau une variable comme indice d'une variable dimensionnée. Cet indice pourrait alors croître progressivement dans une boucle, ce qui vous permettrait de sortir toutes les valeurs contenues dans la variable dimensionnée.

Exemple : sortir les 100 noms qui ont été stockés :

```
10 dim st$(100)
...
100 for n=1 to 100
110 print st$(n)
120 next n
```

Supposons que vous ayez stocké 100 noms différents dans la variable dimensionnée ST\$, ces quatre lignes suffiraient tout à fait pour les faire tous sortir sur l'écran.

Si nous n'avions pas recours au dimensionnement, nous pourrions quand même sortir ces 100 noms mais il nous faudrait utiliser une ligne d'instruction différente pour chaque variable à sortir :

```
10 print st$
20 print st1$
30 print st2$
...
```

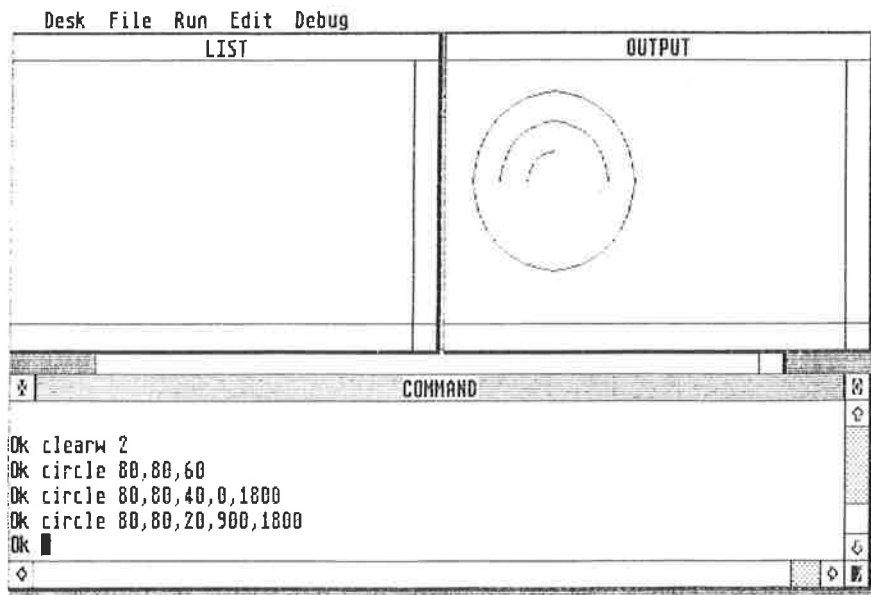
Un supplice sans pareil.

### 3.6. INSTRUCTIONS GRAPHIQUES INTERESSANTES DU BASIC ST

Nous connaissons maintenant les instructions BASIC les plus importantes. Avant de taper de longs programmes, étudions brièvement quelques instructions graphiques intéressantes que nous vous invitons simplement à essayer en laissant libre cours à votre imagination. Nous nous intéresserons ensuite à quelques aides pratiques qui facilitent la programmation.

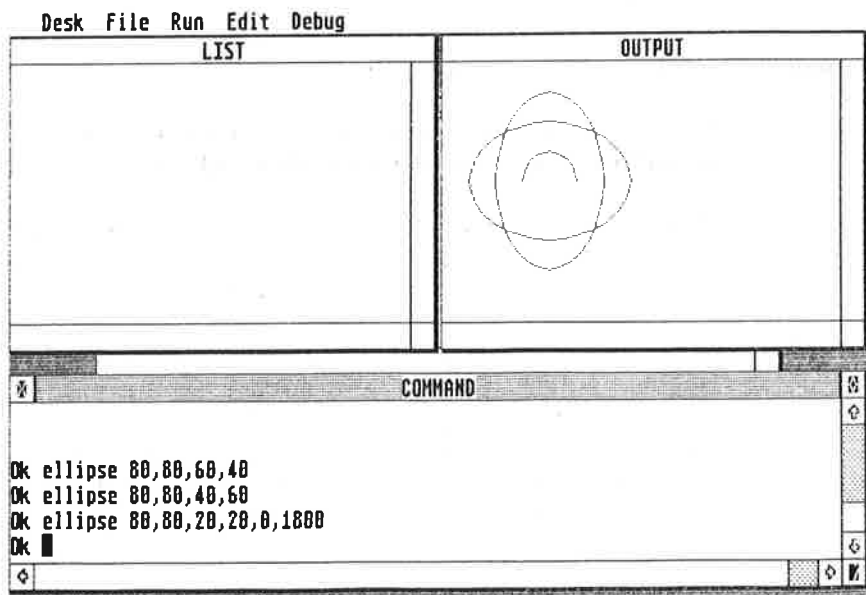
#### 3.6.1. CIRCLE A,B,C,D,E

Cette instruction vous permet de faire dessiner un cercle sur l'écran OUTPUT. A et B représentent respectivement les coordonnées sur l'axe des X et sur l'axe des Y du centre du cercle; C déterminer le rayon du cercle alors que D et E indiquent le début et la fin d'une section de cercle



### 3.6.2. ELLIPSE A,B,C,D,E,F

Cette instruction dessine une ellipse sur l'écran OUTPUT. A et B représentent les coordonnées X et Y du rayon X, C et D déterminent le rayon Y, E et F indiquent le début et la fin d'une section d'ellipse.



### 3.6.3. OPENW A

Instruction d'ouverture d'une fenêtre donnée (0 = EDIT, 1 = LIST, 2 = OUTPUT, 3 = COMMAND).

### 3.6.4. CLOSEW A

Instruction de fermeture d'une fenêtre donnée (0 = EDIT, 1 = LIST, 2 = OUTPUT, 3 = COMMAND).

### 3.6.5. FULLW A

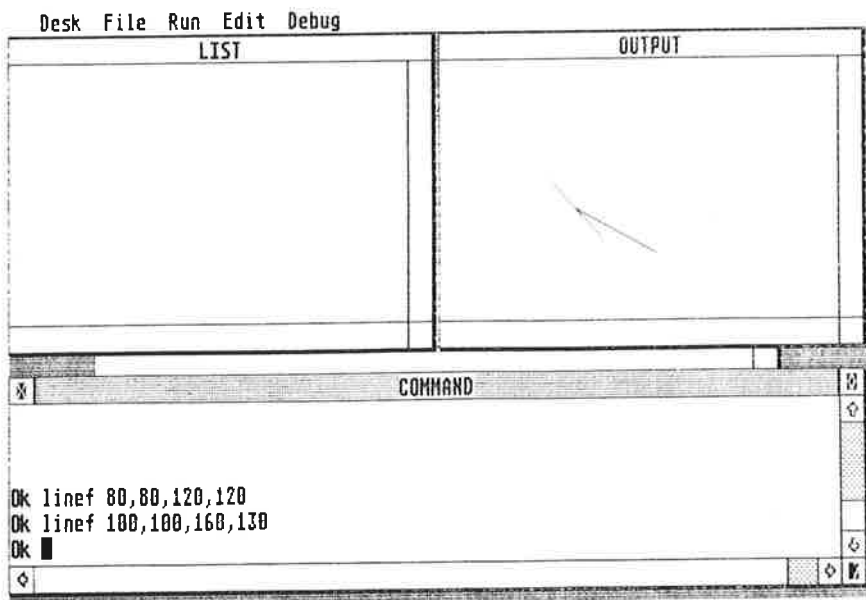
Instruction d'ouverture d'une fenêtre donnée sur l'écran tout entier (0 = EDIT, 1 = LIST, 2 = OUTPUT, 3 = COMMAND).

### 3.6.6. GOTOXY A,B

Le curseur de texte est positionné dans l'emplacement défini par les coordonnées d'écran A et B.

### 3.6.7. LINEF A,B,C,D

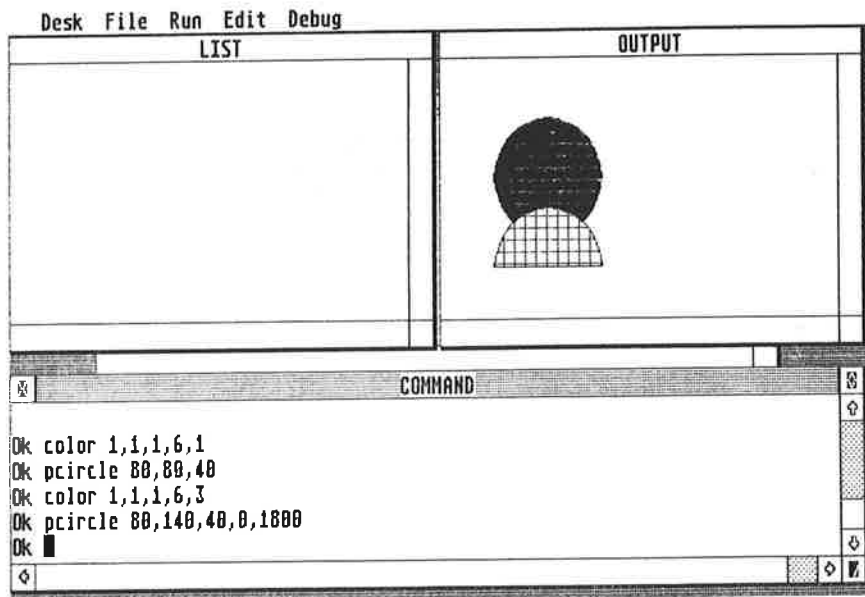
Dessin d'une ligne, des coordonnées A,B aux coordonnées C,D, sur l'écran OUTPUT.



### 3.6.8. PCIRCLE A,B,C,D,E

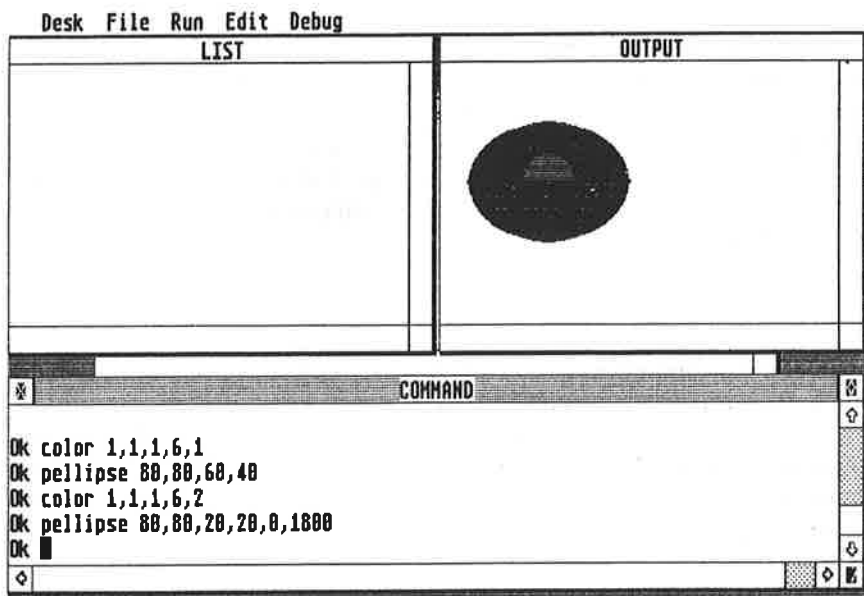
Dessiner un cercle sur l'écran OUTPUT, A et B indiquent les coordonnées X et Y du centre, C détermine le rayon du cercle, D et E indiquent le début et la fin d'une section de cercle. Le cercle ou la section de cercle sont peints avec la couleur indiquée auparavant, dans une instruction COLOR: COLOR couleur de texte, couleur de remplissage, couleur de ligne, style, indice).





### 3.6.9. PELLIPSE A,B,C,D,E,F

Cette instruction dessine une ellipse sur l'écran OUTPUT. A et B représentent les coordonnées X et Y du rayon X, C et D déterminent le rayon Y, E et F indiquent le début et la fin d'une section d'ellipse. L'ellipse ou la section d'ellipse sont peints avec la COLOR indiquée auparavant (voir 3.6.8.).



### **3.7. AIDES PRECIEUSES**

#### **3.7.1. Aides à la programmation**

Les instructions suivantes vous rendront de grands services dans votre travail de programmation. Elles doivent être entrées en mode direct, sans numéro de ligne.

#### **AUTO**

Après que cette instruction ait été entrée, la programmation s'effectuera avec numérotation automatique des lignes, c'est-à-dire que vous n'aurez plus à entrer vous-même les numéros de ligne en ordre croissant; chaque fois que vous terminerez une ligne en appuyant sur la touche d'entrée, le prochain numéro de ligne sera automatiquement affiché.

Si vous faites suivre l'instruction AUTO d'un nombre, la numérotation sortie automatique de numéros de ligne commencera à partir du numéro de ligne correspondant à ce nombre. AUTO 100 fera donc commencer la numérotation automatique à la ligne 100. Si vous ajoutez encore une virgule et un autre nombre, cette valeur sera interprétée comme intervalle entre les numéros de ligne. Vous pouvez sortir de cette fonction en appuyant sur les touches 'CONTROL' et 'G'.

### DELETE

Suppression de lignes de programme. DELETE vous permet de supprimer une ligne unique, par exemple DELETE 20 pour supprimer la ligne 20, ou un plus grand nombre de lignes, par exemple DELETE 10-100 pour supprimer toutes les lignes de 10 à 100 incluses. DELETE -20 supprimera toutes les lignes du début du programme jusqu'à la ligne 20 incluse.

### LIST

Affichage du programme sur l'écran. LIST, suivi d'un numéro de ligne permet aussi de faire afficher sur l'écran un ligne bien précise. Par exemple, LIST 10 ne fera afficher que la ligne 10; LIST 10-100 affichera sur l'écran les lignes de 10 à 100 incluses.

### RENUM

Renumérotation rétroactive des lignes en mémoire. Cette instruction peut être complétée par trois valeurs ou paramètres : RENUM 10,100,5 signifiera: renuméroter le programme à partir de la ligne 100 qui deviendra la ligne 10, les autres lignes devant être numérotées de 5 en 5.

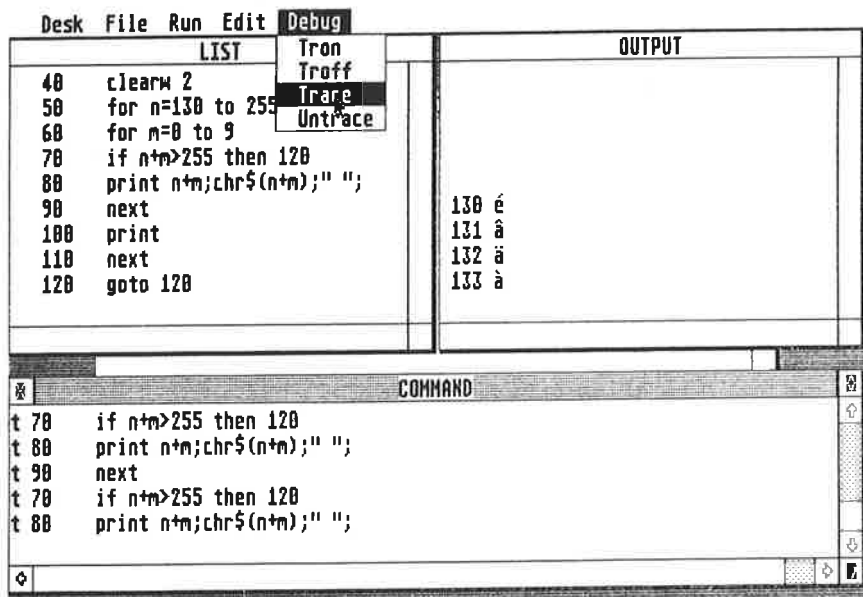
100 100  
5 5

### 3.7.2. Débugger

Quelques mots sur les aides au 'débuggage', c'est-à-dire à la détection et à l'élimination des erreurs.

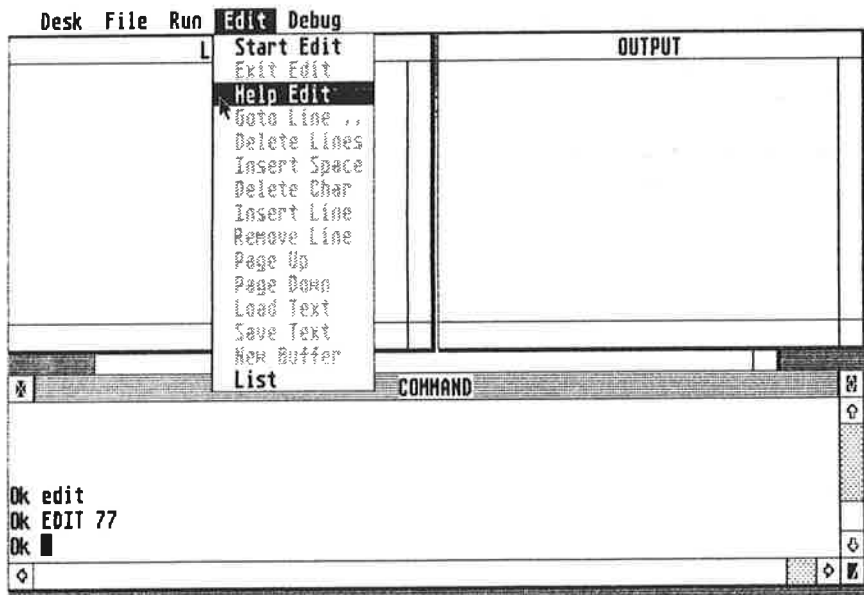
1. On peut interrompre le programme en cours avec 'Control-G'. On peut ensuite faire poursuivre l'exécution du programme avec CONT ou simplement en appuyant sur la touche d'entrée (Return ou Enter).
2. On peut interrompre et arrêter définitivement le programme en cours avec 'Control-C'. Il n'est plus possible de reprendre l'exécution du programme.
3. L'instruction TRON vous permet de suivre l'exécution du programme ligne par ligne grâce à l'affichage du numéro de la ligne actuellement exécutée. TRON est désactivé par TROFF.
4. L'instruction TRACE vous permet de suivre l'exécution du programme ligne par ligne grâce à l'affichage non pas seulement du numéro de la ligne actuellement exécutée mais de toute la ligne. TRACE est désactivé par UNTRACE.
5. L'instruction FOLLOW A affiche sur l'écran COMMAND la valeur actuelle de la variable A pendant toute la durée de l'exécution du programme.

Les instructions TRACE, UNTRACE, TRON et TROFF se trouvent dans le menu Debug mais elles peuvent bien sûr également être entrées au clavier.



### 3.7.3. EDIT vous aide dans le débogage

Vous vous êtes certainement déjà demandé pourquoi nous avons jusqu'ici aussi systématiquement délaissé la fenêtre EDIT du BASIC. La fenêtre EDIT est une sorte de papier brouillon sur lequel vous pouvez corriger votre programme à votre guise. Vous appelez la fenêtre EDIT et par là-même le mode EDIT soit en entrant EDIT au clavier soit en marquant par un clic le champ 'Start Edit' du menu Edit. EDIT 20 est aussi possible, dans ce cas seule la ligne 20 sera affichée dans la fenêtre EDIT. Vous trouvez dans le menu EDIT toute une série d'instructions avec lesquelles on peut faire un excellent travail :



Ces instructions signifient, de haut en bas :

Lancement du mode EDIT

Sortie du mode EDIT

Fenêtre d'aide pour le mode EDIT (voir l'illustration)

Aller en ligne ... (numéro)

Supprimer ligne

Insérer un espace

Supprimer caractère

Insérer ligne

Supprimer ligne

Tourner une page vers le début (du listing)

Tourner une page vers la fin (du listing)

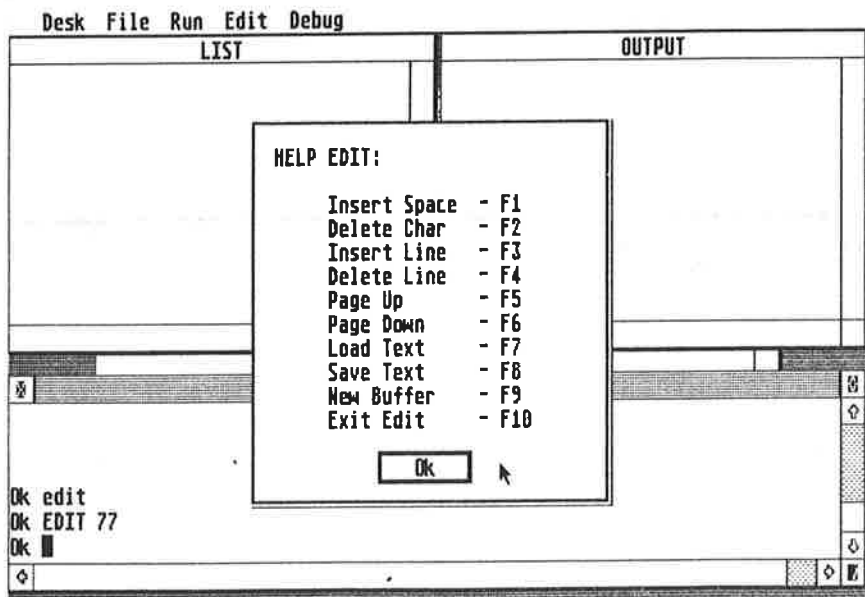
Charger le contenu d'une mémoire provisoire

Sauver le contenu d'une mémoire provisoire

Supprimer mémoire provisoire

Lister le programme

On peut en outre éditer (corriger) les programmes au moyen des touches de fonction. Avec l'instruction "Help Edit" (dans le menu Edit), vous pouvez faire afficher la liste de ces touches de fonction.

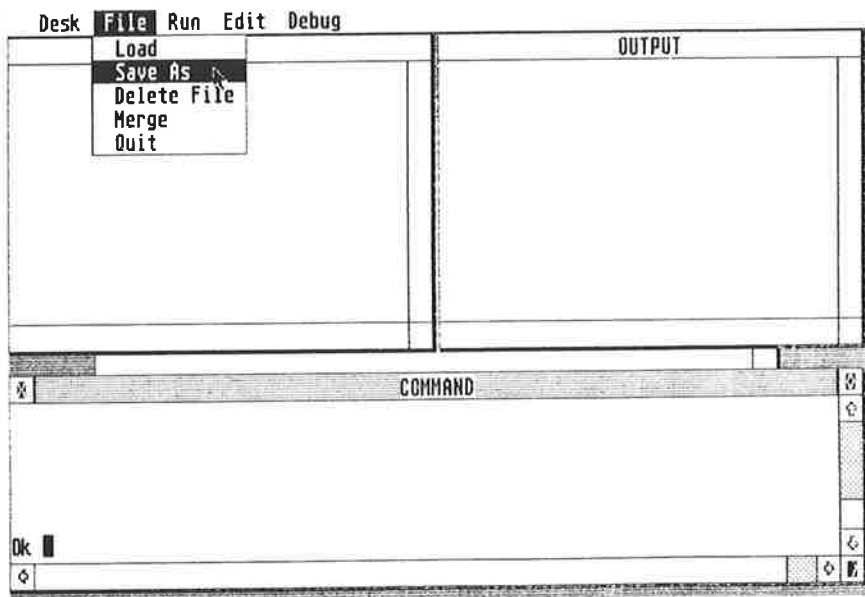


L'édition est très facile. Après avoir appelé la fenêtre EDIT, vous y trouvez le listing du programme. Vous pouvez alors entreprendre des corrections ligne par ligne. Ce qui est modifié sera toujours affiché en gris. Avant de confirmer la modification d'une ligne avec la touche d'entrée, vous pouvez examiner votre travail, le sauvegarder (Save Text) et aussi le traiter en utilisant les fonctions indiquées ci-dessus. Le mieux est donc que vous écriviez un petit programme. Vous pouvez peut-être taper tout simplement un des programmes des sections précédentes. Essayez ensuite les possibilités d'édition sur ce programme. Vous verrez que le ST se révèle, ici aussi, très pratique.

### 3.8. LE TRAVAIL AVEC LE LECTEUR DE DISQUETTE

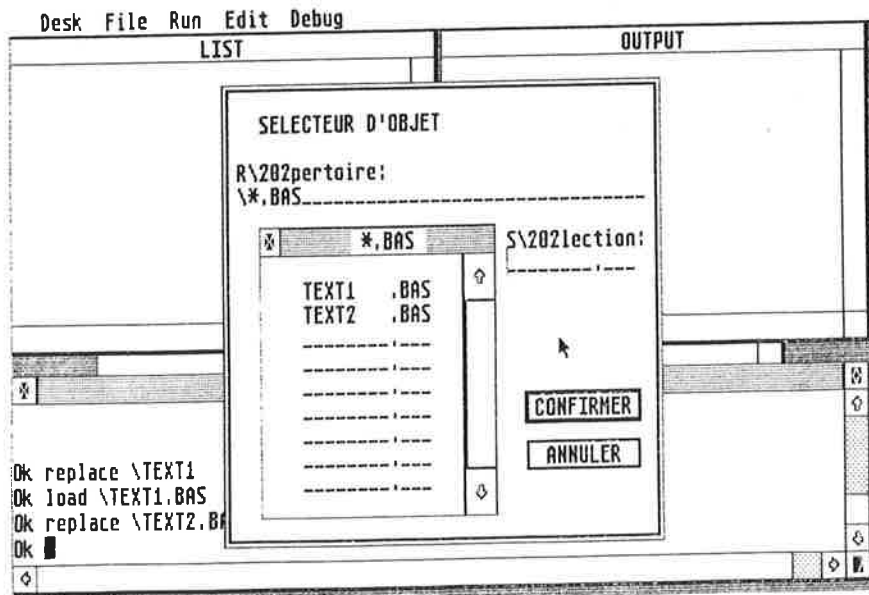
Vous vous souvenez que nous avons vu plus haut qu'il fallait toujours formater une disquette avant de pouvoir l'utiliser. Une fois que vous disposez d'une disquette formatée, vous pouvez y placer non seulement les langages de programmation et le système d'exploitation mais aussi des programmes personnels comme par exemple ceux que vous aurez tapés à partir de ce livre. Le menu pull down FILE vous offre toutes les possibilités nécessaires pour cela.

Avez-vous tapé un programme que vous avez l'intention d'utiliser plus d'une fois ? Alors appelons tout simplement le menu File.





### 3.8.1. LOAD et SAVE



Avez-vous une disquette formatée sous la main ? Alors vous pouvez maintenant marquer 'SAVE AS' (sauver sous le nom) par un clic. Nous pouvons taper le nom dont nous voulons doter notre programme. Tapez par exemple TEST1 et sous le mot 'sélection' apparaîtra le nom choisi. Il suffit maintenant de marquer par un clic le champ OK et votre programme sera sauvegardé. Vous pourrez alors le rappeler à tout moment.

Comment ? La fonction LOAD (=charger) fait à nouveau apparaître l'image ci-dessus qui nous est maintenant familière. Vous pouvez maintenant voir votre programme à gauche dans la fenêtre. Si vous avez déjà sauvegardé plusieurs programmes, sélectionnez le programme voulu avec la souris, faites un clic et il apparaîtra dans le champ sélection. Si vous marquez maintenant le champ OK par un clic, votre programme sera chargé en mémoire.

### **3.8.2. DELETE FILE, MERGE et QUIT**

**DELETE FILE** nous permet de supprimer sur la disquette un programme qui y a été sauvegardé auparavant. Cette fonction correspond au fait d'amener un symbole sur la corbeille à papier. L'image familière apparaît à nouveau, nous marquons par un clic le fichier à supprimer qui apparaît sous 'sélection' et qui sera supprimé après la confirmation OK.

**MERGE** permet de fusionner un programme se trouvant encore sur la disquette avec le programme actuellement en mémoire. Attention avec cette instruction !!! Après exécution de cette fonction, vous n'avez plus qu'un seul programme qui contient en lui-même les deux programmes indépendants.

Cela vous permet donc d'accrocher un programme à la suite d'un autre. Exemple : le programme 1 contient les numéros de ligne 10 à 150. Le second programme devra commencer par un numéro de ligne supérieur ou égal à 160, sinon vous obtiendrez une purée de programme qu'il vous sera difficile de dénouer.

**QUIT** vous permet de quitter le BASIC et de revenir au bureau GEM.

Bien, vous connaissez donc maintenant un certain nombre des instructions les plus importantes du fantastique BASIC ST. Faites vous-même quelques essais avec ces instructions. Nous sommes bien sûr loin de pouvoir traiter de toutes les instructions de ce BASIC très riche. Si vous avez pris goût au BASIC, nous ne pouvons que vous recommander de compléter vos connaissances avec des ouvrages spécialisés sur ce sujet. Nous n'en dirons donc pas plus dans cet ouvrage consacré aux débuts sur l'ATARI ST.

## **3.9. LES PREMIERS PROGRAMMES**

Tout va bien jusqu'ici ? Alors nous allons vraiment pouvoir commencer. Vous trouverez dans les pages suivantes des programmes BASIC qui se composent presque exclusivement du vocabulaire que nous avons appris dans les sections précédentes. Le listing de chaque programme est suivi d'une explication de lignes détaillée.

En plusieurs endroits vous rencontrerez telle ou telle instruction que nous n'avons pas encore vue dans cet ouvrage. Dans ce cas, nous sommes entré un peu plus dans le détail dans le texte d'explication.

Pour quelqu'un dont le BASIC est la langue de programmation quotidienne, les listings que nous vous présentons ne posent aucun problème.

Mais le problème, avec les longs programmes, est surtout que les débutants en informatique se lancent dans la copie de listings de plusieurs pages sans rien comprendre du tout de ce qu'ils vont recopier pendant des pages entières. Ce ne devrait donc déjà plus être votre cas si vous avez étudié ce livre. Il ne faut pas s'étonner en tout cas que la recherche des erreurs soit une tâche pratiquement impossible. Or nous sommes tous, vous aussi, exposés à faire des fautes de frappe. Mais on fait bien sûr d'autant plus de fautes qu'on comprend moins ce qu'on tape. Malheureusement, un ordinateur ne pardonne pas la moindre faute de frappe.

