VG 5000

MICRO ORDINATEUR

MANUEL D'UTILISATION

Sommaire

CHAPITRES

ANNEXES

ATTENTION : *Les exemples de programmes de ce manuel ont été imprimés sur une machine qui intervertit les* Ø *et les* O

Dans les exemples : Ø = *Zéro* O = *lettre*

Sur votre écran : O = *Zéro* Ø = *lettre*

1

Spécifications VG 5000
Micro ordinateur familial 8 bits

Généralités

Micro processeur	Z80A (4 Mhz)
Générateur de caractères	EF9345 EFCIS
Mémoire morte (ROM)	18 K octets
Mémoire vive (RAM)	24 K octets

(dont 13758 octets disponibles pour l'utilisateur).

Langage

Basic Microsoft résident.
81 instructions et fonctions Basic.

INSTRUCTIONS BASIC :

&"..." - ABS - ACTION - AND - ASC - ATN - AUTO - CALL - CHR$ - CLEAR - CLOAD - CONT - COS - CSAVE - CURSORY - CURSORX - DATA -DEF FN - DELIM - DIM - DISPLAY - EG - END - ET - EXP - FRE - FOR ... STEP - NEXT - GOSUB - RETURN - GOTO - GR - IF ... GOTO - IF ... THEN - INIT - INPUT - INT - KEY - LEFT$ - LEN - LET - LIST - LLIST -LOAD - LOG - LPOS - LPRINT - MID$ - NEW - NOT - ON ... GOSUB - ON ... GOTO - OR - PAGE - PEEK - PLAY - POKE - POS - PRINT - READ - REM - RENUM - RESTORE - RIGHT$ - RND - RUN - SAVE - SCREEN - SCROLL - SETEG - SETET - SGN - SIN - SOUND - SPC - SQR - STICKX - STICKY - STOP - STORE - STR$ - TAB - TAN - TX - USR - VAL.

Clavier

63 touches type AZERTY à déplacement.
Touche MAJUSCULES - minuscules bloquées.
Touche contrôle [CTRL] permettant l'accès direct aux 33 fonctions BASIC préprogrammées et à 10 lettres accentuées.
Commandes à accès direct : PRINT - LIST - RUN - effacement écran - effacement ligne - effacement caractère - insertion ligne - insertion caractère - arrêt de déroulement de programme.

Affichage

25 lignes x 40 caractères
Matrice caractères 8 x 10
Définition image 80000 points
Couleurs 8
110 caractères ASCII texte
128 caractères graphiques
192 caractères définissables par l'utilisateur.

Son

255 sons programmables
Synthétiseur musical 4 octaves.

Entrée / Sortie

Prise péritélévision DIN 8 broches
Prise magnétophone DIN 8 broches
Prise alimentation DIN 5 broches
Sortie BUS 2 x 25 contacts (bord de carte).

Accessoires livrés avec l'appareil

Alimentation séparée
Cordon péritélévision AG8374
Cordon magnétophone
Manuel d'utilisation.

Accessoires en option

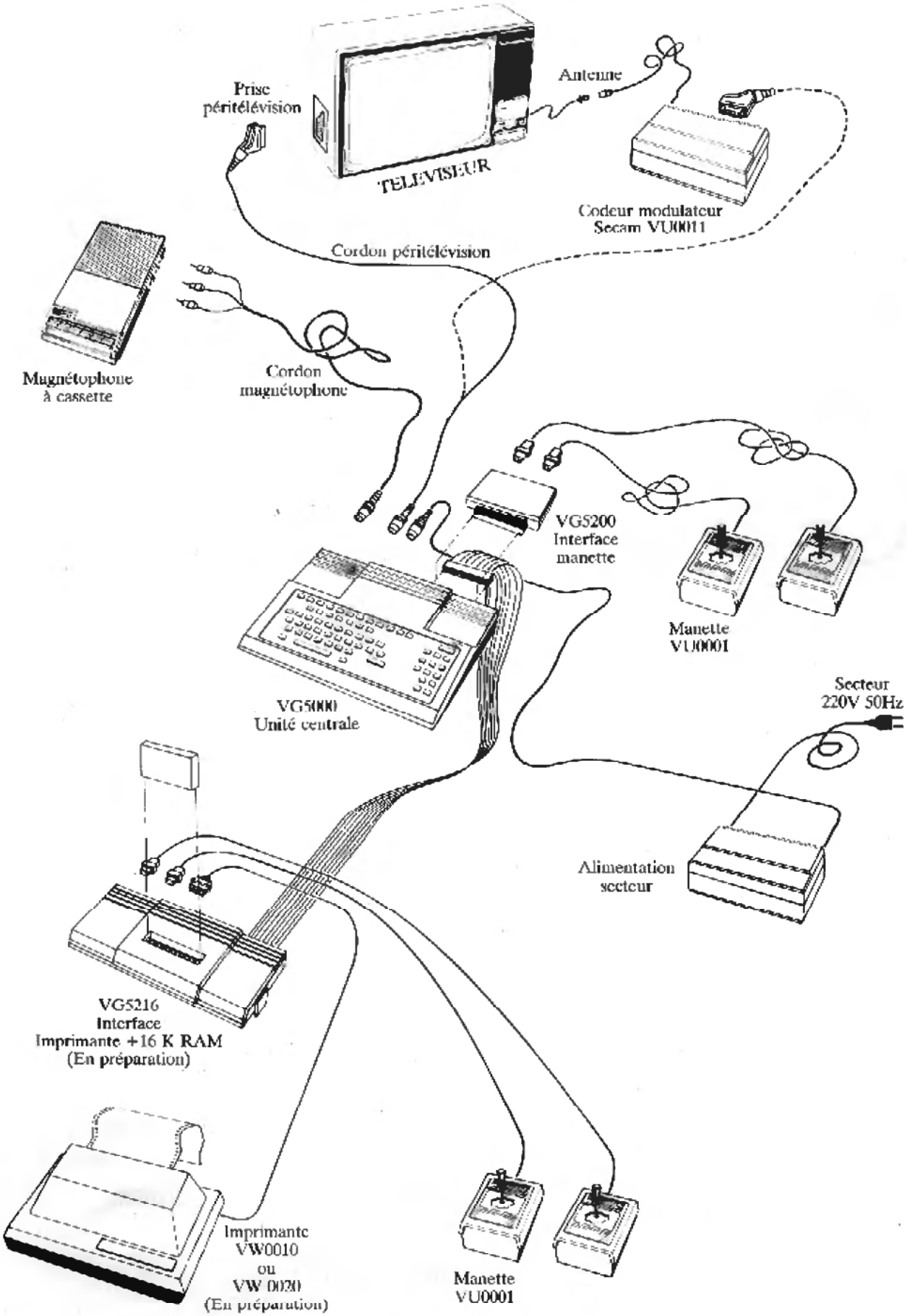
VG5200 Interface manette
VU0001 Manette
VU0011 Codeur modulateur Secam
VG5216 Module d'extension mémoire et périphériques (en préparation)

Caractéristiques de l'alimentation

Secteur 220 V ± 10% 50 Hz
Tensions régulées 12 V = 100 mA
5 V = 1,5 Amp.

