SANCO

ORDINATEUR PORTABLE

MODELE 8300

NOTICE D'EMPLOI

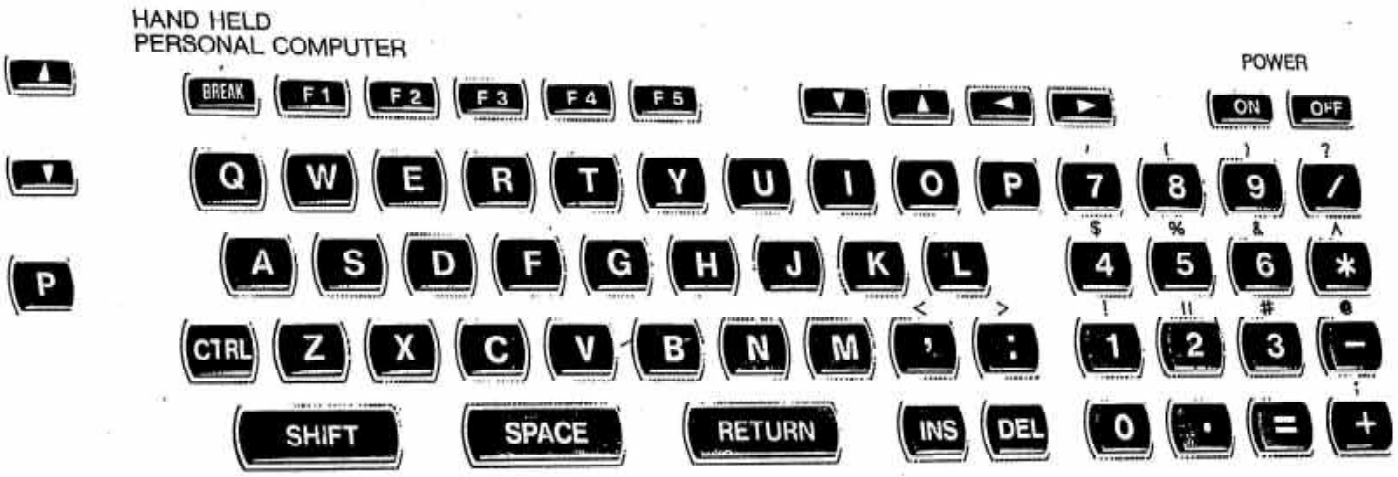
TABLE DES MATIÈRES

CHAPITRE 2 : LES COMMANDES ET LES INSTRUCTIONS

I. SPÉCIFICATIONS

Langage de programme	:	BASIC	
Capacité	:	CPU	CMOS 8 bit
		Système ROM	20 koctets
		Capacité de mémoire adressable (RAM)	6 koctets
		Aire système	1.7 koctets
		Aire utilisateur	4.3 koctets
Fonction de mise au point	:	Déplacement du curseur (◀, ▶) Insertion (INS) Suppression (DEL) Ligne précédente et suivante (▼, ▲)	
Protection de la mémoire	:	Soutien de piles CMOS rechargeable (protège les programmes et les données)	
L'affichage	:	Cristaux liquides Largeur de 48 caractères (24 × 2)	
Touches	:	62 touches, y compris : Alphabétiques Numériques de fonction définissable par l'utilisateur Pré-programmées	
Source d'alimentation	:	6:O V, c.c. 4 piles sèches (type UM-3, AA, ou R6)	
Dimensions	:	Longueur = 199 mm Largeur = 96 mm Epaisseur = 26 mm	
Poids	:	410 g (avec les piles)	
En option	:	Interface imprimante/cassette (TP-83) Longueur = 329 mm Largeur = 117 mm Epaisseur = 51 mm Module d'extension de mémoire (de type à prise, RAM 4 koctets TCR-40, RAM 8 koctets TCR-80)	
Autres	:	Connectable directement avec des imprimantes (10", 15", etc.) du type Centronics parallèle	

II. DISPOSITION DU CLAVIER



III. CARACTÉRISTIQUES DU CLAVIER ET DE L'AFFICHAGE

A. LE CLAVIER

Touches 'ON , OFF' (marche , arrêt)

Ces touches servent à allumer ou éteindre l'appareil. Pour économiser le courant, l'appareil s'éteindra automatiquement au bout de sept minutes si rien est saisi au clavier, à moins qu'un programme ne soit en cours d'exécution.

Touches Alphabétiques

Les touches alphabétiques vous permettent d'introduire des instructions et des données. En outre, elles peuvent être utilisées pour désigner des «zones mémoire de stockage» desquelles et auxquelles vous pourrez sauvegarder ou extraire des données.

Touches Numériques et Touches d'Opérations Arithmétiques

Avec ces touches, vous pourrez introduire des chiffres pour les calculs. Les touches [+] [−] [*] [/] signalent à l'appareil de faire respectivement l'addition, la soustraction, la multiplication et la division.

Touche 'SHIFT'

Cette touche met en action les fonctions inscrites au-dessus de nombreuses touches non-alphabétiques. Par exemple, pour écrire le point-virgule, appuyez d'abord sur la touche [SHIFT] , et puis sur la touche [+]

Touches de Fonctions

Vous pouvez assigner les commandes fréquemment utilisées à ces touches, jusqu'à 10 fonctions au maximum.

Exemple : a) Pour écrire le mot "PRINT" dans F1,
tapez : KEY1,"PRINT" [RETURN]
b) Pour relire ceci, appuyez sur F1.

Pour lire les touches F6 à F10, appuyez sur la touche "SHIFT" et les touches F1 et F5 respectivement. Par exemple, pour lire le contenu de la touche F10, appuyez sur les touches "SHIFT" et F5.

Touche "BREAK"

Cette touche arrête l'exécution du programme.

Touches de déplacement du curseur

[◀] Déplace le curseur d'un caractère vers la gauche

[▶] Déplace le curseur d'un caractère vers la droite

Caractères de Control (CTL)

CTL + D	Supprime le caractère sur la position du curseur
CTL + E	Supprime tous les caractères vers la droite du curseur
CTL + ▶	Affiche les caractères suivant les premiers 48 caractères
CTL + ◀	Affiche les premiers 48 caractères si plus de 48 caractères étaient entrés
CTL + L	Efface l'affichage

Caractères de "SHIFT"

SHIFT + ▶	Déplace la donnée affichée par PRINT vers la gauche
SHIFT + ◀	Déplace la donnée affichée par PRINT vers la droite.
DEL	Supprime le caractère à la gauche du curseur
INS	Déplace vers la droite d'un espace tous les caractères qui se trouvent à droite du curseur et laisse la position du curseur vide
▲	Affiche le contenu du numéro de ligne précédent. (Pour l'imprimante, monte-papier)
▼	Affiche le contenu du numéro de ligne suivant. (Pour l'imprimante, descend-papier)

Touche RETURN

Au fur et à mesure que vous les écrivez,les lettres et les nombres apparaissent sur l'affichage. Cependant l'appareil n'entrera en action qui si vous lui indiquez que vous avez fini d'écrire. Vous pouvez faire cela en appuyant sur la touche RETURN. A ce moment, l'ordinateur examinera les caractères écrits pour vérifier si la forme est correcte. Certaines erreurs, mais pas toutes, causeront le refus de votre entrée.

N'OUBLIEZ PAS : d'appuyer sur la touche RETURN chaque fois que vous désirez entrer une commande ou une donnée.

B. L'AFFICHAGE

Appuyez sur ON. La partie «vitrée» de l'ordinateur s'appelle l'affichage. Il correspond à l'illustration suivante :

```
10:INPUT ''LIST SIZE?'';A
```

Sur l'affichage, vous devriez voir le symbole de guidage , quelques mots ou abréviations.

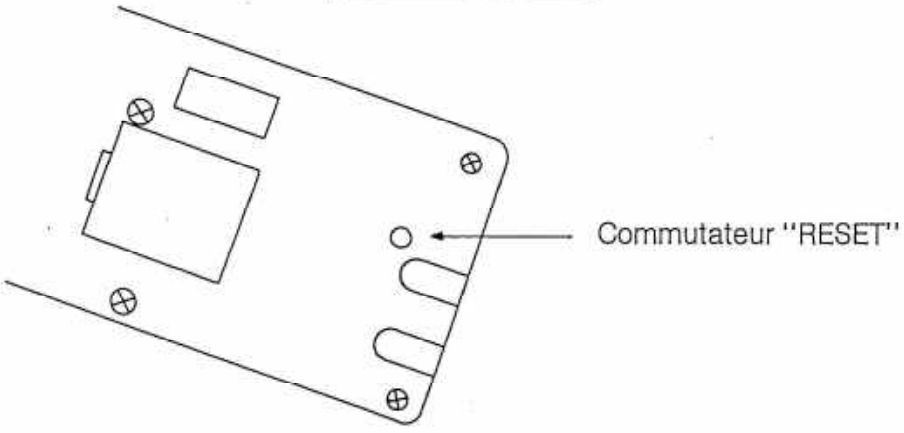
Le Curseur est le symbole de guidage.

Sur le côté extrême gauche de l'affichage, vous trouveriez le symbole ; il est là pour vous inciter à parler à l'appareil. Quand il est affiché, cela veut dire que l'appareil n'a pas de projets pour l'immédiat et qu'il attend vos ordres. **Tapez un caractère de votre choix.** Il remplace le --- à la gauche de l'affichage tandis qu'à la droite du caractère apparaît un trait de soulignement. Ceci s'appelle «curseur». Au fur et à mesure que vous appuyez sur des touches ce curseur avance à travers l'affichage, indiquant où va apparaître le caractère suivant. Tapez votre nom et suivez le mouvement du curseur.

C. Le commutateur "RESET"

Appuyez sur le commutateur "RESET" au dos de l'appareil si l'affichage disparaît ou si les touches deviennent sans effet à cause des fausses opérations ou des chocs externes.

DOS DE L'APPAREIL



IV. INTERFACE IMPRIMANTE/CASSETTE

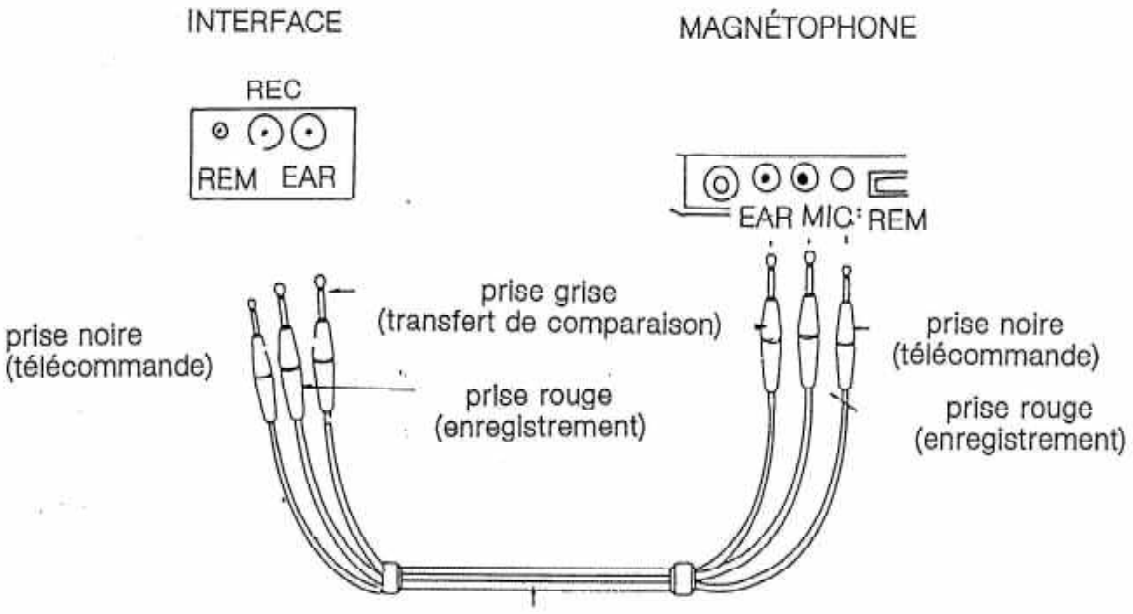
A. L'ALIMENTATION

L'unité d'interface imprimante/cassette fonctionne avec 4 piles NI-CAD rechargeables et intégrés dans l'appareil. Il s'avère donc nécessaire de recharger les piles après le déballage de l'appareil.

La vérification des charges des piles est faite automatiquement et le message «NO BATTERY» s'affiche quand les piles commençent à être épuisées.

B. LE RACCORDEMENT AVEC UN MAGNÉTOPHONE

Premièrement, raccordez l'unité d'interface imprimante/cassette et l'appareil et puis connectez le magnétophone avec l'interface de la façon indiquée par le schéma ci-dessous.



C. LES SPÉCIFICATIONS DU MAGNÉTOPHONE

Ci-dessous vous trouverez une description des conditions minimum pour qu'un magnétophone puisse être branché sur l'unité d'interface.