Encore deux indications :

1. En entrant les programmes suivants sur votre ATARI ST, n'oubliez surtout pas d'actionner la touche d'entrée à la fin de chaque ligne.
2. Dans les explications des programmes ligne par ligne, nous avons systématiquement ignoré les lignes REM.

### 3.9.1. Agenda téléphonique



Le programme numéro 1 vous montre comment votre ST peut être utilisé comme agenda téléphonique.

Le programme fonctionne de la façon suivante :

On vous demande d'abord le nombre de correspondants à entrer. Vous entrez ensuite les noms et les numéros de téléphone. Vous pourrez ensuite entrer un nom quelconque et votre ST retrouvera le numéro de téléphone correspondant.

```
10 REM agenda téléphonique
20 CLEARW 2
30 INPUT "Combien de numéros de téléphone ";A
40 REM Dimensionnement pour les numéros et les noms
50 DIM NA$(A),NU$(A)
60 CLEARW 2
70 REM Création de l'agenda téléphonique
80 FOR N=1 TO A
```

```
90 INPUT "Nom ";NA$(N)
100 INPUT "Numéro ";NU$(N)
110 PRINT
120 NEXT N
130 REM Recherche de numéros de téléphone
140 CLEARW 2
150 INPUT "Quel numéro cherchez-vous";A$
160 FOR N=1 TO A
170 REM Nom trouvé alors sortie du numéro
180 IF A$=NA$(N) THEN GOSUB 210
190 NEXT N
200 GOTO 140
210 REM Sous-programme 'sortie du numéro'
220 PRINT "Le numéro de téléphone: ";NU$(N)
230 PRINT
240 INPUT "Veuillez appuyer sur la touche d'entrée ";B$
250 RETURN
```

Examinons maintenant le programme ligne par ligne :

Ligne 20 : L'écran OUTPUT est vidé.

Ligne 30 : Le nombre de numéros de téléphone qui devront être entrés par la suite est placé dans le tiroir A. Ce nombre correspond donc également au nombre de correspondants.

Ligne 50 : On fixe ici le nombre de variables qu'on utilisera pour les numéros de téléphone, NU\$(A), et pour les correspondants, NA\$(A). Bien que les numéros de téléphone se composent surtout de chiffres, nous avons dû les stocker dans des variables de texte à cause du trait d'union éventuel (1-51 42 32 23) et du 0 qui ne doit pas disparaître s'il est le premier chiffre d'un numéro (05 14 32 23).

Ligne 60 : Vider à nouveau l'écran OUTPUT.

Lignes 70-120 : Demander le nom du correspondant (ligne 90) et son numéro de téléphone (ligne 100).

Comme il ne s'agit pas ici de demander le nom et le numéro d'un seul correspondant, nous avons choisi la méthode d'une boucle FOR ... NEXT pour faire exécuter plusieurs fois la même fonction mais avec des contenus de variables différents. Le nombre de parcours de la boucle est fixé grâce à la valeur que vous avez entrée dans le tiroir A en ligne 30.

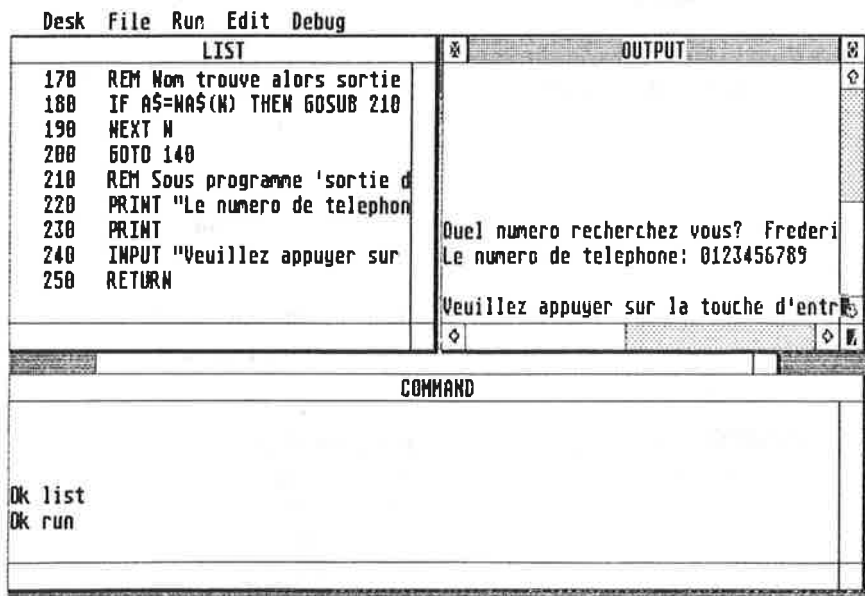
On vous demande donc des données autant de fois que nécessaire, jusqu'à ce que la valeur entrée pour la variable A soit atteinte.

Lignes 130-150 : Après vidage de l'écran, on vous demande le nom du correspondant voulu. Ce nom est stocké dans la variable A\$.

Lignes 160-190 : Une boucle FOR ... NEXT est à nouveau utilisée ici pour examiner tous les noms déjà sauvegardés. Si un de ces noms (NA\$(N)) correspond totalement au nom du correspondant voulu, le programme saute au sous-programme en lignes 210 et suivantes.

Que la recherche ait été ou non couronnée de succès, la ligne 200 lance à nouveau la même procédure d'interrogation.

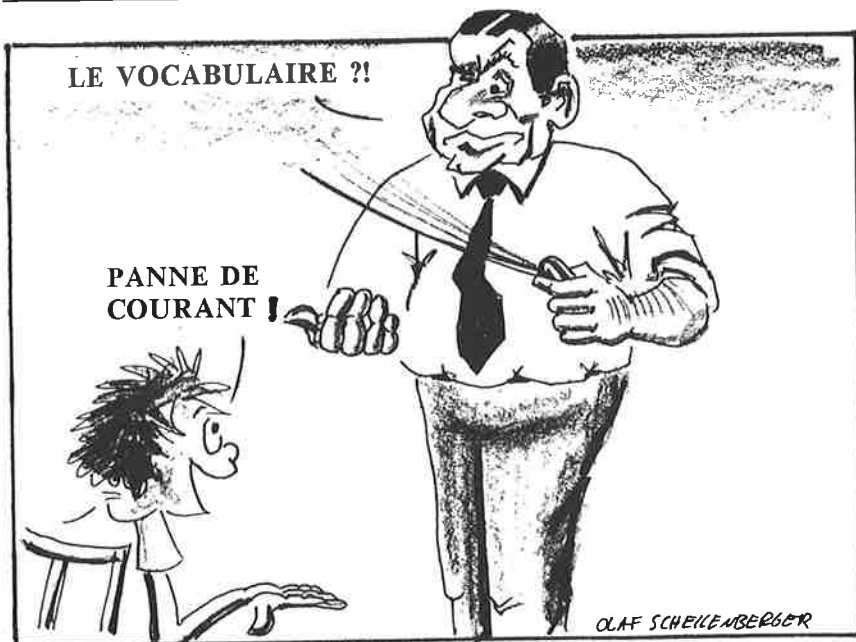
Lignes 210-250 : On saute à ce sous-programme uniquement si la variable A\$ qui contient le correspondant voulu et la variable NA\$(N) qui contient un des correspondants 'stocké' sont identiques. Dans ce cas, la ligne 220 sort le numéro de téléphone voulu.



La ligne 240 remplit une fonction particulière :

Elle bloque l'exécution du programme jusqu'à ce que vous ayez appuyé sur la touche d'entrée. Il pourrait arriver sinon que votre ST retourne trop rapidement au programme principal, vidant ainsi l'écran OUTPUT (ligne 140), avant que vous ayez eu le temps de lire le numéro de téléphone sorti sur l'écran.

### 3.9.2. Programme de vocabulaire



Ce programme transforme votre ST en entraîneur pour l'apprentissage du vocabulaire.

On vous demande d'abord le nombre de mots à stocker puis on vous demande d'entrer un mot anglais suivi du mot français correspondant.

Vous pouvez ensuite choisir si votre ST doit partir des mots français ou anglais pour l'interrogation.

Il tirera ensuite des mots au hasard et vous demandera d'entrer l'équivalent dans l'autre langue.

```
260 10 REM Programme de vocabulaire
    20 CLEARW 2
    30 INPUT "Combien de mots ";A
    40 DIM A$(A),B$(A)
    50 CLEARW 2
```



```
60 FOR N=1 TO A
70 INPUT "Mot français ";A$(N)
80 INPUT "Mot anglais ";B$(N)
90 PRINT
100 NEXT N
110 CLEARW 2
120 B$=""
130 INPUT "Français - anglais (oui) ";B$
140 IF B$="oui" THEN GOSUB 200 ELSE GOSUB 280
150 PRINT
160 B$=""
170 INPUT "Encore une fois (non) ";B$
180 IF B$="non" THEN END
190 GOTO 110
200 B=RND*A+1
210 CLEARW 2
220 PRINT A$(B);
230 AN$=""
240 INPUT AN$
250 IF AN$=B$(B) THEN PRINT "Juste!"
260 IF AN$<>B$(B) THEN PRINT "Faux!"
270 RETURN
280 B=RND*A+1
290 CLEARW 2
300 PRINT B$(B);
310 AN$=""
320 INPUT AN$
330 IF AN$=A$(B) THEN PRINT "Juste!"
340 IF AN$<>A$(B) THEN PRINT "Faux!"
350 RETURN
```

Examinons maintenant le programme ligne par ligne :

Ligne 20 : Vidage de l'écran OUTPUT.

Ligne 30 : Le nombre de mots à entrer par la suite dans chaque langue est placé dans le tiroir A.

Ligne 40 : On fixe ici le nombre de variables de texte pour les mots français (A\$(A)) et anglais (B\$(A)).

Ligne 50 : On vide à nouveau l'écran OUTPUT.

Lignes 60-100 : On demande ici le mot français (ligne 70) et son équivalent anglais (ligne 80). Vous pouvez naturellement remplacer le français et l'anglais par n'importe quelles autres langues.

Comme nous ne demandons pas seulement un mot mais plusieurs, nous utilisons ici, comme dans le programme précédent, une boucle FOR ... NEXT. On demande les données autant de fois que nécessaire, jusqu'à ce que la valeur de la variable A soit atteinte.

Lignes 110-130 : Après le vidage de l'écran, on vous demande si l'interrogation doit se faire dans le sens français-anglais, auquel cas vous devez entrer 'oui' ou plutôt dans le sens anglais-français, auquel cas vous devez entrer n'importe quoi d'autre que 'oui'.

Votre réponse est stockée dans la variable de texte B\$.

Ligne 140 : On décide ici à quel sous-programme votre ST doit sauter. Si vous avez répondu 'oui' en ligne 130, votre ST sautera au sous-programme en ligne 190, c'est-à-dire à l'interrogation français-anglais, sinon il sautera au sous-programme en ligne 260, c'est-à-dire à l'interrogation anglais-français.

Comme vous le voyez nous utilisons ici l'instruction IF ... THEN ... ELSE que nous connaissons déjà. La ligne de programme 140 pourrait être traduite en français de la façon suivante : si B\$="oui" alors sauter au sous-programme en ligne 190, sinon sauter au sous-programme en ligne 260.

Avec notre interrogation IF ... THEN, il y a toujours deux possibilités : soit l'affirmation avec IF est vraie, auquel cas l'instruction immédiatement à la suite de THEN est exécutée, soit elle est fausse, auquel cas l'instruction à la suite de ELSE (sinon) sera exécutée. ELSE est cependant facultatif. Si vous l'omettez dans une instruction IF et si l'affirmation est fausse, l'exécution du programme se poursuivra à la ligne de programme suivante.

Lignes 150-190 : Après l'interrogation sur un mot, une ligne vide est produite sur l'écran en ligne 150, pour plus de clarté, puis on vous demande, en ligne 170, si vous voulez faire encore un exercice. Suivant votre entrée, le programme se termine en ligne 180, avec la nouvelle instruction END, qui signifie que l'ordinateur doit arrêter ici l'exécution du programme, ou bien le programme continue à l'interrogation en ligne 190.

Desk File Run Edit Debug		LIST	OUTPUT
270	RETURN		Mot Francais ? chien
280	B=RND*A+1		Mot Anglais ? dog
290	CLEARW 2		
300	PRINT B\$(B);		Mot Francais ? chat
310	ANS=""		Mot Anglais ? cat
320	INPUT AN\$		
330	IF AN\$=A\$(B) THEN PRINT "Jus		Mot Francais ? livre
340	IF AN\$>A\$(B) THEN PRINT "Fa		Mot Anglais ? book
350	RETURN		Mot Francais ? ■

COMMAND	
Ok list	
Ok run	

Lignes 200-220 : Voici à nouveau une nouvelle instruction : RND.  
Cette instruction produit un nombre aléatoire, c'est-à-dire tiré au hasard, qui sera compris entre 0 et 1.

La formule en ligne 200 a donc pour but de multiplier la valeur aléatoire, puis de lui ajouter 1, pour obtenir un nombre qui ait la même probabilité de représenter chacun des mots stockés en mémoire.

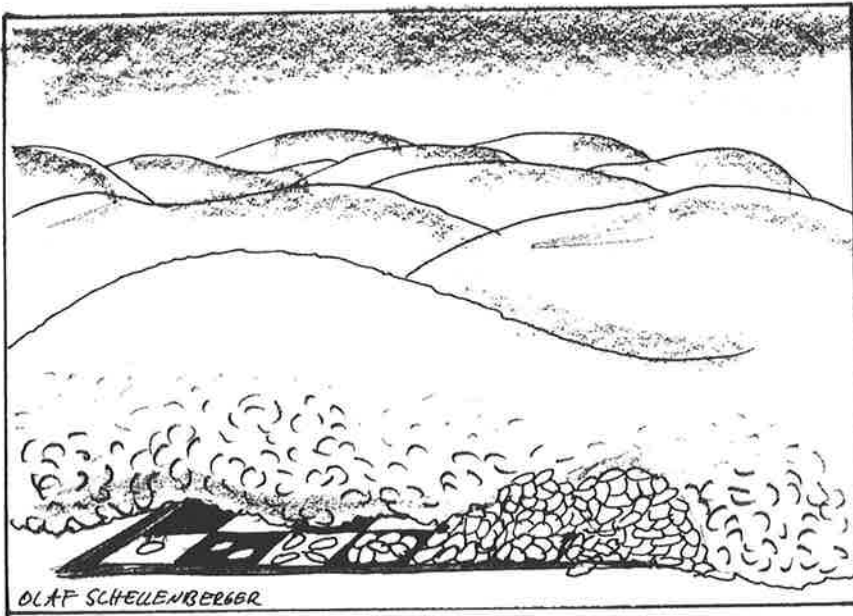
L'écran est ensuite vidé en ligne 210 et la ligne 220 sort le mot tiré au sort. Le point-virgule à la suite de l'instruction PRINT indique que la réponse à la question doit commencer directement à la suite de la sortie sur l'écran.

Lignes 230-270 : On examine dans ces lignes si vous avez répondu juste à la question posée, auquel cas les variables AN\$ et B\$(B) doivent avoir le même contenu, ou si vous avez donné une réponse erronée, auquel cas il n'y aura pas d'identité, en ligne 260, entre les variables AN\$ et B\$(B). On retourne ensuite au programme principal en ligne 270.

Lignes 280-300 : La formule en ligne 280 a donc pour but de multiplier la valeur aléatoire, puis de lui ajouter 1, pour obtenir un nombre qui ait la même probabilité de représenter chacun des mots stockés en mémoire. L'écran est ensuite vidé en ligne 290 et la ligne 300 sort le mot tiré au sort. Le point-virgule à la suite de l'instruction PRINT indique que la réponse à la question doit commencer directement à la suite de la sortie sur l'écran.

Lignes 310-350 : On examine dans ces lignes si vous avez répondu juste à la question posée, auquel cas les variables AN\$ et A\$(B) doivent avoir le même contenu, ou si vous avez donné une réponse erronée, auquel cas il n'y aura pas d'identité, en ligne 340, entre les variables AN\$ et A\$(B). On retourne ensuite au programme principal en ligne 350.

### 3.9.3. Le problème de l'échiquier



Le programme numéro 3 a pour but de vous montrer comment votre ST peut résoudre un problème de calcul en quelques secondes.

Le célèbre problème de l'échiquier tire son origine d'une légende selon laquelle un roi aurait un jour promis à un sage de lui offrir ce qu'il voudrait.

Le sage demanda alors un échiquier avec des grains de blé disposés de la façon suivante :

1 grain de blé sur la première case de l'échiquier, 2 grains sur la seconde, 4 sur la troisième, 8 sur la quatrième et ainsi de suite jusqu'à la 64ème case. Chaque case, à partir de la seconde, devrait donc recevoir deux fois le nombre de grains de blé placés sur la case précédente.

Bien sûr, à première vue, le roi crut ce sage très modeste. Un roi n'est pas à un millier de grains de blé près ?! Si vous connaissez déjà cette histoire, vous savez pourtant qu'il eut une grande surprise et qu'il fut bien incapable d'exaucer ce vœu. Vous n'en serez pas autrement étonné une fois que le programme aura terminé son travail et que vous aurez vu sur l'écran l'énormité du nombre de grains de blé obtenu. Sachez, en tout cas, que la production mondiale de blé actuelle ne suffirait pas à égaler ce nombre. Le sage avait donc exprimé un vœu irréalisable.

```
10 REM Le problème de l'échiquier
20 CLEARW 2
30 REM L'échiquier a 64 cases
40 FOR N=1 TO 64
50 PRINT "Case ";N
60 REM Formule d'élévation à la puissance
70 A=2^(N-1)
80 PRINT A;"grains de blé"
90 REM Le nombre total de grains est dans le tiroir B
100 B=B+A
110 PRINT "En tout ";B;" grains de blé"
120 PRINT
125 FOR I=1 TO 500: NEXT I
130 NEXT N
```

Le nombre de grains de blé deviendra tellement astronomique sur la fin que le nombre de grains de blé à placer sur la case 64 ne pourra plus être affiché. La dernière indication vous sera donc fournie pour la case 63. Vous pouvez aisément imaginer quel nombre immense on aurait pour la case 64.

Voici l'explication du programme :

Ligne 20 : Vidage de l'écran OUTPUT

Ligne 40 : On ouvre ici une boucle FOR ... NEXT pour pouvoir calculer toutes les cases l'une après l'autre.

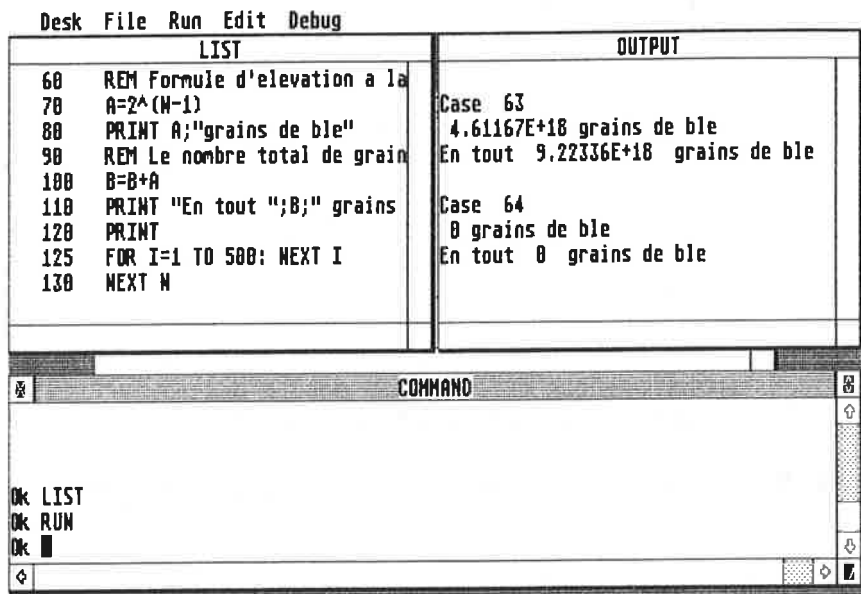
Ligne 50 : Cette ligne sert à informer le spectateur attentif car elle sort sur l'écran le numéro de la case actuellement traitée.

Ligne 70 : Cette formule de calcul permet d'obtenir le nombre de grains de blé sur une case donnée. La flèche vers le haut (^) est le signe d'élévation à la puissance.

Ligne 80 : On sort ici le nombre de grains de blé que recevra la case actuellement traitée.

Ligne 100 : On stocke dans la variable B le nombre de grains de blé et on additionne à ce nombre la valeur obtenue avec les cases déjà occupées.

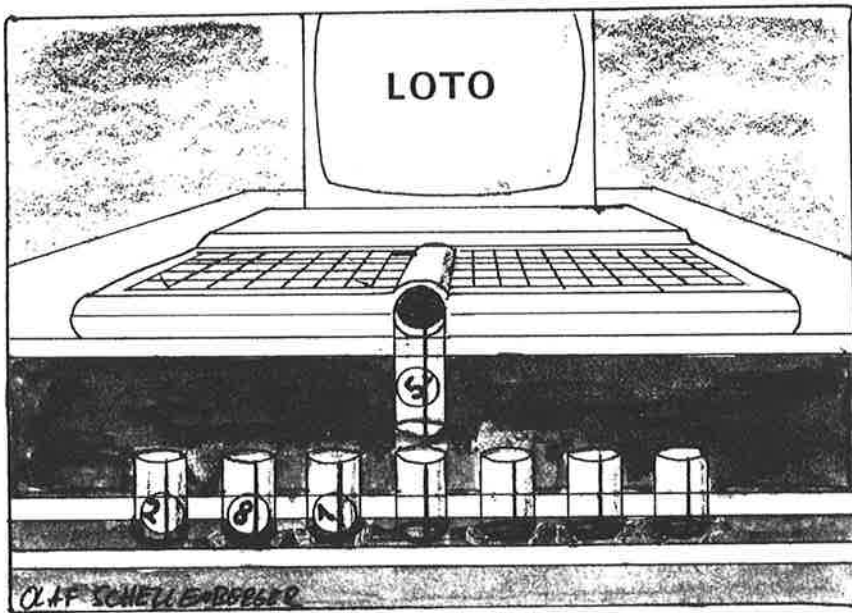
Ligne 110 : On sort ici le nombre total de grains de blé sur l'échiquier.



Lignes 120-130 : En ligne 120, une ligne vide est sortie sur l'écran OUTPUT, pour plus de clarté. En ligne 130, on saute à nouveau à la ligne 40 de façon à ce que les 64 cases soient calculées l'une après l'autre. Entre temps, la ligne 125 produit une pause avec une boucle FOR ... NEXT tournant à vide. Cette pause nous donne le temps de lire les indications qui nous sont données à l'écran.



### 3.9.4. Les chiffres du loto



Le programme numéro 4 a pour but de vous montrer comment le ST peut vous aider à remplir une grille de chiffres du loto, SANS GARANTIE bien entendu !

Dans ce programme, nous avons tout naturellement fait à nouveau appel à la fonction aléatoire RND du langage BASIC.

Le programme se déroule de la façon suivante: On vous demande d'abord si vous voulez jouer selon le principe du loto français (6 nombres plus un nombre complémentaire de 1 à 49) ou selon le principe du loto allemand du mercredi (7 nombres plus un nombre complémentaire de 1 à 39).

On tire ensuite 6 ou 7 nombres qui doivent cependant être comparés aux nombres déjà tirés avant d'être sortis pour que le même nombre ne puisse pas sortir plus d'une fois. Les nombres de la chance sont ensuite sortis sur l'écran.

```
10 REM nombres du loto
20 CLEARW 2
30 INPUT "Loto français (oui) ";A$
40 IF A$<>"oui" THEN GOTO 180
50 FOR N=1 TO 6
60 A%(N)=RND*49+1
70 NEXT N
80 FOR N=1 TO 6
90 FOR M=1 TO 6
100 IF N=M THEN GOTO 120
110 IF A%(N)=A%(M) THEN GOTO 50
120 NEXT M
130 NEXT N
140 FOR N=1 TO 6
150 PRINT A%(N);
160 NEXT N
165 PRINT
170 GOTO 30
180 FOR N=1 TO 7
190 A%(N)=RND*38+1
200 NEXT N
210 FOR N=1 TO 7
220 FOR M=1 TO 7
230 IF N=M THEN GOTO 250
240 IF A%(N)=A%(M) THEN GOTO 180
250 NEXT M
260 NEXT N
270 FOR N=1 TO 7
280 PRINT A%(N);
290 NEXT N
295 PRINT
300 GOTO 30
```

Examinons maintenant le programme de plus près, ligne par ligne :

Lignes 10-30 : Après que l'écran ait été vidé en ligne 20, vous pouvez choisir entre le loto français et le loto allemand.

Si vous entrez 'oui' en ligne 30, votre ST comprendra que vous voulez jouer au loto français. Si vous entrez n'importe quoi d'autre, par exemple 'non', votre ST comprendra que vous voulez jouer au loto allemand et il sautera donc, en ligne 40, à la section de programme commençant en ligne 180.

Lignes 50-70 : Ces lignes produisent les 6 nombres tirés au hasard pour le loto français. Pour stocker ces nombres, nous avons choisi des variables numériques qui ne peuvent recevoir que des nombres entiers.

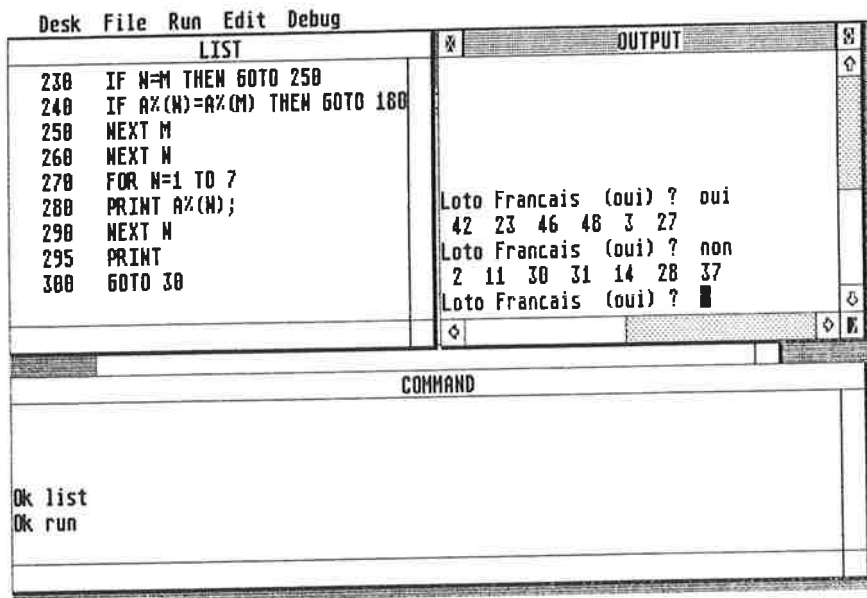
Nous multiplions la variable  $A\%(N)$  par 49 puisqu'il s'agit d'obtenir des nombres compris entre 1 et 49.

Lignes 80-130 : Ce bloc de lignes compare tous les nombres aléatoires produits entre eux. Si deux nombres sont égaux, en ligne 110, on produit de nouveaux nombres aléatoires en sautant à la ligne 50.

Lignes 140-170 : On sort ici les six nombres différents que vous pourrez utiliser pour faire votre loto. Bonne chance !

Lignes 180-200 : Ces lignes produisent les sept nombres aléatoires pour le loto allemand. Pour stocker ces nombres, nous avons choisi des variables numériques qui ne peuvent recevoir que des nombres entiers.

Nous multiplions la variable  $A\%(N)$  par 38 puisqu'il s'agit d'obtenir des nombres compris entre 1 et 38.



Lignes 210-260 : Ce bloc de lignes compare tous les nombres aléatoires produits entre eux. Si deux nombres sont égaux, en ligne 240, on produit de nouveaux nombres aléatoires en sautant à la ligne 180.

Lignes 270-300 : On sort ici les sept nombres différents que vous pourrez utiliser pour faire votre loto en Allemagne. Bonne chance !

### 3.9.5. Programme de conversion

Le cinquième programme est en fait constitué de 4 sous-programmes. Ce programme vous montre comment le ST peut mémoriser des conversions : de FF en Dollars, de Dollars en FF, de cm en inch (pouces) et de inch en cm. Il serait bien sûr possible d'étendre la liste à volonté.