2

Configuration VG 5000

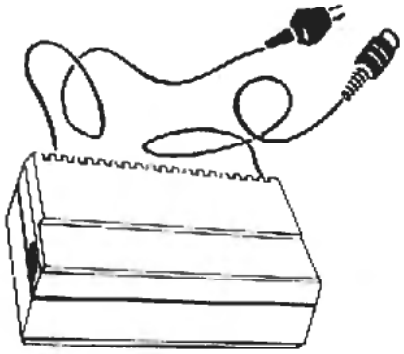


3

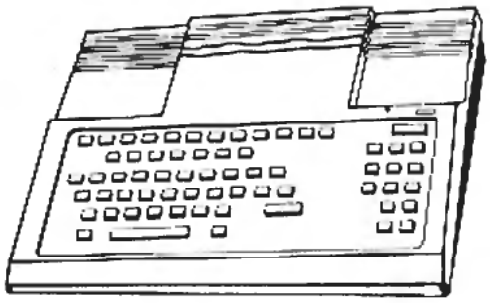
Mise en service du VG5000

Présentation

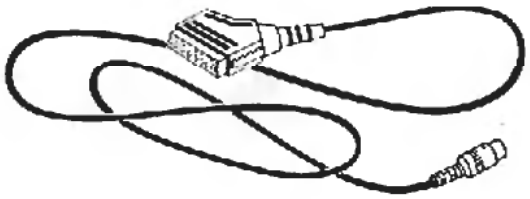
Le VG5000 se compose de quatre parties :



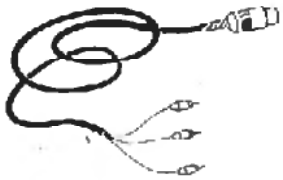
- l'alimentation (fig. 1)



- l'unité centrale (fig. 2)

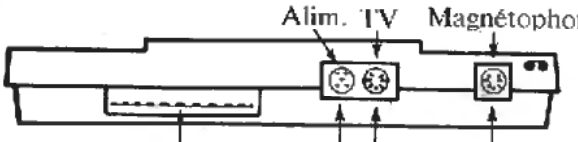


- un câble péritélévision (fig. 3)



- un cordon de raccordement pour magnétophone cassette (fig. 4)

L'unité centrale comprend sur le dessus le clavier et sur la face arrière 4 possibilités de connexions :

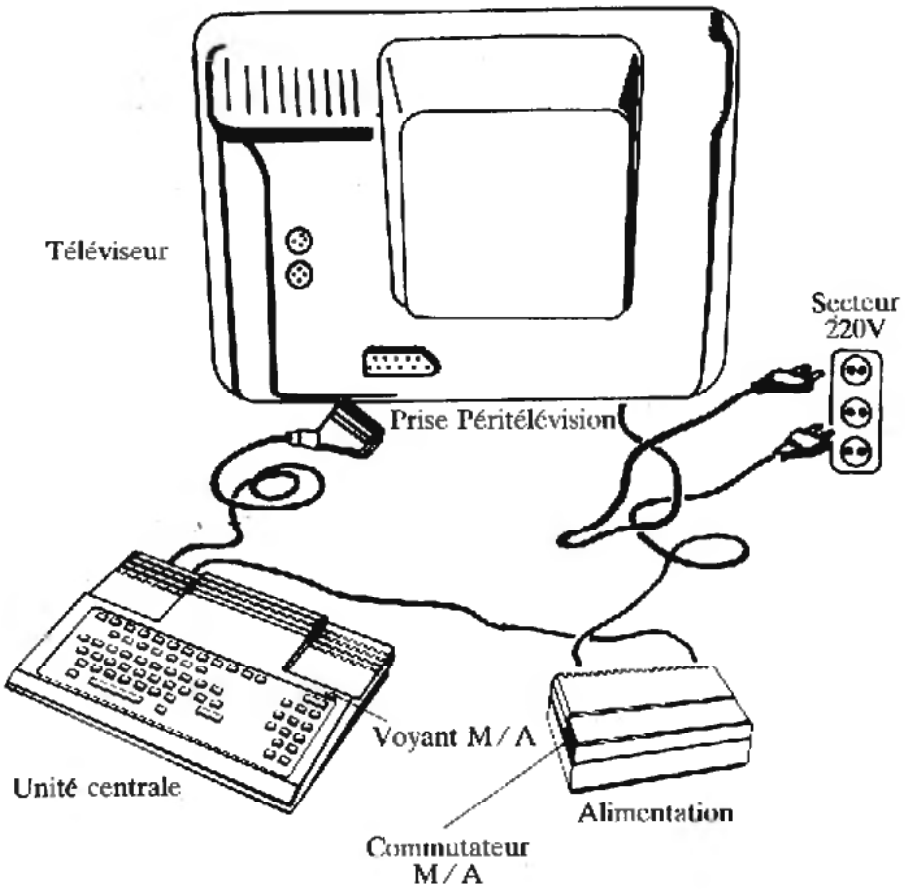


Vue face arrière (fig. 5)

- 1 prise DIN 8 broches pour brancher le magnétophone à cassette.
- 1 prise DIN 8 broches pour connecter le câble péritélévision.
- 1 prise DIN 5 broches pour l'alimentation.
- 1 connecteur 50 contacts pour brancher des périphériques - interface manette - module d'extension, etc.

Les branchements

1. sortir l'unité centrale et l'alimentation de l'emballage
2. placer l'unité centrale face à soi



3. brancher le cordon péritélévision sur la prise 8 broches marquée "TV" sur la figure 5
4. brancher l'autre extrémité du câble péritélévision sur la prise péritélévision de votre téléviseur (si celui-ci ne comporte pas de prise péritélévision - téléviseur acheté en France avant 1980 - procurez-vous chez votre vendeur un CODEUR MODULATEUR SECAM VU0011)
5. brancher le câble muni d'une fiche 5 broches venant de l'alimentation dans la prise 5 broches marquée "ALIM" sur la fig. 5.
6. brancher le câble secteur de l'alimentation sur une prise secteur 220 V 50 Hz.

- Pour le branchement du magnétophone, veuillez vous reporter au chapitre 12.

Mise en service

- Mettez en marche votre téléviseur.
- Actionnez le commutateur marche / arrêt situé sur le côté de l'alimentation

Le voyant situé sur la console s'allume et vous voyez apparaître sur votre écran :

```
"VG 5000 BASIC VERSION 1.0
13758 OCTETS DISPONIBLES
OK!
```

Ceci signifie que votre ordinateur est prêt à fonctionner et que la mémoire disponible est de 13758 *octets*.
Ok! signifie que vous pouvez commencer à travailler.

Si
```
"VG 5000 BASIC VERSION 1.0
 13758 OCTETS DISPONIBLES
 OK!
```
n'apparaît pas, arrêtez l'alimentation, attendre environ 2 secondes puis mettre en marche à nouveau.

D'une manière générale, chaque fois que vous voyez apparaître Ok!
ceci signifie que l'ordinateur a terminé son travail et qu'il attend un ordre de votre part.

Le rectangle rouge qui clignote à gauche de l'écran est le "CURSEUR".
Il indique la prochaine position d'affichage.
- Réglez la lumière de votre téléviseur.
- Réglez le volume sonore du téléviseur.
  VG5000 émet un "Bip" à chaque fois que l'on appuie sur une touche.

4

Conseils et précautions à prendre

Votre VG5000 n'est pas fragile ; mais vous devez tout de même en prendre soin.