	Conditions à remplir
1. Type de magnétophone	N'importe quel magnétophone à cassette, micro-cassette, ou à bande ouverte peut être utilisé tant qu'il satisfait aux conditions énoncées ci-dessous.
2. Prise d'entrée	Le magnétophone doit posséder une mini-prise d'entrée "MIC". Ne jamais utiliser la prise "AUX".
a. Impédance d'entrée	La prise d'entrée doit avoir une basse impédance (200 - 1000 ohms).
b. Niveau d'entrée minimum	Inférieur à 3mV ou — 50 dB.
3. Prise de sortie	Une mini-prise "EXT" (haut-parleur externe), "MONITOR", "EAR" ou un équivalent.
a. Impédance de sortie	Doit être inférieure à 8 ohms.
b. Niveau de sortie	Supérieur à 1V (sortie maximum en pratique supérieur à 100 mW).

Au cas où la miniprise fournie n'est pas compatible avec les prises d'entrée ou de sortie de votre magnétophone, vous devrez vous procurer un raccord d'adaptation.

NOTE :

• Certains magnétophones ne peuvent pas être raccordés dû à l'incompatibilité des spécifications. D'autres ne donnent pas des résultats satisfaisants parce que de longues années d'usage ont créé chez eux des distortions, des bruits parasites et des faiblesses de puissance.

• Précautions à prendre lors de l'utilisation du magnétophone.

1) Pour tout transfert ou modification, utilisez toujours le magnétophone sur lequel l'enregistrement a été effectué. Une négligence de cette précaution peut rendre le transfert ou la modification Impossible à effectuer.

2) Gardez les têtes du magnétophone propres pour éviter une augmentation du niveau de distortion ou une diminution du niveau d'enregistrement.

3) Evitez d'utiliser des bandes à réponse en fréquences très basses ou encore des bandes froissées ou égratignées.

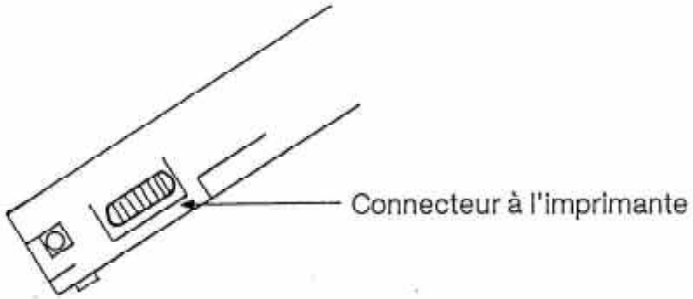
D. LE RACCORDEMENT AVEC L'IMPRIMANTE

L'appareil peut être branché directement avec n'importe quelle imprimante (10", 15", etc) de type parallèle Centronics. Le connecteur sur l'unité d'interface imprimante/cassette est configuré suivant le câblage ci-dessous :

Pin N°	1	$\overline{\text{STRB}}$ (STROBE)
	2	DATA 1
	3	DATA 2
	4	DATA 3
	5	DATA 4
	6	DATA 5
	7	DATA 6
	8	DATA 7
	9	DATA 8
	11	BUSY
	13 - 23	GND (GROUND)

Utilisez le câble de raccordement pour connecter l'imprimante avec l'interface imprimante/cassette et aussi connectez le Pin 24 du connecteur de l'unité d'interface à la terre - GND (Pin 13 - Pin 23).

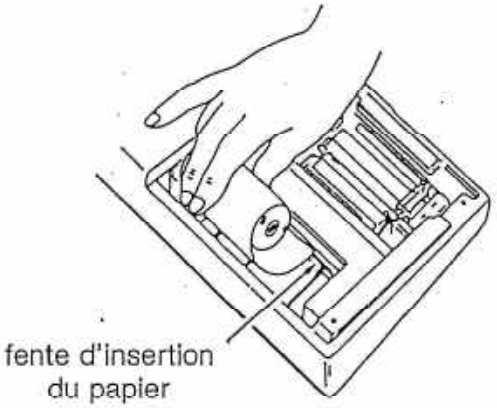
L'UNITÉ D'INTERFACE IMPRIMANTE/CASSETTE



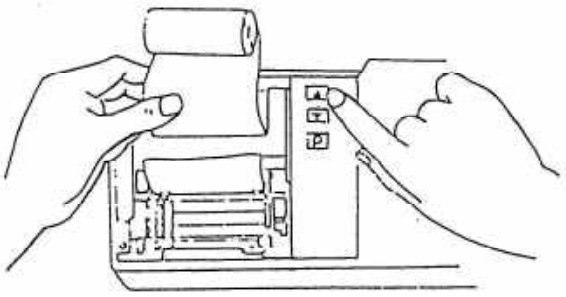
E. LE CHARGEMENT DU PAPIER

1) Pour ôter le couvercle de l'imprimante, levez la partie haute du couvercle.

2) Coupez proprement l'extrémité du rouleau de papier, puis introduisez-la dans la fente d'insertion du papier. (Un pli ou un gondolage quelconque peut empêcher l'insertion du papier).

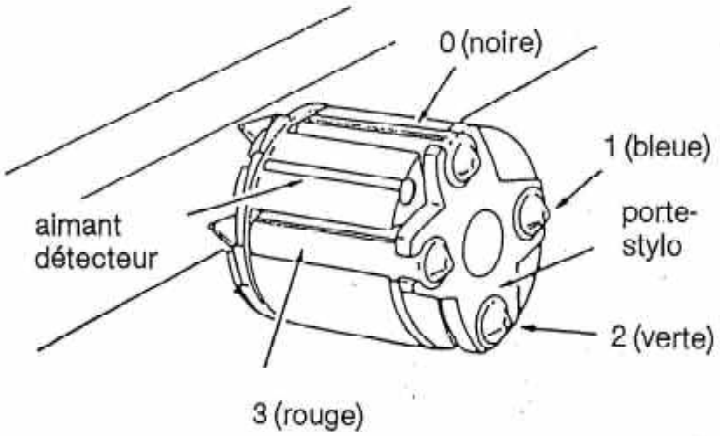


3) Appuyez sur la touche ON pour allumer l'appareil, puis sur la touche ▲ pour effectuer l'alimentation en papier. Laissez le papier sortir de 3 à 5 cm de l'imprimante.



F. LE REMPLACEMENT DES STYLOS

Quatre stylos différents peuvent être montés sur cette unité. Le schéma ci-dessous montre les positions des stylos :



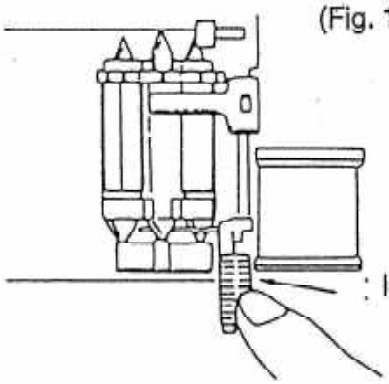
L'instruction COLOR positionne le stylo choisi sur l'aimant détecteur. Les stylos ayant les numéros 0, 1, 2, et 3 à partir de l'aimant comme indiqué sur la figure ci-contre. (Le porte-stylo tourne dans le sens inverse des aiguilles d'une montre pour amener le stylo choisi en haut).

Pour monter ou remplacer les stylos, suivez les instructions ci-dessous :

1) Appuyez la touche P . Ceci amène l'imprimante dans la position de remplacement des stylos.

2) Pour retirer le stylo, appuyez sur le levier de retrait de stylo. Cette opération fait sauter le stylo situé en haut du porte-stylo. Note : pour empêcher le stylo de,retomber dans le boîtier de l'imprimante, retenez-le légèrement (fig. 1).

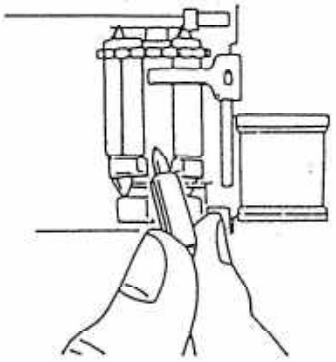
(Fig. 1)



: levier de retrait de stylo

3) Mettez en place un nouveau stylo (Fig. 2).

(Fig.2)



4) Appuyez sur la touche [▲] et tapez «COLOR 1» et «RETURN» pour installer ou retirer le stylo suivant. Le porte-stylo revient sur la gauche et tourne jusqu'à ce que le stylo suivant arrive à la position supérieure. Retirez le stylo et le remplacer avec un nouveau stylo comme décrit ci-dessus.

5) Après l'installation ou le remplacement des stylos, appuyez sur la touche [▲] . Ceci libère l'imprimante de sa position de remplacement de stylo et le stylo revient sur la gauche. Note : pour un usage correct, montez toujours 4 stylos. L'omission d'un seul stylo est suffisante pour perturber les changements de couleur.

Entretien des stylos

Retirez les stylos de l'imprimante si vous ne vous servez pas de l'imprimante. Capuchonnez-les avant de les ranger dans leur étui. L'encre risque de se dessécher si les stylos sont laissés sur l'imprimante pendant un temps inactif prolongé ou s'ils sont laissés à l'air libre.

G. L'UTILISATION DU MAGNÉTOPHONE

1. La commande CSAVE

Avec la commande CSAVE qui enregistre votre programme, vous devez donner à votre programme un «nom de fichier». Ceci pour des raison de référence. La longueur de ce nom de fichier ne doit pas dépasser 6 caractères. Pour enregistrer votre programme avec un nom de fichier, tapez :

CSAVE «nom de fichier»

Votre programme sera enregistré sous le nom que vous spécifiez. Vous pouvez le nommer comme vous voulez pourvu qu'il soit facile à s'en souvenir. Si la longueur du nom dépasse 6 caractères, l'excès sera ignoré. Plus d'un programme peut être enregistré sur chaque cassette et l'appareil les «sautera» jusqu'à ce qu'il trouve la place pour enregistrer le programme courant (voir 2.1.7 pour CSAVE).

2. La commande CLOAD

Avec CLOAD qui rappelle votre programme enregistré sur cassette, vous devez fournir un «nom de fichier». Tapez comme ci-dessous :

CLOAD «nom de fichier»

Quand «nom de fichier» est retrouvé sur la bande cassette, le message «FOUND nom de fichier» s'affichera et quand le chargement en mémoire est terminé, le symbole de guidage s'affichera (voir 2.1.4 pour CLOAD).

V. SPÉCIFICATIONS DU LANGAGE 'BASIC'

Les commandes

APPEND , CHAIN , CLEAR , CLOAD , CONSOLE , CONT , CSAVE , DELETE , LIST , LLIST , LOCK , NEW , RENUM , RUN , UNLOCK

Les instructions

BEEP , BOX , COLOR , DATA , DEFFN , DEFINT/SNG/STR , DIM END , ERASE , FOR - NEXT , GOSUB - RETURN , GOTO , IF - THEN - ELSE , LET , LINE , MOVE , MOTOR , ON ERROR GOTO , ON - GOSUB , ON - GOTO , OPTION BASE , ORIGIN , POKE , RANDOMIZE , READ , RLINE , REM , RESTORE , RESET , RESUME , ROTATE, STOP, SCALE, TRON/TROFF, INPUT, INPUT#, LOCATE, LPRINT, PRINT, PRINT, PRINT#, PRINT USING, PAUSE, WAIT

Les fonctions

KEY , KEYLIST , ABS , ATN , COS , EXP , FIX , INT , LOG , RND , SGN , SIN , SQR , TAN , TAB , ASC , CHR$, HEX$, LEFT$, LEN , MID$, OCT$, RIGHT$, STR$, VAL , ERR , ERL , FRE , PEEK , INKEY$

CHAPITRE 1 : INFORMATION GENERALE DU BASIC

1 . 1 . Les modes d'opération

Quand la machine est mise en marche, le symbole ">" s'affiche indiquant que le BASIC est au niveau de commande et que la machine est prête à accepter les commandes. L'appareil peut donc être utilisé en deux modes suivants : direct et indirect.

En mode "direct", les instructions et les commandes sont exécutées immédiatement après leur validation par la touche RETURN. Les résultats sont affichés, mais les instructions ne sont pas gardées en mémoire. Ce mode est utilisé pour la mise au point des programmes. Il est aussi utile quand vous vous servez de l'appareil comme une calculatrice pour des calculs rapides sans le besoin de créer des programmes complets.

En mode "indirect", vous créez des programmes. Les instructions du programme sont dans ce mode toujours précédées par des numéros de ligne. Le programme complet est gardé en mémoire et sera exécuté par la commande "RUN", dans l'ordre des numéros de ligne.

1 . 2 . Le format de ligne

Une ligne de programme commence toujours par un numéro de ligne et se termine par un retour chariot (touche RETURN). Elle peut contenir jusqu'à 255 caractères et respecte le syntaxe suivant :

nnn (numéro de ligne) instruction BASIC [:Instruction BASIC...]...
.....RETURN

Plusieurs instructions peuvent être mises sur la même ligne si elles sont séparées par le symbole ":".