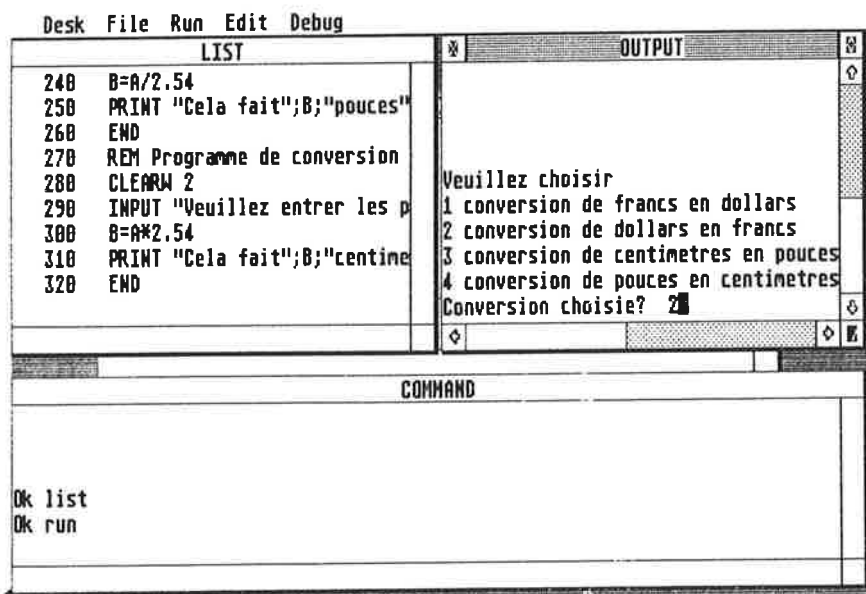
Le programme se distingue par son ergonomie puisqu'un menu permet de choisir quel type de conversion on désire effectuer. On entre le numéro de la conversion souhaitée et le sous-programme correspondant est appelé.

```
10 REM menu
15 CLEARW 2
20 PRINT "Faites votre choix :"
```

```
260 END
270 REM Conversion inch/cm
280 CLEARW 2
290 INPUT "Longueur en inch ";A
300 B=A*2.54
310 PRINT "Cela fait";B;"cm"
320 END
```

Examinons le programme ligne par ligne :

- 10-70      Menu principal. Le nombre entré (1 à 4) est rangé dans la variable X à l'aide de l'instruction INPUT.
- 80          Ceci est une nouvelle instruction. Suivant que X vaut 1, 2, 3 ou 4, un branchement est effectué vers l'un des numéros de ligne indiqués après le GOTO.
- 90-140     L'écran de sortie est effacé (ligne 100), puis l'instruction INPUT (ligne 110) attend le nombre de francs à convertir. Cette valeur est ensuite divisée par le cours courant du Dollar (le Dollar vaut actuellement environ 5.70 FF). Le résultat est affiché en ligne 130 sur l'écran OUTPUT.
- 150-200    L'écran de sortie est effacé (ligne 160), puis l'instruction INPUT attend le nombre de Dollars à convertir. Cette valeur est ensuite multipliée par le cours courant du Dollar. Le résultat est affiché en ligne 190 sur l'écran OUTPUT.
- 210-260    L'écran de sortie est effacé (ligne 220), puis l'instruction INPUT attend le nombre de cm à convertir. Cette valeur est ensuite divisée par le nombre de inch qu'il y a dans un cm (2.54). Le résultat est affiché en ligne 250 sur l'écran OUTPUT.



270-320 L'écran de sortie est effacé (ligne 280), puis l'instruction INPUT attend le nombre de inch à convertir. Cette valeur est ensuite multipliée par le nombre de inch qu'il y a dans un cm. Le résultat est affiché en ligne 310 sur l'écran OUTPUT.

### 3.10. VOCABULAIRE DE BASE DU BASIC DE L'ATARI ST

**CLEARW n :** Efface l'un des écrans-fenêtre :

n=0 : écran EDIT

n=1 : écran LIST

n=2 : écran OUTPUT

n=3 : écran COMMAND

**CONT :** Reprise de l'exécution d'un programme interrompu par 'Control'-'G' ou par l'instruction STOP.

**END :** Termine l'exécution d'un programme (ne peut être reprise avec CONT).

**FOR...NEXT** : Marqueur de début et de fin d'une boucle de programme ; à chaque passage la variable utilisée est incrémentée de 1.

**GOTO** : Branchement vers la ligne indiquée.

**GOSUB...RETURN** : Saut vers un sous-programme se trouvant à la ligne indiquée. Celui-ci doit se terminer par **RETURN** qui provoque un retour au programme appelant.

**IF...THEN...ELSE** : Exécution conditionnelle d'instructions (IF=si, THEN=alors, ELSE=sinon).

**INPUT** : Attente et lecteur d'une valeur dans une variable. **INPUT** peut afficher du texte.

**LIST** : Affiche les lignes du programme sur l'écran **LIST**.

**NEW** : Efface le programme de la mémoire de l'ordinateur.

**PRINT** : Sortie à l'écran de texte ou du contenu de variables.

**REM** : Ce qui est écrit après cette commande n'est pas interprété et peut servir à commenter un programme.

**RND** : Produit un nombre aléatoire compris entre 0 et 1.

**RUN** : Lance l'exécution du programme.

**STOP** : Arrêt de l'exécution du programme. Peut-être poursuivi avec **CONT**.

### **3.11. PERSPECTIVES**

Vous avez maintenant eu un petit aperçu des possibilités de **ST-BASIC**. Le langage **BASIC** convient tout à fait à de petites applications, mais pour des programmes plus complexes vous trouverez probablement l'éditeur peu pratique, les programmes peu lisibles et difficiles à superviser et l'exécution trop lente.



Depuis pas mal de temps déjà, divers éditeurs proposent des BASIC ayant des possibilités nettement supérieures au ST-BASIC. Nous vous en présentons deux ici.

Le BASIC le plus connu et le plus répandu est probablement le GFA-BASIC, distribué en France par Micro Application. Du à sa grande popularité, de nombreux ouvrages ont été publiés à son sujet, notamment aux éditions Micro Application.

Le GFA-BASIC se distingue par un excellent éditeur. Les programmes sont écrits sans numéros de lignes, les références se font par étiquettes. La programmation est beaucoup plus structurée. Par ailleurs l'exécution de la plupart des instructions (il y en a beaucoup plus) a été accélérée. Enfin il existe un compilateur qui accélère encore le programme et qui permet de le lancer sans passer par BASIC, par un simple double-clic.

Le deuxième BASIC avec des possibilités quasi identiques est le BASIC Omikron. Il est parfois encore plus rapide que le GFA-BASIC et attache une grande importance à la précision des calculs. L'éditeur demande une période d'adaptation, mais il accepte la programmation avec ou sans numéros de lignes. Ce BASIC peut également être compilé. Cette version de BASIC est beaucoup moins répandue que le GFA-BASIC, les ouvrages à son sujet sont également rares.

Outre le BASIC, il existe d'autres langages de programmation très intéressants pour le ST. Ainsi il existe quelques très bons compilateurs pour les langages Pascal et C. Avec un bon outil on peut aussi programmer en assembleur sur le ST.



## **4. LE LOGO DE L'ATARI ST**

### **4.1. COMPARAISON ENTRE LE LOGO ET LE BASIC ST**

Après avoir longuement traité du langage de programmation BASIC ST, nous allons nous intéresser au second langage de programmation fourni avec votre ATARI ST.

Le langage de programmation s'appelle DR LOGO et il se distingue du BASIC, entre autres, par les points suivants :

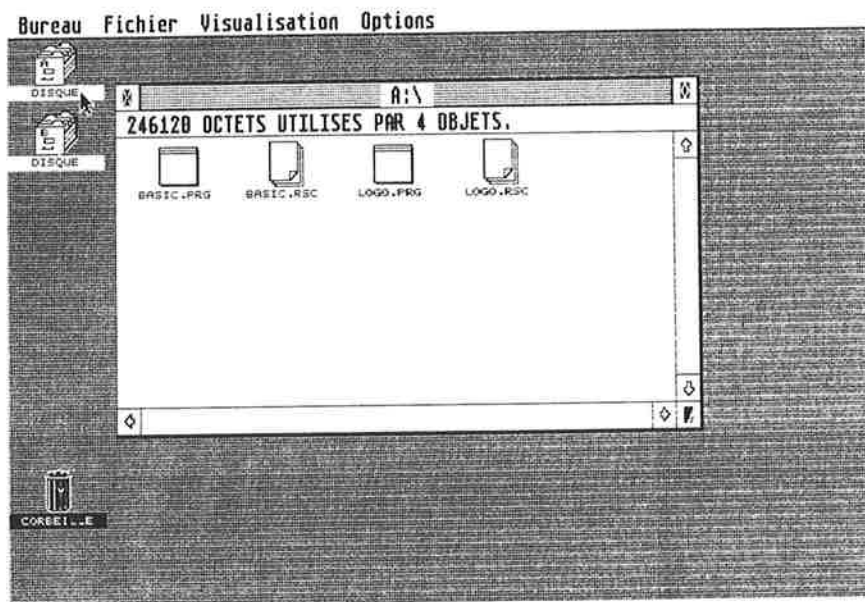
1. Les programmes LOGO n'ont pas de numéros de ligne.
2. L'écran LOGO sur le ST n'est divisé qu'en deux fenêtres.
3. Un programme LOGO se compose de nombreux petits sous-programmes LOGO que vous pouvez appeler à tout moment après les avoir définis.
4. La sortie sous LOGO se fait en grande partie sur l'écran graphique et non, comme en BASIC, sur l'écran de texte.
5. A côté du curseur de texte rectangulaire, le LOGO possède également un curseur graphique, en forme de triangle, qu'on appelle 'tortue' (turtle en anglais).

Nous ne vous raconterons pas en détail comment est apparu le langage de programmation DR LOGO car cela dépasserait largement le cadre de cet ouvrage. Sachez toutefois que, si vous souhaitez approfondir cette matière, il existe, chez DATA BECKER et MICRO APPLICATION, un ouvrage spécialement consacré au LOGO sur l'ATARI ST. Nous essaierons donc plutôt, dans ce chapitre, de découvrir deux bonnes douzaines d'instructions LOGO en prenant des exemples dans différents domaines. Nous essaierons bien sûr également d'en savoir plus sur le bon usage de DR LOGO.

## 4.2 CHARGEMENT DU LOGO

Après que vous ayez chargé le système d'exploitation GEM/TOS de la disquette dans votre ST, prenez à nouveau la disquette 'Language Disk' et faites-en afficher le catalogue à l'écran par un double clic sur la souris.

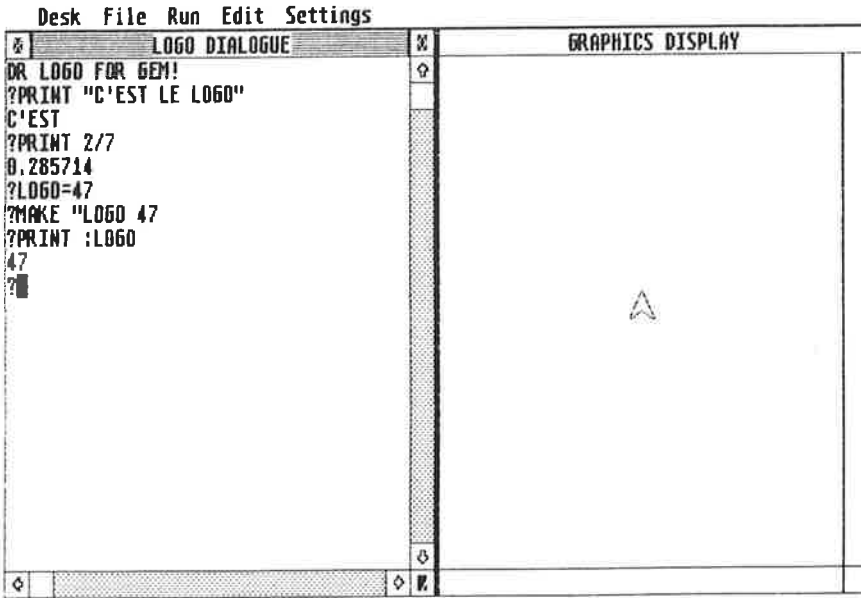
Sur le 'Language Disk' dont nous disposons figurent deux programmes dont le nom fait penser au LOGO. Un programme, symbolisé par une forme de bloc, appelé 'LOGO.PRG' et un fichier de données, représenté par une feuille au coin recourbé, appelé 'LOGO.RSC'.



Il y a peut-être sur votre disquette encore d'autres fichiers tels que par exemple des programmes LOGO tout prêts marqués par l'abréviation finale 'LOG'. Nous ne nous en occuperons pas plus dans ce chapitre car notre but n'est pas ici de développer des programmes géants, ce qui dépasserait l'objet d'un livre pour débuter, mais bien plutôt de découvrir les instructions de base du LOGO.

Chargez donc le langage de programmation DR LOGO de la disquette dans votre ST en amenant la flèche de la souris sur 'LOGO.PRG' puis en faisant un double clic.

Peu de temps après l'exécution du double clic, la flèche de la souris se transforme en une abeille travailleuse, ce qui indique que l'opération de chargement est en train. Peu de temps après, c'est terminé: le LOGO a été chargé de la disquette dans la mémoire de votre ordinateur ST et il attend maintenant la première entrée d'une instruction.



L'écran sur le côté gauche porte le titre 'LOGO DIALOGUE' ce qui signifie : c'est ici qu'aura lieu l'entrée et une partie de la sortie. L'écran sur le côté droit s'appelle 'GRAPHICS DISPLAY', affichage du graphisme.

La ligne de menu porte également d'autres indications : Desk, File, Run, Edit et Settings. Sur la moitié droite de l'écran, à l'intérieur du GRAPHICS DISPLAY, nous trouvons le triangle de dessin dont nous avons déjà parlé au début de ce chapitre.

Dans le côté de dialogue, DR LOGO nous salue ainsi: 'DR LOGO FOR GEM !'.

Notez d'ailleurs que DR n'est pas du tout un titre universitaire quelconque mais un jeu avec l'abréviation de la société Digital Ressearch qui a non seulement développé le système d'exploitation graphique, GEM, de votre ST mais aussi le langage de programmation LOGO qui vous est également fourni avec votre ST et avec lequel nous allons faire connaissance dans ce chapitre.

### **4.3. LE VOCABULAIRE DE BASE DU LOGO**

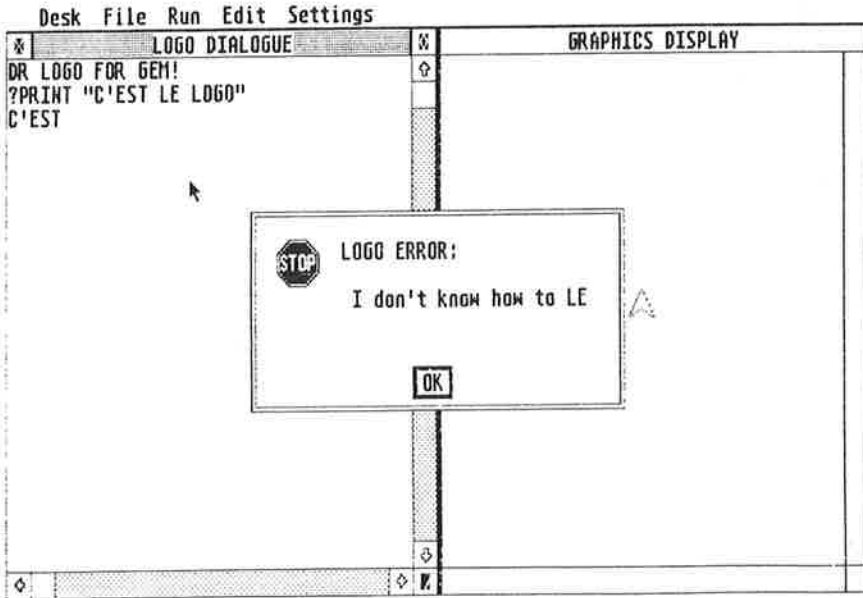
Comme DR LOGO nous salue très poliment, nous allons essayer de faire tout de suite quelque chose de correct et de sensé avec ce langage de programmation.

#### **4.3.1. PRINT**

Vous souvenez-vous du chapitre sur le BASIC où nous avons appris l'instruction PRINT? PRINT vous permet de sortir sur l'écran des phrases ou le résultat de calculs. Est-ce également possible en LOGO ? Pour le savoir, entrez :

**PRINT "C'EST LE LOGO"**

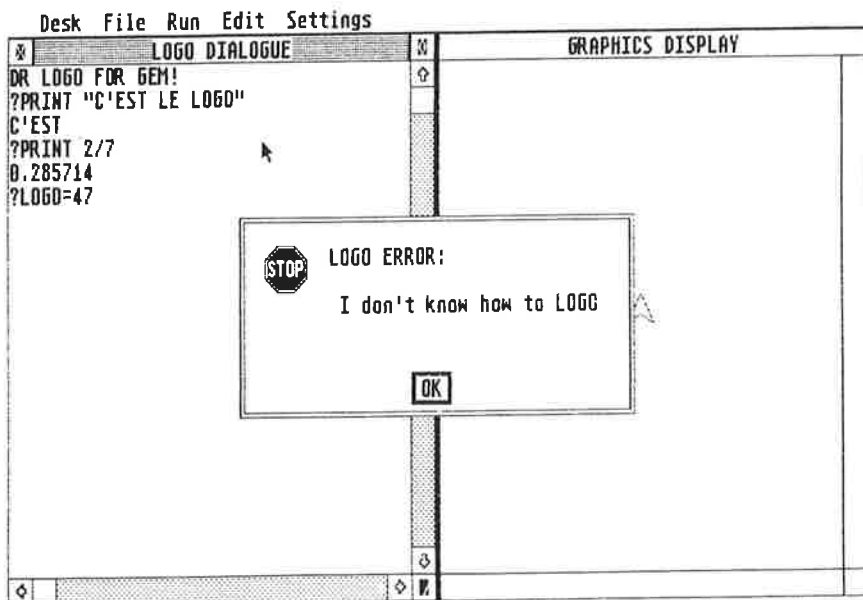
et appuyez ensuite sur la touche d'entrée (Enter ou Return).



Nous obtenons déjà un premier résultat : sans que vous ayez actionné la touche Caps Lock ni tenu la touche SHIFT enfoncée, DR LOGO a tout sorti systématiquement en majuscules sur l'écran de dialogue. Mais que signifie donc le message d'erreur dans le milieu de l'écran? DR LOGO n'a pas compris ce que signifie le mot "LE". En effet, l'instruction PRINT avec les guillemets ne s'applique qu'au mot qui est placé immédiatement à la suite des guillemets. Le premier mot a donc bien été sorti sur l'écran de dialogue directement à la suite de notre instruction.

Comment se présentent, avec DR LOGO, les opérations de calcul, quelle est leur précision ?

PRINT 2/7



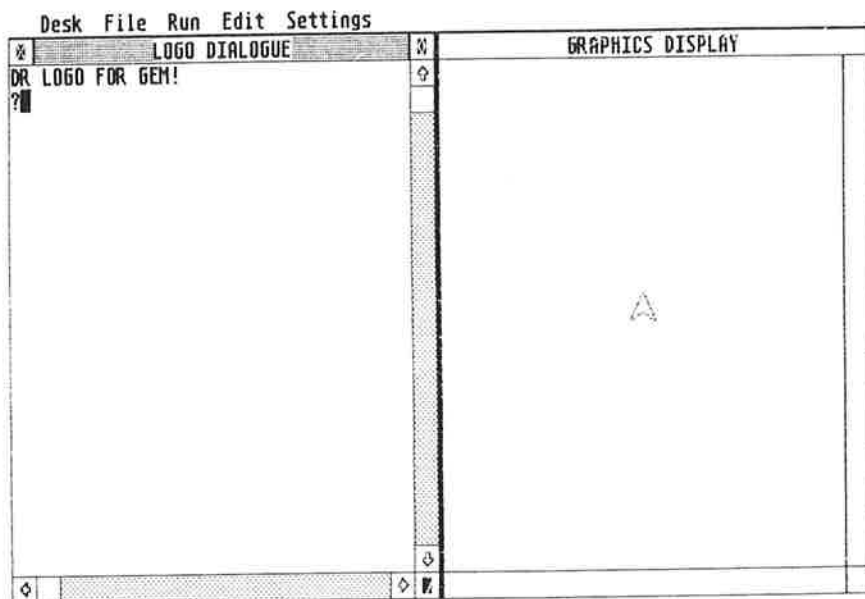
Non seulement l'instruction PRINT a été comprise par DR LOGO mais le calcul a bien été effectué et son résultat nous a été sorti sur l'écran avec 6 chiffres après la virgule. Rappelez-vous à ce sujet de ce que vous nous avons dit, au sujet du BASIC, sur les différentes précisions de calcul possibles. En l'occurrence, DR LOGO s'est donc comporté exactement comme le BASIC ST pour des calculs en simple précision.

#### 4.3.2. MAKE

Est-il possible, sous DR LOGO aussi, d'affecter des valeurs variables à des tiroirs dans la mémoire de l'ordinateur ? Essayons d'abord de le faire comme nous y étions habitués avec le BASIC ST :

LOGO = 47





Certes, les messages d'erreur de DR LOGO sont particulièrement agréables parce qu'ils sont détaillés. Il n'est, malgré tout, jamais très plaisant de constater qu'on a fait des erreurs. Qu'est-ce qui n'allait pas ici ? C'est tout simple : DR LOGO possède un vocabulaire dans lequel ne figure pas le mot LOGO. DR LOGO a donc cherché dans sa mémoire si le programme LOGO existait déjà. Il vous a ensuite informé de l'échec de cette recherche. DR LOGO est vraiment un langage de programmation de grande classe, non ?

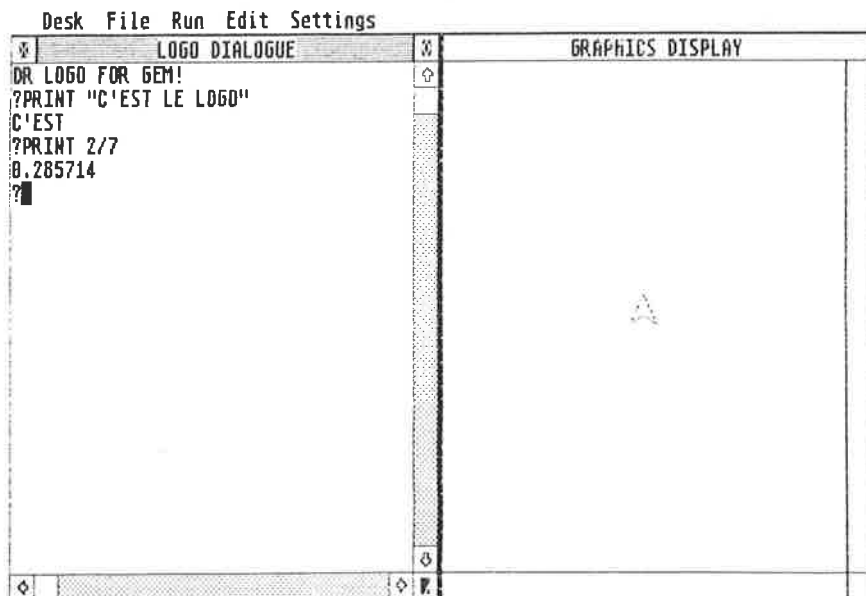
Lorsque vous affectez une valeur à un tiroir sous DR LOGO, vous devez prendre en compte deux choses : l'affectation de valeur doit être précédée du mot 'MAKE' qui veut dire faire.

D'autre part, l'affectation ne doit pas se faire avec le signe égale mais avec les guillemets et un espace.

MAKE "LOGO 47

La variable peut ensuite être à nouveau interrogée avec l'instruction PRINT à condition toutefois que vous n'oubliez pas de faire précéder le nom de la variable d'un double point.

### PRINT : LOGO



### 4.3.3. CS et CT

Vous souvenez-vous encore de l'instruction pour vider les différents écrans en BASIC ST ? Bravo ! C'est effectivement 'CLEARW' avec un nombre de 0 à 3. DR LOGO dispose bien sûr également d'instruction pour vider les deux écrans disponibles.

#### CS

vous permet de vider l'écran graphique, GRAPHICS DISPLAY, et

#### CT

vous permet de vider l'écran de texte, LOGO DIALOGUE.

Bien que nous vous ayons promis de ne pas vous ennuyer avec de longs récits sur l'histoire du LOGO, il nous faut malgré tout en parler un petit peu. Comme nous l'avons déjà indiqué, le langage de programmation LOGO a des rapports très étroits avec les représentations graphiques mais, comme nous l'avons vu dès le début de ce chapitre, le LOGO ne se limite pas à cela.

#### **4.3.4. FORWARD, BACK, RIGHT et LEFT**

LOGO a été développé à l'origine pour pouvoir familiariser les enfants à la pensée logique, grâce à l'emploi des moyens les plus simples possibles. Comme les enfants, à l'âge de la maternelle, représentent leur monde de préférence avec du papier et des crayons de couleur, on imagina un langage de programmation qui travaille le plus possible avec des représentations graphiques.

Pour représenter les différents travaux logiques, on utilisa une tortue en carton à travers la carapace de laquelle on pourrait placer un crayon pour dessiner pendant les déplacements de la tortue. On pouvait exécuter une suite de pas logiques consécutifs ('va vers l'avant', 'va vers la gauche',...) à travers des déplacements de la tortue mais on pouvait, aussi et surtout, faire répéter plus tard ces mêmes pas sur le sol, comme pour une araignée tissant sa toile.

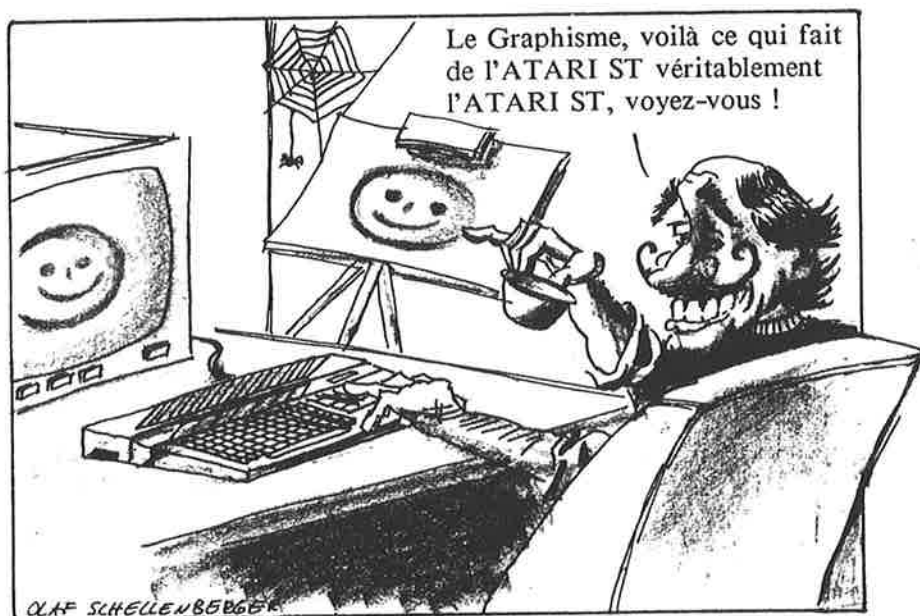
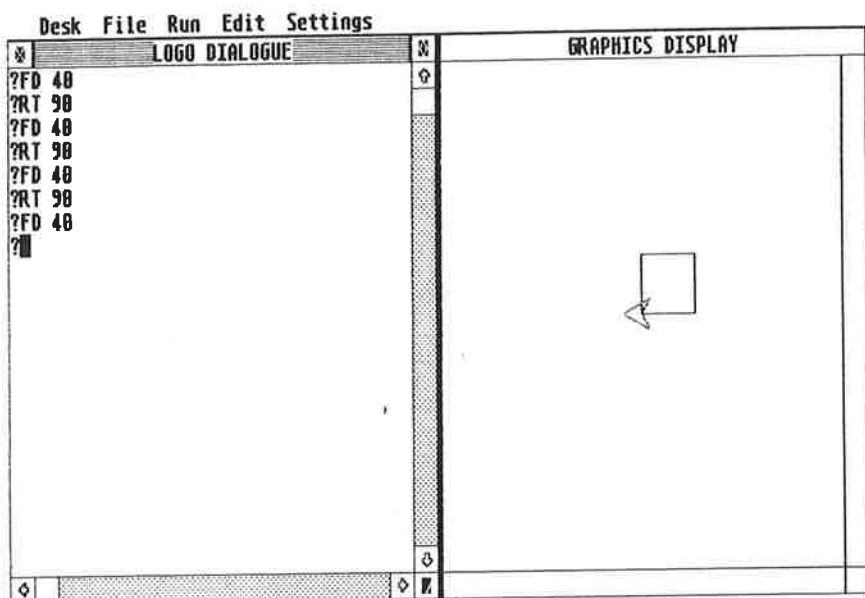
Examinez le GRAPHICS DISPLAY. En son milieu se trouve pour le moment un triangle. Ce triangle est la tortue. Comment pouvons-nous déplacer cette tortue ou en changer la direction ?