- Les touches du clavier sont étudiées pour établir un contact indépendamment de la force avec laquelle on appuie. Il est donc inutile d'appuyer de toutes ses forces.
- Une erreur de manipulation du clavier ne peut endommager votre VG5000 ; tout juste pouvez-vous perdre le programme que vous avez en cours.
- Ne laissez pas couler de liquide sur le clavier.
- Si vous devez nettoyer votre clavier, utilisez une peau de chamois ou un chiffon légèrement humide (Alcool, trichlo, etc. sont déconseillés).
- Ne branchez sur les prises situées à l'arrière de l'appareil que les extensions ou périphériques conseillés par le fabricant. C'est une garantie de bon fonctionnement et de non détérioration de votre appareil.
- Faites attention de ne pas introduire de corps étrangers (pièces de monnaie, trombone, etc.) à l'intérieur de l'appareil, en particulier par la prise 2 x 25 contacts située à l'arrière.

Des orifices de ventilation ont été prévus pour le refroidissement nécessaire de votre ordinateur. Ne mettez rien sur ces orifices afin de maintenir une circulation d'air correcte.

Tenez-le éloigné de toute source de chaleur (poêles, radiateurs ou ensoleillement direct).

Certains animaux domestiques, comme les chats, ont l'habitude de choisir des endroits chauds pour se reposer. Assurez-vous qu'ils ne choisissent pas votre ordinateur ! Ils peuvent perdre des poils qui peuvent pénétrer dans l'appareil et provoquer ainsi un mauvais fonctionnement.

Ne mettez jamais vos doigts sur les points de contact des connecteurs, car cela pourrait provoquer une corrosion exagérée.

Disposez vos câbles de raccordement de telle sorte que personne ne puisse se prendre les pieds dedans.

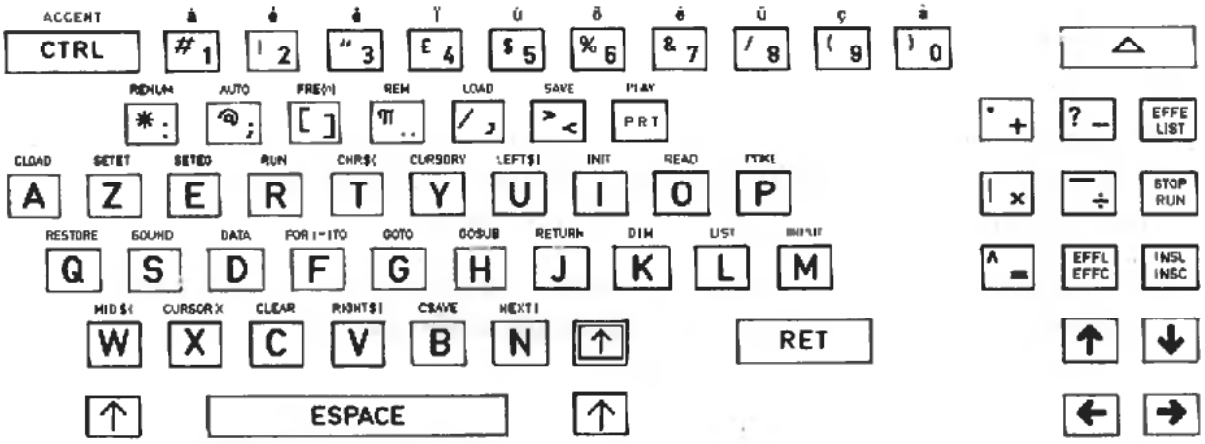
Quand vous débranchez, ne tirez pas sur le câble, mais prenez fermement la fiche elle-même.

Ne branchez aucun accessoire ou périphérique sans avoir pris la précaution de couper l'alimentation de la console.

5

Connaissance du clavier

Avant de vous lancer dans le monde fascinant de la programmation, nous vous conseillons de vous familiariser avec votre clavier.
Celui-ci a été étudié spécialement pour simplifier au maximum l'écriture d'un programme et pour faciliter toutes les corrections, suppressions ou adjonctions qui sont inévitables quand on débute dans la programmation.

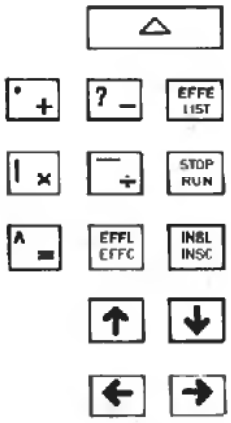


Le clavier que vous avez devant vous comporte en fait 3 claviers. Comme vous pouvez le constater, de nombreuses touches possèdent deux marquages (par exemple, les touches 1 à 0). De plus, certaines ont un marquage situé juste au-dessus d'elles sur le pupitre. Cela signifie que certaines touches peuvent avoir 3 fonctions.

1re fonction : En enfonçant simplement la touche : le caractère marqué en bas à droite de la touche s'inscrit sur l'écran (exemple : 1 - 2 - 3 - : - ; - = , etc.)

2e fonction : En appuyant d'abord sur l'une des deux touches [↑] qui encadrent la barre ESPACE et sur la touche désirée.
Le caractère indiqué à la partie supérieure de la touche s'inscrit alors sur l'écran (exemple : $ - £).
A noter la particularité des touches alphabétiques A - Z... qui ne possèdent qu'un seul marquage. Ces touches, malgré tout, ont deux possibilités : elles donnent soit un caractère majuscule, soit un caractère minuscule.
Il suffit d'appuyer une fois sur la touche (blocage en minuscule) [↑] (à la droite de la touche N).
Lorsqu'on passe en minuscule, le curseur qui est représenté par un rectangle rouge devient un simple trait horizontal clignotant.
Si on appuie une deuxième fois cette touche, on revient en MAJUSCULE (le curseur redevient un rectangle rouge).
Il est possible, lorsqu'on est "bloqué" en minuscule, d'écrire une lettre majuscule en appuyant en même temps sur une des deux touches [↑] qui encadrent la barre ESPACE. Lorsqu'on relâche une de ces deux touches, on revient en minuscule.

3e fonction : Afficher automatiquement une instruction BASIC. L'instruction est indiquée au-dessus de la touche et on y accède en appuyant d'abord sur la touche [CTRL] (Contrôle) et sur la touche désirée.
L'instruction BASIC choisie s'inscrit alors en toutes lettres directement sur l'écran sans qu'on ait à taper le mot en entier au clavier, ce qui permet de gagner un temps considérable et éviter de nombreuses fautes.
La touche [CTRL] permet également d'obtenir les lettres accentuées représentées au-dessus des touches 1 à 0.



A droite du clavier, 14 touches ont été regroupées à part. On y trouve 5 touches avec les signes arithmétiques +, −, ×, ^ (puissance), =, ainsi que le point, les tirets vertical et horizontal, et le point d'interrogation. Les autres touches servent essentiellement à l'édition, c'est-à-dire à la correction de l'affichage sur l'écran.
Comme pour les autres touches, on accède à la fonction supérieure de ces touches en appuyant simultanément sur une des 2 touches [↑] (shift).

4 touches EFF (effacement) et INS (insertion) permettent :

[EFFL EFFC] : Effacement caractère.
En appuyant sur la touche, on efface le caractère placé juste avant le curseur. Toute la partie de ligne située à droite du curseur se décale d'un caractère vers la gauche.