Exemple :

100 SCALE 4 : MOVE (20,0) RETURN

100	SCALE 4	MOVE (20,0)	RETURN
numéro de ligne	instruction	instruction	retour chariot

1 . 3 . Les numéros de lignes

Chaque ligne de programme commence par un numéro de ligne. Les numéros de ligne indiquent la séquence de stockage du programme en mémoire et ils sont utilisés comme indicateurs pour les branchements (par GOTO) ou la mise en page. Les numéros doivent se situer entre 0 et 65529. Un point (.) peut être utilisé pour indiquer (ou supprimer) la ligne courante lors de l'utilisation des commandes LIST et DELETE.

1.4. Le jeu de caractères

Le jeu de caractères se compose de l'alphabet, des chiffres, et autres caractères spéciaux. Les caractères alphabétiques sont les lettres en majuscules tandis que les chiffres sont de 0 à 9.

1.5. Les constantes

Il existe deux types de constantes : chaîne de caractères et numérique.

1.5.1 Les constantes à caractères

Une constante de caractères est une chaîne de 255 caractères au maximum encadrée par des guillemets (").

"HELLO" "25,00F" "NOMBRE D'EMPLOYÉS"

1.5.2 Les constantes numériques

Les constantes numériques peuvent être positives ou négatives et sont d'un des types listés ci-dessous :

TYPE	DESCRIPTION
Entier	Nombres entiers entre −32768 et +32767. Exemple : A% = 32767
Réel	Nombres réels positifs ou négatifs Exemple : A = 35.54
Virgule flottante	Nombres positifs ou négatifs représentés en forme exponentielle. Ils se composent d'une nantisse suivie de la lettre E et l'exposant. L'exposant doit se situer entre −38 et +38. Exemples : 235.988 E−7 est égale à .0000235988 2359 E 6 est égale à 2359000000
Hexadécimal	Nombres préfixés par "&H". Ces nombres seront affichés en décimal. Exemple : 10 X = &H76 : PRINT X RUN 118
Octal	Nombres préfixés par "&O" ou "&". Ils sont aussi préfixés en décimal. Exemple : 10X = &0347 : PRINT X RUN 231

1.5.3 La simple précision

Les constantes en simple précision sont toutes les valeurs numériques ayant sept chiffres ou moins, en forme exponentielle, ou ayant pour caractère de fin le point d'exclamation (!). A l'affichage, seulement six des sept chiffres seront affichés.

Exemple :

46.8
−0.9 E −06 3489.0 22.5!

1.6. Les variables

1.6.1 Les variables et les caractères de déclaration

Les noms des variables peuvent être de n'importe quelle longueur ; cependant, seulement les deux premiers caractères seront pris en compte. Le premier caractère du nom doit être un caractère alphabétique ; les autres caractères peuvent rester alphanumériques.

Un nom de variable ne peut pas être un mot réservé et ne contient pas un mot réservé. Par exemple, BFOR est illégale parce qu'elle contient le mot réservé FOR. Les mots réservés sont les commandes, les instructions, et les fonctions.

Les variables représentent des valeurs numériques ou à caractères. Les noms des variables à caractères ont pour dernier caractère le symbole "$" (A$ = "nom"). Le symbole "$" est ce qu'on appelle le caractère de déclaration du type de variable ; il déclare ici que la variable est à caractères. Utilisez un des trois caractères ci-dessous pour déclarer le type de variable :

%	Variable en entier
!	Variable en simple précision
$	Variable à caractères

Si vous omettez le caractère de déclaration, la variable sera considérée comme une variable simple précision.

Exemples :

MINIMUM!	Déclare une valeur en simple précision.
LIMIT%	Déclare une valeur entière.
N$	Déclare une valeur à caractères.
ABC	Déclare une valeur en simple précision.

1.6.2 Les tableaux indicés (ou dimensionnés)

Les variables peuvent aussi être déclarées comme des tableaux avec indices. Par exemple, A(10) se réfère à un tableau d'une dimension de dix éléments, chaque élément étant référé de A(1) à A(10). A(5,5) déclare un tableau de deux dimensions ayant 25 éléments en 5 lignes et 5 colonnes. Les éléments sont référés par A(1,1) à A(5,5).

1.7 La conversion du type de précision

S'il le faut, le BASIC convertit une constante numérique d'un type à un autre. Si vous essayez de convertir une variable à caractères en numérique ou vice versa, l'erreur «TYPE MISMATCH» s'affichera.

Respectez les règles suivantes lors des conversions des constantes numériques.

a. Si vous avez des constantes de types différents, la constante sera stockée au type déclaré par le nom de la variable.

Exemple :
```
10 A% = 23.42
20 PRINT A%
RUN [RETURN]
23
```

b. Pendant l'évaluation des expressions, tous les opérandes dans une opération arithmétique ou relationnelle sont convertis, et les résultats retournés, au même degré de précision que celui de l'opérande le plus précis.

Exemple :
```
10 D = 6/7
20 PRINT D
RUN [RETURN]
.857143
```

c. Les opérations logiques convertissent les opérandes en entier et retournent un résultat entier. Les opérandes doivent se situer entre −32768 et +32767 sinon l'erreur «OVERFLOW» se produit.

Exemple :
```
10 PRINT 8.123 OR 24
RUN [RETURN]
24
```

d. Quand une valeur en virgule flottante est convertie en entier, la partie fractionnelle est tronquée.

Exemple :
```
10 C% = 55.88
20 PRINT C%
RUN [RETURN]
55
```

1.8. Les expressions et les opérateurs

Une expression est une constante à caractères ou numérique qui, après avoir été combinée, produit un seul résultat. Les opérateurs, utilisés pour effectuer ces opérations mathématiques ou logiques, sont divisés en quatre catégories : arithmétique, relationnel, logique et fonctionnel.

Exemple :

3.14 , "PI" , 5 - 8 , A+B , M$, X<(T−1)

1.8.1 Les opérateurs arithmétiques

Ces opérateurs, dans leur ordre de priorité, sont listés dans le tableau ci-dessous.

OPÉRATEUR	OPÉRATION	EXEMPLE
∧	Elévation à une puissance	2 ∧ 4
−	Négation	−2
*,/	Multiplication et division en virgule flottante	2 * 4 4/2.5
MOD	Modulo division entière	5 MOD 2
+,−	Addition et soustraction	4+2 , 4−2

Utilisez les parenthèses pour changer l'ordre des opérations. Les opérations à l'intérieur des parenthèses sont évaluées en priorité. A l'intérieur des parenthèses, l'ordre de priorité normal est maintenu.

Exemples :

a. Expression algébrique	b. Expression BASIC
X + 2Y	X + 2*Y
$X - \frac{Y}{X}$	X − Y/X
$\frac{X + Y}{Z}$	(X + Y)/Z
$(X^2)\ Y$	(X ∧ 2)*Y
XYZ	X * Y * Z
X(−Y)	X * (−Y)

1.8.2 Les opérateurs relationnels

Les opérateurs relationnels comparent deux valeurs et en décident l'ordre d'exécution du programme. Le résultat de la comparaison est vrai (1) ou faux (0). Le BASIC met à votre disposition les opérateurs relationnels suivants :

OPÉRATEUR	RELATION	EXPRESSION
=	Egalité	X = Y
< > ou > <	Inégalité	X < > Y
<	Inférieur à	X < Y
>	Supérieur à	X > Y
< = ou = <	Inférieur ou égal	X < = Y
> = ou = >	Supérieur ou égal	X > = Y

Exemples :

```
IF SIN(X) < 0 THEN GOTO 100
IF I MOD  J < > 0 THEN X=X +1
```

Note :

Si les opérateurs arihmétiques et relationnels sont combinés dans une expression, l'arithmétique sera effectuée d'abord. Par exemple, l'expression "X + Y < (T − 1)/Z" est vraie si le résultat de "X + Y" est inférieur au résultat de "(T − 1)/Z".

1.8.3 Les opérateurs logiques

Le BASIC fournit des opérateurs logiques pour vous permettre d'effectuer de la modification de configuration binaire, des opérations Booléennes, ou de tester des relations multiples. Comme pour les opérateurs relationnels, l'opérateur logique retourne une valeur vraie (1) ou fausse (0). Les opérations logiques sont évaluées après les opérations arithmétiques et relationnelles. Les exemples suivants montrent le résultat des opérations logiques. Les opérateurs sont arrangés dans leur ordre de priorité.

Exemples :

NOT	X	NOT X
	1	0
	0	1

AND	X	Y	X AND Y
	1	1	1
	1	0	0
	0	1	0
	0	0	0

OR	X	Y	X OR Y
	1	1	1
	1	0	1
	0	1	1
	0	0	0

XOR	X	Y	X XOR Y
	1	1	0
	1	0	1
	0	1	1
	0	0	0

Une utilisation des opérateurs logiques est de connecter deux ou plusieurs opérateurs relationnels pour en faire des décisions sur l'ordre d'exécution du programme. Par exemple :

IF D<200 AND F<4 THEN 80.
Toutes les deux conditions devraient être vraies pour avoir le branchement sur la ligne 80.

IF I>10 OR K<0 THEN 50.
A moins que toutes ces deux conditions soient fausses, le programme se branche sur la ligne 50.

Les opérateurs logiques convertissent leurs opérandes en valeurs entières de 16 bits, en forme de complément à 2, et situant entre les limites de −32768 et +32767. L'opération donnée est évaluée sur les entiers dans la manière de bit par bit ; il est donc possible d'utiliser les opérateurs logiques pour tester les octets pour une configuration de bit particulière. Dans la pratique, vous pouvez utiliser l'opérateur AND pour masquer tout sauf un bit d'un octet d'état du port de l'appareil, ou vous pouvez utiliser l'opérateur OR pour combiner deux octets pour en créer une valeur binaire particulière.

Exemples :

63 AND 16 = 16

63 est égale à 111111, 16 est égale à 10000, donc une opération AND de bit par bit donnera 10000.

10 OR 10 = 10

10 est égale à 1010, donc 1010 OR 1010 = 1010.

1.8.4 Les opérations sur les chaînes de caractères

Les chaînes de caractères peuvent être comparées entre elles de la même manière que pour les nombres. Les comparaisons des chaînes de caractères se font en prenant chaque caractère à partir de la gauche et en le comparant avec les codes ASCII un par un. Si les codes ASCII sont identiques, les chaînes seront considérées comme égales. Si les codes se diffèrent, les codes inférieurs sont considérés comme plus petite en valeur numérique que les codes supérieurs. Une chaîne plus courte est plus petite en valeur. Les espaces devant et derrière sont significatifs.

Exemples :

"AAN" < "AB" "FICHIER" = "FICHIER"

"X&" > "X" "CL" > "CL" "SMYTH" < "SMYTHE"

B$ < "9/12/80" où B$ = "8/12/80"

Les chaînes peuvent être concatenées en utilisant le symbole "+".

Exemple :

```
10 A$ = "FICHIER"   : B$ = "CLIENT"
20 PRINT A$ + B$
30 PRINT "NOUVEAU" + A$ + B$
```

RUN [RETURN]

FICHIER CLIENT

[RETURN]

NOUVEAU FICHIER CLIENT

1.9 La mise en page sur l'affichage

Les caractères entrés au clavier sont d'abord reçus par «l'éditeur» du BASIC, puis affichés à la position du curseur. Les entrées au clavier ne sont interprétés que lorsque vous tapez la touche RETURN. Les lignes qui commençent avec des numéros seront gardées en mémoire en tant que lignes de programme ; tandis que celles sans numéros seront interprétées et exécutées immédiatement en mode direct.

1.10 Les messages d'erreurs

Quand le BASIC détecte une erreur, l'exécution du programme s'arrête et un message d'erreur s'affichera. En mode direct, le format du message est "ERROR # nn (numéro d'erreur)". En mode indirect, le format sera "ERROR # nn (numéro d'erreur) IN nn (numéro de ligne)". Voir page 56 pour le détail sur chaque numéro d'erreur.

CHAPITRE 2 : LES COMMANDES ET LES INSTRUCTIONS

Dans la syntaxe des commandes et instructions qui suit, les remarques ci-dessous s'appliquent.

a. Entrez les éléments écrits en lettres alphabétiques majuscules comme décrit.

b. Les éléments renfermés par < > sont à désigner.

c. Les éléments renfermés par [ ] sont entrés par option. Lorsqu'ils sont omis, les valeurs par défaut, ou les valeurs précédentes s'appliquent. Valeurs par défaut sont des valeurs définies par le BASIC.

d. En plus des "<>"et "[]", les symboles tels que ",", "( )", ";", "–", etc. devront être mis correctement dans la position désignée.

e. Les éléments ayant le symbole (...) peuvent être répétés dans la limite de la longueur de la ligne programme.

f. Les éléments renfermés par { } feront l'objet du choix de l'un d'entre eux.