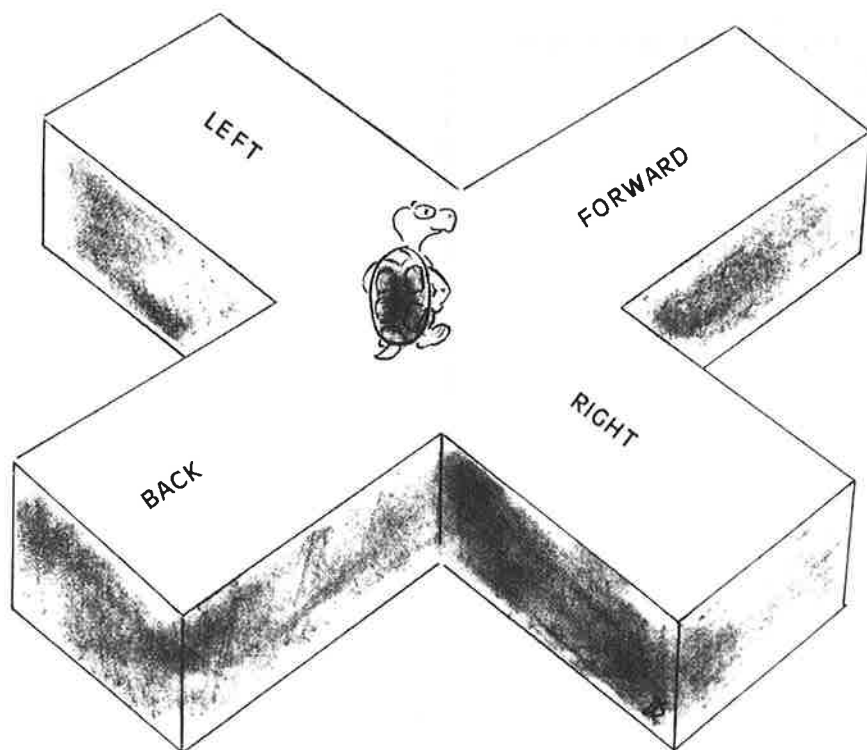


FORWARD=	aller en avant
BACK =	aller en arrière
RIGHT =	tourner à droite
LEFT =	tourner à gauche

Avec les instructions de déplacement 'FORWARD' (FD en abrégé) et BACK, il nous faut indiquer de combien de points notre tortue doit se déplacer. Si nous donnons une nouvelle direction, avec RIGHT (RT en abrégé) ou LEFT (LT en abrégé), nous devons également ajouter à cette instruction un nombre qui indiquera sur combien de degrés la tortue doit tourner. C'est pourquoi la suite d'instructions permettant de tracer un carré devrait par exemple se présenter ainsi :

FD 40 RT 90 FD 40 RT 90 FD 40 RT 90 FD 40



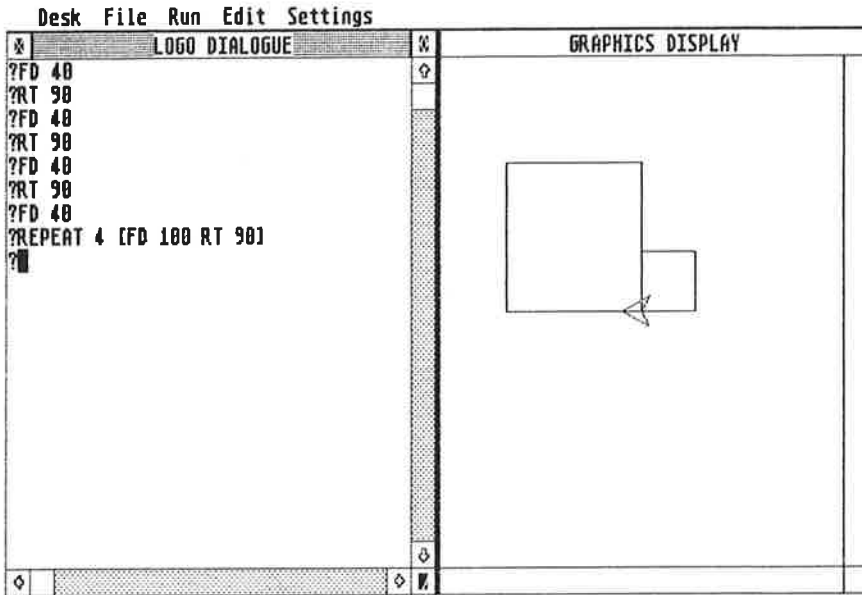


#### 4.3.5. REPEAT

Bon, peut-être êtes-vous déjà fatigué d'entrer autant de fois les mêmes instruction ? Ne serait-il pas possible, avec DR LOGO, d'utiliser une instruction GOTO comme celle qui fait merveille dans le BASIC ST ?

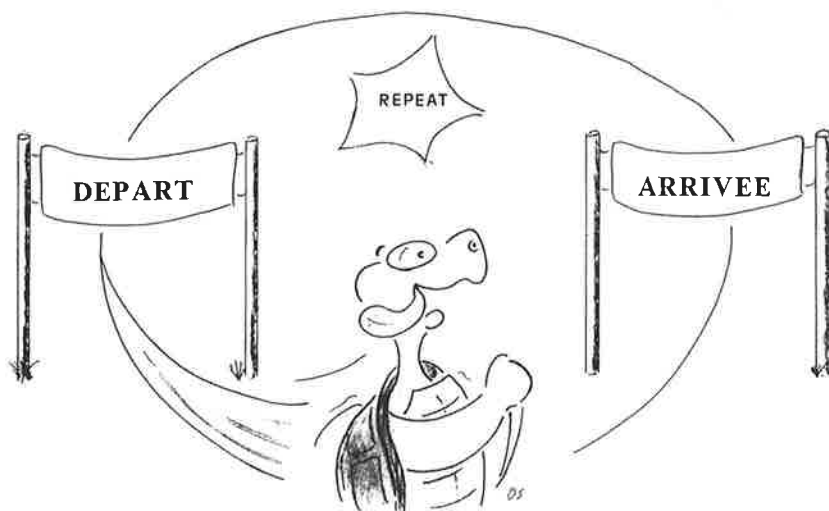
Effectivement il existe aussi en LOGO quelque chose qui ressemble à l'instruction GOTO. C'est l'instruction REPEAT (répéter). Nous allons essayer cette instruction en dessinant un second carré qui aura cette fois, cependant, des côtés beaucoup plus longs.

**REPEAT 4 [FD 100 RT 90]**



Par rapport à notre dessin avec 7 différentes instructions, cette second série d'instructions, y compris REPEAT, était vraiment un jeu d'enfant, non ? N'oubliez pas de placer entre crochets tout ce qui devra faire partie de la boucle REPEAT! Vous pouvez également intégrer la variable que nous avons déjà définie auparavant, LOGO (d'une valeur de 47), dans la boucle REPEAT :

REPEAT 4 [FD :LOGO RT 90]



#### 4.3.6. TO

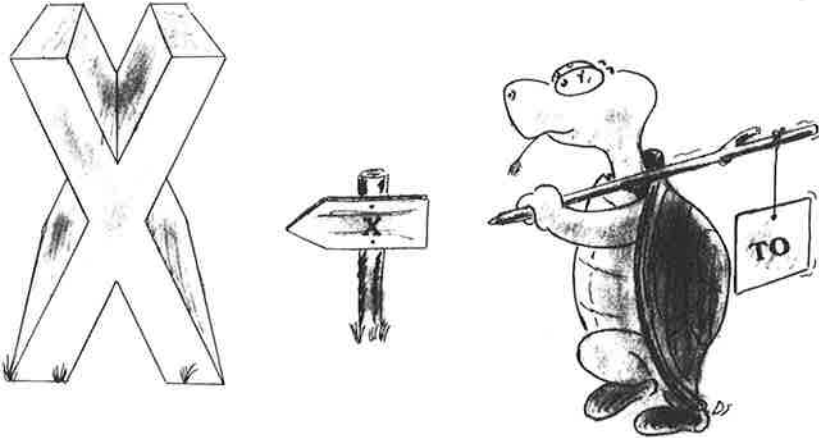
Lorsqu'au début de ce chapitre nous nous demandions en quoi DR LOGO diffère du BASIC ST, nous avons indiqué que les programmes DR LOGO ne possèdent pas de numéros de ligne mais qu'ils se composent, au lieu de lignes, de plusieurs sous-programmes.

Qu'est-ce qu'un sous-programme dans le langage de programmation DR LOGO ? L'imbrication des différentes instructions FD 100 et RT 90 que nous avons réalisée avec REPEAT constituerait-elle déjà un sous-programme ?

Un sous-programme DR LOGO peut être encore plus court qu'une boucle REPEAT. Il suffit que vous commenciez l'entrée du sous-programme par le mot TO et que vous la terminiez par END. Entre 'TO nom du sous-programme' et END, vous pouvez intégrer autant de pas de programmation que vous le souhaitez.



L'avantage des sous-programmes avec DR LOGO, c'est que les sous-programmes pourront ensuite être appelés par leur nom, directement ou par d'autres sous-programmes.



Nous allons tout d'abord créer deux mini-sous-programmes :

```
TO VIDETEXTE  
CT  
END
```

```
TO VIDEGRAPHIQUE  
CS  
END
```

Que s'est-il passé sur l'écran pendant l'entrée de ces deux sous-programmes LOGO et quel est l'effet produit par ces programmes ?

Après la définition du sous-programme 'VIDETEXTE', le point d'interrogation au début de la ligne suivante s'est transformé en une flèche ou, si vous préférez, en un symbole 'supérieur à'. Ce caractère d'entrée subsista jusqu'à ce que vous terminiez le sous-programme par l'instruction END.

L'effet de ces deux sous-programmes c'est que vous pouvez maintenant employer à votre guise soit l'instruction LOGO proprement dite CS, soit la nouvelle instruction définie, le sous-programme LOGO, VIDEGRAPHIQUE. Ces deux instructions ont maintenant exactement la même fonction : elles vident l'écran graphique dès que vous les avez entrées puis appuyé sur la touche d'entrée.

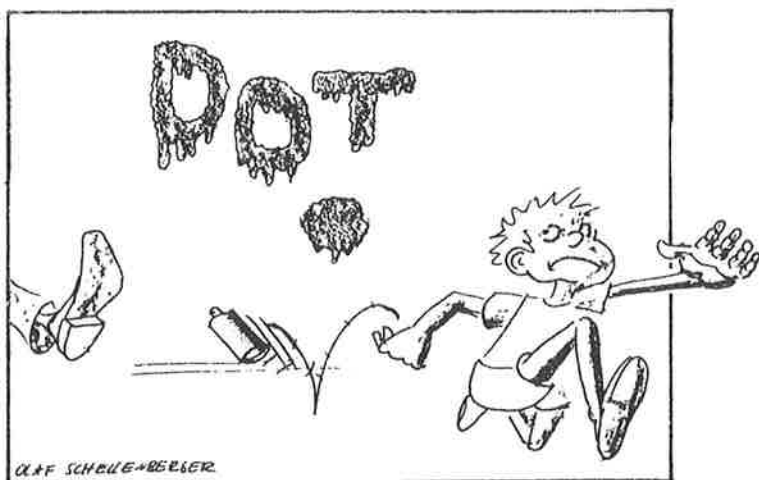
DR LOGO dispose donc ainsi de la possibilité de créer soi-même autant de nouvelles instructions que l'on veut. Cela peut vous permettre, par exemple, de traduire en français le vocabulaire de DR LOGO, comme nous venons de le faire.

#### 4.3.7. PENUP et PENDOWN



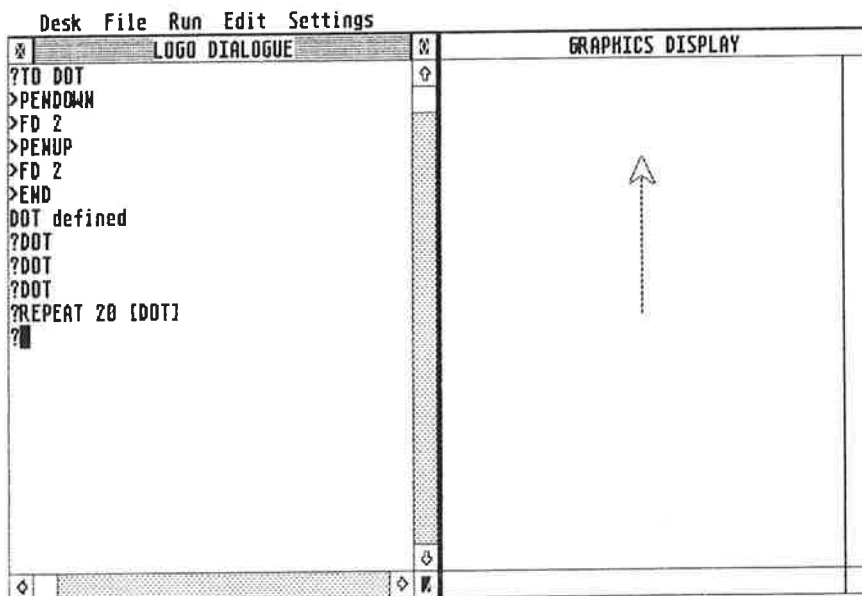
Nous avons parlé de l'histoire du développement du langage de programmation DR LOGO. Nous vous avons parlé également des enfants de la maternelle qui enfoncent un crayon dans leur tortue en carton pour qu'elle dessine 'en même temps qu'eux'. Ils peuvent bien sûr également enlever le crayon de la carapace de la tortue. C'est exactement à cela que servent les instructions **PENUP** (relever le crayon pour qu'on ne dessine pas) et **PENDOWN** (abaisser le crayon: le crayon touche le papier et, par conséquent, en ce qui nous concerne, au fur et à mesure que vous déplacerez le triangle de dessin sur l'écran, vous tracerez vraiment un dessin sur l'écran **GRAPHICS DISPLAY**).

Nous allons maintenant réaliser un petit programme qui utilisera les instructions **PENUP** et **PENDOWN**. Ce programme devra nous permettre de dessiner uniquement un point sur l'écran **GRAPHICS DISPLAY**, de façon à ce que nous puissions réaliser progressivement une ligne en pointillé. Ce sous-programme s'appellera **DOT** :



```
TO DOT
PENDOWN
FD 2
PENUP
FD 2
END
```

Vous avez maintenant défini ce qu'on appelle une 'procédure', appelée DOT. Lancez le programme en entrant son nom, DOT. Tapez ce nom plusieurs fois. Une ligne en pointillé apparaîtra alors peu à peu sur l'écran. Si vous ne voulez pas entrer au clavier autant de fois la même instruction DOT, vous pouvez naturellement utiliser l'instruction REPEAT dont nous avons déjà parlé :



#### 4.4. AUTRES INSTRUCTIONS LOGO

Vous connaissez maintenant le vocabulaire de base de DR LOGO. Vous allez découvrir, dans les pages suivantes, encore quelques autres instructions DR LOGO qui permettent toutes d'obtenir facilement de grands effets.

Bien entendu, vous pouvez intégrer ces instructions dans des sous-programmes de plus grande envergure que vous réaliserez vous-même en vous servant du vocabulaire de base que nous avons étudié jusqu'ici.

#### **4.4.1. BOX**

**BOX X Y longueur hauteur**

Cette instruction nous permet de dessiner un rectangle. Essayez de vous représenter l'écran graphique comme un système de coordonnées dont le point zéro, l'origine, serait dans le coin inférieur gauche. C'est de là que partent l'axe des X vers la droite et l'axe des Y vers le haut. Vous pouvez déterminer n'importe quel point avec deux coordonnées.

Les valeurs effectives que vous devez entrer, à la place des lettres X et Y, à la suite de l'instruction BOX, sont donc ce qu'on appelle des paramètres qui indiquent la situation du coin inférieur gauche du rectangle. Viennent ensuite les valeurs pour la longueur et la hauteur du rectangle. Si vous choisissez une même valeur pour la hauteur et la longueur, vous obtenez un carré.

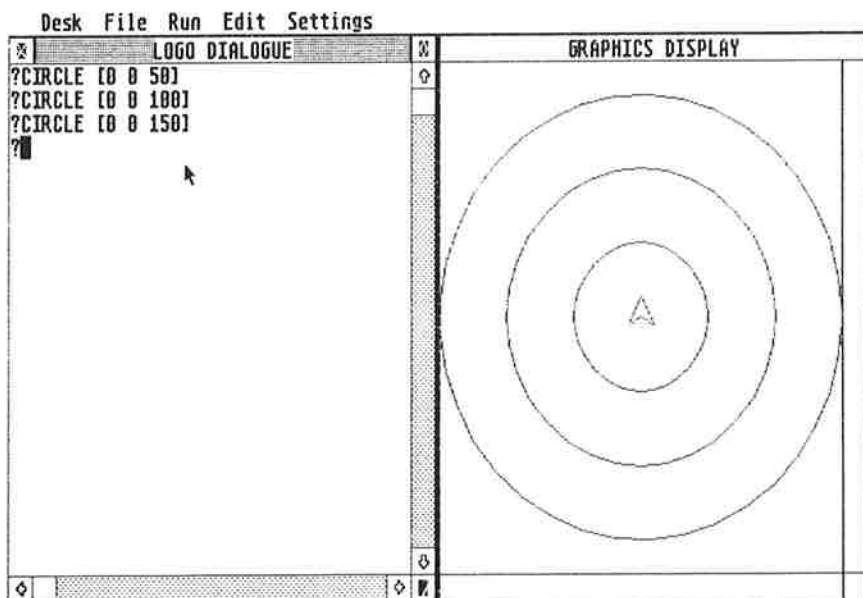
#### **Une remarque importante :**

Avec les instructions LOGO, les valeurs placées à la suite d'une instruction doivent être séparées par un espace et non, comme en BASIC, par une virgule !

#### **4.4.2. CIRCLE**

**CIRCLE A B C**

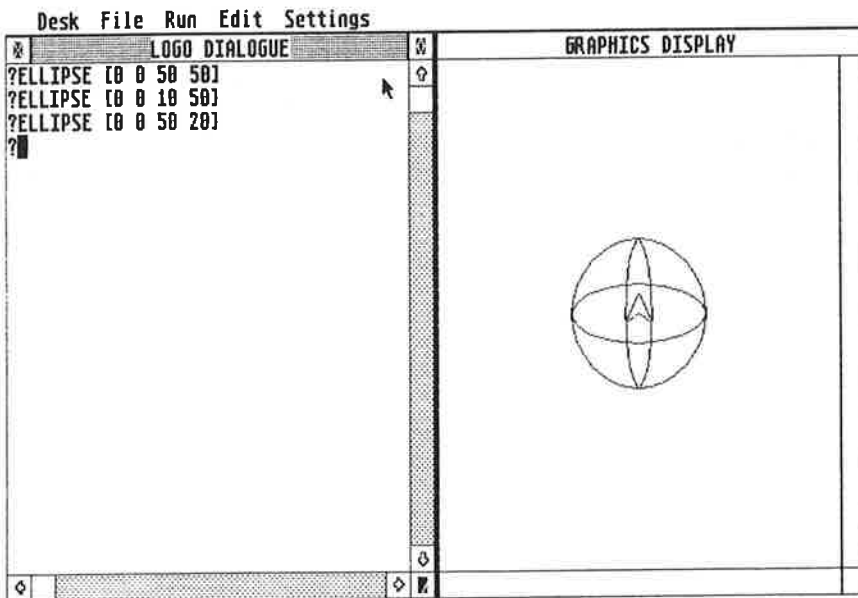
Cette instruction dessine un cercle dont le centre aura A pour coordonnée X et B pour coordonnée Y avec un rayon de C.



#### 4.4.3. ELLIPSE

##### ELLIPSE A B C D

Cette instruction DR LOGO ressemble à CIRCLE mais les deux rayons à entrer, le rayon X C et le rayon Y D définissent dans quelle mesure le cercle doit se transformer en ellipse. En effet, si C et D sont identiques, vous obtiendrez à nouveau un cercle.



#### 4.4.4. HIDE TURTLE et SHOW TURTLE

HIDE TURTLE (HT en abrégé)

SHOW TURTLE (ST en abrégé)

Après avoir entré l'instruction HIDE TURTLE, vous verrez disparaître du GRAPHICS DISPLAY le triangle de dessin. Vous pourrez le faire réapparaître avec SHOW TURTLE. Les fonctions de dessin sont cependant exécutées même si la tortue est invisible.

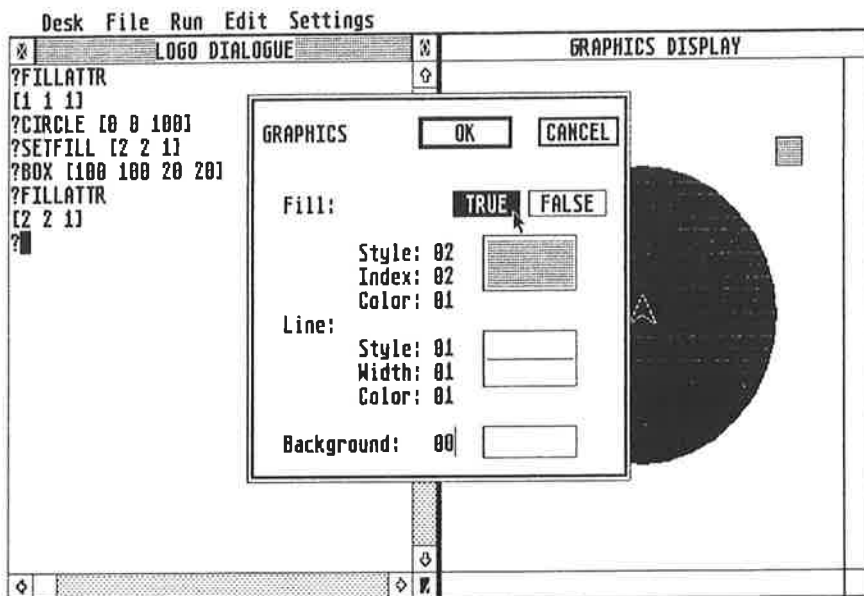
#### 4.4.5. FILLATR ou SETFILL

FILLATR A B C

SETFILL A B C

Vous pouvez peindre des surfaces fermées, telles que les cercles par exemple, avec un modèle déterminé.

Pour cela, choisissez, dans le sous-menu sous 'Settings', le point 'Graphics' et faites ensuite, avec la souris, un clic sur le mot 'TRUE' pour l'instruction Fill que vous verrez alors apparaître. Vous pourrez alors sélectionner un des nombreux modèles que DR LOGO met à votre disposition. L'instruction FILLATTR vous permet d'appeler l'état actuel de la couleur de remplissage et SETFILL A B C vous permet de modifier les paramètres en fonction de ce que vous souhaitez; A définit le style de dessin et peut prendre les valeurs de 0 à 4; B représente l'index et peut prendre les valeurs de 0 à 12. Si vous possédez un moniteur couleur, vous pouvez enfin sélectionner jusqu'à 16 couleurs définies avec le paramètre C.





#### 4.4.6. SHUFFLE et SORT

Ce sont deux instructions qui n'ont rien à voir avec le graphisme de DR LOGO. SHUFFLE vous permet de mélanger des valeurs numériques indiquées entre crochets, à la suite de cette instruction. SORT vous permet à nouveau de trier ces valeurs dans l'ordre croissant.

#### 4.4.7. MOUSE, NODES et TURTLEFACTS

MOUSE vous indique quelles sont les coordonnées de la position de la flèche-souris et si un des deux boutons d'impression est actionné. NODES vous indique la place mémoire encore libre ; DR LOGO calcule la place mémoire en NODES, c'est-à-dire en noeuds. TURTLEFACTS vous informe sur le triangle graphique.

### 4.5. BREF RECAPITULATIF DU VOCABULAIRE LOGO DE BASE

**BACK :** Déplacement en arrière du triangle, du nombre de points indiqué, sur GRAPHICS DISPLAY

**CS :** Vider l'écran GRAPHICS DISPLAY

**CT :** Vider l'écran LOGO DIALOGUE

**END:** Marque de la fin d'une procédure DR LOGO commencée avec TO.

**FORWARD ou FD :** Déplacement en avant du triangle, du nombre de points indiqué, sur GRAPHICS DISPLAY

**LEFT ou LT :** Rotation sur la gauche du triangle, du nombre de degrés indiqué, sur GRAPHICS DISPLAY

**MAKE :** Affectation d'une valeur à une variable (tiroir).

Par exemple : MAKE "A 55: la variable A recevra la valeur 55.

**PENUP ou PU :** A partir de maintenant, le triangle de dessin pourra être déplacé sur le GRAPHICS DISPLAY sans dessiner.

**PENDOWN** ou **PD** : A partir de maintenant, le triangle de dessin laissera une trace lors de chaque mouvement.

**PRINT** : Sortie de variables et de textes sur l'écran LOGO DIALOGUE; si des textes sont écrits après des guillemets à la suite de cette instruction LOGO, on ne sortira que les lettres placées entre les guillemets et le premier espace. Si vous voulez sortir des variables, celles-ci doivent être précédées d'un double point. Les nombres ou les résultats de calculs peuvent être sortis directement, sans être précédés d'une instruction PRINT.

**REPEAT** : Exécution répétée de la suite d'instructions indiquée, par exemple REPEAT 10 [FD 10] déplacera le triangle de dessin sur le GRAPHICS DISPLAY dix fois de dix points vers l'avant.

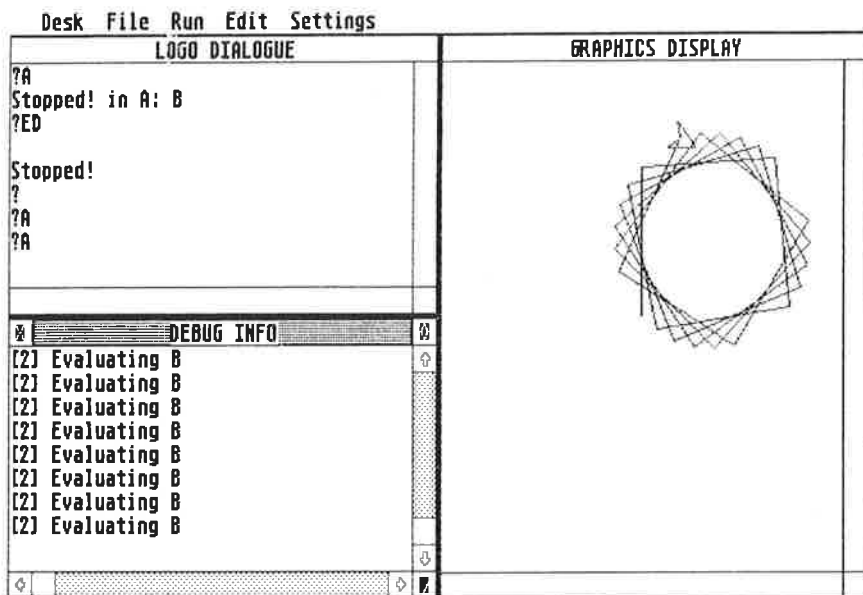
**RIGHT** ou **RT** : Rotation sur la droite du triangle, du nombre de degrés indiqué, sur le GRAPHICS DISPLAY

**TO** : Le nom entré à la suite de cette instruction deviendra ensuite ce qu'on appelle une procédure qui aura le même effet qu'une instruction prédéfinie de DR LOGO.

#### **4.6. LE DETECTEUR D'ERREURS DE DR LOGO**

1. Interrompre le programme en cours en appelant la fonction Pause du menu déroulant 'RUN'. L'exécution du programme peut être reprise grâce à 'Continue', dans le même menu.
2. Interruption et arrêt définitif du programme avec Control-G. Aucune reprise de l'exécution du programme n'est possible.
3. On peut sélectionner l'instruction WATCH dans le menu déroulant 'Settings' (confirmation par un crochet). Cette instruction a pour effet d'indiquer dans la fenêtre d'information, pendant le déroulement du programme, dans quelle procédure l'ordinateur se trouve à un moment donné, à quelle procédure il saute et combien de sauts ont déjà été effectués entre différentes procédures. En faisant à nouveau un clic sur WATCH, Cette aide à la recherche des erreurs est à nouveau désactivée.

4. L'instruction TRACE affiche, dans la fenêtre 'Debug-Info' dans quelle procédure on se trouve pour le moment. Désactivation par un nouveau clic dans la fenêtre 'Settings'.





## **5. PENDANT UN MOMENT DE TRANQUILLITE**

Nous avons maintenant découvert ensemble l'ATARI ST. Nous avons renoncé jusqu'ici à vous ennuyer avec des informations théoriques de fond compliquées dont nous pensions qu'elles n'étaient peut-être pas très intéressantes pour le moment.

Il est cependant tout à fait possible que vous n'ayez pas tout compris dans certain des chapitres précédents parce que vous saviez encore trop peu de choses sur ce qui se passe réellement dans un ordinateur.

Vous seriez par ailleurs certainement intéressé de savoir comment on en est venu à développer cette super-machine qu'est l'ATARI ST et ce que cette machine vous permettra de faire dans l'avenir. Ce sont ces problèmes que nous voudrions éclairer dans ce chapitre. Le mieux serait donc que vous lisiez ce dernier chapitre un jour où vous aurez véritablement un peu de temps devant vous et où vous aurez l'esprit suffisamment libre pour vous concentrer sur toutes les choses intéressantes qui se sont faites et qui se font toujours autour de l'ATARI ST.

### **5.1. LE JARGON TECHNIQUE DE L'INFORMATIQUE**

#### **5.1.1. Comme le temps a passé vite...**

Si vous passez un jour par Munich, après avoir visité quelques tavernes et usines de bière, ne manquez pas d'honorer le Deutsches Museum d'une visite. Une section de ce musée est consacrée à l'évolution de l'informatique. Vous y trouverez une grosse machine de calcul construite en 1941 par l'Allemand Konrad Zuse. Cette machine fut le premier ordinateur librement programmable. Elle a à peu près les dimensions d'une grosse armoire murale et pourtant sa puissance de calcul était extrêmement faible comparée aux ordinateurs actuels.



A quoi sont dûes à la fois la faible efficacité et l'énorme taille de cette machine ? Toutes deux ont une cause unique : le mode de fonctionnement mécanique de cet ordinateur.

Nous reviendrons dans ce chapitre sur le mode de calcul des ordinateurs. Insistons ici sur la mécanique : l'ordinateur Zuse travaillait avec un grand nombre de composants électro-mécaniques qui, branchés en chaîne, étaient à même de retenir des informations par un grand nombre d'états allumé et éteint.

Tout ce qui est réalisé électroniquement de nos jours dans un ordinateur fonctionnait de façon électro-mécanique dans l'ordinateur Zuse. Ce monstre était dominé par un cliquetis continu de commutateurs électro-mécaniques (relais). Vous imaginez aisément qu'avec un tel nombre de commutateurs l'usure était très élevée. Le taux d'erreur dans les calculs était par conséquent également beaucoup plus élevé qu'il ne l'est de nos jours puisque les ordinateurs ne commettent pas d'erreur en général. Il arrivait donc que le résultat du calcul indiqué par de petites lampes soit inexact.

Les punaises aimaient bien habiter dans cet ordinateur. Elles s'y introduisaient souvent, ce qui faisait que les commutateurs ne fonctionnaient plus correctement et que l'ordinateur fournissait des résultats incorrects. C'est d'ailleurs de cette époque que date l'expression 'debugger' pour la recherche des erreurs d'un programme. bug signifie en effet punaise.