[↑] + [EFFL EFFC] : Effacement ligne.
Toute la partie de ligne située après le curseur est effacée.
Pour effacer une ligne de programme, placer le curseur après le numéro de ligne à effacer - effacer la ligne puis appuyer sur [RET]

[↑] + [EFFE LIST] : Effacement écran.
Tout l'écran est effacé.

[INSL INSC] : Insertion d'un caractère.
On crée un espace pour ajouter éventuellement un caractère là où se trouve placé le curseur.

[↑] + [INSL INSC] : Insertion de ligne.
Pour créer un espace libre entre deux lignes.

Les quatre touches inférieures marquées d'une flèche permettent de déplacer le curseur de haut en bas et de gauche à droite ou vice et versa.

Particularités de quelques touches :

Touche [RET] : Elle sert à valider une commande ou une ligne de programme (une ligne d'écran contient 40 caractères, mais une ligne informatique peut contenir 127 caractères sur plusieurs lignes).

Touche [EFFE LIST] : Elle permet de lister automatiquement le programme à partir de la dernière ligne précédemment listée par l'instruction LIST.

Touches [↑] + [STOP RUN] : Lors du déroulement d'un programme, il peut être nécessaire de stopper celui-ci pour reprendre la main (l'ordinateur exécute complètement l'instruction en cours avant de redonner la main, ce qui peut demander un certain temps, en particulier lorsqu'on utilise l'instruction PLAY). Il est donc nécessaire de maintenir les deux touches en appui jusqu'à ce que le message "Arrêt en xx" (numéro de ligne) apparaisse sur l'écran, indiquant à quel niveau le programme a été arrêté.

Touche [STOP RUN] : Elle permet de lancer directement un programme. Appuyez sur [STOP RUN] puis sur [RET]. Cette touche permet également d'avancer ligne par ligne lorsqu'on est dans la fonction LIST.

Touche [PRT] (Abréviation de PRINT) : Cette touche, située à droite de la deuxième rangée, permet d'écrire l'instruction PRINT directement sur l'écran (On peut également utiliser le [↑] + [? _]).

Touche [π] : Ne donne pas la lettre grecque Pi, mais la valeur 3,14159.

Touche [> <] : Ces signes correspondent aux fonctions "plus grand que" (>) ou "plus petit que" (<).

Touches [CTRL] : Arrête l'exécution d'un programme même lorsque la touche [STOP RUN] est inactive (Fonction PLAY).
Elle remet à zéro toutes les variables du programme. C'est pourquoi elle n'est pas à accès direct, et elle n'est active qu'en appuyant d'abord sur [CTRL] et [△].

Touches [↑] + [^ _] : Le signe ^ signifie "élévation à la puissance" vu sur l'écran ce signe est représenté par ↑ .

Nota : Le signe x sera représenté sur l'écran par *
Le signe ÷ sera représenté sur l'écran par /

Remarques valables pour toutes les touches :
Si vous maintenez une touche appuyée pendant plus d'une seconde, le caractère correspondant à cette touche s'inscrit automatiquement jusqu'à ce que vous relâchiez la touche.

Après avoir pris connaissance des caractéristiques du clavier, nous vous conseillons maintenant de le manipuler.

Essayez ce qui suit :

1. Effacez l'écran en appuyant sur les touches [↑] + [EFFL LIST]
2. Déplacez à l'aide des touches [↑] [→] [↓] [←] le curseur vers le centre de l'écran.
3. Ecrivez votre nom en majuscules, puis votre prénom en minuscules.
4. Descendez d'une ligne [↓]
5. Ramenez le curseur sur la gauche.
6. Ecrivez le nom de votre rue suivi de votre code postal.
7. Insérez le nom de votre ville entre votre rue et le code postal.
   Utilisez la touche [INSL INSC] (insertion de caractère) plusieurs fois de suite.
8. Effacez le code postal [↑] + [EFFL EFFC] (effacement ligne).
9. Ajoutez un espace au milieu du nom de votre ville. Utilisez la touche [ESPACE]
10. Supprimez cet espace avec [EFFL EFFC] (effacement caractère)
11. Descendez d'une ligne sans caractère
12. Appuyez sur [CTRL] + [S] puis tapez 5, 150, 25 - sans oublier les virgules.
13. Appuyez sur [RET] et écoutez.

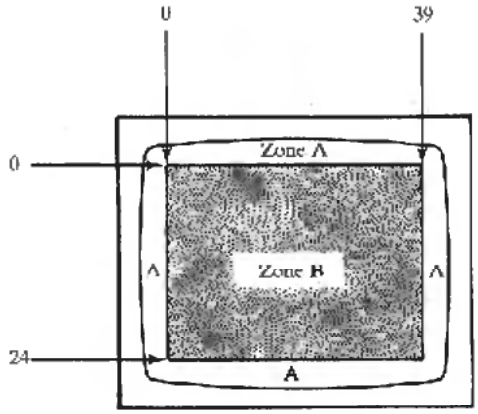
Vous devez avoir maintenant un aperçu de la manipulation de votre clavier. Nous vous conseillons de continuer à vous entraîner en regardant bien ce qui se passe à l'écran.

La bonne connaissance de l'action de chacune des touches du clavier vous sera d'une grande utilité dans la réalisation de vos programmes.

6

L'écran

Le clavier est le moyen que nous avons pour indiquer à l'ordinateur ce que nous voulons qu'il fasse. L'ÉCRAN sert à vérifier ce que nous avons demandé à l'ordinateur de faire et à sa réponse.



L'affichage sur l'écran se décompose en 25 lignes (0 à 24) de 40 caractères (0 à 39).

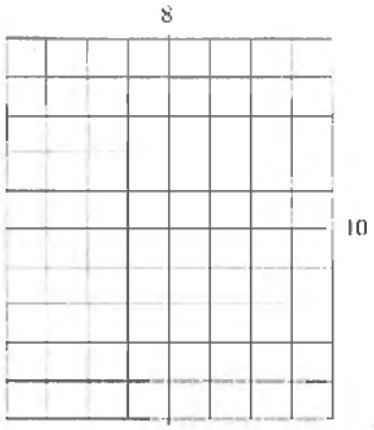
Ce qui permet d'afficher 1000 caractères dans un cadre légèrement inférieur à la surface de l'écran du téléviseur ou du moniteur (1).

Il y a donc deux zones, A et B qui peuvent chacune prendre une teinte différente.

Essayez de taper : INIT 3,4 puis [RET]. La zone A devient bleue alors que la zone B devient jaune.

Toutes les inscriptions graphiques que vous ferez se feront à l'intérieur de la zone B.

Chaque caractère affiché à l'écran peut se décomposer en une matrice de 8 x 10 c'est-à-dire 80 points.



La définition de l'image est donc :

80 points x 1000 caractères = 80000 points.

Nous verrons plus en détail ce point au chapitre "JEU DE CARACTÈRES DÉFINISSABLES PAR L'UTILISATEUR".

(1) NOTA : Il ne faut pas confondre ligne d'écran qui permet d'afficher 40 caractères sur une seule ligne et ligne informatique qui peut comporter jusqu'à 127 caractères et de ce fait utilise plusieurs lignes d'écran.
C'est pourquoi lorsqu'une ligne informatique est plus longue qu'une ligne d'écran, les lignes d'écran suivantes sont soulignées d'un trait.
Si une ligne a plus de 127 caractères, la ligne ne peut être validée tant que les caractères en trop n'ont pas été supprimés.