2.1. LES COMMANDES

2.1.1 APPEND

APPEND charge un programme, enregistré sur bande cassette, en mémoire et le combine, dans le cas échéant, avec celui qui était déjà en mémoire. APPEND diffère de CLOAD par le fait qu'elle ne détruit pas le programme qui était en mémoire (voir 2.1.4 pour CLOAD). Dès que la commande APPEND se termine, le BASIC revient au niveau de commande. Avec APPEND, vous pouvez combiner plus de deux programmes. Cependant, assurez-vous que les numéros de lignes du programme qui entre en mémoire ne soient pas inférieurs à ceux du programme qui était déjà en mémoire. L'ignorance de cette condition peut provoquer des résultats ambigus.

Syntaxe :

APPEND <"nom du programme">

Exemple :

APPEND "ABCD"

Note :

Si le courant est coupé ou un BREAK est exécuté lors de la commande APPEND, le programme qui est en cours de chargement sera invalide.

2.1.2 CHAIN

CHAIN charge le programme désigné, de la cassette en mémoire et commence son exécution à partir du numéro de ligne désigné. Dans le cas où le numéro de ligne n'est pas spécifié, l'exécution démarrera à partir du début du programme. Le programme qui était en mémoire sera détruit avant l'exécution de CHAIN. Cependant les variables ne sont pas détrultes. Au cas où une erreur de lecture se produisait pendant l'exécution de CHAIN, le programme et les données en mémoire seront rendus invalides.

Syntaxe

CHAIN <"nom du programme">[, <numero de ligne>]

Exemple

CHAINE "ABCD", 100

Note :
Si une erreur de lecture bande se produisait pendant le chargement du programme, le programme et les données stockés en mémoire seront rendus invalides.

2.1.3 CLEAR

CLEAR met à zéro toutes les variables numériques et annule les variables à caractères. Ses paramètres en option réserve l'espace des chaînes de caractères.

Syntaxe

CLEAR [<espace chaine>]

Exemples

CLEAR	Met à zéro et annule les variables numériques et variables chaînes respectivement.
CLEAR 500	Idem à CLEAR mais réserve 500 octets de mémoire pour l'espace chaîne.

2.1.4 CLOAD

CLOAD charge un programme en mémoire à partir de la cassette. Lorsque CLOAD est exécutée, le système recherche sur la bande le programme spécifié. Spécifiez donc le même nom qui était désigné lors de la sauvegarde du programme. Le message «FOUND nom du programme» s'affiche dès que le programme recherché est trouvé, et le «>» reviendra quand le chargement sera terminé. Il est possible qu'il y ait d'autres programmes sur bande qui se sont situés avant le programme recherché ; dans ce cas, le message «SKIP nom de programme» s'affichera pour chacun de ces programmes.

Opération

a. Raccorder l'interface et le magnétophone avec le câble approprié. Connecter les «EAR» des deux unités par les prises grises, et les «REM» par les prises noires.

b. Appuyez sur la touche PLAY du magnétophone.

c. Ecrivez sur l'appareil :

```
CLOAD "nom du programme"
```

d. Appuyez sur la touche RETURN.

La commande CLOAD? compare le programme qui se trouve en mémoire avec le programme ayant le même nom sur bande. S'ils sont les mêmes, le «>» apparaîtra ; sinon, une erreur s'affichera. Les programmes ne sont pas modifiés par cette commande.

Syntaxe

```
CLOAD <"nom du programme">
CLOAD ? <"nom du programme">
```

Exemples :

CLOAD "TEST"	Le programme TEST1 est recherché sur la bande et sera chargé en mémoire.
CLOAD? "TEST1"	Le programme TEST1 sur la bande est comparé avec celui actuellement en mémoire.

2.1.5 CONSOLE

CONSOLE affiche les contenus des touches de fonctions (F1 à F10). Pour afficher ces contenus, entrez «1» (ou un chiffre entre 1 à 9) ; pour les faire disparaître, entrez «0». L'appareil est au départ pré-sélectionné avec PRINT dans F1, INPUT dans F2, GOTO dans F3, LIST dans F4, RUN dans F5, CSAVE dans F6, KEY dans F7, CLOAD dans F8, CONSOLE dans F9 et CONT dans F10. Ces fonctions peuvent être remplacées par des nouvelles qui seront entrées au clavier. Lorsque l'appareil est allumé, les contenus de F1 à F5 sont affichés en bas de l'affichage ; ceux de F6 à F10 peuvent être affichés à leur tour quand la touche SHIFT est appuyée.

Exemple :

```
CONSOLE , 1
```

```
>
PRIN  INPU  GOTO  LIST  RUN
```

2.1.6 CONT

CONT reprend l'exécution d'un programme après un arrêt d'exécution suite aux instructions STOP ou END. L'exécution continuera à partir du point d'arrêt.

Syntaxe

```
CONT
```

Exemple :
```
10 FOR I=1 TO 100
20 PRINT I
30 STOP
40 NEXT I
```

Opération au clavier	Affichage
RUN [RETURN]	1
[RETURN]	BREAK IN 30
CONT [RETURN]	2
[RETURN]	BREAK IN 30
CONT [RETURN]	3

2.1.7. CSAVE

CSAVE enregistre sur bande cassette le programme qui est actuellement en mémoire. Vous pouvez l'utiliser en mode direct ou indirect.

Syntaxe

```
CSAVE <"nom du programme">
```

Opération :

a. Raccordez l'interface cassette et le magnétophone par le câble fourni. Connectez «REG» de l'interface au «MIC» du magnétophone avec les prises rouges du câble. Faites de même pour «REM» des deux unités avec les prises noires.

b. Appuyez sur la touche RECORD du magnétophone.

c. Ecrivez sur l'appareil :

CSAVE "nom du programme"

d. Appuyez sur la touche RETURN.

Note :
Il n'y a pas d'affichage particulier durant l'enregistrement du programme ; le symbole «>» s'affichera dès que l'enregistrement sera terminé.

2.1.8 DELETE

DELETE supprime de la mémoire les lignes du programme, que vous aurez spécifiées. Le point (.) peut être utilisé pour dénoter la ligne courante.

Syntaxe

DELETE <numéro de ligne> [– <numéro de ligne>].

Exemple :

DELETE 40	Supprime la ligne 40
DELETE 40 – 100	Supprime les lignes de 40 à 100
DELETE – 40	Supprime toutes les lignes jusqu'à et y compris la ligne 40
DELETE 40 –	Supprime toutes les lignes à partir de 40 jusqu'à la fin. (Y compris la ligne 40).

2.1.9 LIST

LIST affiche une ligne désignée du programme. Si le numéro de ligne n'est pas spécifiée, la ligne ayant le plus petit numéro sera affichée. Par exemple : LIST 100 affichera le contenu de la ligne 100 du programme. Le point (.) peut être utilisé pour indiquer la ligne courante.

2.1.10 LLIST

LLIST imprime tout ou une partie du programme actuellement en mémoire à l'imprimante. Appuyez sur la touche BREAK pour arrêter l'édition.

Syntaxe

LLIST { [numéro de ligne] - [numéro de ligne] }

Exemples :

LLIST	imprime tout le programme en mémoire.
LLIST 500	imprime seule la ligne 500.
LLIST 150–	imprime toutes les lignes à partir de 150 jusqu'à la fin.
LLIST – 150	imprime toutes les lignes du début jusqu'à la ligne 150.
LLIST 50 – 100	imprime toutes les lignes entre 50 et 100.

2.1.11 LOCK

LOCK retient le programme en mémoire. Lorsqu'elle est exécutée, les commandes qui peuvent changer le programme telles que NEW, CLEAR, ERASE, et CLOAD deviennent inactives. Si ces commandes étaient entrées, le message «WRONG PROGRAM CHANGE» s'afficherait. La destruction du programme par une manipulation erronée du clavier est donc empêchée. Pour libérer LOCK, entrez la commande «UNLOCK» (voir 2.1.15 pour UNLOCK).

2.1.12 NEW

NEW supprime le programme en mémoire et efface toutes les variables.

Syntaxe

NEW

Note :
La commande NEW est souvent exécutée en mode direct avant qu'un nouveau programme soit chargé. Après son exécution, le BASIC retourne au niveau de commande.

2.1.13 RENUM

RENUM renumérote les lignes du programme et change toutes les références aux numéros de lignes, désignés dans les instructions GOTO, GOSUB, THEN, ON ... GOTO, ON... GOSUB et ERL.

Syntaxe

RENUM [nouveau numéro [, ancien numéro] [, incrémentation]]

Exemple :

a) RENUM — Remumérote le programme entier en commençant par la ligne 10 avec incrémentation de 10 en 10.

Avant	Après
1 PRINT 1	10 PRINT 1
2 PRINT 2	20 PRINT 2
3 PRINT 3	30 PRINT 3
4 PRINT 4	40 PRINT 4

b) RENUM 2 — Renumérote le programme complet en commençant par la ligne 2 avec incrémentation de 10 en 10.

Avant	Après
10 PRINT 1	2 PRINT 1
20 PRINT 2	12 PRINT 2
30 PRINT 3	22 PRINT 3
40 PRINT 4	32 PRINT 4

c) RENUM 100,,50 — Renumérote le programme complet en commençant par la ligne 100 avec incrémentation de 50.

Avant	Après
2 PRINT 1	100 PRINT 1
12 PRINT 2	150 PRINT 2
22 PRINT 3	200 PRINT 3
32 PRINT 4	250 PRINT 4

d) RENUM 300,150,50 — Change ligne 150 à 300 et toutes les autres lignes avec incrémentation de 50 en 50 (les lignes avant 150 ne sont pas changées)

Avant	Après
100 PRINT 1	100 PRINT 1
150 PRINT 2	300 PRINT 2
200 PRINT 3	350 PRINT 3
250 PRINT 4	400 PRINT 4

2.1.14 RUN

RUN exécute, le programme actuellement en mémoire. Si un numéro de ligne est spécifié, l'exécution débutera de cette ligne.

Syntaxe

RUN [<numéro de ligne>]

Exemple :

RUN 100 commence l'exécution du programme en mémoire à la ligne 100.

2.1.15 UNLOCK

UNLOCK libère le programme qui est «LOCK», et permet le changement du programme.

Syntaxe

UNLOCK

Après son exécution, les commandes NEW, CLEAR, ERASE, CLOAD et la mise en page par numéros de lignes deviennent actives.

2.2 LES INSTRUCTIONS GÉNÉRALES

2.2.1 BEEP

BEEP fait sonner brièvement l'avertisseur sonore.

Syntaxe

BEEP <entier>, <entier>

Exemple :

BEEP 3,10

Note :
Le premier nombre représente le son musical et le deuxième nombre, la durée du son. Il y a 32 sons musicaux disponibles et la durée peut être désignée par un entier positif qui va de 1 à 256. La relation entre les entiers et les sons musicaux est décrite ci-dessous. La durée est de + 10 secondes et la valeur par défaut est à 4. Lorsque le longueur est 0, l'erreur «ILLEGAL FUNCTION CALL» s'affichera.

Entier	Son	Fréquence	Entier	Son	Fréquence
0	nul		16	Sol#	833 Hz
1	Fa	349 Hz	17	La	880
2	Fa#	370	18	La#	933
3	Sol	393	19	Si	992
4	Sol#	414	20	Do	1042
5	La	440	21	Do	1116
6	La #	466	22	Ré	1179
7	La	492	23	Ré#	1250
8	Do	525	24	Mi	1330
9	Do#	553	25	Fa	1389
10	Ré	584	26	Fa#	1488
11	Ré#	625	27	Sol	1563
12	Mi	658	28	Sol #	1645
13	Fa	702	29	La	1736
14	Fa#	735	30	La #	1838
15	Sol	781	31	Si	1953

2.2.2 BOX

BOX imprime un carré avec les lignes et couleurs désignées. Le type de ligne peut être sélectionné en spécifiant un nombre entre 0 à 15. 0 produira une ligne pleine tandis que 15 produira une ligne pointillée. Quand une sélection est omise, la désignation précédente prendra effet. Les types de couleur varient de 0 à 3. Lorsque sa désignation n'est pas mentionnée, la couleur du présent stylo sera choisie (pour la désignation des couleurs, voir 2.2.4 : COLOR).

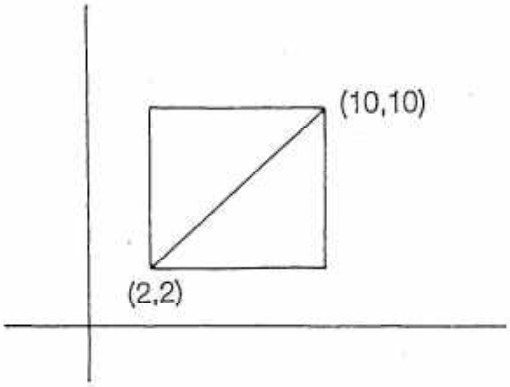
Syntaxe

BOX [(X1,Y1)] [– (X2,Y2)] , [type de ligne] , [type de couleur]

Ceci imprime le carré ayant la ligne diagonale entre les coordonnées (X1,Y1) et (X2,Y2) avec (X1,Y1) comme point d'origine.