Restons un peu dans l'histoire de l'informatique. Nous avons déjà indiqué que l'ordinateur Zuse était trop lent et surtout trop gros. C'est quelques années plus tard que jaillit la solution : du relais qui s'usait si facilement on passa aux transistor, un composant électronique beaucoup plus petit qui avait exactement la même fonction que le relais mais qui ne travaillait plus mécaniquement mais purement électroniquement : une de ces trois pattes est chargée de représenter l'état allumé ou éteint.

On fabriqua bientôt un ordinateur dont les cases mémoire étaient constituées par des transistors et la différence de taille par rapport à l'ordinateur Zuse fut déjà considérable.

Mais les inventeurs continuèrent leur travail. On eut finalement l'idée de dessiner simplement sur le papier, en deux dimensions, les voies électriques qui relient les transistors entre eux. Avec un nombre de généralement plusieurs milliers de transistors, on imagine aisément la taille que pouvait atteindre ce dessin. Sur le plan de la miniaturisation, c'était donc plutôt une régression.

Mais on eut alors une excellente idée : on photographia le gigantesque dessin représentant les liaisons électriques. Avec une pellicule de qualité suffisante, on pouvait alors réduire à quelques centimètres carrés un dessin de plusieurs mètres carrés.

Un procédé chimique spécial permet alors de rendre conductrice la photographie des circuits des transistors. Ce qu'on appelle maintenant un 'chip' fait alors le même travail que plusieurs milliers de transistors. Pas totalement : il faut d'abord que notre chip entre en contact avec le monde extérieur : par rapport au minuscule réseau intérieur de conducteurs, on dispose tout autour du chip des canaux conducteurs qui font l'effet de gigantesques troncs d'arbre. N'importe qui peut alors réaliser la connexion avec l'extérieur, y compris peut-être la connexion avec un autre chip.

Comparé aux relais et aux transistors, notre chip est, de plus, devenu considérablement plus rapide : jusqu'à 100000 additions peuvent être exécutées en une seconde. C'est qu'en effet le réseau de circuits étroitement imbriqués comporte des distances considérablement plus réduites que celles, étalées sur le volume d'une armoire murale, d'un ordinateur tel que celui de Zuse.

Où en est l'évolution de ces chips jusqu'à nos jours ? On travaille actuellement au développement de ce qu'on appelle les méga bit chips, c'est-à-dire des chips comportant un million de fonctions de transistor sur la surface d'une tête d'épingle. C'est le dernier cri de l'évolution technique qui est toutefois encore absent de votre ordinateur ST. Ce dernier comporte néanmoins de nombreux chips de 256 kilo bits qui ne sont donc que quatre fois moins puissants.



### 5.1.2. Différents chips

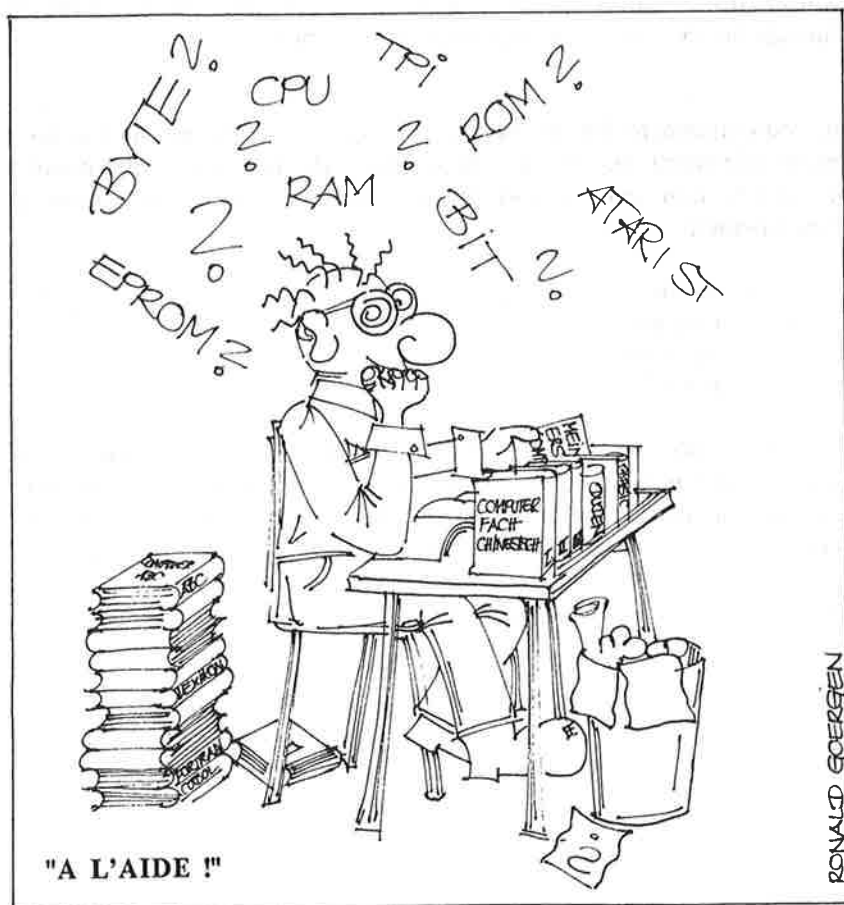
Il y a de nombreuses possibilités d'utiliser des chips. Selon le but recherché, selon que des informations sont ou ne sont pas déjà stockées de manière fixe sur le chip, nous pouvons acheter des chips contenant déjà des programmes tout prêts ou bien utiliser des chips dont la mémoire est librement programmable.

Pour bien comprendre les termes techniques, il faut avoir présent à l'esprit l'importance de la notion de 'mémoire' qui sert donc à former de nombreux termes informatiques. Il existe les types de chips suivants:

ROM  
PROM  
EPROM  
RAM

ROM est l'abréviation de 'Read Only Memory' ou 'mémoire pour lecture uniquement'. La ROM contient les programmes tout prêts qu'on aura à utiliser immédiatement. Sur votre machine à laver, par exemple, vous pouvez choisir entre différents programmes. Vous n'avez bien sûr pas à taper ces programmes une fois que vous avez acheté votre machine à laver, ces programmes sont déjà compris dans le prix de vente.

Sur votre ST, il n'y a rien en ROM pour le moment. C'est pourquoi vous devez charger aussi bien le système d'exploitation que les langages de programmation de la disquette dans la mémoire de l'ordinateur. Il est prévu de placer en ROM le système d'exploitation TOS/GEM et peut-être aussi le BASIC. Il est possible que ce soit déjà fait au moment où vous lirez ces lignes.



Une ROM est un chip comme ceux dont nous avons parlé jusqu'ici. Toutefois les transistors ou les différents circuits les reliant ont déjà été définis par les développeurs ou programmeurs une fois pour toutes. Si vous disposiez donc d'une ROM avec TOS et GEM, vous n'auriez donc aucune possibilité de modifier ces circuits. Vous ne pourriez qu'en lire le contenu sans rien y changer.

Dans la phase préalable de développement d'ordinateurs, on utilise pas immédiatement des ROMs car les coûts de développement seraient considérablement majorés par la nécessité de détruire en permanence les ROMs non encore correctement programmées des phases de développement.

C'est pourquoi on utilise des EPROMs (Erasable Programmable ROM = ROM supprimable et programmable). Elles permettent à l'utilisateur de définir lui-même la structure des circuits puis de les effacer à nouveau s'ils ne lui conviennent pas.

Si vous disposez, en plus de votre ST, de l'extension requise, vous pouvez, vous aussi, programmer vous-même des EPROMs. Vous pourriez ainsi stocker sur une EPROM un programme que vous avez écrit vous-même, comme par exemple le programme d'agenga téléphonique de ce livre. Vous pourriez alors disposer instantanément de ce programme après avoir allumé votre ordinateur. Il "suffirait" pour cela que vous le transcriviez en langage machine et que vous y intégriez les routines du système d'exploitation nécessaires.

Il y a encore les PROMs qui ressemblent aux EPROMs, sans le E qui signifie Erasable = supprimable. Par conséquent, une fois qu'une PROM a été programmée, on ne peut plus rien y changer. Pour modifier le contenu, on doit programmer une nouvelle PROM.

Les chips les plus importants d'un ordinateur sont les RAMs qui peuvent retenir toutes les données entrées mais qui peuvent aussi les oublier à nouveau totalement, soit lorsqu'on leur en donne l'ordre, soit à la suite d'une coupure de courant. Pensez à votre magnétoscope, par exemple. Vous pouvez programmer les heures d'enregistrement autant de fois que vous le voulez.

Cela n'est possible parce qu'il comporte, lui aussi, des mémoires librement programmables ou RAMs.

RAM est en effet l'abréviation de Random Access Memory = mémoire à accès sélectif, c'est-à-dire mémoire à laquelle on peut accéder, au choix, en lecture ou en écriture.



La RAM est donc notre mémoire librement programmable. C'est là que nous pouvons placer nos données, nos lignes de programmes ou même nos lettres d'amour. Mais attention, si vous voulez conserver éternellement ces données, il vous faut laisser l'ordinateur allumé, de façon à ce que ses RAMs restent également allumées, ou bien transférer leur contenu sur d'autres moyens de stockage magnétique tels que les disquettes ou le disque dur. C'est comme lorsque vous écoutez de la musique : vous ne pouvez retenir entièrement tout un morceau de musique si vous ne l'avez écouté qu'une fois. Mais si vous enregistrez ce morceau sur une bande magnétique, vous pourrez le rappeler à votre souvenir autant de fois que vous le souhaitez.

### 5.1.3. Au fond, les ordinateurs ne sont pas intelligents

Vous rappelez-vous comment fonctionnait l'ordinateur Zuse ? On alignait à la suite les uns des autres de nombreux relais ou commutateurs qui ne pouvaient représenter que les états "courant passe" ou "courant ne passe pas". Une fois le calcul effectué, de petites lampes affichaient le résultat.

Mais comment un ordinateur fait-il, et pas seulement notre ST, pour travailler avec seulement ces deux états 'allumé' et 'éteint' ? Dans la préhistoire, nos ancêtres développèrent arbitrairement différents systèmes numériques. Le plus répandu et celui qui finalement l'emporta sur les autres était le système décimal qui est à la base des mots que nous utilisons pour désigner les nombres. Ce système décimal comporte dix chiffres différents :

0 1 2 3 4 5 6 7 8 9

Lorsque vous comptez, vous commencez donc par la gauche, par le 0. Que faites-vous lorsque vous arrivez au 9 ? Evidemment, vous recommencez par le chiffre le plus à gauche, mais vous le faites précéder d'un 1. Vous obtenez ainsi le nombre 10. Vous conservez le 1 jusqu'à ce que vous arriviez à nouveau à 9. Vous recommencez à nouveau à 0 mais avec un 2 comme premier chiffre, ce qui donne bien sûr 20. Et ainsi de suite, indéfiniment.

Quand on arrive à 9 pour les deux premiers chiffres, c'est-à-dire à 99, ces deux chiffres passent à 0 et on place à nouveau un 1 devant ces deux chiffres. On procède ainsi pour les centaines, puis les milliers, etc...

Pour nous, être humains, ce système décimal est sans doute celui qui nous permet le plus facilement de retenir et d'utiliser des nombres. Il n'empêche que le choix de ce système a quelque chose de totalement arbitraire : pourquoi utilisons-nous dix chiffres différents plutôt que douze, soixante ou deux par exemple ?

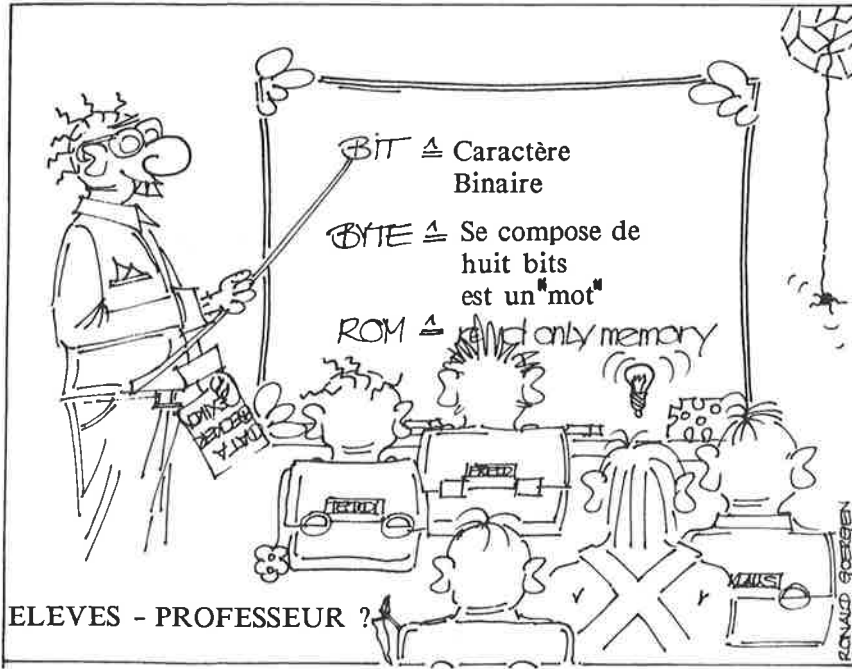


Lorsque les hommes inventèrent les ordinateurs, ils se demandèrent comment apprendre un système numérique, notre système décimal de préférence, à l'ordinateur.

On développa des commutateurs qui pouvaient distinguer plus de deux états : il passe beaucoup de courant, il passe peu de courant, il ne passe pas de courant. Malheureusement, il arrivait que l'ordinateur ne fasse pas la différence entre peu et beaucoup de courant. Le résultat obtenu était alors incorrect. On ne supporterait pas aujourd'hui d'avoir une calculatrice de poche qui fournisse des résultats aussi incertains.

On convint donc finalement de ne faire travailler l'ordinateur qu'avec deux états différents : allumé ou éteint. On pouvait alors être totalement certain que ce système fonctionnerait correctement! Les mathématiciens avaient déjà donné, dans leurs théories, un nom à un système numérique ne comportant que deux états, 'allumé' et 'éteint' ou 'oui' et 'non' ou encore '0' et '1'. Ils appelaient un tel système 'système binaire'.

Mais comment représenter des nombres importants lorsqu'on ne dispose que des chiffres 0 et 1 ?



La procédure est en fait la même qu'avec le système décimal: le plus petit nombre binaire est 0. Vient ensuite 1. Comme nous ne disposons pas, comme avec le système décimal, de chiffres supérieurs à 1, il nous faut, dès après 1, placer un 1 devant le 0 ce qui nous donne le prochain nombre : 10. Bien sûr, il ne faut pas lire dix mais un-zéro car il s'agit de l'écriture binaire de 'deux'!

Nous pouvons alors augmenter à nouveau de 1 le plus petit chiffre ce qui nous donne le nombre binaire 11. Les deux chiffres utilisés sont déjà parvenus chacun à leur valeur maximum, comme tout à l'heure avec 99 en système décimal, et il nous faut à nouveau ajouter devant un chiffre supplémentaire. Nous obtenons 100. Les valeurs suivantes seront 101, 110 et 111.

A première vue, ce système numérique, le système binaire, peut vous sembler très compliqué mais il fonctionne, en réalité, exactement comme le système décimal qui vous est si familier: lorsqu'un chiffre parvient à sa valeur maximum, 9 dans le système décimal, 1 dans le système binaire, on écrit tout simplement un chiffre supplémentaire devant ce chiffre ou bien on augmente de 1 le chiffre précédent s'il y en a déjà un.

Comparons maintenant les premiers nombres tels qu'ils se présentent dans les deux systèmes numériques :

Décimal	Binaire
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111

Bien entendu, s'il nous fallait, à nous pauvres humains, utiliser ce système, il nous faudrait des heures pour compter jusqu'aux centaines ou aux milliers. Il existe heureusement une méthode plus rapide pour compter en système binaire ou plutôt pour écrire des nombres binaires à partir des nombres décimaux qui nous sont familiers, c'est-à-dire pour convertir des nombres décimaux en nombres binaires. Il suffit pour cela d'utiliser les mathématiques. Comme le système binaire comporte deux chiffres, la solution nous sera fournie par les différentes puissances de deux :

$$\begin{aligned}2^0 &= 1 \\2^1 &= 2 \\2^2 &= 4 \\2^3 &= 8 \\2^4 &= 16 \\2^5 &= 32 \\2^6 &= 64 \\2^7 &= 128\end{aligned}$$



A partir de ces puissances de deux, nous pouvons écrire la table suivante dans laquelle nous pourrions placer n'importe quel nombre binaire :

=====						
....2^5	2^4	2^3	2^2	2^1	2^0	
=32	=16	=8	=4	=2	=1	
=====						
			1	1	0	0
=====						
			8	+	4	=12

On peut alors commencer à additionner. Partout où il y a un 0 dans le nombre binaire, on peut continuer à lire les chiffres suivants mais partout où l'on trouve un 1 on additionne la puissance de deux correspondante, c'est-à-dire le nombre de la seconde ligne. Avec le nombre binaire 1100, cela nous donne  $8 + 4 = 12$ .

Un autre exemple :

$$111 \text{ binaire} = 2^2 + 2^1 + 2^0 = 4 + 2 + 1 = 7$$

C'est ainsi qu'on peut représenter avec huit chiffres binaires des valeurs allant jusqu'à 255 et avec seize chiffres binaires des valeurs allant jusqu'à 65535.

Nous n'en dirons pas plus sur le système binaire. L'essentiel est que vous en ayez compris le principe car dans votre travail avec le ST vous rencontrerez toujours ce système binaire. Cela ne vaut d'ailleurs pas seulement pour votre ST mais aussi pour toutes les machines ou appareils sur le marché qui comportent un ordinateur intégré. Seul ce système numérique permet en effet d'apprendre à une machine à opérer des calculs. La logique informatique repose sur des opérations de calcul et les opérations mathématiques logiques en système binaire sont à la base du fonctionnement des ordinateurs.

Avant que nous ne nous replongions dans la pratique, encore une indication importante : outre le système binaire et le système décimal, il existe encore beaucoup d'autres systèmes numériques.

Il y a par exemple un système numérique comportant huit états différents :

0 1 2 3 4 5 6 7 = le système octal

Il existe même un système numérique comportant seize états différents :

0 1 2 3 4 5 6 7 8 9 A B C D E F = le système hexadécimal

Le principe de fonctionnement de ces systèmes reste le même que celui que nous avons étudié pour les systèmes décimal et binaire.

Notez par ailleurs que vous trouverez en annexe de ce livre de petits programmes BASIC qui vous permettent de faire effectuer facilement par votre ST des conversions en différents systèmes numériques.

#### 5.1.4. Que veut dire 512 kilo-octets ?

Vous vous souvenez de ce qu'est la RAM ? La RAM est une mémoire de votre ordinateur dans laquelle vous pouvez placer vos programmes, textes, etc... Lorsque vous éteignez votre ST ou lorsqu'une panne de courant se produit, toutes les informations contenues dans la mémoire RAM disparaissent.



Les ATARI ST sont équipés de 512 Koctets, ou plus, de mémoire programmable. Les autres ordinateurs sont également affublés d'un tel qualificatif, un peu à la manière des chevaux pour les automobiles. Que signifie l'abréviation Koctet ou K ?

Chaque chiffre d'un nombre binaire est appelé bit (BInary digiT = chiffre binaire). Un bit peut donc prendre les valeurs 0 ou 1. Un nombre de 8 bits est appelé un octet, un tel nombre peut prendre 256 (0 à 255) valeurs différentes, tout comme avec deux chiffres décimaux on peut former 100 valeurs différentes (00 à 99).

L'octet est donc l'unité de base qui permet de mesurer la taille d'une mémoire. Comme tout ce qui touche à l'ordinateur est compté en binaire, les nombres en question sont donc des multiples de 2. Pour cette raison, en informatique un "kilo" ne vaut pas 1000 mais 1024 (2 puissance 10), c'est une convention. Un kilo-octet vaut donc 1024 octets, un kilo-bit vaut 1024 bits.

Ainsi il existe des puces mémoires de 256 Kbits, ce sont elles que l'on retrouve dans notre ST. Si nous prenons 8 de ces puces, nous obtenons 256 Koctets, il en faut donc 16 pour arriver à 512 K, la mémoire des 520 ST/M, et 32 pour arriver à 1024 Koctets (ou 1 Moctet) comme c'est le cas des 1040 STF.

Les machines plus récentes MEGA ST utilisent les nouvelles mémoires de 1 Mbit (1024 Kbits), il en faut donc 16 pour obtenir les 2 Moctets du MEGA ST2, et 32 pour obtenir les 4 Moctets du MEGA ST4.

Nous savons maintenant que 512 K correspondent à 524288 octets ; avouez qu'il est plus simple de dire "512 K". Cette taille mémoire correspond à la mémoire RAM. Lorsque nous tapons du texte, chaque caractère occupe un octet de la RAM, nous savons donc combien de caractères notre ST peut emmagasiner.

Que ce soient 524288, 1048576, 2097152 ou même 4194304 caractères, il y a peu de risque que les programmes que vous pourrez écrire remplissent cette mémoire. Il ne faut toutefois pas oublier que le BASIC lui-même occupe également une partie de la RAM.

Cette profusion de mémoire est particulièrement précieuse lorsque l'on a de grandes quantités de texte à manipuler. Le présent livre par exemple est constitué d'environ 500000 caractères, il pourrait tenir en entier dans la mémoire du ST ! A raison d'environ 2000 caractères par page, notre ST pourrait contenir un livre de 260 pages, et un MEGA ST4 plus de 2000 pages.



### **5.1.5. La Fonction du microprocesseur**

Nous avons maintenant appris à évaluer la taille de notre mémoire, nous savons aussi qu'il y a des ROMs et des RAMs mais cela n'est pas encore tout à fait suffisant. L'ordinateur doit disposer d'un 'coeur' qui soit capable de gérer les nombreuses places mémoire mais aussi de convertir les données décimales entrées dans l'ordinateur en système binaire, c'est-à-dire dans le système numérique propre de l'ordinateur.

Ce composant électronique est appelé microprocesseur ou unité centrale, Central Processing Unit ou CPU en anglais. Notez cependant qu'on préfère parfois distinguer l'unité centrale du microprocesseur. On étend alors la notion d'unité centrale pour désigner sous ce terme l'unité constituée par la RAM, la ROM et le microprocesseur. Sachez-le pour éviter toute confusion si vous rencontrez ce terme dans la littérature technique.

Le microprocesseur est, lui aussi, un chip qui comprend de la ROM, qui contient son langage propre, le langage machine, mais aussi de la RAM, qui contient un certain nombre de ce qu'on appelle des registres où s'effectue tout le travail de traitement réalisé par l'ordinateur. Le microprocesseur intégré dans votre ST est le microprocesseur 68000 de la firme américaine Motorola. Il s'agit d'un microprocesseur très puissant et très répandu.

Vous souvenez-vous encore de la section que nous avons consacrée au mode de calcul des ordinateurs avec des uns et des zéros ? Vous souvenez-vous aussi qu'avec huit uns on représente le nombre 255 et qu'avec seize uns on représente même le nombre 65535 ? Un 1 ou un 0 est appelé bit. Le 68000 est ce qu'on appelle un microprocesseur 16/32 bits parce qu'il peut traiter au même moment des nombres binaires jusqu'à 65535 en décimal. Il existe aussi des microprocesseurs 8 bits tels que le Z80A qu'on trouve sur les ordinateurs MSX ou les Amstrad CPC ou aussi le Commodore 128. Avec ses environ 600 instructions, il est beaucoup plus difficile à programmer que le 68000 bien qu'il soit loin d'être aussi puissant.

Alors que le 68000 ne peut traiter que des données d'une longueur de 16 bits 'seulement' (0 à 65535 en décimal), son bus d'adresse est doté de 24 canaux soit 24 bits ou des nombres décimaux de 0 à 16777215. Ce n'est qu'ainsi qu'il devient possible d'utiliser une mémoire de 512 kilo-octets. Avec 16 canaux d'adresse (16 chiffres binaires) on ne peut appeler 'que' 65536 octets.

Encore deux choses :

1. La prochaine génération d'ordinateurs travaillera, dans quelques années, en grande partie avec des processeurs 32 bits. Après avoir lu ce chapitre, vous comprenez peut-être mieux pourquoi un microprocesseur 32 bits est beaucoup plus puissant que notre 68000. Un 32 bits peut traiter simultanément des nombres de 0 à plus de 4 milliards alors qu'un 16 bits comme notre 68000 ne peut traiter au même moment 'que' des nombres de 0 à 65535. Mais il s'écoulera encore pas mal de temps avant que les microprocesseurs 32 bits inondent le marché aux prix des microprocesseurs 16 bits actuels tels que le 68000.

2. Essayez de retenir encore un autre composant du microprocesseur: la ALU (Arithmetic Logic Unit) qui est la véritable calculatrice du microprocesseur.

Le microprocesseur a lui aussi son langage de programmation. Celui-ci ne doit toutefois pas être confondu avec les langages de programmation que nous avons déjà étudiés dans cet ouvrage. Nos langages de programmation BASIC et LOGO se composent d'instructions qui sont aisément compréhensibles lorsqu'on dispose de connaissances élémentaires en anglais. Même si on ne connaît d'ailleurs rien à l'anglais, il n'est pas difficile d'associer une fonction à un mot donné, comme PRINT pour écrire sur l'écran, LIST pour lister un programme sur l'écran ou FORWARD pour faire avancer le curseur de dessin.

Parce que le BASIC et le LOGO sont aussi faciles (!?), on dit que ce sont des langages de programmation évolués. Si l'on veut par contre programmer directement le microprocesseur, on doit disposer pour cela de beaucoup plus de connaissances.

En effet, pour reconstituer la fonction d'une simple instruction BASIC ou LOGO, il faut un grand nombre d'instructions d'organisation de la mémoire très abstraites. L'intérêt de programmer directement en langage machine est toutefois qu'on peut ainsi écrire des programmes pouvant être jusqu'à plusieurs centaines de fois plus rapides qu'en BASIC ou en LOGO.

## **5.2. PERSPECTIVES D'AVENIR**

Vous savez maintenant beaucoup de choses sur votre ATARI ST. Mais vous avez peut-être déjà pensé à étendre et à améliorer encore votre installation informatique.

L'ATARI ST dispose d'un grand nombre de connexions qui permettent de le connecter à presque tout ce qu'on trouve actuellement sur le marché. De plus, il existe déjà de nombreux programmes tout prêts pour votre ST, avec lesquels vous pouvez faire du traitement de texte, de la gestion de fichier, des jeux, etc... Nous souhaitons vous donner maintenant un petit aperçu de tout ce que vous pouvez faire avec votre ST.

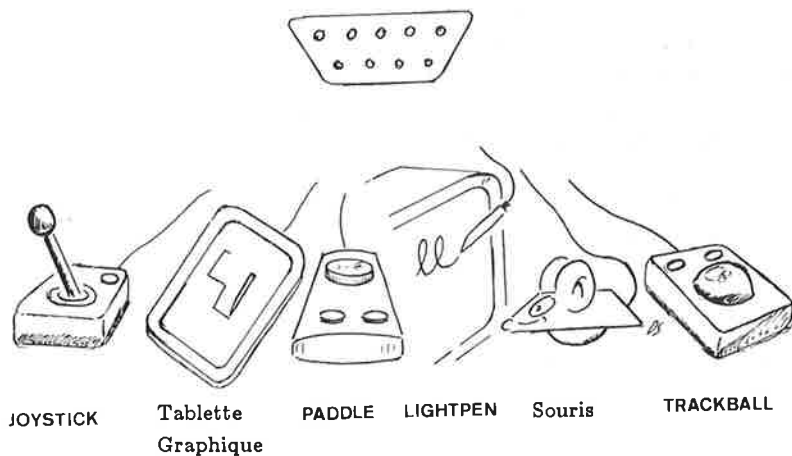
### **5.2.1. Connexions à gogo**

Comme nous le savons déjà, un ordinateur se compose entre autre de ROM, de RAM et du microprocesseur. Alors que la ROM comprend des informations ineffaçables et que la RAM fournit la place mémoire libre (pour nos programmes et textes), ce qu'on appelle le microprocesseur, le cœur du ST, veille à ce que la communication entre RAM et ROM mais aussi la communication avec l'extérieur fonctionne sans problème.

Suivant le modèle de ST dont vous disposez, vous avez constitué vous-même votre installation ou bien vous avez acheté un paquet ST qui se compose de l'ordinateur, du lecteur de disquette, du moniteur et de la souris.

En plus de ces différents éléments qui constituent le minimum nécessaire pour une installation ST, vous pouvez encore connecter de nombreux autres appareils à votre ATARI ST. Commencez tout de suite à faire des économies car, en lisant les pages suivantes, vous aurez certainement le désir d'agrandir votre installation !

### 5.2.1.1. Deux ports de joystick



### Le joystick

Les actions de notre ST peuvent être partiellement 'télécommandées' au moyen d'un joystick. Pour cette raison tous les ST disposent sur leur coté droit, ou sous la machine, de deux connecteurs (dont un est déjà utilisé par la souris).

Ces deux connecteurs sont prêts à accueillir deux joysticks (ou manches à balai), mais aussi d'autres périphériques comme les paddles (joystick analogique) ou une tablette graphique.

Le joystick permet de lire 8 directions différentes (N, NE, E, SE, S, SW, W, NW).



L'ordinateur peut donc reconnaître 8 positions du joystick espacées par intervalles de 45 degrés.

En poussant le manche du joystick dans la direction voulue, vous pouvez piloter les mouvements d'un objet placé sur l'écran. Bien entendu, vous pouvez utiliser le joystick avec ses boutons pour des jeux ordinaires mais il est également possible de l'employer dans des applications plus sérieuses comme par exemple de dessiner une image qui sera sauvegardée sur disquette lorsque vous appuierez sur le bouton. Vous pouvez d'ailleurs imaginer vous-même à quoi vous voulez employer ces instruments de télé-commande.