7

Fonctionnement du VG5000

Votre ordinateur VG5000 peut fonctionner suivant deux modes :

Le mode direct ou calculateur
Le mode indirect ou programme.

Mode direct : En mode direct, c'est l'exemple que vous avez tapé précédemment (SOUND 5, 150, 25). L'ordinateur exécute l'ordre mais ne le conserve pas en mémoire. Toutes les instructions ne sont pas toutes utilisables en mode direct.

Mode indirect : C'est le mode programme. On indique à l'ordinateur une suite d'instructions à exécuter. En principe on place une instruction par ligne et chaque ligne est numérotée.
Lorsqu'une ligne, c'est-à-dire une instruction, a été écrite on la *valide* en appuyant sur la touche [RET]. Le curseur se place alors au début de la ligne suivante mais contrairement au mode direct, rien ne se passe.
Par exemple : tapez
10 PRINT 5 + 2 puis [RET]
Le curseur se place sous le 1 de 10. L'ordinateur attend soit l'ordre d'exécuter, soit l'entrée d'une autre instruction.
Pour exécuter la ligne 10, tapez [STOP RUN] puis [RET]
7
Ok!
apparaît sur l'écran.
Si vous retapez [STOP RUN] puis [RET], l'ordinateur vous redonne
7
Ok!
Il a conservé l'instruction en mémoire.
Vous pouvez aussi ajouter une nouvelle instruction.
Tapez par exemple :
20 PRINT 9 + 3 puis [RET]
Effectuez [STOP RUN] puis [RET]
Sur l'écran apparaît
7
12
Ok!
Il exécute la ligne 10 puis la ligne 20.

Vous venez de réaliser un mini programme.

Attention : Ne donner pas un ordre d'exécution sur une ligne où il subsiste des caractères.

Assurez-vous quand vous tapez par exemple [STOP RUN] qu'il n'y a rien sur la ligne.
Sinon vous verrez apparaître un message d'erreur et l'instruction ne s'exécutera pas.

8

Conseils aux débutants

et peut-être aux autres aussi

Notre intention n'est pas de faire de ce manuel un cours de programmation ; d'excellents livres existent pour cela chez votre vendeur ou en librairie.

Ce que nous voulons c'est vous donner quelques renseignements et quelques conseils qu'il ne faut pas oublier quand on programme.

1er postulat :
Un ordinateur ne sait rien faire tant qu'on ne lui a pas indiqué comment faire et ne fait rien tant qu'on ne lui a pas demandé.

2e postulat :
Un ordinateur ne peut exécuter un ordre que si précédemment on lui a donné toutes les informations nécessaires à l'exécution de cet ordre.
Sur une ligne vide, taper par exemple :

```
PRINT X + 2 : X = 5 puis [RET]
2
OK !
```

La réponse est 2 car l'ordinateur ne peut afficher la valeur de X + 2 puisqu'il ne connait pas encore x qui est donné ensuite (en l'absence de valeur pour x, il lui donne la valeur zéro).
Par contre essayez :

```
X = 5 : PRINT X + 2 puis [RET]
7
OK !
```

L'ordinateur peut maintenant exécuter l'ordre.
Lorsque vous écrirez un programme mettez-vous toujours à la place de la machine : puis-je exécuter ces instructions dans l'ordre dans lequel elles m'ont été données ?"

3e postulat :
Lorsque quelque chose d'inattendu se passe sur l'écran ou dans l'exécution d'un programme il y a toujours une explication logique ; ce n'est généralement pas une panne du matériel mais une erreur dans le programme.

Dans tout ce qui va suivre vous verrez apparaître des instructions BASIC, nous vous conseillons à chaque fois de vous reporter à la description de cette instruction, au chapitre 18.

Avant de taper un nouveau programme, assurez-vous toujours que la mémoire est vide en tapant LIST. S'il reste quelque chose, tapez NEW puis [RET]

9

Langage et programmation

Le langage

Pour pouvoir converser avec l'ordinateur, vous avez à votre disposition un clavier dont nous venons de voir les particularités et l'écran de télévision qui sert à vérifier que l'ordinateur a bien compris ce que vous lui avez dit par l'intermédiaire du clavier. Ce dernier permet également à l'ordinateur d'afficher les résultats qu'il a obtenus après application de vos directives.

Le langage BASIC, comme tout autre langage informatique, permet de mettre en rapport l'homme et la machine - en définissant un certain nombre de mots que la machine traduira grâce à son dictionnaire interne (Mémoire ROM).

Le langage utilisé dans le VG5000 est le langage BASIC MICROSOFT (1). Il se compose de 81 mots ou instructions BASIC. Toutes ces instructions sont définies et expliquées au chapitre 18.

En les combinant astucieusement et surtout d'une manière logique (rappelez-vous le 2e postulat) vous pourrez réaliser des programmes vous permettant de faire vos propres jeux, de résoudre des calculs, ou de faire des images qui pourront être de véritables tableaux de maître !!...

Quelques instructions BASIC

Voici quelques exemples de mots BASIC que le VG5000 peut comprendre et qui seront pour lui des instructions qu'il devra exécuter :

PRINT	Signifie : Affiche ce qui vient après sur l'écran (vous avez déjà utilisé cette instruction).
GOTO (N° de ligne)	Va exécuter le programme à partir de telle ligne.
FOR I = 1 TO X	Pour I variant de 1 à X exécute la séquence qui suit x fois. La fin de la séquence doit être suivie d'une instruction NEXTI qui signifie donne à I la valeur suivante et ainsi de suite jusqu'à la valeur X.
RUN	Exécute ce que l'on vient de lui demander.
INPUT	L'ordinateur attend qu'on lui donne une valeur qu'il affectera à une variable. Dans ce cas, l'ordinateur met un ? sur l'écran et s'arrête : il attend *que vous entriez une valeur au clavier.*
INIT X, Y	Indique à l'ordinateur d'effacer l'écran et de lui donner une teinte de fond qui dépendra de la valeur donnée à X. Par exemple pour X=5 l'écran deviendra magenta (mauve). La valeur donnée à Y attribuera une teinte au bord de l'écran. Par exemple, pour Y = 3, le bord deviendra jaune.

(1) Tous les langages BASIC ne sont pas équivalents suivant les ordinateurs. Certains ordinateurs possèdent des mots que ne possèdent pas les autres. Un programme prévu pour une machine devra donc être adapté pour fonctionner sur une autre.

Un programme

A l'aide de ces quelques mots on peut déjà réaliser un programme. Par exemple demander à l'ordinateur d'afficher 10 fois automatiquement sur l'écran le mot "Bonjour" en changeant à chaque fois la teinte de l'écran.

```
10 INPUT A                    RET
20 INIT A,3                   RET
30 FOR I=1 TO 10              RET
40 PRINT"BONJOUR"             RET
50 NEXT I                     RET
60 GOTO 10                    RET
```

Pour lancer ce programme, taper RUN puis [RET]
Un ? apparaît sur l'écran.
Taper 1 puis [RET]
BONJOUR s'inscrit 10 fois sur l'écran sur un fond rouge, bord jaune
Un autre ? apparaît.
Taper 2 puis [RET]
et observer ce qui se passe.