Exemple :

BOX (2,2) - (10,10) , 0, 3



Le carré à gauche est imprimé en ligne pleine et en couleur rouge

2.2.3 CIRCLE

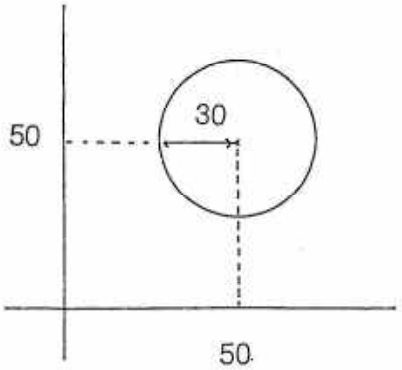
CIRCLE imprime un cercle avec son centre au point (X,Y) et un rayon d'une longueur désignée. La couleur peut être spécifiée en option.

Syntaxe

CIRCLE [(X,Y)] , <rayon> [,<type de couleur>]

Exemple :

CIRCLE (50,50), 30,2



2.2.4 COLOR

COLOR change le code couleur par défaut de l'imprimante. Il y a quatre couleurs possibles (0 à 3). Les numéros de couleur sont : 0 - noire, 1 - bleue, 2 - verte et 3 - rouge. Si vous entrez «COLOR TEST», les chiffres 0 à 3 seront imprimés dans leur couleur respective.

Syntaxe

COLOR n (numéro de couleur : 0 à 3)

Exemple :

```
COLOR TEST
0 (en noir), 1 (en bleu), 2 (en vert), 3 (en rouge)
```

2.2.5 DATA

DATA fournit des données numériques et des constantes de chaîne à caractères, à l'instruction READ. L'intruction READ lit les éléments de l'instruction DATA dans l'ordre de leur occurence. Les éléments contenus dans DATA peuvent être considérés comme formant une liste continue. Une instruction DATA peut contenir autant de constantes qu'une ligne de programme le permet et elle peut être placée n'importe où dans le programme.

Syntaxe

DATA <constante> [, <constante>]...

Exemple :

```
10 REM PROGRAMME POUR LIRE UNE LISTE
20 REM DE VARIABLES ET LES IMPRIMER
30 FOR I=1 TO 3
40 READ A,B,C
50 PRINT A ; B ; C
60 NEXT I
70 DATA 1,2,3
80 DATA 4,5,6
90 DATA 7,8,9
100 END

RUN
1 2 3
[RETURN]
4 5 6
[RETURN]
7 8 9
```

2.2.6 DEFFN

La fonction DEFFN est utilisée pour définir des nouvelles fonctions. Le nom de la fonction doit être précédé par FN, et les variables dans sa liste de paramètres, doivent être séparées par une virgule (,). Les fonctions à définir peuvent être numériques ou en chaîne à caractères, mais leurs valeurs doivent correspondre à celles des paramètres. Exécutez une instruction DEFFN pour définir une fonction avant son appel.

Syntaxe

DEFFN <nom de la fonction> [(<liste d'arguments>)]
= <définition de la fonction>

Exemple :

```
* Définition d'une fonction numérique
10 DEFFNB (X,Y)=X/Y * 100
20 I=20 : J=5
30 T=FNB (I,J)
40 PRINT T
50 END
RUN
400
```

2.2.7 DEF + INT/SNG/STR

DEF + INT/SNG/STR déclare un type de variable avec sa gamme de premier caractère. DEFINT est pour les entiers, DEFSNG est pour la simple précision, ou DEFSTR est pour les chaînes à caractères.

Syntaxe

DEF <Type> <Gamme de caractères>
[<Gamme de caractères>]

Type = INT, SNG ou STR

Exemple :

10 DEFINT L-P Toutes variables commençant par L, M, N, O et P deviennent des entiers.

20 DEFSTR A Toutes variables commençant par la lettre A deviennent des chaînes à caractères.

2.2.8 DIM

L'instruction DIM attribue la place mémoire pour le stockage des tableaux et des matrices. Par défaut, les variables indicées sans leurs définitions par DIM, auront 10 éléménts. Si une valeur supérieure au maximum défini, est référée, l'erreur «SUBSCRIPT OUT OF RANGE» s'affichera. A chaque instruction DIM, les valeurs des tableaux sont mises à zéro.

Syntaxe

DIM <variable> (<valeur minimum> [,<valeur maximum>...])

Exemples :

```
10 DIM A (20)
```

```
10 DIM A (5)
20 FOR I=1 TO 5
30 A (I)=I
40 PRINT A ;
50 NEXT I
RUN
0 0 0 0 0
```

2.2.9 END

END termine l'exécution du programme et retourne l'appareil au niveau de commande après avoir fermé tous les fichiers. Elle peut apparaître n'importe où dans le programme. Elle est facultative à la fin du programme.

Syntaxe

```
END
```

Exemple :

```
520 IF K >1000 THEN END ELSE GOTO 20
```

2.2.10 ERASE

ERASE supprime les tableaux antérieurement définis par DIM. Après une telle suppression, vous pouvez re-dimensionner le tableau et l'espace libéré pourra donc être utilisé pour d'autres projets.

Syntaxe

```
ERASE <variable tableau> [,<variable tableau>...]
```

Exemple :

```
10 DIM A (10)
20 FOR I=1 TO 10
30 PRINT A (I) ;
40 NEXT I
50 ERASE A
60 DIM A (5)
70 FOR I=1 TO 5
80 PRINT A (I) ;
90 NEXT I
RUN
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0
```

2.2.11 FOR ... NEXT

La commande FOR ... NEXT fait boucler les instructions qu'elle entoure pour un nombre de fois donnée. FOR ouvre la boucle et doit définir le nom de variable, la première valeur, la dernière valeur et aussi le pas d'incrémentation (par défaut 1). NEXT ferme la boucle et renvoie le contrôle à l'instruction FOR. Ce processus continue jusqu'à l'atteinte de la valeur maximum.

Syntaxe

FOR <nom variable> = X (valeur de départ) TO Y (valeur de fin)
STEP Z (incrémentation)

NEXT <nom variable>

Exemple :

```
10 REM ITERATION SIMPLE
20 FOR I=1 TO 5
30 PRINT I ;
40 NEXT I
RUN
1 2 3 4 5
```

2.2.12 GOSUB/RETURN

GOSUB donne l'ordre au programme de transférer l'exécution à une sous-routine désigné. Les sous-routines sont utilisés lorsqu'une même série d'instructions est utilisée plusieurs fois par le programme. Dès que les instructions de la sous-routine sont exécutées, le RETURN renvoie la suite de l'exécution à la ligne qui suit le GOSUB. Il peut y avoir plusieurs RETURNS qui correspondent à un seul GOSUB. Les sous-routines peuvent emboîter d'autres sous-routines. Le nombre d'emboîtement est limité par la capacité de la mémoire.

Syntaxe

GOSUB <numéro de ligne>

Exemple :

```
10 PRINT "AVANT SOUS-ROUTINE J=";J
20 GOSUB 60
30 PRINT
40 PRINT "APRES SOUS-ROUTINE J=";J
50 END
60 J=J+5
70 RETURN
RUN
AVANT SOUS-ROUTINE J=0
RETURN
RETURN
APRES SOUS-ROUTINE J=5
```

2.2.13 GOTO

GOTO inconditionnellement branche l'exécution du programme à un numéro de ligne désigné.

Syntaxe

GOTO <numéro de ligne>

Exemple :

```
10 READ R
20 PRINT "R=";R;
30 A=3.14 *RΛ2
40 PRINT "AIRE = " ; A
50 GOTO 10
60 DATA 5,7,12
RUN
R=5                AIRE=78.5
[RETURN]
R=7                AIRE=153.86
[RETURN]
R=12               AIRE=452.16
[RETURN]
ERROR #4 IN 10
```

2.2.14 IF...THEN...[ELSE]

IF choisit une route particulière pour l'exécution du programme, basé sur les conditions établies dans l'expression logique. Utilisez AND si plusieurs conditions sont nécessaires pour accomplir un résultat désiré.

Syntaxe

IF <expression logique>[<AND expression logique>].....

{... GOTO< numéro de ligne>
ou THEN <expression> / <numéro de ligne>}

ELSE <expression> / <numéro de ligne>

Exemple :
```
10 INPUT "AGE ET SALAIRE" ;AGE,SAL
20 IF AGE=65 AND SAL<7000 THEN PER= .05ELSE PER=.1
30 DIS=300*PER
40 PRINT
50 PRINT "VOTRE REDUCTION EST"
60 PRINT USING "F# #.# #";DIS
```

Opération au clavier		Affichage
RUN	[RETURN]	AGE ET SALAIRE?
64,8000		AGE ET SALAIRE? 64,8000
[RETURN]	[RETURN]	VOTRE REDUCTION EST
[RETURN]		F30.00

2.2.15 LET

LET affecte une valeur ou la valeur d'une expression à une variable. Son usage est facultatif.

Syntaxe

[LET] <variable> = <valeur>

Exemple :

Les exemples suivants ont la même signification

```
10 LET D$="BONJOUR"        10D$="BONJOUR"
20 LET A=100               20 A=100
30 LET B=8^2               30 B=8^2
40 LET SOM=A+B             SOM=A+B
```

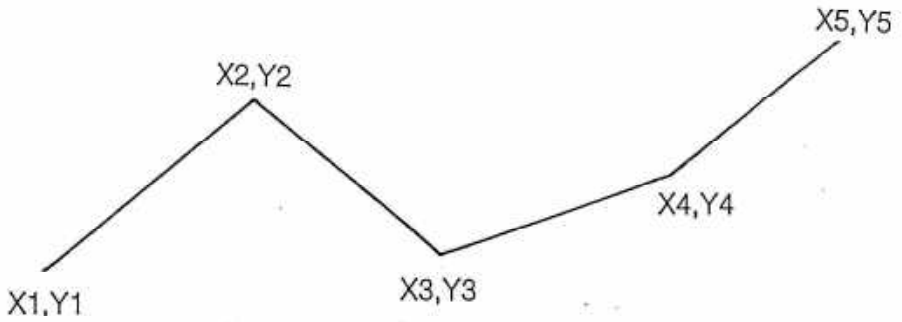
2.2.16 LINE

La commande LINE imprime une ligne reliant deux points spécifiés. Facultativement, la couleur peut être spécifiée.

Syntaxe

LINE [(X1,Y1)] –(X2,Y2) [... –(X7,Y)],
<type de ligne> [,type de couleur]

Exemple :



Note :
Si (X1,Y1) est omis. Le point d'origine sera la coordonnée. Les types de lignes varient entre 0 à 15. 0 désigne une ligne solide et 15 est la ligne pointillée. X et Y peut être au nombre maximum de 7. Les types de couleurs varient de 0 à 3.

2.2.17 MOVE

MOVE déplace le stylo vers la coordonnée spécifiée avec le stylo dans la position dégagée.

Syntaxe

MOVE [(X,Y)]

Exemple :

MOVE (100,100)

Note :
Si (X,Y) est omis, le déplacement sera vers le point d'origine.

2.2.18 MOTOR

MOTOR contrôle le moteur du magnéto-cassette. Pour un fonctionnement correct de MOTOR, raccordez les câbles du magnéto-cassette correctement et mettez en marche sa touche PLAY. Connectez les trois fils de couleur du câble de raccordement comme suit :

COULEUR	POSITION
Noire	Prise télécommande
Grise	Prise écouteur
Rouge	Prise microphone

Spécifiez par un "0" pour arrêter le moteur. Toute valeur supérieure à "0" mettra le moteur en marche. MOTOR, utilisé sans sa spécification, met le moteur en marche ou l'arrête dépendant de son dernier état.

Syntaxe

MOTOR [<paramètre>]

Exemples :

MOTOR 1	Mettre en marche le moteur
MOTOR 0	Arrête le moteur

2.2.19 ON ERROR GOTO

ON ERROR GOTO attrape toutes les erreurs qui peuvent se produire pendant l'exécution du programme. Lorsque une erreur se produit, contrôle rend la main à la routine de traitement d'erreur spécifiée par le GOTO. L'instruction ON ERROR GOTO Ø annule l'effet du routine de traitement d'erreur, et les erreurs postérieures s'afficheront normalement. Si l'instruction ON ERROR GOTO Ø est exécutée dans un routine de traitement d'erreur, le message d'erreur système approprié est affiché et contrôle retourne en mode direct.

Syntaxe

ON ERROR GOTO <numéro de ligne>

Exemple :
```
10 ON ERROR GOTO 100
20 FOO I=1 TO 3
30 PRINT I
40 NEXT I : STOP
100 PRINT "VOUS AVEZ COMMIS UNE ERREUR"
110 END
RUN
VOUS AVEZ COMMIS UNE ERREUR
```

2.2.20 ON GOSUB/ON GOTO

ON GOSUB/ON GOTO branche l'ordre d'exécution sur un numéro parmis plusieurs numéros de lignes. Le branchement se fait par l'ordre de spécification des numéros des lignes. Par exemples, si la valeur de l'expression est 3, contrôle passera sur le numéro de ligne spécifié par la troisième valeur sur la liste des numéros.

Syntaxe

ON <expression> { GOSUB / GOTO. } <numéro de ligne> [,numéro de ligne]...