### Tablette graphique

Le dernier exemple d'application, le dessin avec un joystick, peut être cependant réalisé de façon beaucoup plus simple et pratique avec ce qu'on appelle une tablette graphique. La tablette graphique doit être également connectée aux ports joystick de votre ST. La tablette graphique est une surface de dessin, de la taille approximativement d'une carte postale, qui représente l'écran en réduction.

Une poignée en plastique est normalement livrée avec la tablette graphique. Cette poignée de dessin doit être placée sur la tablette graphique et vous pouvez alors réaliser à l'écran les plus beaux dessins en appuyant simplement la poignée sur la tablette graphique. Celle-ci transmet à l'ordinateur le point d'écriture correspondant exactement à l'endroit où vous dessinez. Vous pouvez utiliser également votre ongle pour dessiner mais attention à ne pas rayer la tablette !

C'est bien sûr beaucoup plus pratique pour dessiner que d'utiliser un joystick dont le manche revient toujours en position centrale lorsque vous le relâchez. Cependant une tablette graphique est beaucoup plus chère qu'un joystick car tous les points de la tablette doivent être testés de façon très précise pour que l'information puisse être transmise à votre ordinateur ST. D'autre part, une tablette graphique ne peut être utilisée que pour le dessin.

Naturellement, la tablette graphique dispose aussi d'un ou deux boutons à presser, soit à l'extérieur de la tablette, soit directement sur la poignée en plastique. De plus, il y aura certainement à l'avenir des logiciels pour les tablettes graphiques qui sortiront sur l'écran une sorte de menu pour les différentes fonctions. Il vous suffira alors d'amener la poignée sur l'endroit de la tablette correspondant à la fonction voulue (par exemple sauvegarde ou choix de couleur ...).

### Paddle

Un troisième instrument, de même type que le joystick, que vous pouvez connecter à un ou aux deux ports joysticks (si vous êtes deux joueurs) : le paddle. Le paddle se compose d'un bouton que vous pouvez tourner pour commander le déplacement d'un objet vers la gauche ou la droite.

Pour mieux comprendre la fonction d'un paddle, voici deux exemples d'utilisation d'un paddle :

1. Vous voulez conduire une voiture sur l'écran. Vous pouvez utiliser pour cela le bouton du paddle comme un volant. Le paddle dispose aussi d'un bouton sur un côté. Si vous appuyez sur ce bouton, votre voiture roulera plus vite. Si vous le relâchez, elle ralentira.
2. Vous voulez faire sauter un petit bonhomme, dans un jeu, sur une balançoire. Si le petit bonhomme tombe bien sur la balançoire, il est renvoyé très haut où il peut atteindre des ballons de baudruche. La pesanteur ramène enfin le bonhomme au sol et vous devez alors ajuster à nouveau la balançoire de façon à ce que le bonhomme puisse y atterrir correctement.

Les deux jeux pourraient être réalisés également avec un joystick mais alors ils ne seraient pas aussi pratiques et leur mode d'emploi ne rappellerait pas autant la vie réelle, surtout pour la course de voitures.

### Track-ball

Il s'agit d'une petite boîte, de la taille d'une soucoupe, que l'on place sur la table. Le nom de cet appareil de commande vient d'une boule (ball en anglais) qui se trouve dans la boîte. La boule dépasse du boîtier de façon à ce qu'elle puisse aisément être mûe dans toutes les directions lorsqu'on place sa main dessus. La fonction de la track-ball est donc la même que celle du joystick, indiquer une des 8 directions possibles, mais la track-ball est toutefois beaucoup plus facile à manier.

### Souris

La souris connectée à votre ST correspond à une technique voisine de celle de la track-ball. C'est au fond une track-ball mais sa boule n'est mise en mouvement que par le fait de déplacer le boîtier n'importe où sur la surface de la table. La boule se trouve en effet sur le dessous du boîtier, comme une roue.

Il va de soi que la souris, aussi bien que la track-ball, disposent au moins d'un bouton de commande qui remplit les mêmes fonctions que celles déjà expliquées en détail pour le joystick. Si la souris ATARI est connectée sur le port joystick 0, les deux touches qu'elles porte sur sa face supérieure peuvent être interrogées pour deux fonctions différentes.

### Lightpen

Vous souvenez-vous de ce que nous avons dit sur la tablette graphique ? Vous utilisez une poignée graphique qui vous permet de réaliser votre créativité sur une tablette de la taille d'une carte postale dotée de cellules sensorielles.

Le lightpen est également un instrument pour le dessin mais il vous permet de dessiner directement sur la surface de l'écran. Ne craignez rien, il n'y a pas de risque de rayer l'écran de votre moniteur ou encore moins de le gribouiller de toutes les couleurs.

Comme son nom l'indique le lightpen ou crayon optique contient une photodiode qui est à même de déterminer sur quel endroit de l'écran vous le pointez.

L'interrogation d'un lightpen sur un moniteur informatique spécial ne pose pas de problème mais c'est déjà très différent si vous le connectez à un téléviseur ordinaire. En effet, les lightpens existant pour d'autres ordinateurs sur le marché des accessoires n'offrent pas toujours des résultats satisfaisants. C'est que l'épaisse paroi de verre de nos téléviseurs dévie les rayons lumineux vers l'extérieur de sorte que le crayon optique apposé sur l'écran ne peint pas là où il devrait mais le plus souvent quelques centimètres plus à gauche ou plus à droite, plus haut ou plus bas que le point de l'écran sur lequel il a été dirigé. Cela marche déjà mieux avec un moniteur couleur dont vous aurez ôté la vitre de contraste (s'il y en a une). Pour le moment toutefois, ce système ne dépasse guère le niveau du jeu ou de la démonstration. L'effet est cependant toujours très impressionnant.

Lorsque ces lignes ont été écrites, nous ne savions pas encore s'il existerait sur le marché, dans un proche avenir, un crayon optique pour les ordinateurs ATARI ST. On ne savait pas non plus si le crayon optique devrait être connecté au port joystick.

Il y a certainement de nombreuses autres possibilités de tirer parti des prises joystick de votre ST, notamment pour les bricoleurs qui veulent contrôler des commandes ou pour les biologistes qui veulent surveiller la mesure d'une eau.

Mais attention, il ne suffit en aucun cas de simplement connecter un de ces périphériques au ST, il faut aussi prévoir un programme qui sache le reconnaître et interpréter les informations transmises. De telles routines sont en général intégrées dans les jeux, pour permettre une utilisation de joysticks, et lorsque l'on achète un crayon optique ou une tablette graphique, le périphérique est en général livré avec un programme de commande.

### 5.2.1.2. Connexion pour les lecteurs de disquette



500 kB



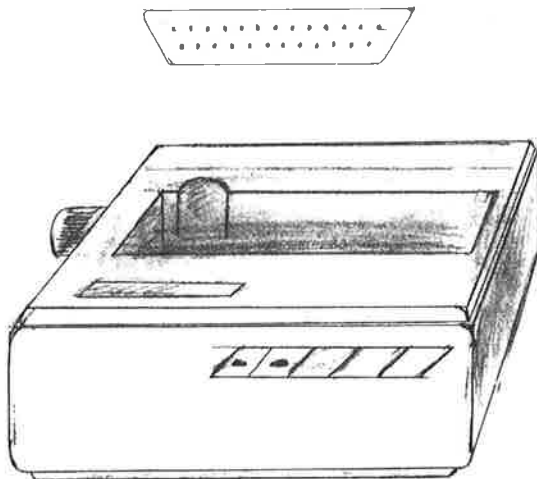
1 MB

Vous pouvez connecter à votre ST un lecteur de disquettes 3.5 pouces. Un deuxième lecteur 3.5 pouces peut être connecté au premier ; le chapitre 1.3 décrit cela en détail. Les lecteurs proposés par ATARI pour le ST ont une capacité de 500 Koctets et 1 Moctet.

### 5.2.1.3. La connexion Centronics

#### IMPRIMANTE

##### PRINTER



Pour la connexion d'une imprimante, ce qu'on appelle un port parallèle Centronics est intégré sur votre ST.

Centronics est le nom de la firme qui a développé cette connexion et qui est par ailleurs une importante firme de fabrication d'imprimantes. Mais que signifie connexion parallèle ?

Vous souvenez-vous encore de notre digression sur les bits et les octets ? 8 bits, c'est-à-dire 8 chiffres d'un nombre binaire, constituent un octet. A travers la connexion Centronics, votre ST ne peut envoyer à la fois que 8 bits, c'est-à-dire 8 signaux électriques (allumé ou éteint) à l'imprimante. Ces signaux sont envoyés l'un à côté de l'autre, au même moment. C'est donc parce que les bits sont envoyés l'un à côté de l'autre qu'on parle de transmission parallèle.

Nous avons appris plus haut que tous les nombres de notre système décimal peuvent également être représentés sous forme de nombres binaires. On a, de même, attribué aux lettres des codes chiffrés qui peuvent être envoyés à l'imprimante sous la forme de nombres binaires, à travers la connexion parallèle.

La transmissions de bits peut également être effectuée de façon sérielle, c'est-à-dire bit après bit. Vous imaginez bien cependant que la transmission sérielle fonctionne beaucoup plus lentement que la transmission parallèle.

On est donc fondé à affirmer que votre connexion parallèle Centronics permet une transmission de données très rapide entre ordinateur et imprimante. C'est d'ailleurs pour cela, entre autre, que tous les ordinateurs personnels importants ont des connexions Centronics.

Encore quelques mots sur les types d'imprimante que vous pouvez connecter ici :

Vous avez certainement déjà entendu parler des imprimantes à aiguille ou imprimantes matricielles qui sont aujourd'hui les plus répandues. Ce système ne produit pas toujours sur le papier des figures de lettres très propres mais vous pouvez par contre compter sur une vitesse d'impression qui est le plus souvent de 50 caractères par seconde et qui est souvent très supérieure. Cela tient au faible travail mécanique que constitue le fait de ramener en arrière ou d'appuyer sur le papier les aiguilles. Avec une machine à écrire ordinaire ou avec une imprimante à marguerite, qui fonctionne un peu comme une machine à écrire, une telle vitesse ne pourrait être atteinte, sauf bien sûr si vous en payez le prix qui peut être très élevé. En effet, à une telle vitesse, les lettres ne tarderaient pas de se bloquer car elles n'auraient même pas le temps de revenir dans leur position de départ.

Outre les imprimantes matricielles et à marguerite, il existe aussi des imprimantes à jet d'encre, des imprimantes thermiques et des imprimantes laser.

Le principe de l'imprimante à jet d'encre est proche de celui de l'imprimante à aiguilles, celles-ci étant remplacées par de minuscules gicleurs qui par une mécanique de haute précision projettent point par point de fines gouttelettes sur le papier.

L'avantage des imprimantes à jet d'encre est leur grand silence de fonctionnement, en revanche elle ne permettent pas de réaliser des copies carbone.

Les imprimantes thermiques sont encore moins bruyantes que leurs homologues à jet d'encre. La tête d'impression développe des zones de chaleur en fonction du caractère à imprimer, à ces endroits le papier spécial noircit. Les imprimantes thermiques sont économiques à l'achat, mais il ne faut pas négliger le prix du papier spécial qu'il faut utiliser.

Finalement nous arrivons aux imprimantes laser qui grâce à cette technique moderne permettent d'imprimer une page complète en quelques secondes, et ce avec une très grande résolution. Atari propose déjà lui-même une imprimante laser avec une excellente résolution et à un prix très avantageux.

Nous avons vu que par l'intermédiaire du connecteur Centronics il était possible de connecter pratiquement n'importe quelle imprimante du marché. Mais tout n'est pas encore réglé.

Bien qu'il soit possible de connecter toute imprimante au standard Centronics, que ce soit une EPSON, une NEC, une PANASONIC, une STAR ou toute autre marque, il faut encore régler le problème du jeu de caractères.

Qu'est-ce qu'un jeu de caractères ? Chaque caractère est stocké sous forme codée dans le ST et il existe donc une table de correspondance entre chaque caractère et son code. Ainsi la lettre 'A' a pour code le nombre 65, l'espace a pour code 32, et ainsi de suite. Pour des raisons évidentes de compatibilité et de clarté, il a été défini il y a déjà un certain nombre d'années une table de codage standard des caractères, appelée code ASCII. Malheureusement ce standard ne porte que sur les caractères dont le code est compris entre 0 et 127.



Si vous comptez faire l'acquisition d'une imprimante, demandez conseil à votre revendeur pour qu'il vous indique quelles imprimantes peuvent être connectées au ST. Il est d'ailleurs probable que de plus en plus de constructeurs proposent des modèles de leurs imprimantes spécialement adaptées au ST (comme par exemple la TAXAN KP 810 ST).

Qu'est-ce qu'un jeu de caractères ? Comme nous l'avons déjà indiqué à plusieurs reprises, votre ST ne travaille que de façon purement mathématique. Même les caractères apparaissant à l'écran sont codés sous forme de codes numériques. C'est ainsi que la lettre A porte le numéro 65, que le caractère espace porte le numéro 32, etc...

Pour qu'il n'y ait pas une trop grande confusion parmi les différents constructeurs d'imprimantes, on s'est mis d'accord, il y a déjà plusieurs années sur un jeu de caractères standard appelé ASCII. Ce standard ne concerne cependant que la représentation des caractères dont les codes sont compris entre 0 et 127.

Si vous voulez donc ajouter une imprimante à votre installation, nous vous recommandons de vous faire conseiller par un commerçant spécialisé qui puisse vous indiquer quel modèle vous pouvez utiliser. Avec le temps, il y aura certainement de plus en plus de constructeurs qui proposeront sur le marché des imprimantes compatibles ST, c'est-à-dire pouvant être utilisées sans problème par votre ordinateur ST.

#### 5.2.1.4. La connexion de cartouche

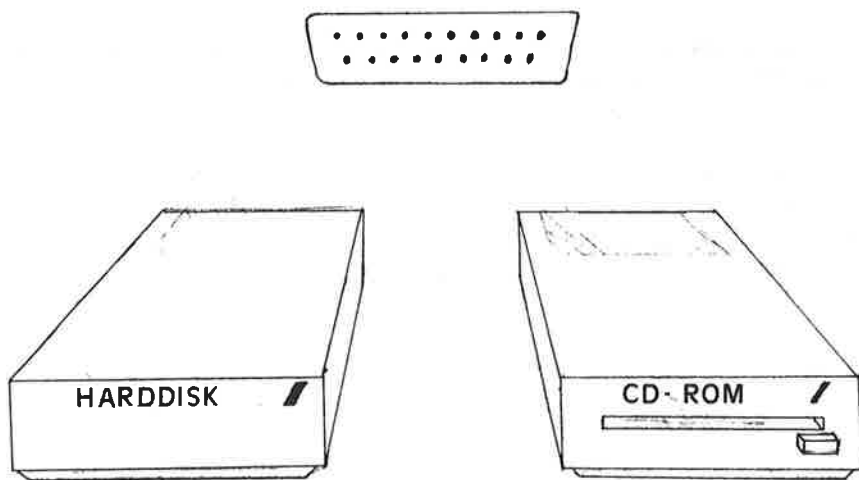
Tous les ATARI ST disposent enfin d'une connexion d'extension également appelé port cartouche (cartridge port). Cette connexion se trouve sur le côté gauche de votre ST.

Une cartouche est un boîtier plastique plat dans lequel figure une platine portant une ROM (vous vous souvenez certainement encore de ce qu'est une ROM). L'avantage des programmes sur cartouche, par rapport aux mêmes programmes sur disquette, est qu'il vous suffit d'enficher la cartouche dans la connexion cartouche de votre ordinateur pour que le programme soit prêt à travailler quelques secondes plus tard.

Un programme sur disquette nécessite, lui, pour son chargement, de nombreuses secondes et il vous faut même parfois attendre plusieurs minutes avant que vous ne puissiez commencer à travailler.

#### 5.2.1.5. La connexion du disque dur

Il est possible de connecter des mémoires de masse encore bien plus rapides que les lecteurs de disquettes sur le ST, comme les lecteurs de disques optiques et CD-ROM.



Pour vous donner un aperçu des vitesses de transmission des données : les disquettes envoient leurs données à 250000 bits par seconde alors que le connecteur pour disque dur permet une transmission à 1000000 bits par seconde (voir BAUD).

### Connexion d'un lecteur de vidéo-disque



Si vous examinez de plus près les lecteurs de vidéo-disques, vous constaterez que le mode d'emploi pour sélectionner des images isolées est très compliqué mais que c'est néanmoins possible.

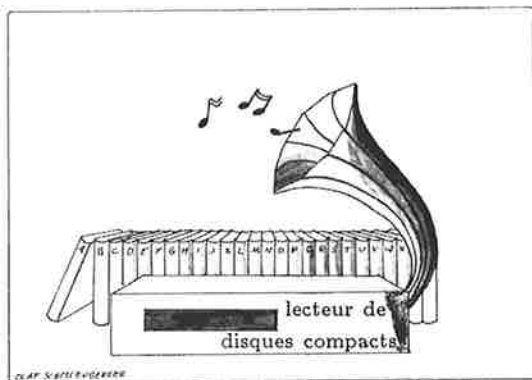
Avec un lecteur de vidéo-disques connecté à votre ST à travers la connexion disque dur, vous pourriez par exemple, dans un garage, faire apparaître sur le moniteur, en une fraction de seconde, une diapositive du modèle voulu.

Vous pourriez ainsi présenter à chaque client une image représentant non seulement le modèle qui l'intéresse mais aussi la couleur de carrosserie et les accessoires voulus. Ce travail de commande avec le ST ne présente pas de problème technique sérieux. Cela nécessiterait cependant le stockage sur un vidéo-disque de diapositives représentant tous les modèles dans toutes les teintes. Ce serait un procédé très coûteux qui ne présenterait certainement d'intérêt que pour un très gros concessionnaire.

Une application plus accessible consisterait par exemple à faire un programme sur la vie des animaux. Vous diriez quelle sorte d'animal vous recherchez (mammifère, vivant en Amérique du Nord). Après que votre ST ait traité toutes ces données, il commanderait le vidéo-disque de façon à ce que la diapositive recherchée apparaisse à l'écran.

Une troisième application possible pour les lecteurs de vidéo-disque avec votre ST : un jeu d'aventure. Un jeu d'aventure est un jeu qui repose sur une action proche de celle d'un livre. Chaque fois qu'une action intéressante commence, l'image correspondante sera affichée sur l'écran. Vous pourriez même affiché une suite d'images, un petit film dont vous pourriez vous-même déterminer le déroulement, peut-être au moyen de votre joystick. Un jeu électronique réaliste qui comblerait toutes les attentes.

### Connexion d'un lecteur de disques compacts, CD



Vous pourriez travailler de manière semblable en connectant à votre ST un lecteur de disques compacts. Pour un musicien, il serait peut-être intéressant de faire commander telle ou telle suite de notes mais ce n'est pas de cela que nous voulons parler. On peut en effet imaginer d'utiliser un disque compact comme une disquette de mémoire de masse.

Alors qu'une disquette ne peut recevoir 'que' 180 pages de format A4, une disque compact pourrait en recevoir 250000.

Que diriez-vous par exemple de pouvoir rapidement consulter, avec votre ST et un lecteur CD, un dictionnaire stocké sur un disque compact ?

Comprenons-nous bien : cette masse d'informations ne serait pas dans la RAM de votre ordinateur ST mais sur le disque CD (ou sur un vidéo-disque d'un lecteur de vidéo-disques). Votre ST n'aurait qu'à commander la recherche des données. Il pourrait lire toutes les informations en quelques secondes ou retrouver ce que vous recherchez grâce à une méthode d'organisation de fichier adaptée. Il ne lui resterait plus alors qu'à sortir sur l'écran les informations voulues.

### Branchement d'un disque dur

Bien entendu, et c'est son rôle premier, vous pouvez aussi brancher un disque dur sur le connecteur pour disque dur. Une condition préalable à cela est toutefois que vous disposiez d'un ST avec TOS en ROM. A côté de quelques disques durs proposés par des constructeurs indépendants, il existe deux modèles chez ATARI, le SH 204 en boîtier métallique gris et le plus récent SH 205 en boîtier plastique, assorti à la nouvelle série des MEGA ST. Les deux modèles sont techniquement équivalents et ne se distinguent, outre par leur boîtier, que dans la mesure où le SH 205 voit son connecteur reconduit, ce qui permet le branchement d'autres périphériques à la suite du disque dur.

Sortez le disque dur soigneusement de son carton, ainsi que toutes les autres pièces qui s'y trouvent. Vous devez vous trouver en présence du disque dur lui-même, du manuel d'utilisation, d'un câble secteur, d'un câble avec deux connecteurs larges et d'une disquette. Avant que nous ne commençons à installer le disque dur, réalisez une copie de sécurité de la disquette. Disposez maintenant le disque dur dans la position qu'il occupera par la suite. Le SH 205 trouve sa place idéale comme support pour le moniteur, avec un MEGA ST il peut servir de support à ce dernier. Tous les appareils étant éteints, reliez le disque dur au ST (connecteur 3, chapitre 1.1, inscription 'harddisk') au moyen du câble fourni. Sur le SH 205 utiliser le connecteur marqué 'harddisk in'. Vissez les vis de fixation pour éviter tout risque par la suite. A présent vous pouvez insérer le câble secteur. Une dernière vérification et on peut commencer.

Avant de pouvoir commencer à travailler avec le disque dur, celui-ci doit auparavant subir quelques préparations. Dans tous les cas il faut que vous soyez suffisamment familiarisé avec le ST pour que des problèmes, comme lancer un programme, n'en soient plus pour vous. Au besoin, si ce n'est déjà fait, lisez le chapitre 2 de ce livre.

Insérez la copie de sécurité de votre disquette dans le lecteur A, puis mettez en marche le disque dur (l'interrupteur se trouve à l'arrière du boîtier). Vous devez entendre le ronronnement du ventilateur, une petite lumière verte doit s'être allumée et une petite lumière rouge doit scintiller légèrement. Sinon vérifiez le câble secteur. Au bout de quelques instants, la lumière rouge doit s'éteindre. Allumez maintenant l'ordinateur, le desktop doit se présenter avec un icône lecteur de disque supplémentaire, portant le numéro C. Cliquez sur la disquette A et examinez son contenu. Vous y trouverez un auto classeur (\*) qui contient un programme driver sans lequel il n'est pas possible de travailler avec le disque dur.

La disquette doit également contenir un programme ayant pour nom 'HDX.PRG'. Lancez-le en cliquant deux fois dessus. Il s'agit d'un petit utilitaire qui permet de préparer le disque dur. Cette procédure ne doit être faite qu'une seule, avant toute utilisation du disque dur ; elle permet toutefois aussi, comme pour les disquettes, d'effacer complètement son contenu.

La barre de menu vous propose trois menus déroulants. Comme d'habitude, le menu 'Desk' propose un petit message d'information, le menu 'File' permet, par l'intermédiaire de 'Quit' de quitter le programme, mais intéressons nous au menu 'Disk'. Comme pour une disquette, il faut commencer par formater le disque dur. Pour cela appelez la fonction 'Format', confirmez avec 'OK'. Une boîte vous demande de choisir une unité. Comme vous ne disposez probablement que d'un seul disque dur, cliquez le point 'UNIT 0' et confirmez avec 'OK'. La boîte suivante vous demande un nom pour le disque dur. L'ATARI ST ne peut gérer que des disques durs d'un maximum de 16 Mo. Comme nous disposons de 20 Mo et que nous voulons les utiliser pleinement, il faut diviser le disque dur en plusieurs parties, appelées partitions. Chaque partition sera considérée par

la suite comme une unité de disquette, avec son propre icône. Cliquez donc 'Partition' et confirmez avec 'OK'. Indiquez à nouveau 'UNIT 0', puis 'OK'. Vous avez la possibilité de créer jusqu'à 4 partitions. Des tailles vous sont proposées, vous pouvez aussi choisir celles qui vous conviennent. Une partition peut être complètement inhibée en fixant sa taille à 0. Veillez à ce que la somme des tailles ne dépasse pas 20 Mo. Ce n'est pas le cas dans les tailles proposées par défaut ; réduisez donc la première partition à 0 puis agrandissez-la à 4 Mo. Lorsque vous êtes satisfait de votre choix, cliquez sur 'OK', les partitions sont alors installées sur le disque.

Vous avez certainement remarqué deux autres rubriques dans la liste. Bien que ce ne soit pas indispensable, nous vous conseillons de prendre la peine de les exécuter. Le premier point, 'Zero', remplit les différentes partitions avec des zéros, celles-ci sont alors complètement effacées. Renouvelez l'opération pour chacune des partitions.

Nous arrivons au dernier point. La fonction 'Markbad' marque les secteurs défectueux d'une partition. Il peut en effet arriver qu'un secteur ne soit pas correctement utilisable. Pour éviter l'utilisation de tels secteurs, ce programme les marque, ce qui évitera tout risque par la suite. Un secteur défectueux par Mo est tout à fait habituel, de sorte que notre disque dur pourrait en recenser une vingtaine.

Voilà, nous sommes arrivés au bout de nos peines. Quittez le programme avec 'Quit' et cliquez sur le disque C. Immédiatement vous voyez apparaître un catalogue vide. Normal, puisque le disque dur ne contient encore aucun fichier. Fermez toutes les fenêtres et examinez votre bureau. Bien que nous ayons créées plusieurs partitions, celles-ci n'apparaissent nulle-part. Pour changer cela, cliquez une fois sur le disque C pour le noircir puis allez dans la fonction 'Déclarer lecteur' (\*) du menu 'Options'. Transformez le C majuscule en un D majuscule et cliquez sur 'Déclarer' (\*). A l'écran vous voyez maintenant apparaître un icône supplémentaire, portant la lettre D. Répétez cette opération jusqu'à ce que toutes les partitions créées apparaissent à l'écran. Pour ne pas avoir à répéter ce procédé après chaque mise en marche du ST, nous allons

sauvegarder cette configuration sur la copie de sécurité de la disquette, au moyen de 'Sauvegarder' (\*).

A présent vous pouvez travailler avec le disque dur comme avec un lecteur de disquettes, mais toutes les opérations sont beaucoup plus rapides. Testez-le par vous-même en copiant les programme `basic.prg` et `basic.rsc` sur le disque dur et en lançant le programme depuis ce disque.

Pour que le disque dur puisse fonctionner, il faut absolument que la disquette boot se trouve dans le lecteur A au moment de la mise en marche du ST. Sur les disquettes boot plus récentes se trouve un programme qui permet de booter depuis le disque dur. Ce programme porte le nom 'Hinstall.prg'. Après l'avoir lancé, vous pouvez dans le menu 'File', avec la fonction 'Install', rendre la partition C 'bootable'. Copiez alors le fichier 'desktop.inf' sur le disque C. Pensez que les accessoires sont lancés à partir de maintenant depuis le disque C. Copiez donc les accessoires nécessaires également sur le disque C.

Remarque importante : le disque dur doit toujours être le premier appareil à être allumé. Le disque dur s'initialise lui-même, ce qui lui prend de 10 à 15 secondes. Pendant ce laps de temps il ne doit être perturbé par aucun signal provenant du ST, sous peine de ne pas fonctionner correctement. Vous le reconnaîtrez au fait que les icônes correspondant manquent, ou bien à ce que le voyant rouge ne s'éteigne pas.

#### 5.2.1.6. Les prises MIDI

Ces prises sont prévues pour y connecter un synthétiseur ou tout autre instrument conforme au standard musical/informatique MIDI. Avec un programme adapté il est alors possible de commander le synthétiseur depuis le ST, les possibilités sont gigantesques.



#### 5.2.1.7. L'interface RS 232

Contrairement à la prise MIDI qui est plutôt réservée aux musiciens, l'interface RS 232 est d'un usage beaucoup plus universel.

Elle permet par exemple de faire communiquer le ST avec un autre ordinateur, à condition bien sûr que celui-ci soit également équipé d'une interface RS 232.

Par ailleurs c'est sur cette prise que l'on connectera un modem (MODulateur DEModulateur), appareil qui transforme les caractères qu'on lui envoie en signaux sonores qui sont ensuite envoyés sur la ligne téléphonique. De l'autre côté un autre modem effectue l'opération inverse et restitue donc les informations à l'ordinateur connecté. On peut ainsi transmettre des informations d'un ordinateur à un autre sur de très grandes distances, via la ligne téléphonique.

En outre il existe des imprimantes utilisant la transmission par RS 232 à la place du connecteur Centronics, on les connectera également ici.

Pour les sociétés il peut être intéressant de noter que le ST peut se comporter comme un terminal. Dans ce cas il sert uniquement à transmettre et à recevoir des informations qui sont traitées par un ordinateur central, qui peut être partagé par plusieurs utilisateurs, chacun ayant un terminal.

Qu'est-ce qui différencie la prise RS 232 de la prise Centronics ? la prise Centronics transmet les données sous forme parallèle, c'est à dire par paquets de 8 bits (ou octet par octet). Dans la prise RS 232 au contraire chaque octet est transmis bit par bit, sur une seule ligne. On parle alors d'une transmission série. Celle-ci est logiquement plus lente que la transmission parallèle, mais nécessite un seul fil (contre 8).

#### 5.2.1.8. La prise moniteur

La prise moniteur de votre ST est capable de reconnaître si l'appareil utilisé est un moniteur monochrome ou couleur. Dans le premier cas l'utilisateur dispose uniquement de la très haute résolution de 640\*400 points, en deux couleurs, dans le cas d'un moniteur couleur on a le choix entre la basse résolution de 320\*200 points en 16 couleurs ou la moyenne résolution de 640\*200 points en 4 couleurs.

Par ailleurs il est possible, à l'aide d'un cable adequat, de brancher le ST sur un téléviseur équipé de la prise PERITEL. On dispose alors des mêmes résolutions qu'avec un moniteur couleur.

#### 5.2.1.9. La prise pour l'alimentation

Dernière dans l'énumération des prises; elle n'en est pas moins importante, puisque c'est par ici que le ST reçoit son énergie. Veillez toujours à ce que la fiche soit correctement enfoncée, une coupure d'alimentation due à un mauvais contact peut avoir des conséquences très facheuses.