Pour sortir de ce programme tapez [↑] + [STOP RUN]

Pour effacer totalement ce programme de la mémoire, taper NEW puis [RET]

Si vous tapez RUN ou LIST, Ok! s'affiche immédiatement sur l'écran signifiant qu'il n'y a plus rien à exécuter dans la mémoire.

Modification d'un programme

Dans l'établissement d'un programme il est souvent nécessaire de modifier soit une valeur, soit une instruction, soit encore d'ajouter une instruction entre deux autres instructions déjà placées dans le programme.

Modifier une valeur dans une instruction.
Procéder simplement comme cela a été indiqué pour corriger un caractère dans un texte. N'oubliez pas d'effectuer RET après la modification sinon la nouvelle valeur n'est pas validée et votre programme conserve l'ancienne valeur.
Vérifier toujours que la modification a bien été acceptée en faisant LIST.
Exemple : `10 PRINT 5 + 2`

Pour changer le 5 en 8 par exemple :
Amener le curseur sur 5
Taper la touche 8
Puis effectuer [RET]
Vérifier que la modification a été acceptée en frappant LIST.

Modifier ou supprimer une ligne de programme.
Amener le curseur juste après le numéro de ligne [↑] + [EFFL EFFC] pour effacer la ligne. Puis retaper ou non une nouvelle instruction.
N'oubliez pas [RET] lorsque vous avez terminé.
Pour effacer une ligne de la mémoire, vous avez aussi la possibilité de taper le numéro de ligne seul, puis [RET] .

Déplacer une ligne de programme.
Il faut agir en deux temps.

1. D'abord recopier la ligne à l'endroit où vous désirez la mettre. Pour cela, amener le curseur sur le 1er chiffre du numéro de la ligne à déplacer, changer le n° de ligne, puis valider la nouvelle ligne en tapant [RET] .
   Vérifier que cette nouvelle ligne est bien placée en faisant LIST.
2. Puis, effacer l'ancienne ligne qui n'a plus lieu d'être. Pour cela, amener le curseur juste *après le numéro de ligne*, entre le numéro de ligne et la première lettre de l'instruction. Appuyez sur [↑] + [EFFL EFFC] , puis sur [RET]
   Vous pouvez aussi sur une ligne vierge taper le numéro de ligne seul puis [RET]

Pour vous aider dans la correction d'un programme, VG5000 possède deux commandes intéressantes : AUTO et RENUM.

AUTO permet le numérotage automatique des lignes.

RENUM permet, lorsque l'on a terminé un programme et que l'on a rajouté des lignes, de renuméroter toutes les lignes pour que tous les numéros de ligne se suivent régulièrement.

10

Définition de quelques termes

Variable

En informatique, on appelle variable une zone de mémoire dans laquelle on peut stocker une donnée numérique ou alphabétique. Le nom donné, AB par exemple, représente pour l'ordinateur l'adresse de la zone. Quand on lui demandera d'afficher AB, il ira chercher dans sa mémoire la valeur qui se trouve dans la zone mémoire AB.

On distingue deux types de variables : les variables numériques et les variables de chaîne.

Variables numériques :
Les variables numériques X ou AB peuvent se voir affecter une valeur numérique par exemple X = 5 ou AB = 137.79.

Variables de chaîne :
Celles-ci peuvent recevoir des données "chaînes de caractères" : (BONJOUR - VG5000 sont des chaînes de caractères). Une chaîne de caractères peut contenir des valeurs numériques mais celle-ci seront considérées comme des caractères et aucune opération mathématique ne peut être effectuée avec elles.

Pour distinguer une variable numérique d'une variable de chaîne on fait suivre la variable de chaîne du signe $
A$ = "BONJOUR"
et les données qui suivent doivent être placées entre guillements ; si ceux-ci sont oubliés un message d'erreur "opérande mal adapté" apparaît.

Variables indicées
Variables numériques ou variables de chaîne peuvent être indicées.
Exemple X(3), AB$(5). Ceci permet de donner à un même nom de variable plusieurs valeurs en fonction de l'indice.
X (3) = 5 AB$(1) = "BONJOUR"
X (4) = 22 AB$(2) = "BONSOIR"
L'indice peut être également une variable AB$(I) où I pourra prendre des valeurs en fonction de calcul (voir les instructions DATA et READ).

VG5000 accepte des variables bi-dimensionnées A (I,J) (voir l'instruction DIM).

NOTA : si un nom de variable a plus de deux caractères, seuls les deux premiers sont pris en compte (MINUTE et MIN sont une seule et même variable MI).
Une variable de chaîne peut contenir jusqu'à 127 caractères.
Un nom de variable doit toujours commencer par une lettre. (6B ne peut être un nom de variable).

Expressions numériques
C'est l'association de constantes numériques et/ou de variables numériques liées entre elles par des opérations mathématiques.
X + 2 - 12 * 3 - 55 * Y / 2 - (25 + 3) * A
sont des expressions numériques.

11

Phases de l'écriture d'un programme

1. L'idée

C'est le départ d'un programme (au début ne soyez pas trop ambitieux). Par exemple, vous décidez de faire un PATCHWORK sur l'écran, c'est-à-dire remplir l'écran d'une multitude de petits rectangles de couleurs différentes et au hasard.

2. Comment réaliser cette idée

a) Il faut écrire à la suite les uns des autres des rectangles.
b) Tirer un nombre au hasard pour choisir les teintes.

3. Le programme peut s'énoncer ainsi

a) effacer l'écran
b) placer le curseur en haut à gauche
c) choisir la teinte du rectangle au hasard
d) imprimer un rectangle avec sa teinte
e) recommencer à choisir une teinte pour afficher un rectangle (c'est-à-dire retourner en c)
f) s'arrêter quand l'écran est rempli.

4. Quelles sont les instructions à votre disposition

PRINT	pour demander d'afficher un caractère
CHR$(xx)	qui permet de définir le caractère ASCII (voir caractère ASCII) qui correspond à celui désiré.
TXT, J, K	pour définir la teinte du rectangle.
RND	tirer un nombre au hasard
INT	partie entière d'un nombre décimal
INIT	pour initialiser l'écran et faire démarrer le premier carré en haut à gauche de l'écran.
FOR...NEXT	permet de faire plusieurs fois la même chose.
GOTO	permet de sauter une partie du programme ou de recommencer sans fin une partie de programme.

Avant de commencer l'écriture du programme, lisez l'explication détaillée de ces instructions.

5. Écriture du programme et commentaires

`10 REM PATCHWORK`	*1re ligne* - Cette ligne ne sert à rien dans le programme ; elle vous permet de vous rappeler le type de programme qui vient ensuite. Elle apparaît en rouge lorsque vous listez le programme.
`20 INIT 4,4`	*2e ligne* (numérotée 20) - Initialise l'écran en bleu et positionne le curseur en haut à gauche.
`30 FOR I = 1 TO 974`	*3e ligne* - Début de la boucle qui permet d'afficher 974 caractères sur l'écran.
`40 A = RND(1)*7`	*4e ligne* - Tirage d'un nombre au hasard qui servira à choisir la teinte (RDN donne un nombre décimal).
`50 B = INT(A)`	*5e ligne* - Comme les teintes vont de 0 à 7 et que ce sont des valeurs entières, INT permet de ne prendre que la partie entière de A.
`60 TX B,0,0`	*6e ligne* - Initialisation en mode texte. B donne la teinte du caractère.
`70 PRINT CHR$(127);`	*7e ligne* - Affichage du caractère dont la valeur ASCII est 127, soit un caractère plein. Le point virgule indique que le prochain caractère sera inscrit juste à la suite.
`80 NEXT I`	*8e ligne* - Fin de la boucle. NEXT I renvoie le programme à la ligne 30 si I n'est pas égal à 974. Quand I est supérieur à 974, on passe à la ligne suivante.
`90 GOTO 10`	*9e ligne* - Indique à l'ordinateur que le programme doit recommencer à la ligne 10.