Exemple :

```
10 PRINT "ENTREZ 1 - POUR SAISIR 2 - POUR CONSULTER 3 -
   POUR IMPRIMER 4 - POUR FIN DE TRAVAIL : " : INPUT A
20 ON A GOTO 50 , 70 , 90 , 100
.
.
.
```

Un 2 entré pour la variable A branchera le contrôle sur la ligne 70.
Un 3 branchera le contrôle sur la ligne 90. Un nombre négatif entré pour A provoquera le message d'erreur "ILLEGAL FUNCTION CALL" sur la ligne 20. Si une valeur supérieure au nombre maximum des numéros sur la liste est entrée, par exemple la valeur 4 dans notre exemple, branchera contrôle sur la ligne qui suit l'instruction ON GOSUB/ON GOTO.

2.2.21 OPTION BASE

OPTION BASE spécifie la valeur minimum d'indice des tableaux indicés.

Syntaxe :

OPTION BASE n (n = 1 ou 0)

Note :
Si l'OPTION BASE est omise, la valeur minimum sera 0. Quand OPTION BASE 1 est exécutée, la valeur minimum d'indice est égale à 1. Cette commande peut être utilisée seulement une fois avant le DIM.

2.2.22 ORIGIN

ORIGIN fixe l'origine dans les commandes BOX, CIRCLE, LINE, etc.

Syntaxe :

ORIGIN

Exemple :

```
20 ORIGIN
30 CIRCLE (10,20), 10, 3
```

2.2.23 POKE

POKE écrit un octet d'information dans la mémoire. L'entier «I» doit être une adresse mémoire comprise entre 0 et 65535. La valeur de «J» est la donnée à être écrite en mémoire, qui doit se situer entre 0 et 255. I et J peuvent être en hexadécimal, octal ou décimal. Utilisez le PEEK pour lire un octet particulier d'information de la mémoire. Les limites de l'entier pour le PEEK sont identiques à celles pour le POKE (voir 3.3.3 PEEK).

Syntaxe

```
POKE I,J
PEEK (I)
```

Exemple

```
10 POKE &H5AOO, &HFF
20 PRINT PEEK (&H5AOO)
RUN
62
```

2.2.24 RANDOMIZE

RANDOMIZE génère une série de nombres aléatoires.

Syntaxe

```
RANDOMIZE [formule]
```

Exemple

```
10 RANDOMIZE
20 FOR I=1 TO 4
30 PRINT RND;
40 NEXT I
50 PRINT
```

Opération au clavier	Affichage
RUN [RETURN]	(0-65529)?
3	(0-65529)? 3
[RETURN]	.88598 .484668 .586328 .119452
RUN [RETURN]	(0-65529)?
4	(0-65529)? 4
[RETURN]	.803506 .162462 .929364 .292443

Si la formule est omise, l'exécution s'arrête et l'appareil vous demandera d'entrer une valeur comprise entre 0 et 65529.

2.2.25 READ

READ attribue les données d'une instruction DATA à ses variables une par une. Elle peut contenir des variables numériques ou chaîne à caractères, mais elles doivent être compatibles aux types des constantes listées dans l'instruction DATA. Une seule instruction READ peut avoir accès sur plusieurs instructions DATA et plusieurs instructions READ peuvent avoir accès sur une seule instruction DATA. Si le nombre de variables dans READ dépasse le nombre de valeurs constantes dans DATA, le message «OUT OF DATA» s'affichera. Si le nombre des constantes de DATA dépasse celui des variables du READ, les constantes en extra sont ignorées.

Syntaxe

READ <variable , variable , ...>

Exemple

```
10 FOR I=1 TO 5
20 READ A
30 PRINT A;
40 NEXT I
50 DATA 1,2,3,4,5,7
RUN
1 2 3 4 5
```

2.2.26 RLINE (Ligne Relative)

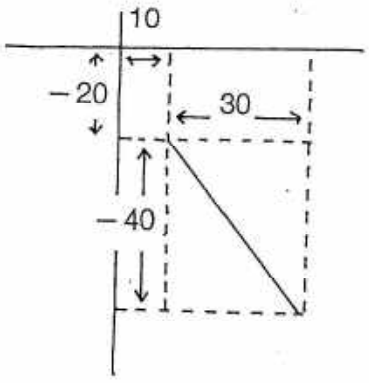
RLINE dessine une ligne sur l'imprimante dans le type et couleur spécifiées.

Syntaxe

RLINE [(X1,Y1)]–(X2,Y2) [...–(X7,Y7)] [,type de ligne, couleur]]

Exemple

```
10 RLINE (10,–20)–(30,–40) , 0 ,1
RUN
```



ligne pleine bleue

Note :

Dans l'Instruction LINE, la position de départ par défaut est l'origine spécifiée, mais dans RLINE la valeur par défaut est la position actuelle du stylo.

2.2.27 REM ou '

REM est une instruction non-exécutable utilisée pour entrer des commentaires dans le programme. Les commentaires peuvent suivre une ligne de programme si vous précédez les commentaires par un ('). L'apostrophe peut être utilisé comme une abréviation de REM.

Syntaxe

REM <commentaires>

Exemple

```
120 REM CALCUL VITESSE MOYENNE
130 FOR I=1 TO 20
140 SUM=SUM+V(I)
```

ou

```
120 FOR I=1 TO 20 'CALCUL VITESSE MOYENNE
130 SUM=SUM+V(I)
140 NEXT I
```

2.2.28 RESTORE

RESTORE libère les éléments d'une instruction DATA pour qu'ils puissent être lus de nouveau par une instruction READ. Si un numéro de ligne est spécifié, l'instruction READ lira les éléments de cette ligne de DATA.

Syntaxe

RESTORE [<numéro de ligne>]

Exemple :

```
10 READ A,B,C
20 RESTORE
30 READ D,E,F
40 RESTORE 90
50 READ G,H,I
60 PRINT A;B;C;D;E;F;G;H;I
70 DATA 1,2,3,4,5
80 DATA 6,7,8,9,10
90 DATA 11,12,13,14,15
RUN
1 2 3 1 2 3 11 12 13
```

2.2.29 RESET

RESET est utilisé pour remettre à l'état initial la position du stylo de l'imprimante. Quand elle est exécutée, le stylo sera déplacé vers l'extrême gauche de l'imprimante sans faire de frappe. L'imprimante va ensuite retourner au mode caractère et fixe les coordonnées de l'origine à la nouvelle position.

Syntaxe

RESET

Exemple

```
10 COLOR 1
20 MOVE(10,20)
30 RLINE(10,-20)-(30,10),0
40 RESET
50 PRINT "TITRE:";X$
```

2.2.30 RESUME

Utilisez RESUME pour continuer l'exécution du programme après avoir traité une erreur par ON ERROR GOTO. Ses options sont les suivantes :

RESUME [0] - reprend l'exécution sur l'instruction qui provoquait l'erreur.

RESUME NEXT - reprend l'exécution sur la ligne qui suit immédiatement la ligne faussée.

RESUME numéro de ligne - reprend l'exécution sur la ligne spécifiée.

Syntaxe

```
RESUME [0]
RESUME NEXT
RESUME <numéro de ligne>
```

Exemple

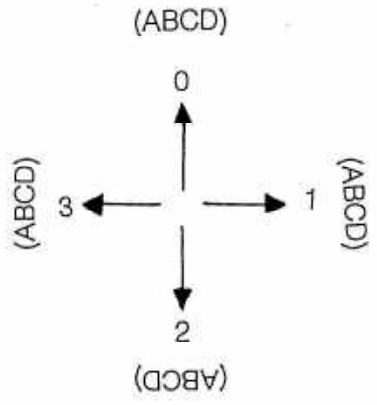
```
10 EN ERROR GOTO 250
        :
        :
250 IF ERR=230 THEN PRINT "RESSAYEZ"
260 IF ERL=90 THEN RESUME 80
```

2.2.31 ROTATE

ROTATE désigne 0 à 3 pour l'inclination des caractères.

Syntaxe

```
ROTATE <entier>
```



Exemple:

```
10 MOVE (100,100)
20 ROTATE 1
30 LPRINT "ABCD"
RUN
```

2.2.32 STOP

STOP arrête l'exécution du programme et retourne en mode direct. Vous pouvez utiliser STOP à n'importe quel endroit dans le programme. Quant STOP est rencontré, le message «BREAK IN LINE nnn» s'affiche avec nnn comme numéro de ligne d'instruction. Pour reprendre l'exécution du programme, tapez simplement «CONT» en mode direct.

Syntaxe

STOP

Exemple :

	Opération	Affichage
10 INPUT A , B , C	RUN	?
20 K=A^2 5.3 : L = B^3/.26	1,2,3 RETURN	BREAK IN 30
30 STOP	PRINT L RETURN	30.7692
40 M = C*K + 100 : PRINT M	RETURN	
	CONT RETURN	115.9

2.2.33 SCALE

La commande SCALE est utilisée pour fixer la taille des caractères. La valeur de SCALE est mise à zéro au départ lorsque l'appareil est mis en marche. Les tailles peuvent aller de 0 qui est la plus petite jusqu'à 15 qui est la plus large.

Syntaxe

SCALE <entier>

Exemple :

```
10 SCALE 0
20 LPRINT «ABCD»
```

2.2.34 TRON/TROFF

TRON trace l'exécution du programme. Quand elle est exécutée, soit en mode direct ou en mode indirect, les numéros des lignes exécutées seront affichés ou imprimés. TROFF (ou NEW) annule l'effet du TRON. Cette disponibilité de trace est utile pour la mise au point des programmes. Elle vous permet de suivre l'exécution ligne par ligne.

Exemple :	Opération	Affichage
10 I = 1	TRON RETURN	
20 J = 2	RUN RETURN	<10>
30 K = 3	RETURN	<20>
40 PRINT I;J;K	RETURN	<30>
50 END	RETURN	<40>
	RETURN	1 2 3
	RETURN	<50>

2.3 LES INSTRUCTIONS D'ENTREE/SORTIE

2.3.1 INPUT

INPUT accepte les données entrées au clavier pendant l'exécution du programme. Quand elle est rencontrée, le système vous demande d'entrer la donnée en affichant le point d'interrogation (?).

Les données individuelles doivent être séparées par des virgules et doivent être au même nombre que le nombre des variables listées après l'instruction INPUT. Si pas assez de données sont entrées, le système affiche "??" et attend les entrées. Si au contraire trop de données sont entrées, le message "EXTRA IGNORED" s'affiche et l'exécution continuera. Les entrées des données doivent correspondre aux mêmes types des variables.

Vous pouvez aussi ajouter une remarque avec l'instruction INPUT pour aider l'entrée des données. Dans ce cas, vous aurez le point d'interrogation après l'affichage de la remarque.

Syntaxe

INPUT ["remarque";] variable , variable ....

Exemples

```
10 INPUT X
20 PRINT X "RACINE CARRE = ";X^2
30 END
```

Opération	Affichage
RUN [RETURN]	?
5 [RETURN]	5 RACINE CARRE = 25

```
10 PI = 3.14
20 INPUT "QUEL EST LE RAYON";R
30 A = = PI*R^2
40 PRINT "L'AIRE DU CERCLE = ";A
50 PRINT
60 GOTO 20
```

Opération	Affichage
RUN [RETURN]	QUEL EST LE RAYON?
7.4 [RETURN]	L'AIRE DU CERCLE = 171.946

2.3.2 INPUT#

INPUT# est utilisée pour lire un enregistrement d'un fichier séquentiel préalablement créé. Elle charge ses variables avec les éléments appropriés de l'enregistrement. Quand un numéro de fichier de "—1" est spécifié, l'appareil lira un enregistrement d'un fichier de la bande cassette. Les variables dans INPUT# doivent correspondre à chaque variable de l'instruction PRINT#.

Syntaxe

INPUT# numéro de fichier, <variable , variable,...>

Exemple:

100 INPUT# —1, A$, X

2.3.3 LOCATE

LOCATE déplace le curseur sur la position spécifiée de l'affichage.

Syntaxe

Locate <entier>

Exemple :

```
10 LOCATE 10
20 PRINT "BONJOUR"
```

Note :

l'entier peut avoir une valeur entre 0 et 255. Lorsque l'entier est supérieur à 48 (affichage des fonctions), le curseur est revenu à zéro.

2.3.4. LPRINT

LPRINT est similaire à PRINT sauf l'impression se fait à l'imprimante.

Syntaxe

LPRINT "<remarque> " [,<élément>]...

Après les commandes – BOX, CIRCLE, COLOR, RLINE, LINE, MOVE, ORIGIN, ROTATE, SCALE -- un RESET doit être fait pour l'impression sur papier.

2.3.5 PRINT

PRINT affiche les valeurs des constantes numériques et de chaîne à caractères, et des expressions. L'endroit où ces valeurs seront affichées est déterminé par la ponctuation utilisée pour séparer les élements à afficher. Si les éléments sont séparés par une virgule (,) chaque valeur sera affichée sur la même ligne avec une intervalle de 18 caractères entre le début de chacune d'elle. Si le (;) est utilisé, les valeurs seront concatenées et affichées sur la même ligne. Une instruction PRINT n'ayant ni le (,) ni le (;) à sa fin, terminera avec un retour chariot.