Peut-être que la lecture de ce chapitre ne vous a-t-elle pas seulement familiarisée avec l'ensemble des nombreux connecteurs dont est équipé le ST, mais cela vous a peut-être aussi donné des idées sur des applications futures de cet ordinateur.

#### 5.2.2. Aspect logiciel

Après une excursion dans le monde matériel (hardware en anglais) gravitant autour du ST, nous allons nous intéresser à l'aspect logiciel de ce qui est et sera proposé pour l'ATARI ST.

Un ordinateur ayant remporté un tel succès a motivé la plupart des éditeurs de logiciels à écrire des programmes pour le ST. Ainsi il existe une multitude de traitements de textes, de questionnaires de fichiers, de tableurs, de logiciels graphiques et de jeux.

L'éventail de produits disponibles est trop grand pour que nous puissions ici donner des indications précises. La presse spécialisée vous sera par contre certainement d'une aide précieuse.

Il existe des programmes utilisant la souris et l'environnement GEM, et d'autres qui n'en tiennent pas compte. Dans tous les cas nous vous conseillons de vous faire faire une démonstration du logiciel avant de l'acheter, pour que vous ayez une idée de ce qui vous attend.

Tant au niveau matériel que logiciel, l'avenir appartient à l'ATARI ST. Pour illustrer cela, nous allons en guise de conclusion faire ensemble quelques réflexions d'ordre général concernant l'ATARI ST.

### **5.3. L'IMPORTANCE DE L'ATARI ST**

Lorsque Jack Tramiel, patron d'ATARI, ouvrit le stand ATARI du CES (salon informatique américain) en janvier 1985, il déclara : "Nous ne vendrons pas d'ordinateurs familiaux. Nous ne vendrons pas de mini-ordinateurs. Nous vendrons des ordinateurs personnels."

Mais que signifie "ordinateur personnel" ? Il y a quelque temps je tenais dans la main un ordinateur de poche d'une firme japonaise qui se disait également ordinateur personnel. Jusqu'à présent cela a toujours été comme ça : comme personne n'a pu se mettre d'accord sur les critères permettant de définir ce que serait un ordinateur personnel, ce terme resta dénué de signification concrète. Jack Tramiel a présenté ses ordinateurs avec le slogan "Power without the price", ce qui signifie "la puissance sans le prix". Cette maxime a au moins le mérite d'être claire et nette.

Qu'y a-t-il de nouveau à la série des ST d'ATARI ? Tout d'abord ils sont livrés d'origine avec une grande quantité de mémoire, 512 Ko et plus, selon le modèle. Par ailleurs ils brillent par l'utilisation du microprocesseur 16 bits 68000 de chez Motorola. Quel avantage pour l'utilisateur ? au lieu de 1000000 opérations par secondes, le ST peut en effectuer 10000000 !

A quoi sert une telle puissance de calcul ? pensez simplement à la taille d'un texte que peut contenir la mémoire du ST. Une opération de recherche d'un mot dans ce texte, qui n'est rien d'autre qu'une suite d'opérations mathématiques au niveau machine, bénéficie pleinement de cet accroissement de puissance. Et ce n'est qu'un exemple parmi tant d'autres.

Mais ce n'est pas tout. La gamme ST englobe une grande quantité de périphériques pouvant lui être adjoints, qui jouissent tous d'un rapport qualité/prix qui fait date dans l'histoire de l'informatique.

Par ailleurs l'ATARI ST est équipé d'origine de connecteurs lui permettant d'accéder au monde extérieur, depuis toutes sortes d'imprimantes jusqu'aux synthétiseurs professionnels. Et puis n'oublions pas ses possibilités graphiques hors du commun : 512 couleurs et une résolution pouvant atteindre 640\*400 points.

Un dernier point qui a contribué au succès du ST : jusqu'à présent, lorsqu'il fallait l'acquisition d'un nouvel ordinateur, le débutant devait commencer par suivre un chemin de croix à travers un dédale d'instructions, uniquement pour charger un bête programme. Jack Tramiel au contraire a placé dans la main de l'utilisateur l'outil qui lui permet de dialoguer en toute simplicité avec la machine.

Vous allez peut-être penser que pour arriver à un tel résultat il a fallu des décennies de développement. Oui et non ; bien sûr le ST a bénéficié d'une longue expérience en matière de développement de technologie de pointe. Mais ce n'est qu'au milieu de l'année 1984 que Jack Tramiel découvrit qu'il était temps de proposer cette technologie de pointe à un prix qui la rend accessible au grand public. C'est à cette date que commencèrent les travaux de développement du ST... sur une table du cuisine. Début 1985 il fut présenté au public au CES de Las Vegas, et depuis il est produit à des centaines de milliers d'exemplaires.

La constante évolution du produit pose aussi des problèmes à ceux qui comme nous écrivent des livres à son sujet. Pendant la phase d'écriture du livre, il nous est parfois apparu des points sur

lesquels nous n'étions pas absolument certains quant à la politique que poursuivrait ATARI. Pour cette raison il peut arriver qu'à la lecture de cet ouvrage vous découvriez des aspects qui auront déjà été modifiés depuis son écriture. Après tout nous ne nous situons qu'au début de l'ère ST et ATARI nous a habitué à des nouveautés surprises, et c'est cela qui caractérise un produit vivant et une entreprise dynamique.

Le meilleur moyen pour rester au courant des nouveautés proposées pour l'ATARI ST est de consulter la presse spécialisée. Il y a en France quelques magazines dédiés à l'ATARI ST qui vous tiendront au courant de toutes les évolutions du produit ST.

Dans l'annexe qui suit vous trouverez quantité d'informations pratiques sur le ST, depuis son jeu de caractères jusqu'à un mini dictionnaire d'informatique, en passant par des programmes de conversions entre les différents systèmes numériques.

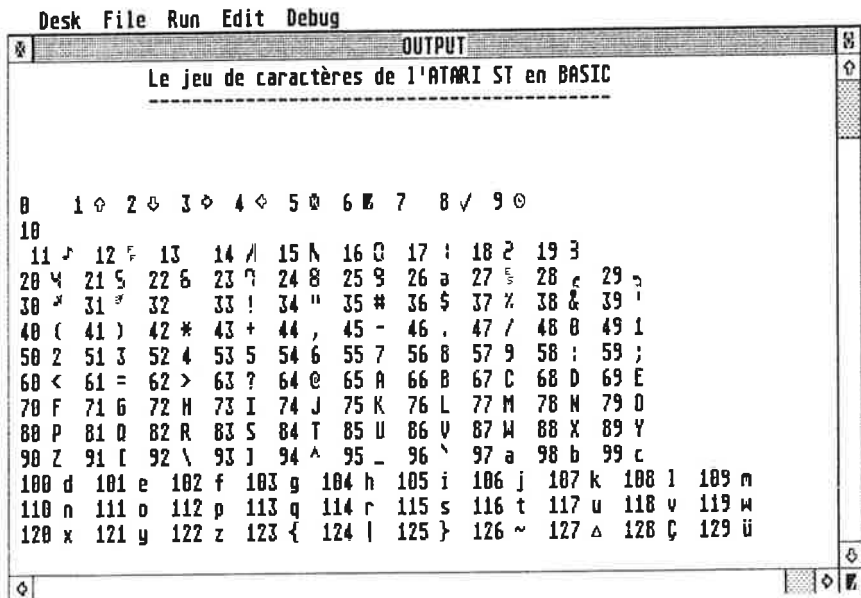


## ANNEXE

### A. LE JEU DE CARACTERES DE L'ATARI ST

Comme vous le voyez, votre ST a un jeu de caractères riche que vous pouvez employer directement au clavier.

## Le jeu de caractères de l'ordinateur ATARI ST en BASIC ST



Le jeu de caractères de l'ordinateur ATARI ST en BASIC ST

Desk File Run Edit Debug									
OUTPUT									
Le jeu de caractères de l'ATARI ST en BASIC									
-----									
130 é	131 â	132 ä	133 à	134 â	135 ç	136 ê	137 ë	138 è	139 ÿ
140 î	141 ï	142 ñ	143 ñ	144 É	145 æ	146 Æ	147 ô	148 ö	149 ð
150 û	151 ù	152 ü	153 ü	154 Ü	155 €	156 £	157 ¥	158 ø	159 f
160 á	161 í	162 ó	163 ú	164 ñ	165 ñ	166 ã	167 õ	168 ì	169 ¨
170 ¨	171 ½	172 ¼	173 ¼	174 «	175 »	176 ã	177 õ	178 ø	179 þ
180 æ	181 œ	182 ã	183 ñ	184 õ	185 ¨	186 ´	187 ¨	188 ¨	189 ø
190 ø	191 ¨	192 ij	193 ij	194 x	195 1	196 1	197 1	198 1	199 1
200 1	201 1	202 1	203 1	204 1	205 1	206 1	207 1	208 1	209 1
210 1	211 1	212 1	213 1	214 1	215 1	216 1	217 1	218 1	219 1
220 1	221 1	222 1	223 1	224 1	225 1	226 1	227 1	228 1	229 1
230 1	231 1	232 1	233 1	234 1	235 1	236 1	237 1	238 1	239 1
240 1	241 1	242 1	243 1	244 1	245 1	246 1	247 1	248 1	249 1
250 1	251 1	252 1	253 1	254 1	255 1				

Comment peut-on donc appeler les nombreux caractères spéciaux ? Vous pouvez d'abord faire sortir les différents caractères sur l'écran avec `PRINT CHR$(...)`. Il existe cependant une autre possibilité qui consiste à produire une grande partie de ces caractères spéciaux directement au clavier, grâce aux combinaisons de touches suivantes :

<u>Code du caractère :</u>	<u>Combinaison de touches correspondante :</u>
----------------------------	--

1 :	Ctrl-A
2 :	Ctrl-B
4 :	Ctrl-D
5 :	Ctrl-E
6 :	Ctrl-F
9 :	Touche TAB et Ctrl-I
11 :	Ctrl+ et Ctrl-K



---

12 :	Ctrl-L et Ctrl-,
14 :	Ctrl-N et Ctrl-.
15 :	Ctrl-O et Ctrl-/
16 :	Ctrl-P et Ctrl-0
17 :	Ctrl-Q et Ctrl-l
18 :	Ctrl-R
19 :	Ctrl-S et Ctrl-3
20 :	Ctrl-T et Ctrl-4
21 :	Ctrl-U et Ctrl-5
22 :	Ctrl-V
23 :	Ctrl-W et Ctrl-7
24 :	Ctrl-X et Ctrl-8
25 :	Ctrl-Y et Ctrl-9
27 :	Touche Esc
28 :	Ctrl- <
29 :	Ctrl- =
30 :	Ctrl- 6
31 :	Ctrl- -

Si vous voulez par exemple sortir sur l'écran le logo ATARI, vous avez deux possibilités :

1. PRINT CHR\$(14);CHR\$(15)
2. Ctrl-N puis appuyer sur Ctrl-0

## **B. PROGRAMMES DE CONVERSION NUMERIQUE**

### **1. Conversion décimal - binaire**

```

10 INPUT"Nombre décimal";A
20 FOR N=15 TO 0 STEP-1
30 IF A=INT(2^N) THEN A$=A$+"1":A=A-INT(2^N):Z=1
40 IF A>INT(2^N) THEN A$=A$+"1":A=A-INT(2^N):Z=1
50 IF Z=0 THEN A$=A$+"0"
60 Z=0:NEXT N
70 PRINT A$

```

*Exercice :* Conversion de 255 décimal en son équivalent binaire

RUN

Nombre décimal ? 255

Résultat : 0000000011111111

### **2. Conversion binaire - décimal**

```

10 INPUT "Nombre binaire";A$
20 FOR N=1 TO LEN(A$)
30 IF MID$(A$,N,1)="1" THEN Z=1
40 IF Z=1 THEN Z=0:A=A+2^(LEN(A$)-N)
50 NEXT N
60 PRINT A

```

*Exercice :* Conversion de 11111111 binaire en son équivalent décimal

RUN

Nombre binaire ? 11111111

Résultat : 255

### 3. Conversion décimal - hexadécimal

*Exercice :* Conversion de 255 décimal en son équivalent hexadécimal

PRINT HEX\$(255)

Résultat : FF

### 4. Conversion hexadécimal - décimal

*Exercice :* Conversion de FF hexadécimal en son équivalent décimal

PRINT &HFF

Résultat : 255

### 5. Conversion décimal - octal

*Exercice :* Conversion de 255 décimal en son équivalent octal

PRINT OCT\$(255)

Résultat : 377

### 6. Conversion octal - décimal

*Exercice :* Conversion de 377 octal en son équivalent décimal

PRINT &o377

Résultat : 255

### 7. Conversion hexadécimal - binaire

```

10 INPUT "Nombre hexadécimal";A$
20 A$="&H"+A$
30 A=VAL(A$):A$=""
40 FOR N=15 TO 0 STEP-1
50 IF A=INT(2^N) THEN A$=A$+"1":A=A-INT(2^N):Z=1
60 IF A>INT(2^N) THEN A$=A$+"1":A=A-INT(2^N):Z=1
70 IF Z=0 THEN A$=A$+"0"
80 Z=0:NEXT N
90 PRINT A$

```

*Exercice :* Conversion de FF hexadécimal en son équivalent binaire

RUN

Nombre hexadécimal ? FF

Résultat : 0000000011111111

### 8. Conversion binaire - hexadécimal

```

10 INPUT "Nombre binaire";A$
20 FOR N=1 TO LEN(A$)
30 IF MID$(A$,N,1)="1" THEN Z=1
40 IF Z=1 THEN Z=0:A=A+2^(LEN(A$)-N)
50 NEXT N
60 A%=A
70 PRINT HEX$(A%)

```

*Exercice :* Conversion de 11111111 binaire en son équivalent hexadécimal

RUN

Nombre binaire ? 11111111

Résultat : 255

---

## C. PETIT LEXIQUE DE L'INFORMATIQUE

### 68000 :

Microprocesseur ou CPU ou unité centrale intégré sur l'ordinateur ST

### BASIC :

Langage de programmation relativement facile à apprendre parce que les erreurs sont immédiatement identifiées et parce que les instructions sont désignées par des mots qu'on peut facilement retenir.

### Baud :

Unité de mesure de la vitesse, en bits par seconde, de transmission des données de l'ordinateur à la périphérie.

### Bit :

Plus petite unité d'information pour l'ordinateur. C'est sur cette unité que repose tout le traitement des données (8 bits = 1 octet = 1 lettre ou 1 caractère).

### Bouton Fire :

Bouton du joystick en principe prévu pour la commande d'un jeu.

### Bug :

Erreur dans le programme (signifie en fait 'punaise' en souvenir des punaises qui provoquaient autrefois des pannes sur les ordinateurs mécaniques).

C :

Langage de programmation rapide qui n'est pas très facile à apprendre car l'élimination des erreurs ne peut être entreprise qu'après la conversion en langage machine, opération qu'on appelle 'compiler'.

Cartouche :

Boîtier plastique dans lequel figure une platine qui peut par exemple contenir un programme.

Centronics :

Société qui a par exemple développé la connexion imprimante dont est doté votre ST.

Chaîne de caractères :

Tiroir ou variable dans lequel des textes peuvent être stockés.

Chip :

Composant électronique fabriqué par photographie qui contient une grande quantité de transistors.

Code de commande :

En appuyant sur des touches, on peut, par exemple, vider l'écran.

Compatibilité :

Possibilité d'échanger matériel (périphériques) ou logiciel (programmes) entre différents ordinateurs.

Connexion parallèle :

Transmission simultanée, en parallèle, de 8 bits (Contraire : sériele = 1 bit à la fois).

Coupleur acoustique :

Appareil pour la transmission des données par téléphone.

CP/M :

Nom d'un système d'exploitation utilisé essentiellement pour les programmes commerciaux.

CPU :

(Central Processing Unit) Coeur de l'ordinateur qui commande tout le traitement des données.

Crayon optique :

Appareil de dessin qui est doté d'une photodiode et permet ainsi de dessiner directement sur l'écran.

Curseur :

Marque de l'emplacement actuel d'écriture sur l'écran.

Débugger :

Eliminer les erreurs d'un programme. (à l'origine : enlever les punaises gênant le bon fonctionnement de l'ordinateur).

Disque dur :

Appareil de sauvegarde et de chargement très rapides de grandes masses de données. (En anglais : Hard disk = disque dur par opposition à Floppy disk = disquette souple).

Disquette :

Disque plastique enduit de particules magnétiques pour la sauvegarde de données. Tailles: 3, 3,5 ou 5,25 pouces.

Editer :

Corriger des erreurs directement sur l'écran.

EPROM :

Composant de mémoire sur lequel des données peuvent être fixées. Il reste cependant possible de supprimer ultérieurement ces données pour en stocker d'autres.

Fenêtres :

Voyez sous Window.

Floppy disk :

(Disquette souple) Nom utilisé pour désigner la disquette ou le lecteur de disquette, par opposition à Hard disk = disque dur.

GEM :

Graphics Environment Manager. Système d'exploitation graphique qui permet de travailler avec jusqu'à 4 fenêtres simultanément.

Graphisme :

Représentation de dessins.

Hardware :

Tous les éléments mécaniques et électroniques de l'ordinateur, par opposition au software ou logiciel.



Icône :

Symbole graphique qui représente une action (par exemple programme sous la forme d'un bloc, lecteur de disquette sous la forme d'une boîte de fiches).

Imprimante :

Appareil pour écrire des données sur le papier.

Imprimante matricielle :

Egalement appelée imprimante à aiguilles parce qu'elle écrit sur le papier des lettres, dessins et graphiques qui peuvent être définis sous la forme d'une matrice.

Interface :

Point de liaison entre l'ordinateur et les appareils connectés (périphérie).

Interface RS 232 :

Liaison avec d'autres ordinateurs; la transmission des données à travers cette interface se fait de façon sériele.

Jeu de caractères :

Jeu de lettres et autres caractères que votre ordinateur maîtrise dès le départ.

Joystick :

Instrument de commande avec manche à balai et bouton de feu ; surtout utilisé pour les jeux.

Kilo-bits :

1024 bits = 128 octets; unité de mesure pour les composants de mémoire.

Kilo-octets :

1024 octets ; unité de mesure de la capacité mémoire d'un ordinateur.

Langage de programmation :

Ensemble d'instructions qui permettent de commander l'ordinateur. Il y a le langage machine qui est très difficile à apprendre parce que c'est directement le langage de l'ordinateur et il y a des langages évolués comme le BASIC ou le LOGO.

Langage machine :

Langage de programmation qui est le véritable langage de travail de votre ordinateur après que les instructions BASIC aient par exemple été traduites dans ce langage.

Lecteur de disquette :

Appareil permettant d'utiliser des disquettes pour y stocker ou y lire des données.

Lightpen :

Voir sous crayon optique.

Logiciel :

Tout ce qui constitue la 'nourriture' des appareils électroniques. Par exemple : langages de programmation, systèmes d'exploitation et programmes tout prêts.

LOGO :

Langage de programmation relativement facile à apprendre parce que les erreurs sont immédiatement identifiées et parce que les instructions sont désignées par des mots qu'on peut facilement retenir. Axé sur le graphisme, "tortue".

Marque d'écriture :

Curseur actuel sur l'écran.

Menu déroulant :

Chaque terme de la ligne supérieure de l'écran cache un menu que vous pouvez dérouler vers le bas (déroulant) après l'avoir touché avec la flèche de la souris.

Message d'erreur :

Message sorti sur l'écran qui indique qu'une erreur s'est produite et qui en donne une analyse sommaire.

Méga :

Représente environ 1 million.

Mémoire :

Composants électroniques qui peuvent retenir des données :  
RAM, ROM, EPROM, PROM.

Microprocesseur :

Voir sous CPU.

Modulateur :

Ce composant convertit l'image produite par l'ordinateur en un signal pouvant être représenté sur un téléviseur.

Moniteur :

Appareil spécial de visualisation des données. Donne une image totalement stable parce que renouvelée avec une plus grande fréquence.

MSX :

Abréviation de Microsoft Super Extended: Nom donné au BASIC et au système d'exploitation des ordinateurs MSX qui sont des ordinateurs compatibles entre eux.

Octet :

Correspond à 8 bits. 1 octet permet par exemple de coder 1 lettre ou 1 caractère. On calcule la mémoire d'un ordinateur en unités de 1024 octets = kilos-octets.

Paddle :

Instrument de commande qui est doté d'un bouton qu'on peut tourner ainsi, en général, que d'un bouton de feu.

Périphérie :

Tous les appareils pouvant être reliés à l'ordinateur.

Port d'extension :

Prise de connexion de votre ST qui sert pour les extensions.

PROM :

Composant de mémoire sur lequel des données peuvent être fixées. Il n'est pas possible de les supprimer ensuite.

RAM :

Composant de mémoire sur lequel des données peuvent être fixées. La RAM se vide lorsqu'elle n'est plus alimentée en courant électrique.

Relais :

Commutateur électromécanique. Remplacé par la suite par les transistors et enfin par les chips.

ROM :

Mémoire pour lecture uniquement qui ne perd pas les données qu'elle contient si on coupe le courant. (On distingue EPROM, PROM et RAM).

RVB :

Abréviation pour Rouge Vert Bleu, les couleurs de base utilisées par les moniteurs.

Son :

Sons et bruits produits par un chip spécial de l'ordinateur.

Souris :

Instrument de commande composé d'un boîtier contenant une boule. Lorsque vous déplacez ce boîtier sur la table, le curseur se déplace sur l'écran.

Synthétiseur :

Composant électronique qui peut restituer des notes et des bruits produits artificiellement.

Système binaire :

Système numérique utilisé par l'ordinateur. Ce système ne connaît que les chiffres 0 et 1.

Système d'exploitation:

Programme qui règle la commande des rapports entre l'ordinateur et la périphérie.

Système décimal :

Système numérique que nous utilisons, avec 10 chiffres de 0 à 9.

Système hexadécimal :

Système numérique qui comporte 16 chiffres, de 0 à 9 puis de A à F.

Système octal :

Système numérique qui comporte huit chiffres, de 0 à 7.

Tablette graphique :

Appareil qui permet de réaliser du graphisme sur une tablette de la taille d'une carte postale. Le graphisme est dessiné avec une poignée plastique.

TOS :

Tramiel Operating System. Système d'exploitation de votre ST.

Touches de fonction :

Touches qui peuvent se voir affecter différentes fonctions par les programmes ou auxquelles une instruction a déjà été affectée. On peut entrer ainsi une instruction plus rapidement que si on devait la taper entièrement.

Transistor :

Élément de commutation électronique (autrefois relais) qui fut ensuite remplacé par le chip qui contient plusieurs milliers de transistors sur la surface d'une tête d'épingle.

Transmission série de données :

Transmission des données effectuée bit par bit (Contraire : parallèle = 8 bits à la fois).

Unité centrale :

Voir sous CPU.

Variable :

Partie de la mémoire ( tiroir ) contenant des données déterminées.

Window :

Un des différents écrans pouvant être représentés simultanément et pouvant éventuellement se chevaucher en partie.

## **D. LES MESSAGES D'ERREUR DU BASIC ST**

Voici une liste des messages d'erreur les plus importants avec leur traduction.

1 : Undefined error = une erreur indéfinissable s'est produite. Le BASIC ST dispose d'un grand nombre de messages d'erreur mais il peut de temps en temps se produire encore une erreur que l'ordinateur ne soit pas capable d'analyser.

2 : Something is wrong = quelque chose est incorrect. Ce message d'erreur est en général sorti lorsque vous avez mal écrit un mot du vocabulaire BASIC, par exemple PINT au lieu de PRINT.

3 : RETURN statement needs matching GOSUB = à chaque instruction RETURN doit correspondre une instruction GOSUB. Vous ne pouvez quitter un sous-programme avec RETURN pour retourner au programme principal si vous n'avez pas quitté auparavant le programme principal avec GOSUB.

6 : Number too large = le nombre entré est trop grand. Le BASIC ST ne supporte 'que' les nombres entre 1E-19 et 1E+18.

7 : Not enough memory = pas assez de place mémoire libre. Que vous possédiez un ST avec 512 K ou même 1 méga de RAM, il peut parfois arriver que la place mémoire ne suffise pas pour un programme que vous aurez entré. Il y a peu de chance toutefois que cela vienne du programme proprement dit. Il est plus probable que cela se produise parce que vous aurez entré une masse de données trop importante.

11 : You cannot divide by zero = vous ne pouvez pas diviser un nombre par zéro. Il s'agit d'une vieille règle arithmétique bien connue qui ne nécessite pas de commentaire particulier.



13 : Types of value do not match = les deux types de valeur employés dans une affectation de valeur à une variable ne correspondent pas. Vous ne pouvez pas affecter à une variable numérique une chaîne de caractères. Il faut utiliser pour cela le type des variables de texte comme A\$ par exemple.

15 : Strings cannot be over 255 characters long = les chaînes ne peuvent comporter plus de 255 caractères. Si vous voulez stocker un texte plus long dans des variables de texte, vous devez utiliser plusieurs variables de texte.

17 : CONT works only in BREAK mode = l'instruction CONT(inuer) (pour faire continuer l'exécution du programme) n'agit que si le programme a été interrompu avec BREAK (par exemple en appuyant simultanément sur les touches 'Ctrl' et 'G' ou si l'instruction END a été rencontrée dans le programme.

23 : Program line too long = cette ligne de programme est trop longue. Une ligne de programme en BASIC ST ne doit pas compter plus de 255 caractères.

30 : Window number invalid = ce numéro de fenêtre n'existe pas. Vous ne pouvez appeler avec les instructions de fenêtre que les numéros 0 à 3 (0=EDIT, 1=LIST, 2=OUTPUT, 3=COMMAND).

53 : File not found on disk drive specified = le programme voulu ne se trouve pas sur la disquette se trouvant pour le moment dans le lecteur de disquette actuel. Lorsque vous voulez charger un programme à partir d'une disquette, avec LOAD"Nom", vous devriez examiner auparavant le catalogue de la disquette.

58 : File exists = ce nom de programme figure déjà sur la disquette actuelle. Si vous voulez à nouveau stocker un programme sous le même nom, cela ne pose pas de problème, l'ancien programme sera remplacé par le nouveau. Vous n'obtiendrez donc ce message d'erreur que lorsque vous tenterez de changer le nom d'un programme pour lui donner celui d'un programme déjà existant. Dans ce cas, il convient donc que vous supprimiez d'abord l'ancien programme.

61 : Disk full = la disquette est pleine. Vous devez supprimer quelques programmes sur cette disquette ou bien prendre une autre disquette.

64 : Invalid filename = le nom de programme choisi n'est pas autorisé. Un nom de programme peut se comporter de jusqu'à 8 caractères et d'une marque de fichier de 3 caractères.

99 : — Break — = interruption du programme. Ce signal est sorti sur l'écran lorsque vous interrompez le programme, par exemple avec Ctrl - G. Vous pouvez ensuite reprendre l'exécution du programme avec CONT.

103 : Invalid line number = le numéro de ligne entré n'est pas autorisé. Les numéros de ligne ne doivent pas être inférieurs à 0 ni supérieurs à 65529.

106 : Line number does not exist = ce numéro de ligne n'existe pas. Vous ne pouvez pas sauter, avec GOSUB ou GOTO par exemple, à un sous-programme ou à un numéro de ligne qui n'existent pas.

107 : Number too large for an integer = la valeur numérique entrée est trop grande pour un nombre entier. Les nombres entiers doivent être compris entre -32767 et 32767.

108 : Input data is not valid, restart input from first item = vous ne devez pas entrer une chaîne de caractères lorsque l'instruction INPUT attend une valeur numérique. Vous devez donc recommencer votre entrée du début.

109 : STOP = interruption définitive du programme. Le programme ne peut plus être relancé qu'avec RUN.

204 : FOR statement needs a NEXT or WHILE needs a WEND = à une instruction FOR doit correspondre une instruction NEXT et à une instruction WHILE doit correspondre une instruction WEND, sinon la boucle de programmation ne peut être exécutée correctement.

205 : NEXT statement needs a FOR or WEND needs a WHILE = à une instruction NEXT doit correspondre une instruction FOR et à une instruction WEND doit correspondre une instruction WHILE, sinon la boucle de programmation ne peut être exécutée correctement.

221 : System error ..., please restart = erreur du système, veuillez relancer le programme.

223 : Too many FOR loops = trop de boucles FOR ... NEXT imbriquées. Il y a une limite à l'imbrication de boucles FOR ... NEXT.

## E. INDEX

68000.....5.1.1.

### A

Adaptation de l'imprimante.....2.2.3.1.  
Affichage Info.....2.2.3.3.  
Afficher.....2.2.3.3.  
Aides au débogage sous LOGO.....4.2.  
Alimentation électrique.....1., 1.5.  
Alternate et Help.....2.7.3.2.  
Alternate et Shift.....2.7.1.1., 2.7.3.2.  
Alternate.....2.7.1.1.  
ALU.....5.1.5.  
Annoncer application.....2.2.3.4.  
Annoncer lecteur de disquette.....2.2.3.4.  
Arithmetic Logic Unit.....5.1.5.  
ASCII.....5.2.1.3.  
AUTO.....3.7.1.

### B

Back-space.....2.7.1.7.  
BACK.....4.3.4.  
BAS.....2.6.3.  
BASIC.....2.7., 3., 5.1.5., Ann.B, Ann.A  
Binaire-décimal.....Ann.B  
Binaire-hexadécimal.....Ann.B  
Bit.....5.1.4.  
Bloc avec souche.....2.6.2.  
Bloc numérique.....3.5.  
BOX.....4.4.1.  
Bug.....5.1.1.