6. Vérification du programme

Ce qui est affiché sur l'écran n'est pas forcément ce qui a été mis en mémoire. Tapez LIST. Vérifiez ligne par ligne qu'il n'y a pas d'erreur. N'avez-vous pas oublié une parenthèse, une virgule, un numéro de ligne, etc.

7. C'est l'heure de vérité

Lancez le programme en tapant RUN (n'oubliez pas [RET]) et regardez.

Pour l'arrêter tapez [↑] + [STOP RUN]

Maintenant, vous pouvez vous amuser en changeant TX en GR et CHR$(127) en CHR$(60) ou autre.

12

Sauvegarde d'un programme sur cassette

Lorsqu'on a réalisé un programme intéressant, il est souvent agréable de pouvoir le sauvegarder pour ne pas avoir à le retaper à chaque utilisation.
Pour permettre cette sauvegarde le VG5000 est équipé d'une sortie (on dit interface) pour le relier à un magnétophone.

Branchement du magnétophone

Utiliser le cordon équipé d'une fiche DIN 8 broches et de 3 fiches "JACK". La fiche DIN 8 broches se connecte à la prise DIN correspondante sur le VG5000. Les fiches Jack se branchent sur le magnétophone.

- la fiche (Jack 3,5 mm) correspondant au fil blanc se connecte sur la sortie casque (EAR) ou (LINE) du magnétophone
- la fiche (Jack 3,5 mm) du fil rouge sur l'entrée Micro (MIC)
- la fiche (Jack 2,5 mm) du fil noir sur la prise télécommande (REM)
- branchez l'alimentation secteur du magnétophone ou mettre des piles (neuves !)
- Introduisez une cassette (vierge de préférence).

Sauvegarde du programme

Tapez le programme suivant (ou un autre)

```
10 INIT 3,3
20 DATA DIMANCHE,LUNDI,MARDI
30 DATA MERCREDI,JEUDI
40 DATA VENDREDI,SAMEDI
50 PRINT: PRINT "QUELLE EST VOTRE DATE DE NAISSANCE"
60 INPUT "JJ,MM,AAAA";J,M,A
70 IF(J<1)OR(J>31) GOTO 110
80 IF(M<1)OR(M>12) GOTO 110
90 IF(A<1800)OR(A>2000) GOTO 110
100 GOTO 130
110 PRINT"     IL Y A UNE ERREUR"
120 PRINT:GOTO 50
130 IF M>2 GOTO 150
140 A=A-1:M=M+12
150 E=INT(A/100)
160 K=A-E*100
170 B=INT(2.6001*(M-2)-.2)+J+K
180 B=B+INT(K/4)+INT(E/4)-2*E
190 B=B-INT(B/7)*7+1
200 RESTORE
210 FOR I=1 TO B
220 READA$
230 NEXT I
240 PRINT "VOTRE JOUR DE NAISSANCE ETAIT UN :"
250 PRINT :PRINT "            ";A$
260 PRINT :GOTO 50
RUN

QUELLE EST VOTRE DATE DE NAISSANCE?
VOTRE JOUR DE NAISSANCE ETAIT UN :
```

vous entrez par exemple 1, 5, 1984 puis RET

(Ce programme permet de connaître le jour d'une date donnée entre l'an 1800 et l'an 2000).

- Lancez-le par RUN pour voir s'il tourne normalement.
- Rembobinez la cassette au début.
- Mettez le magnétophone en position enregistrement : appuyez sur les touches "REC" et "PLAY" du magnétophone.
- Frappez au clavier : CSAVEL puis RET. Cette commande a pour but de mettre le magnétophone en marche pendant quelques secondes pour passer la bande amorce de la cassette avant de faire l'enregistrement.
- Mettez le compteur du magnétophone à zéro.
- Frappez CSAVE "VG5000" puis RET. (VG5000 sera le nom du programme ; vous pourriez très bien l'appeler AZERTY). Le curseur disparaît. Le magnétophone se met en marche et s'arrête après quelques secondes. Ok! et le curseur réapparaissent.
- Recommencez encore une fois en plaçant le curseur sur la même ligne CSAVE et en appuyant sur RET. Lorsque le magnétophone s'arrête, appuyez sur la touche STOP du magnétophone de manière à ne plus être en position "Enregistrement" et d'éviter ainsi par une fausse manœuvre d'effacer le programme que vous venez d'enregistrer.

Lecture d'un programme enregistré

- Rembobinez la cassette au début - 000 du compteur.
- Videz la mémoire par NEW puis RET.
- Placez le magnétophone en position "Lecture" en appuyant sur la touche "PLAY".
- Mettez le potentiomètre de volume au maximum.
- Frappez CLOAD "VG5000" au clavier puis RET. L'ordinateur donne l'ordre au magnétophone de se mettre en marche par la télécommande et se met lui-même en attente de réception du programme "VG5000". Le bord de l'écran devient rouge et reste rouge tant qu'il n'a rien trouvé. Dès qu'il a trouvé le programme "VG5000" le bord devient vert et sur l'écran apparaît
  Trouvé : VG5000.

Quand le programme commence à entrer dans la mémoire, le bord vert de l'écran redevient turquoise mais est traversé par des barres noires qui semblent défiler de bas en haut.

La vitesse de ce mouvement dépend du magnétophone mais doit être régulière ; si le défilement n'est pas régulier, essayez de diminuer le volume jusqu'à ce qu'il devienne régulier.

Si la vitesse est régulière, ceci indique que le programme en principe rentre correctement dans la mémoire. Lorsque le chargement est terminé, vous voyez réapparaître : Ok! et le curseur.

Vous pouvez alors, soit lancer le programme (RUN) soit le lister (LIST).

Si la vitesse n'a pas été régulière, il y a de fortes chances pour que le programme ne soit pas rentré correctement dans la mémoire. Un message "Mauvais fichier" apparaît sur l'écran. Il faut alors recommencer au début ou lire le deuxième programme qu'on a pris soin d'enregistrer à la suite du premier.

D'autres messages d'erreur peuvent apparaître :

- *Annulé - Repositionnez la bande*

Ce message apparaît lorsque la touche STOP du clavier a été enfoncée ou que le magnétophone a été arrêté pendant l'enregistrement.

- *Sortie de mémoire*

Ce message apparaît si un fichier contient plus d'informations que ne peut en contenir la mémoire - ce qui peut arriver si la mémoire n'a pas été correctement vidée avant l'enregistrement.
Essayez de nouveau ou encore éteignez et rallumez le VG5000.