Le point d'interrogation (?) peut être utilisé comme une abréviation de PRINT.

Syntaxe

PRINT "remarque" ; <élément> [,(ou ;)élément] [,(ou ;)élément]...

Exemple 1:

```
10 FOR I=1 TO 6
20 PRINT I
30 NEXT I
```

Exemple 2:

```
10 FOR I=1 TO 6
20 PRINT I;I+1;
30 NEXT I
```

Opération	Affichage
RUN RETURN	1
RETURN	2
RETURN	3
RETURN	4

Opération

RUN RETURN

Affichage

1 2 2 3 3 4 4 5 5 6 6 7

Note:
Un PRINT utilisé tout seul fera un saut de ligne.

2.3.6. PRINT#

La commande PRINT# est utilisée pour écrire un enregistrement sur un fichier séquentiel. Un numéro de fichier de " − 1" devrait être utilisé pour écrire sur une bande cassette.

Syntaxe

PRINT# numéro de fichier, <variable>...

Exemple:

```
100 A% = 10 : B$ = "XYZ"
110 PRINT# −1 , A% , B$
```

2.3.7 PRINT USING

Cette commande permet à l'utilisateur de formatter ses sorties. Les codes de formats, se composant de symboles spéciaux, sont enfermés entre des guillemets et indiquent à l'appareil comment afficher l'information désirée.

2.3.7.1 Les données en chaîne à caractères.

Il y a deux caractères de formattage de texte disponibles :

!	spécifie que seul le premier caractère d'une chaîne donnée, sera affiché.
"& (espaces) &"	affiche le nombre spécifié de caractères d'une chaîne à caractères, déterminé par 2+le nombre d'espaces situant entre les deux "&".

Syntaxe 1

PRINT USING "caractère de formattage";variable chaîne; variable chaîne...

Exemple:

```
10 A$ = "BONNE"
20 B$ = "JOURNEE"
30 PRINT USING "!";A$;B$
40 PRINT USING "!& &";A$;B$
50 PRINT USING "&(5 espaces)&";A$;B$;"!!"
```

Opération	Affichage
RUN [RETURN]	BJ
[RETURN]	BJOU
[RETURN]	BONNE JOURNEE!!

2.3.7.2 Les données numériques

Il existe plusieurs caractères de formattage pour les valeurs numériques.

Syntaxe

PRINT USING "chaîne de format";variable;variable...

Exemple:

\# Représente chaque chiffre d'une zone. Le point décimal peut être mis à toute position de la zone. Des zéros seront placés à la droite du point décimal si cette place n'est pas remplie et la place non-remplie à gauche du point sera remplie par des espaces. Les nombres sont arrondis quand il le faut.

PRINT USING "# # # . # #";1. , .2 , 3.456
donnera
1.00 0.20 3.46

\+ Affiche le signe plus (+) ou moins (−) quand il est placé au début ou à la fin de la chaîne de format.

\− Imprime les valeurs négatives avec le symbole (−) à la fin s'il est spécifié à la fin de la chaîne de format.

PRINT USING "+ # # # . # #";111.22 , 222.33 , −444.44
donnera
+111.22 +222.33 −444.44

PRINT USING "# # # . # # −";111.22 , 222.33 , −444.44
donnera
111.22 222.33 444.44−

** Remplit tous les espaces en tête par des astérisques. Elles représentent aussi deux positions de chiffres.
PRINT USING "**###.##"; 12.39 , −0.9 , 765.1 , 1.0
donnera
12.39 −0.90 *765.10 ***1.00

$$ Idem à ** mais elles impriment le $ à la place du *.

**$ Combine les deux fonctions ci-dessus.

, Place une virgule après chaque groupe de trois chiffres à la gauche du point décimal. Quand elle est placée à la fin de la chaîne de format, la virgule sera affichée à sa position.
PRINT USING "#####,.##"; 12345.5 donnera 12,345.50
PRINT USING "######.##,"; 12345.5 donnera 12345.50,

^^^^ Spécifie la notation exponentielle quand elle est placée à la fin de la chaîne de format.
PRINT USING "###. ##^^^^"; 123.56 donnera 12.36E + 01

% Elle est affichée par le système lorsque une valeur est supérieure au format spécifié.
PRINT USING "##.##"; 132.22 donnera %123.22

2.3.8. PAUSE

La commande PAUSE est similaire à la commande PRINT. Elle affiche la valeur pour une période de temps fixée par l'instructeur WAIT.

Syntaxe

PAUSE [USING <symbole de format> ;] expression

Exemple:

```
10 A=5 : B$="123"
20 WAIT A*6
30 PAUSE B$
RUN
123
```

Dans l'exemple ci-dessus, le 123 est affiché pour 3 secondes. La durée du temps est calculée par le nombre désigné dans WAIT, plus 10.

2.3.9 WAIT

WAIT est utilisée avec PAUSE, pour spécifier le temps d'affichage de la valeur d'une expression.

Syntaxe

WAIT <entier>

2.4. LES INSTRUCTIONS DE FONCTION

2.4.1. KEY

KEY stocke au maximum 15 caractères de chaîne ou de contrôle dans les touches de fonction. Les caractères qui ne peuvent pas être entrés au clavier seront remplacés par l'Instruction CHR$(n). Ces CHR$(n)s sont ajoutées à la suite par le (+).

Syntaxe

KEY <numéro de la touche de fonction> , <chaîne à caractères>

Exemple:

KEY 5, "RETURN" + CHR$(13)

2.4.2 KEY LIST

KEY LIST affiche les contenus des touches de fonction.

CHAPITRE 3 LES FONCTIONS

Les fonctions peuvent être appelées de tous les programmes sans la nécessité de les redéfinir.

Les arguments de ces fonctions sont toujours renfermés entre des parenthèses :

X et Y - représente les expressions numériques
I et J - représente les expressions entières
A$ et B$ - représente les expressions en chaîne à caractères

Si vous fournissez une valeur à virgule flottante lorsque un entier est spécifié, l'entier sera le résultat avec troncage de la partie fractionnelle.

3.1. LES FONCTIONS NUMERIQUES

3.1.1. ABS

Cette fonction retourne la valeur absolue de l'expression X.

Syntaxe

ABS (X)

Exemple:

```
PRINT ABS(7 * (-5))
35
```

3.1.2 ATN

Cette fonction retourne l'arc-tangent de X en radians. Le résultat est dans les limites de $-\pi/2$ à $\pi/2$. L'expression X peut être de n'importe quel type numérique, mais l'évaluation est toujours faite en simple précision.

Syntaxe

ATN(X)

Exemple:

```
10 INPUT X
20 PRINT ATN (X)
```

Opération		Affichage
RUN	RETURN	?
3	RETURN	1.24905

3.1.3 COS

COS retourne le cosinus de X en radians. Le calcul de COS (X) se fait en simple précision.

Syntaxe

COS(X)

Exemple:

```
10 X=2 * COS(.4)
20 PRINT X
RUN
1.84212
```

3.1.4. EXP

EXP retourne le "e" à la puissance de X, qui doit être inférieur à 87.3366. Si la capacité est dépassée, le message "OVERFLOW" s'affichera.

Syntaxe

EXP(X)

Exemple:

```
10 A=5
20 PRINT EXP (A-1)
RUN
54.5982
```

3.1.5 FIX

FIX retourne la partie entière de X. FIX(X) est équivalente à SGN(X) * INT(ABS(X)). La différence principale entre FIX et INT est que FIX retourne un entier plus grand que X lorsque celui-ci est négatif.

Syntaxe

FIX (X)

Exemple:

```
PRINT FIX(58.75)   PRINT FIX(-58.75)
58                 -58
```

3.1.6 INT

INT renvoie la plus grande valeur entière qui ne dépasse pas X.

Syntaxe

```
INT(X)
```

Exemple

```
PRINT INT(99.89)
99
PRINT INT(-12.11)
-13
```

3.1.7 LOG

LOG renvoie le logarithme naturel de X. Cette dernière doit être supérieure à zéro.

Syntaxe

```
LOG(X)
```

Exemple

```
PRINT LOG(45/7)
1.86075
```

3.1.8 RND

RND renvoie un nombre aléatoire entre 0 et 1. La valeur de I variera ce nombre renvoyé comme suit:

I = 0	génère le même nombre pour une valeur donnée de I
I >0	génère des nombres entre 0 et 1

Syntaxe

```
RND(1)
```

Exemple

```
10 FOR I=1 TO 5
20 PRINT INT(RND(1)*100);
30 NEXT I
RUN
24 30 32 51 5
```

3.1.9 SGN

SGN renvoie le signe de la valeur de X.

Syntaxe

```
SGN(X)
```

Si X>0, SGN(X) renvoie 1
Si X=0, SGN(X) renvoie 0
Si X<0, SGN(X) renvoie -1

Exemple

```
ON SGN(X) +2 GOTO 100 , 200 , 300
```

Contrôle branchera sur 100 si X est négative, à 200 si X est 0, et à 300 si X est positive.

3.1.10 SIN

SIN renvoie le sinus de X en radians. SIN(X) est calculée en simple précision.

Syntaxe

```
SIN(X)
```

Exemple

```
PRINT SIN(9)
.412118
```

3.1.11 SQR

SQR renvoie la racine carrée de X.

Syntaxe

```
SQR(X)
```

Exemple

```
10 FOR X=10 TO 25 STEP 5
20 PRINT X , SQR(X)
30 NEXT X
```

Opération	Affichage
RUN [RETURN]	10 3.16228
[RETURN]	15 3.87298
[RETURN]	20 4.47214
[RETURN]	25 5

3.1.12 TAN

TAN renvoie le tangent de X en radians. TAN(X) est calculée en simple précision. Si TAN dépasse en capacité, le message "OVERFLOW" s'affichera et l'exécution arrêtée.

Syntaxe

```
TAN(X)
```

Exemple

```
10 G=5
20 Y=G*TAN(6) /2
30 PRINT Y
RUN
-.727516.
```

3.1.13 TAB

TAB renvoie des espaces jusqu'à la position I. Si la position actuelle d'impression est au-delà de I, TAB n'aura pas d'effet. La position 0 est celle de l'extrême gauche. I doit être entre 0 et 255. TAB peut être utilisée seulement dans PRINT, LPRINT et PAUSE.

Syntaxe

TAB(I)

Exemple

```
10 PRINT "NOM" TAB(15) "PRENOM":PRINT
RUN
NOM             PRENOM
    <---------->
    15 espacements
```

3.2 LES FONCTIONS DE CARACTERES

3.2.1 ASC

ASC renvoie le code ASCII du premier caractère de la chaîne X$.

Syntaxe

ASC(X$)

Exemple

```
10 X$ = "TEST"
20 PRINT ASC(X$)
RUN
84
```

3.1.2 CHR$

CHR$ renvoie le caractère dont le code ASCII est I. Vous l'utilisez normalement pour imprimer des caractères spéciaux.

Syntaxe

CHR$(I)

Exemple

```
PRINT CHR$(66)
B
```

3.2.3 HEX$

HEX$ renvoie la chaîne à caractères qui représente la valeur hexadécimale de l'argument décimal. X est tronquée à l'entier avant que HEX$(X) soit évaluée.

Syntaxe

HEX$(X)

Exemple

```
10 INPUT X
20 A$ = HEX$(X)
30 PRINT X "DECIMAL EST" A$ "HEXADECIMAL"
RUN
? 32
32 DECIMAL EST 20 HEXADECIMAL
```

3.2.4 LEFT$

LEFT$ renvoie la chaîne de I caractères à partir de la gauche DE X$. I doit se situer entre 0 et 255. Si I est supérieure à LEN(X$), la chaîne de caractères complète de X$ sera renvoyée. Si I = 0, la chaîne nulle (longueur 0) est renvoyée.

Syntaxe

```
10 A$ = "CMT BASIC"
20 B$ = LEFT$(A$,5)
30 PRINT B$
RUN
CMT B
```

3.2.5 LEN

LEN renvoie le nombre de caractères de X$. Les caractères non-imprimables et les espaces sont comptés aussi.

Syntaxe

```
LEN(X$)
```

Exemple

```
10 X$ = "CMT BASIC"
20 PRINT LEN(X$)
RUN
9
```

3.2.6 MID$

MID$ renvoie une chaîne de J caractères de X$, en commençant par le 1e caractère. I et J doivent se situer entre 0 et 255. Si vous omettez J ou s'il y a moins de J caractères à la droite du Ie caractère, tous les caractères à droite du Ie caractère seront renvoyés. Si I est inférieure à LEN(X$), MID$ renverra une chaîne nulle.

Syntaxe

```
MID$(X$ , I [,J])
```

Exemple

```
10 A$ = "BON"
20 B$ = "JOUR SOIR"
30 PRINT A$;MID$(B$,6,4)
RUN
BONSOIR
```

3.2.7 OCT$

OCT$ renvoie une chaîne de caractères qui représente la valeur octale de l'argument décimal. X est arrondie à un entier avant que OCT$(X) soit évaluée.