## C

Caps Lock.....	2.7.1.4.
Caractères supplémentaires.....	2.7.4.2.
Cartouche.....	5.2.1.4.
Catalogue de la disquette.....	2.2.2.
Central Processing Unit.....	5.1.5.
Centronics.....	5.2.1.3.
Champ de contrôle.....	2.7.4.2., 4.4.1.
Chiffres décimaux.....	3.5.2.
Chiffres.....	5.2.1.3.
Chip.....	5.2.1.
CHR\$(.....	Ann.A
CIRCLE (BASIC).....	3.6.1.
CIRCLE (LOGO).....	4.4.2.
Classeur.....	2.2.3.4.
Clavier.....	2.7.1.5.
CLEARW.....	3.4.7.1., 3.4.
Clic de touche.....	2.2.3.1.
Cloche.....	2.2.
CLOSEW.....	3.6.4.
Clr/Home.....	2.7.3.4.
Codage.....	5.2.1.7.
Coin caoutchouc.....	4.2., 4.3.3.
Coin de fermeture.....	3.9.3.
COLOR.....	4.4.3.
Connexion d'imprimante parallèle.....	2.6.3.
Connexion disque dur.....	5.2.1.5.
Connexion imprimante.....	5.2.1.3., 5.2.1.7.
Connexion moniteur.....	1.2., 5.2.1.8.
Connexion secteur.....	5.2.1.3.
CONT.....	3.7.2.
CONTINUE.....	3.7.2., 3.10.
Control.....	2.7.1.2.
Conversion centimètres-pouces.....	3.9.5.
Conversion de systèmes numériques.....	5.1.3.
Conversion Dollars-Francis.....	3.9.5.
Conversion Francis-Dollars.....	3.9.5.
Conversion pouces-centimètres.....	3.4.3.

Copie.....	2.2.3.1.
Copier/suppression de programmes.....	4.3.1.
Copier/supprimer fichier.....	2.4., 3.8.2.
Corbeille à papier.....	5.2.1.1.
Coupleur acoustique.....	2.2.3.1., 5.2.1.7.
CPU.....	5.1.5.
CS.....	4.3.3.
CT.....	4.3.3.

## D

Débuggage en BASIC.....	3.7.
Debugger.....	3.7.3., 4.6., 5.3.
Décimal-binaire.....	Ann.B
Décimal-hexadécimal.....	Ann.B
Décimal-octal.....	Ann.B
Décoder.....	5.2.1.7.
DELETE (BASIC).....	3.7.1., 3.8.2.
Delete (touche).....	2.7.1.8.
Desk.....	2.2.3.1.
Desktop GEM.....	2.1., 2.2.3.1., 2.2.3.4.
Digital Research.....	4.2., 5.2.2.3.
DIM.....	3.5.4.
Dimensionnement.....	3.5.4.
Disque dur.....	2.2.2.3.
Disquette système d'exploitation.....	2.1., 2.7., 3.1., 4.2.
Disquette.....	2.3.
DOC.....	2.6.3.
DOT.....	4.3.7.
Double clic sur la souris.....	2.2.2.
Double précision.....	3.5.2.
DR.....	4.2.

## E

Ecran initial.....	2.2.3.4.
ELLIPSE (BASIC).....	3.3.1.
ELLIPSE (LOGO).....	3.6.2.
ELSE.....	4.4.3.
Emulateur VT 52.....	3.9.2.
Enter.....	3.4.3.
EPROM.....	2.7.4.1., 3.3.1.
Esc.....	Ann.D

## F

Fenêtre COMMAND.....	3.2.
Fenêtre EDIT.....	3.2., 3.4.7.1., 3.7.3.
Fenêtre LIST.....	3.7.1., 3.7.3.
Fenêtre OUTPUT.....	2.2.3.2., 2.6.1.
Fenêtres BASIC.....	3.2., 3.4.7.1.
Fenêtres en BASIC.....	2.5., 3.2., 4.2.
Fenêtres.....	Ann.D
Fermer une fenêtre.....	3.2.
Fermer.....	2.2.2.5.
Fichier.....	2.2.3.2.
FILLATR.....	3.6.4.
Fin du BASIC.....	3.8.2.
Fixation des couleurs.....	2.7.1.6.
Fixer valeurs standard.....	5.1.
FOLLOW.....	4.4.5.
Fonction d'heure.....	5.2.1.3.
Fonction de date.....	2.2.3.1.
FOR...NEXT.....	3.7.2.
Formater.....	3.4.4.
FORWARD.....	2.3.1.
Frappe des touches.....	2.7.
FULLW.....	4.3.4.

## G

GEM.....	3.5.2.
GOSUB.....	2.2.3.1.
GOTO.....	3.4.5.
GRAPHICS DISPLAY.....	3.4.2.

## H

Help & Alternate.....	2.7.3.5., 2.7.3.7.
Help.....	5.2.1., 5.2.2.
Hexadécimal-binaire.....	2.7.3.7.
Hexadécimal-décimal.....	Ann.B
HIDETURTLE.....	5.1.3.
Home/clr.....	4.4.3.

## I

Icônes/symboles graphiques.....	5.2.2.2.
IF...THEN.....	2.6.
IF...THEN...ELSE.....	3.4.3.
Imprimante laser.....	2.3.2., 2.4.6.2.
Imprimante matricielle.....	5.1.5.
Imprimante à aiguilles.....	4.4.7.
Imprimante.....	5.2.1.3.
Imprimantes thermiques.....	3.4.3., 3.10.
Imprimantes à jet d'encre.....	5.2.1.3.
Imprimantes à marguerite.....	4.4.7.
Imprimer l'écran.....	2.2.3.4., 2.7.3.7.
Index comme image ou comme texte.....	3.9.5.
Informations Desktop.....	2.2.3.1.
INPUT.....	2.2.3.1.
Insert.....	3.4.1.3.
Instructions graphiques BASIC.....	3.6.
Interface parallèle.....	5.2.1.3., 5.2.1.7.
Interface RS 232.....	2.2.3.1.
Interface sérieielle.....	5.2.1.3., 5.2.1.7.
Interface.....	2.2.3.2.
Interruption de programme.....	5.2.2.2.



## J

Joystick.....2.7.3.3.

## K

K octets.....5.1.4.

K.....5.2.1.1.

Kilo-bits.....5.1.4.

Kilo-octets.....5.1.4.

## L

Langage de programmation évolué.....2.7.3.4.

Langage de programmation.....2.4.

Langage machine.....5.2.1.1.

Lecteur de disques compacts.....5.2.1.5.

Lecteur de disquette.....1.3., 2.1., 2.3., 2.2.3.4.

Lecteur de vidéo-disque.....5.2.1.5.

LEFT.....5.2.1.3.

Lightpen.....4.3.4.

Lignes de commentaire.....5.1.2.

Lignes de remarques.....5.1.2.

LINEF.....5.2.1.1.

LIST.....3.6.7.

LOG.....3.2., 3.4.7.1.

Logiciel.....4.4.6.

LOGO DIALOGUE.....4., 5.2.2.2.

LOGO.....4.2., 3.

## M

Machine à écrire.....5.2.1.3., 5.2.1.7.

MAKE.....4.5.4.3.

Matériel.....5.2.1.5.

Mémoire disque dur.....5.2.1.5.

Mémoire.....1.4., 5.2.1.1.

Mémoire.....2.2.3.3., 4.4.6.

Menu de déplacement.....3.5.

Menus pull down.....5.1.2.

Message d'erreur.....	1.2., 5.2.1.8.
Messages d'erreur BASIC.....	Ann.D
Messages d'erreur.....	3.4.6.
Microprocesseur.....	5.2.1.6.
MIDI.....	3.4.6.
Modem.....	5.1.5.
Moniteur couleur.....	2.2.3.1.
Moniteur noir et blanc.....	3.5.
Moniteur.....	5.2.1.7.
MOTOROLA.....	1.2., 5.2.1.8.
MOUSE.....	5.1.5.

## N

NEW.....	2.2.3.2.
NEXT.....	3.4.7.2.
NODES.....	3.4.4.
Nombres entiers.....	2.7.2.
Noms de variables.....	3.5.
Nouveau classeur.....	1., 1.5.

## O

Octal-décimal.....	2.2.3.2.
Octet.....	5.1.4.
OPENW.....	5.1.3.
Opérateurs arithmétiques.....	5.1.2.
Options.....	3.6.3.
Ordinateur personnel.....	3.4.3.7.
Ouverture.....	4.4.7.

## P

Paddle.....	3.2.
PAUSE.....	5.2.1.2.
PCIRCLE.....	4.6.
PELLIPSE.....	3.6.8.
PENDOWN.....	3.6.9.
PENUP.....	3.4.3.7.
Péritel.....	3.4.1.2., 3.10.

Pile de papiers (fichier).....	2.4.
Place mémoire.....	5.1.
Point décimal.....	2.7.4.2.
Ports joystick.....	5.2.1.1.
PRINT (BASIC).....	2.7.3.2.
PRINT (LOGO).....	3.4.1.2.
Programme d'agenda téléphonique.....	2.2.3.1.
Programme de chiffres du lotto.....	4.6.
Programme de conversion.....	2.2.3.1.
Programme de vocabulaire.....	2.2.2.1.
Programme du problème de l'échiquier.....	5.2.1.8.
Programmes BASIC.....	3.9.
PROM.....	2.7.1.2., 3.7.2.
Protection contre l'écriture.....	2.7.1.

## Q

QUIT.....	2.2.3.
-----------	--------

## R

RAM.....	3.8.2.
Réglage de la RS 232.....	5.1.2.
Relais.....	2.7.4.2.
REM.....	5.1.1.
RENUM.....	3.4.6.
REPEAT.....	3.7.1.
RETURN (BASIC).....	2.7.1.9.
Return (touche).....	4.3.5.
RIGHT.....	2.2.3.1.
RND.....	4.3.4.
ROM.....	3.9.2.
RUN.....	2.2.3.1., 5.2.1.7.
RVB.....	3.3.1.

## S

Sauvegarde du travail.....	2.2.3.4.
Scrolling.....	1.2., 5.2.1.8.
SETFILL.....	5.2.1.3., 5.2.1.7.
Shift et Alternate.....	2.7.1.3., 2.7.3.2., 4.1.
Shift.....	4.4.5.
SHOWTURTLE.....	2.7.3.2.
SHUFFLE.....	4.4.4.
SORT.....	5.2.2.
Sortir fichier.....	2.6.3.
Souris.....	5.2.1.3.
Suppression.....	3.9.4.
Symboles graphiques/icônes.....	2.6.
Synthétiseur.....	1.5., 2.1.
System disk.....	5.2.1.6.
Système binaire.....	5.1.3.
Système binaire.....	5.1.3.
Système décimal.....	5.1.3.
Système hexadécimal.....	Ann.B
Système octal.....	Ann.B
Systèmes numériques.....	5.1.5.

## T

Tab.....	2.1.
Tablette graphique.....	4.3.2.
Terminal.....	3.9.1.
THEN.....	5.2.2.3.
Tiroirs/variables.....	2.2.3.2., 2.3.
TO.....	5.2.1.3.
TOS.....	4.3.6.
Touches d'édition.....	2.7.3.
Touches de fonction.....	3.6.5.
Touches-flèches avec Shift/Alternate.....	2.7.3.1.
Touches-flèches.....	5.3.
TRACE (BASIC).....	2.1., 2.2.3.1., 2.2.3.4., 2.6.
TRACE (LOGO).....	3.7.3.
Track-ball.....	4.6.

Traitement de texte.....	3.5.3.
Tramiel.....	5.2.1.1.
Transistor.....	5.3.
Transmission de données.....	5.2.1.5., 5.2.1.7.
Transmission parallèle de données.....	5.2.1.3., 5.2.1.7.
Transmission sérielle de données.....	2.2.2.4.
Tri.....	4.4.6.
TROFF.....	5.3.
TRON.....	3.7.1.
TTP.....	3.7.1.
Tueur de punaises (débuggage).....	2.2.2.1.
TURTLEFACTS.....	2.2.3.4.

## U

Undo.....	3.9.5.
Unité centrale.....	2.7.4.
UNTRACE.....	2.7.3.6.

## V

Variable numérique.....	Ann.B
Variables de texte.....	5.2.1.7.
Variables/tiroirs.....	3.7.2.
Virgule.....	5.1.4.
Vocabulaire de base BASIC.....	3.4., 3.10.

## W

WATCH.....	3.7.2., 4.6.
Windows.....	4.6.

## Z

Z80A.....	2.5., 3.2.
Zuse.....	5.1.5.

Les meilleurs ouvrages sur ATARI ST. Plus de 15 titres traitant clairement et complètement de tous les sujets essentiels et recouvrant tous les degrés d'attente. Une offre logicielle exceptionnelle regroupant les grands standards tels que: CALCOMAT 2, SUPERBASE 2 et bien sûr GFA 3.0 et tout son environnement.



#### LE LIVRE DU GEM

Comprendre et utiliser GEM efficacement, et profiter de son immense bibliothèque... Avec le LIVRE DU GEM, toutes les informations fondamentales sont à portée de main: la programmation de GEM en assembleur, C

ou GFA basic; l'accès aux ressources, menus déroulants, description des routines... Un formidable outil de développement pour votre ATARI ST. Réf. ML 139. 179 F. 462 P. ML 239. 279 F avec la disquette. ●



#### BIEN DEBUTER SUR ST

Pour prendre un bon départ et gagner du temps, ce livre aborde simplement l'installation de votre machine, l'utilisation du TOS et de GEM, la souris, l'écran, le clavier, le Basic et le Logo. Créez vos premiers

programmes et maîtrisez les différentes configurations de l'ATARI ST. Réf. ML 156. 129 F. 244 P



#### DISQUETTE ET DISQUE DUR

Tirez le maximum de votre lecteur de disquettes ou disque dur. Vous saurez tout sur la structure d'une disquette, l'emplacement des programmes, l'accès direct aux données par le biais du

langage machine ou Basic. Profitez aussi de nombreux utilitaires (copie, formatages spéciaux, etc...). Réf. ML 172. 179 F. 483 P. Réf. ML 272. 279 F avec la disquette. ●



#### LE LANGAGE MACHINE SUR ST

Voici une introduction aisément compréhensible à la programmation du surpuissant microprocesseur 68000. Programmes de jeux hyper rapides, animations graphiques époustouflantes... toutes ces applications font appel au langage machine. LE LANGAGE MACHINE SUR ST dévoile les clefs de la programmation en assembleur, de la simple manipulation de bits à l'intégration de vos programmes dans d'autres langages évolués. Mais aussi: la structure du microprocesseur, les étapes de la programmation en assembleur, de nombreux programmes exemples disponibles en option sur la disquette. Réf. ML 141. 149 F.



## LE LIVRE DE 1ST WORD PLUS

LE LIVRE DE 1ST WORD vous assure une prise en main efficace du logiciel, jusqu'à une parfaite maîtrise des commandes. De la mise en forme des textes à la préparation de documents pour l'impression, de très nombreux exemples permettent la mise au point de documents élaborés. La disquette propose plusieurs drivers d'imprimantes, une banque d'images ainsi que des fichiers au format SUPERBASE et CALCOMAT pour optimiser le publipostage. Réf. ML 616. 299 F. avec la disquette. ▲

## LE GRAND LIVRE DE L'ATARI ST

Bien connaître son ATARI ST n'empêche pas cependant de rencontrer des problèmes en apparence insolubles. LE GRAND LIVRE DE L'ATARI ST est le livre de référence de cette machine. Un ennui d'imprimante, de boot, des informations à connaître sur le disque RAM... Résolvez tous vos problèmes en un tour de main en apprenant à maîtriser parfaitement votre ST. Pour acheter un logiciel ou ajouter un accessoire, connaître le Hardware ou les systèmes d'exploitation, vous ne trouverez de meilleur conseiller. Réf. ML 530. 199 F. Réf. ML 630. 299 F avec la disquette. ● (à paraître novembre).



## BIEN DEBUTER EN GFA BASIC (versions 2.0 et 3.0 incluses)

Ce livre aborde tous les aspects du langage, de la simple mise en route jusqu'à la gestion des outils de GEM. Vous y trouverez tout ce qui concerne la programmation en GFA:

les variables, les procédures, les fonctions paramétrables, le traitement des données, l'accès aux disquettes... De très nombreuses informations pour parfaire l'apprentissage de ce fabuleux langage ou passer en toute quiétude de la version 2.0 à la version 3.0. Réf. ML 527. 129 F. (à paraître octobre).



## TRUCS ET ASTUCES II

Ce best seller des livres sur ST rassemble dans sa nouvelle version les meilleurs trucs et astuces que vous pourrez tirer de votre machine. Sur la disquette, une foule d'utilitaires indispensables tels que: GEM STARTER, remplacer

les bombes par des messages d'erreurs, des trucs pour les formulaires gérés avec l'AES, la création de fichiers ACC, la programmation du son par les interruptions... Réf. ML 629. 299 F avec la disquette. ▲ (à paraître novembre).





## APPLICATIONS SOUS SUPERBASE

Ce précieux ouvrage contient près de 40 applications "clés en main", disponibles sur la disquette et modifiables à tout moment. Que vous soyez animateur d'un club, agriculteur... ce livre

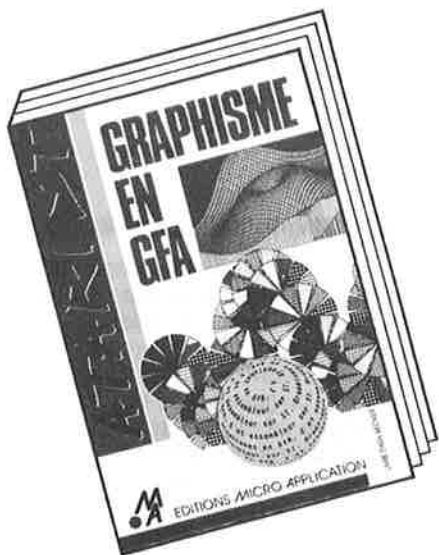
contient toutes les solutions à vos problèmes de gestion. Une excellente méthode pour aborder la construction et l'organisation de votre base de données. Réf. ML 617. 349 F avec la disquette. ▲



## DEVELOPPER EN GFA BASIC 2.0

Ecrit par l'auteur du GFA Basic, cet ouvrage permet de tirer le maximum de l'interpréteur et du compilateur. Tous les sujets indispensables que les programmeurs attendaient sont enfin traités:

hardcopy, emploi des routines AES, chargement des polices avec GEM... et en plus une multitude de nouveaux utilitaires. Réf. ML 295. 299 F. 256 p. avec la disquette. ▲



## LE LIVRE DU GFA BASIC

L'ouvrage indispensable pour exploiter au maximum votre langage favori. Toutes les commandes de l'interpréteur et du compilateur sont expliquées. De puissantes procédures, notamment en graphisme et en animation décuple-

rons votre efficacité dans la programmation en GFA Basic. Réf. ML 185. 199 F. 656 p. Réf. ML 285. 299 F avec la disquette. ●



## TRUCS ET ASTUCES EN GFA BASIC 2.0

Pour exploiter les fantastiques possibilités de ce langage, il est indispensable de connaître les astuces des professionnels du GFA Basic: un éditeur d'icônes pour créer ses symboles, programmation

de GEM, Ressources Construction Set sous GFA Basic... Réf. ML 299. 269 F. 400 p. avec la disquette. ▲



## LA BIBLE ST

460 pages pour exploiter à fond la puissance de votre ATARI ST. Découvrez la structure du Hardware: le BIOS, GEM, et les adresses importantes des routines; une description détaillée des interfaces

(port d'extension, interface souris, V24...). Réf. ML 142. 199 F 462 P.

## GRAPHISME EN GFA, C ET ASSEMBLEUR

La référence absolue des infographistes sur ST. Plus de 800 pages d'informations précieuses, commentées et détaillées. GRAPHISME EN GFA, C ET ASSEMBLEUR est une porte ouverte aux délires les plus fous, en 2 ou 3 dimensions, en animation. Sujets traités: programmation sous GEM (fenêtres, menus); algorithmes de tracé de lignes, cercles, projections; création de polices de caractères... Réf. ML 502. 249 F. 872 p. Réf. ML 602. 349 F avec 2 disquettes. ●



De par sa puissance et sa convivialité, GFA s'est imposé comme "le" langage sur ATARI ST. Aujourd'hui GFA surpasse GFA avec la version 3.0 dont la richesse et l'environnement renforce encore sa suprématie.

### GFA BASIC 3.0

Totalement réécrit, avec des performances considérablement accrues, le GFA Basic 3.0 surpasse tous les Basic conçus pour l'ATARI ST. L'éditeur a été modifié: l'écran principal affiche désormais de nouveaux symboles: heure en temps réel (modifiable dans la barre de menu), compteur de ligne asservi au curseur, témoin de verrouillage des majuscules et du bloc numérique. De plus, un petit sigle ATARI permet par simple clic d'accéder aux accessoires de bureau de GEM sans quitter l'application en cours. Dans ce mode vous définirez 4 formats différents pour les mots clés et variables de l'éditeur: suffixes optionnels pour les variables, première lettre en capitale pour les mots clés et variables... Afin de gagner du temps dans les développements, de nouveaux outils vous faciliteront la tâche: le bloc numérique permet le déplace-



ment du curseur, l'appel des fonctions peut être effectué entre autre par la combinaison ALT/bloc numérique. 200 nouvelles fonctions: fonctions mathématiques, opérations sur les bits, interruptions, nouveaux types de variables... Les nouvelles fonctions liées à la définition des procédures ou des variables font du GFA 3.0 un langage structuré sans en subir les contraintes. Des excès de vitesse: pour certaines instructions comme PSET, FOR... NEXT ou INPUT le gain de temps va de 100 à 400%! En moyenne, le GFA 3.0 non compilé tourne 50% plus vite que son prédécesseur. Réf. ST 027. 750 F.



### PROGRAMMATION EN GFA BASIC 3.0

C'est le complément indispensable pour réaliser des applications poussées capables de tirer parti des fabuleuses possibilités de la version 3.0. Cette bible du programmeur exploite les nouvelles instructions du langage telles que: programmation structurée, nouveaux types de variable, instructions Line-A, routines assembleur, bibliothèque AES... La disquette offre de très nombreux exemples et une dizaine de programmes utilitaires allant de la sortie de textes en assembleur à la création d'icônes. Réf. ML 638. 349 F avec la disquette. ▲

## GFA BASIC 2.0

Le GFA Basic s'est imposé comme le langage standard sur ST. Rapide, puissant et simple, il représente plus de 200 commandes pour programmer la souris, les zones d'alerte, les menus déroulants... Ses caractéristiques: programmation structurée, interface GEM ultra rapide, interpréteur compact, éditeur puissant, calcul sur 11 chiffres... Instructions: REPEAT, UNTIL, DO, LOOP, CIRCLE, FILL, TIMER, XBIOS... Bien sûr, les programmes créés en GFA Basic peuvent être compilés avec GFA COMPILATEUR. Réf. ST 012. 495 F.

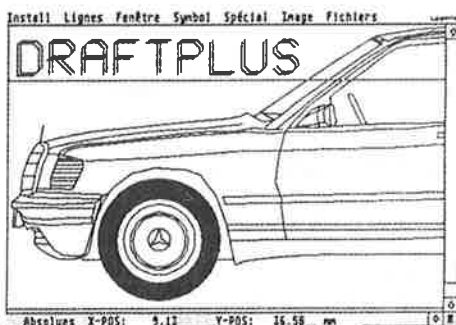
## GFA COMPILATEUR 2.0

Pour accélérer vos programmes en GFA Basic au niveau du C, Turbo Pascal ou Assembleur, une seule solution: le GFA COMPILATEUR. Il décuple la vitesse des applications quasi instantanément, sans passer par le Run Time ou autres bibliothèques de programmes, sans aucun LINKER, en générant du code machine compact et très performant. Simple d'utilisation, il vous fera bénéficier de la formidable rapidité du ST (ne compile que le GFA Basic). Réf. ST 013. 295 F.



## GFA DRAFT PLUS

GFA DRAFT PLUS est un programme professionnel de CAO en deux dimensions permettant de réaliser schémas de connexion, desins techniques, croquis ou plans. Capacités: création de dessins sur 255 plans; lignes variant de 0.3 à 4.5 mm en noir ou en couleur (sur imprimante, écran ou traceur)... GFA DRAFT PLUS possède des guides-lignes (aimantation), une grille de positionnement variable, de nombreuses fonctions: motifs, hachures, agrandissement, déformation, rotation, zoom, symboles liés aux touches de fonctions. Réf. ST 017. 990 F.



## GFA ASSEMBLEUR

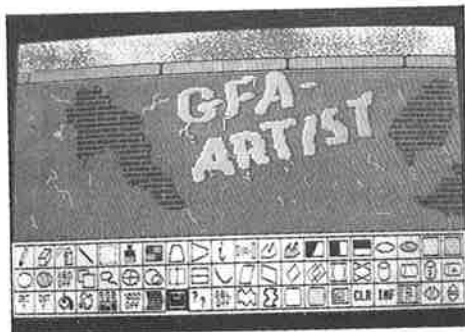
Programmes de jeu, animations graphiques hyper rapides... GFA ASSEMBLEUR a été conçu pour satisfaire tous les développeurs. L'éditeur corrige instantanément les erreurs. Accessible avec la souris, il permet la recherche et le remplacement de zones programmables, de nombreux réglage et de très nombreux raccourcis au clavier. Il offre une fonction très appréciable d'autocorrection pendant la saisie qui évite les mauvaises surprises à l'exécution. L'assemblage est conditionné et récursif et possède une liste impressionnante de caractéristiques comme le compactage des fichiers pour un gain de place appréciable. Avec son debugger le GFA ASSEMBLEUR est un outil unique! Réf. ST 033. 750 F. (à paraître novembre).

# L'ÉNERGIE MICRO



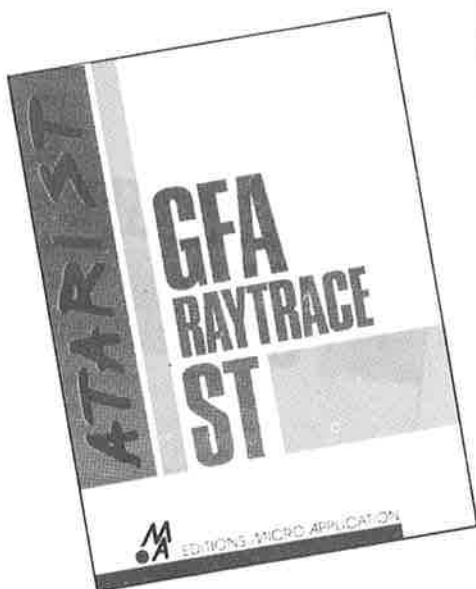
### GFA ARTIST: ANIMEZ VOTRE TALENT.

Les graphismes les plus fous s'animent, les lutins se déforment en suivant les trajectoires indiquées... GFA ARTIST est stupéfiant. Sa puissance permet la projection sur une sphère, un cylindre, une perspective... GFA ARTIST est un véritable studio d'animation: création de séquences avec arrière-plan, animation en temps réel... Ses nombreux utilitaires (éditeur de caractères, logiciel de fusion des films) font de lui une merveille en animation graphique. (configuration minimum: 1 Mo de mémoire, TOS en ROM, écran couleur). Réf. ST 026. 495 F. Version ATARI 520 disponible en novembre.



### GFA RAYTRACE

Quoi de plus fascinant que ces superbes images pleines d'ombres et de perspectives que traversent d'énormes billes polies comme des miroirs... GFA RAYTRACE est le premier logiciel permettant de développer de telles applications sur ST. Puissant et convivial, vous pouvez mettre au point une image animée facilement et instantanément. Vous disposez de nombreux éclairages, fonctions d'animation et de dessins, et vous pourrez charger les images créées sous d'autres logiciels pour les retravailler par la suite. Réf. ST 028. 495 F.



Achevé d'imprimer en décembre 1988  
sur les presses de l'imprimerie Laballery  
58500 Clamecy  
Dépôt légal : décembre 1988  
N° d'impression : 812001



PAULY/SCHEPERS/SCHULZ

# TRUCS ET ASTUCES

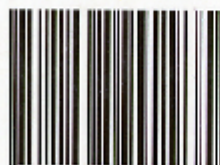
Tout ce que vous ne pensiez pas pouvoir faire avec votre ATARI ST, TRUCS ET ASTUCES va vous le permettre !

Fabriquer ses accessoires, lancer un programme résident, convertir des images, éviter les 'bombes' ou modifier la fréquence du processeur... Voici une véritable mine d'astuces, dont nombreuses sont inédites, présentées sous forme de programmes en code compilé, langage C, Assembleur et GFA Basic.

Si vous n'êtes pas un adepte des discours théoriques, vous pourrez immédiatement exploiter ou modifier tous les programmes du livre et de la disquette, et par la suite approfondir vos connaissances techniques dans les domaines les plus variés : hardware, software, langages, périphériques, entrées/sorties...

## Principaux sujets traités :

- Installation du GDOS et de ses accessoires.
- Dissimuler, reformater, coder des fichiers...
- Accélérer la gestion de vos disquettes : le floppy-speeder.
- Affecter les fonctions au clavier, créer des raccourcis clavier.
- Recherche rapide et paramétrable des fichiers sur disque.
- Conversion et insertion des images graphiques aux formats BitMap, Degas...
- Des routines graphiques hyper-rapides.
- Créer et installer vos propres accessoires de bureau.
- Comment renommer un dossier.
- Reset, extinction automatique d'écran par une simple touche.
- Réaliser des hardcopies d'écran, des scrolling doux...
- Programmer en multi-tâche, détourner les interruptions du 68000...



9 782868 991935

Réf. : ML 651. Prix : 299 F  
246/ISBN : 2 86899 193 9/ISSN : 0980-1928

EDITIONS MICRO APPLICATION

58, RUE DU FAUBOURG-POISSONNIÈRE  
75010 PARIS. TÉL. : (1) 47 70 32 44