Syntaxe

```
OCT$(X)
```

Exemple

```
PRINT OCT$(24)
30
```

3.2.8 RIGHT$

RIGHT$ renvoie une chaîne de I caractères à partir de l'extrême droite de X$. (voir aussi 3.2.4. LEFT$)

Syntaxe

```
RIGHT$(X$,I)
```

Exemple

```
10 X$ = "BONJOUR"
20 PRINT RIGHT$(X$,4)
RUN
JOUR
```

3.2.9 STR$

STR$ renvoie une chaîne qui représente la valeur numérique de X.

Syntaxe

```
STR$(X)
```

Exemple

```
PRINT "$" + STR$(15 + 3)
$18
```

3.2.10 VAL

VAL renvoie la valeur numérique de la chaîne X$. Si le premier caractère de X$ n'est pas + , − , & , ou un caractère, VAL(X$) renvoiera 0.

Syntaxe

```
VAL(X$)
```

Exemple	Opération	Affichage
10 INPUT A$	RUN RETURN	?
20 PRINT VAL(A$) + 32	25 RETURN	57

3.2.3 LES FONCTIONS GENERALES

3.3.1 ERR , ERL

ERR et ERL sont utilisées dans les sous-routines de traitement des erreurs.

Lorsqu'une erreur est signalée, contrôle se branche sur la routine de traitement des erreurs (s'il existe) et le code d'erreur est stocké dans la variable ERR tandis que le numéro de ligne où l'erreur a été causée est stocké dans la variable ERL. Vous utilisez normalement ERR et ERL dans l'instruction IF... THEN pour contrôler le déroulement du traitement des erreurs.

Syntaxe

```
IF ERR = code d'erreur THEN....
IF ERL = numéro de ligne THEN....
```

Exemple

```
10 ON ERROR GOTO 500
.
.
.
.
500 REM routine de traitement des erreurs
510 IF ERR = 4 THEN RESUME 100
520 IF ERL = 150 THEN RESUME 150
530 X = 15
540 RESUME 0
.
.
```

3.3.2 FRE

Les arguments de FRE sont des arguments fictifs. Si l'argument est 0 (numérique), FRE renvoie le nombre d'octets non-utilisés de la mémoire. S'il est une chaîne de caractères, FRE renvoie le nombre d'octets libres de l'espace chaîne.

Syntaxe

```
FRE(0)
FRE(X$)
```

Exemple

```
PRINT FRE(0)
4034
```

3.3.3 PEEK

PEEK renvoie l'octet (entier décimal entre 0 et 255) lu d'une adresse mémoire I, qui doit se situer entre 0 et 65536. PEEK est l'instruction complémentaire de POKE.

Syntaxe

```
PEEK(I)
```

Exemple

```
PRINT PEEK(741)
52
```

3.4 FONCTION ENTREE/SORTIE

3.4.1 INKEY$

Avec INKEY$, un caractère est renvoyé dès qu'une touche clavier est appuyée. La chaîne nulle est renvoyée si aucune touche n'est appuyée.

VI. LES MESSAGES D'ERREURS

Code	Message d'erreur	Signification
1	NEXT without FOR	Les instructions de FOR et NEXT ne correspondent pas. (trop de NEXT).
2	Syntax error	Une ligne est rencontrée avec une séquence incorrecte des caractères (exemple, faute d'orthographe). Dans le cas de INPUT — 1, ce code veut dire que les données sur la cassette contiennent une erreur.
3	RETURN without GOSUB	Les instructions RETURN et GOSUB ne correspondent pas.
4	Out of data	Il n'y a pas d'Instructions DATA pour lire les instructions READ.
5	Illegal function call	Un paramètre qui est dépassé en capacité est introduit dans une fonction arithmétique ou chaîne.
6	Overflow	Le résultat du calcul est trop large en capacité pour être représenté en BASIC.
7	Out of memory	Un programme est trop gros ou Il a trop de variables tableaux indices.
8	Undefined line number	Une référence a été faite sur un numéro de ligne qui n'existe pas.
9	Subscript out of range	Un élément d'un tableau indicé est référencé avec un indice qui est dehors de la dimension définie.
10	Redimensioned array	Deux instructions DIM sont données pour le même tableau.
11	Division by zéro	Division par zéro dans une expression.
12	Illegal direct	Une instruction qui est illégale en mode direct vient d'être exécutée en mode direct.
13	Type mismatch	Incompatibilité entre deux valeurs (entre valeur numérique et chaîne à caractères).

14	Out of string space	Le nombre des variables à caractères dépasse la capacité d'espace chaîne allouée.
15	String too long	Une chaîne de caractères est trop longue (plus de 255 caractères).
16	String formula too complex	Une expression de chaîne à caractères est trop longue ou trop complexe. (Un emboîtement illegal des niveaux de parenthèses, etc...).
17	Can't continue	Une commande CONT non-exécutable est rencontrée. (Le pointeur du programme est détruit, etc...).
18	Undefined User Function	Une fonction d'utilisateur est appelée sans sa définition (par l'instruction DEF).
19	No RESUME	Un sous-routine de traitement d'erreurs est entré, mais l'instruction RESUME n'y est pas présente.
20	RESUME without error	Une instruction RESUME est rencontrée avant l'entrée dans un sous-routine de traitement d'erreurs.
21	Unprintable error	Un message d'erreur qui n'est pas disponible pour l'erreur qui existe.
22	Missing Operand	Une expression contient un opérateur sans opérande qui le suit.
24	Tape read ERROR	Une erreur est rencontrée lors de la lecture de la bande cassette.
25	Bad File Data	Les données du fichier sur mémoire de masse sont mal formattées.
26	Wrong Program Change	Une tentative est faite pour changer le programme après l'exécution de LOCK.

VII. CARACTÈRES DÉFINISSABLES PAR L'UTILISATEUR

Il existe dix caractères (50 octets) à partir de l'adresse hexa FCAE qui sont disponibles pour l'utilisateur. Ils correspondent aux codes ASCII de E0 à E9 et de F0 à F9.

Lorsque les 5 octets à partir de l'adresse FCAE sont chargés par POKE, les caractères graphiques peuvent être affichés sur l'affichage par PRINT et CHR$. Sept bits en code ASCII sont utilisés et le 8e bit est ignoré.

Exemple

```
10 POKE &HFCAE , 1        : POKE &HFCAF , 2
20 POKE &HFCBO , 3        : POKE &HFCB1 , 4
30 POKE &HFCB2 , 5
40 PRINT CHR$ (&HEO)
RUN
```

```
• o • o •
o • • o o
o o o • •
o o o o o
o o o o o
o o o o o
o o o o o
```

Ce caractère sera affiché.

LA DÉFINITION DE LA LARGEUR DE L'IMPRESSION

Valeur Initiale : 18
Adresse : FC34

Si vous voulez changer la largeur autre que celle indiquée ci-dessus, faites-la par POKE. Lorsque les données sont imprimées continuellement par "," dans LPRINT, l'impression peut être faite dans la largeur définie.

VIII. TABLE DES CODES ASCII

Positions des bits supérieurs
b7, b6, b5

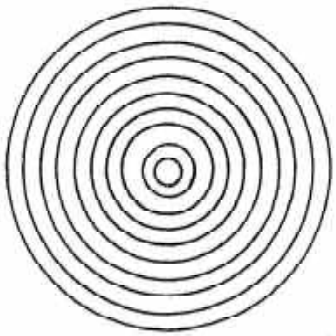
Positions des
bits inférieurs
b4, b3, b2, b1

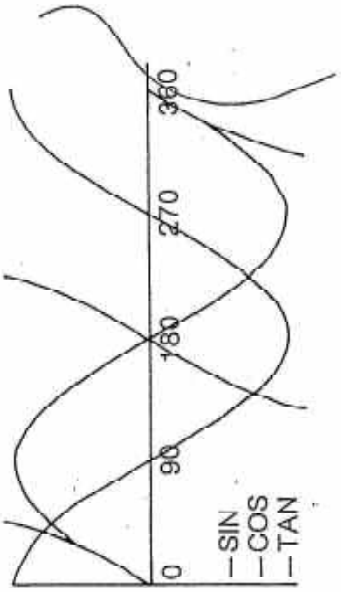
	000	001	010	011	100	101
Hexa decimal	0	1	2	3	4	5
0000	0		SPACE	0	@	P
0001	1		!	1	A	Q
0010	2		"	2	B	R
0011	3		#	3	C	S
0100	4		$	4	D	T
0101	5		%	5	E	U
0110	6		&	6	F	V
0111	7		'	7	G	W
1000	8		(	8	H	X
1001	9		)	9	I	Y
1010	A		*	:	J	Z
1011	B		+	;	K	[
1100	C		,	<	L	¥
1101	D		−	=	M	]
1110	E		.	>	N	^
1111	F		/	?	O	—

IX. EXEMPLES DES PROGRAMMES EN BASIC

Exemple 1

DÉMONSTRATION
TPC—8300






```
10 SCALE 1:MOVE(0,
0)
20 COLOR 1
30 LPRINT "DEMONST
RATION"
40 SCALE 2:MOVE(0,
-50)
50 COLOR 2
60 LPRINT "TPC-830
0"
70 RESET
80 ' ***CIRCLE***
90 MOVE (107,-120)
:ORIGIN
100 FOR R=10 TO 10
0 STEP 10
110 GOSUB 160
120 CIRCLE(0,0),R,
C
130 NEXT
140 MOVE(-107,-110
):RESET
150 GOTO 180
160 C=C+1:IF C>3 T
HEN C=0
170 RETURN
180 '*SIN*COS*TAN*

190 MOVE(0,-360):R
ESET:SCALE 1
200 ROTATE 3:MOVE(
107,0):ORIGIN
210 LINE(-105,0)-(
105,0),0,0:LINE(0,
0)-(0,375)
220 GOSUB 460
230 FOR D=1 T0 3
240 K=57.2958:X1=0
:Y1=0
250 FOR Y2=0 TO 36
0 STEP 10
260 ON D GOTO 340,
390,360
270 LINE(-X1,Y1)-(
-X2,Y2),0,D
280 X1=X2:Y1=Y2
290 NEXT Y2
300 NEXT D
310 GOSUB 520
320 MOVE(0,0): RES
ET
330 GOTO 1000
340 RD=Y2/K:X2=SIN
(RD)*100:GOTO 270
350 RD=Y2/K:X2=COS
(RD)*100:GOTO 270
360 IF Y2=60 GOTO
410
370 IF Y2=240 GOTO
440
380 RD=Y2/K:X2=TAN
(RD)*100:GOTO 270
390 IF Y2=0 THEN X
1=100
400 GOTO 350
410 Y2=120:MOVE(12
0,120)
420 RD=Y2/K:X2=TAN
(RD)*100
430 GOTO 280
440 Y2=300:MOVE(12
0,300)
450 GOTO 420
460 MOVE(20,5):LPR
INT "0"
470 MOVE(20,80):LP
RINT "90"
480 MOVE(20,165):L
PRINT "180"
490 MOVE(20,255):L
PRINT "270"
500 MOVE(20,345):L
PRINT "360"
510 RETURN
520 LINE(50,10)-(5
0,30),0,1:MOVE(58,
40):LPRINT "SIN"
530 LINE(70,10)-(7
0,30),0,2:MOVE(78,
40):LPRINT "COS"
540 LINE(90,10)-(9
0,30),0,3:MOVE(98,
40):LPRINT "TAN"
550 RETURN
```

Exemple 2

Programme pour une mélodie musicale.

```
1000 BEEP 19,2
1010 BEEP 19,2
1020 BEEP 19,2
1030 BEEP 19,2
1040 BEEP 17,2
1050 BEEP 15,2
1060 BEEP 15,2
1070 BEEP 14,2
1080 BEEP 12,2
1090 BEEP 12,2
1100 BEEP 15,2
1110 BEEP 19,2
1120 BEEP 24,2
1130 BEEP 24,2
1140 BEEP 24,2
1150 BEEP 24,2
1160 BEEP 22,2
1170 BEEP 20,2
1180 BEEP 20,2
1190 BEEP 19,2
1200 BEEP 17,2
1210 BEEP 17,2
1220 BEEP 19,2
1230 BEEP 20,2
1240 BEEP 19,2
1250 BEEP 20,2
1260 BEEP 19,2
1270 BEEP 23,2
1280 BEEP 20,2
1290 BEEP 19,2
1300 BEEP 19,2
1310 BEEP 17,2
1320 BEEP 15,2
1330 BEEP 15,2
1340 BEEP 14,2
1350 BEEP 12,2
1360 BEEP 14,2
1370 BEEP 14,2
1380 BEEP 14,2
1390 BEEP 14,2
1400 BEEP 15,2
1410 BEEP 14,2
1420 BEEP 12,2
1430 BEEP 12,2
1440 BEEP 12,2
1450 BEEP 12,2
```

Imprime Au Japon

Impression-Création HAECK - 451.58.77