Quelques conseils pour la sauvegarde d'un programme sur cassette audio

Il faut savoir que si la sauvegarde d'un programme informatique, qui n'a rien à voir avec un enregistrement de la parole ou de la musique, est possible sur bande magnétique, cette méthode présente des difficultés en particulier dues à la diversité des magnétophones et de tous les paramètres qui interviennent. Aussi faut-il prendre beaucoup de précautions.
Voici quelques conseils qui, vous permettront d'enregistrer et surtout de relire vos programmes.

1. Tous les magnétophones standard à cassette conviennent en principe ; éviter les magnétophones HIFI ou stéréo qui possèdent des systèmes DOLBY ou autres, qui causent des distorsions du signal.
Demander conseil à votre vendeur ; il vous renseignera utilement.

2. Le réglage du potentiomètre de volume en général n'est pas critique, et peut au départ être placé au maximum. Si dans ces conditions le programme n'est pas chargé, recommencez plusieurs fois la lecture en réduisant le volume jusqu'à ce que le programme soit trouvé.
Lorsque vous aurez trouvé le programme, repérez bien la position du potentiomètre de volume ; il restera en général valable pour tous les autres enregistrements que vous ferez.
Le potentiomètre de TONE ou TONALITÉ, quand il existe, doit être mis au maximum d'aigües.
Si vous n'arrivez pas à relire un programme qui a été enregistré, en particulier ceux qui ont été enregistrés sur un autre magnétophone, le bord de l'écran reste rouge et le magnétophone ne s'arrête pas ; alors vérifiez ou faites vérifier les points suivants :

- l'azimuth de tête
- l'encrassement des têtes
- L'emplacement du magnétophone par rapport à son environnement
- il faut que celui-ci soit éloigné d'au moins 80 cm du téléviseur ou du moniteur
- éviter des tables métalliques ou parties métalliques importantes à proximité.

Utilisez des cassettes ULTRA FERRO 30 mn de préférence ou des cassettes 15 mn spéciales pour micro-ordinateurs.
Les cassettes C-60 conviennent également.
Évitez les cassettes Ferro.

Magnétophone sans télécommande

Au cas ou vous utilisez un magnétophone qui ne possède pas de prise télécommande "REM" ou si celle-ci ne fonctionne pas (certains magnétophones ont une prise télécommande inversée, ne la branchez pas) vous pouvez quand même vous servir de votre magnétophone en effectuant manuellement les mises en route et les arrêts.

Procédez de la manière suivante :

Pour un enregistrement :

Après avoir tapé CSAVE "nom du programme", mettez en marche le magnétophone en position enregistrement. Enfoncez "REC" + "PLAY" puis sur la touche RET (attention à la bande amorce).

Pour une lecture (chargement) :

Faire l'inverse. Tapez CLOAD "nom du programme". Tapez RET avant de lancer le magnétophone en appuyant sur la touche "PLAY".

Dans les deux cas, n'arrêtez le magnétophone que lorsque Ok! apparaît sur l'écran.

NOTA :

Si vous voulez vérifier qu'un programme existe bien sur une cassette ou si vous recherchez un programme qui se trouve au milieu d'une cassette, il est possible de repérer le programme enregistré en écoutant, c'est-à-dire en débranchant les fiches EAR (casque) du magnétophone et en mettant le magnétophone sur "PLAY" - volume au maximum - et d'écouter ce qu'il y a d'enregistré sur la bande.

Tant qu'il n'y a rien on entend un "bruit de fond" et dès qu'on arrive sur un programme on entend, au début un sifflement aigu, puis quelques secondes après le bruit "très particulier" (et surprenant au début) des octets qui défilent.

Avec un peu d'expérience vous reconnaîtrez facilement le début d'un programme. Naturellement, un compteur est plus pratique !

13

Arithmétique et fonction mathématique

VG5000 reconnaît un certain nombre d'instructions arithmétiques, mathématiques, logiques et numériques.

1. Opérateurs arithmétiques

Ils déterminent :

Les opérations arithmétiques : +, −, X, ÷, ∧ (puissance)
Les opérations logiques : NOT, AND, OR
Les opérations d'affectation ou d'égalité : =
Les tests de comparaisons : =, <, >, <=, >=, <>.

Exemple :
```
? 40 / 5 + 3
11
? 343↑(1 / 3)
7
```

(343 élevé à la puissance 1/3)

Dans toutes les fonctions mathématiques les parenthèses sont très importantes. Reprenez l'exemple précédent et supprimez les parenthèses.

Exemple :
```
PRINT343↑1/3
114,333
```

L'ordinateur comprend : 343 élevé à la puissance 1 et le tout divisé par 3 = 114,333.

Remarquez que la notation des valeurs décimales se fait suivant la méthode anglo-saxonne où la partie décimale est séparée de la partie entière par un point et non une virgule.

Lorsque vous voulez entrer une valeur décimale, vous devez séparer la partie décimale par *un point.*

2. Opérateurs logiques

Les opérations ET, OU et NON logiques sont effectuées par VG5000.
Les instructions correspondantes sont AND, OR et NOT (voir ces instructions).
Elles sont utilisées pour prendre une décision lorsque deux évènements doivent survenir ou non en même temps.
Tous les opérateurs de comparaisons peuvent être utilisés.

3. Opérateurs de comparaison = ; > ; <

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

Ces opérateurs sont en général utilisés dans des instructions de décisions.
Exemple : si X>38 alors afficher "X Supérieur à 38"
(voir les instructions IF ... THEN et IF ... GOTO)

Égal (=) sert aussi à affecter une valeur ou une expression à une variable.

Exemple : X = 5
A = B * C
B$ = "VG5000"

4. Priorités des opérateurs

Si VG5000 a une expression arithmétique à calculer, par exemple :

```
PRINT 25/5 + 6 = 15/3 * 6↑2
```

il faut savoir qu'il effectue les calculs suivant les priorités suivantes :

1. les expressions entre parenthèses
2. Élévation à une puissance
3. Négation
4. * ;/
5. + ; −
6. = ; <> ; < ; > ; <= ; >=
7. Not
8. AND
9. OR

L'expression précédente sera calculée ainsi :

```
6↑2 = 36
15/3 X 36 = 180
25/5 = 5
```

enfin 5 + 6 – 180
– 169

Dans le cas d'expressions plus complexes dans lesquelles on utilise des parenthèses, tous les calculs entre parenthèses sont exécutés en priorité par rapport aux priorités précédentes.
Dans l'exemple ci-dessus, si 5 + 6 est mis entre parenthèses (5 + 6) le calcul est fait en priorité. L'expression à calculer devient alors

25/11 - 15/3*6 ↑ 2
- 177.727

5. Précisions des calculs

VG5000 effectue les calculs avec 6 chiffres significatifs.
PRINT 250 / 11
22.7273

Si les nombres dépassent les limites 999999 ou 0,01, ils sont arrondis et exprimés en notation scientifique. Pour exprimer un nombre en notation scientifique, on exprime ce nombre en puissance de 10.
25000000 s'exprime 2.5 10^7.

En informatique 10 puissance 7 s'écrit E+7 (exposant + 7)
(l'exposant peut être négatif)
Essayez : PRINT 25000000
2.5 E+7

Si les chiffres à afficher sont inférieurs à E-38 (10^{-38}) ou supérieurs à 1,7E + 38 ($1{,}7 \times 10^{38}$), il y a dépassement de capacité (appelé Infini machine).

6. Fonctions mathématiques

VG5000 possède un certain nombre d'instructions qui permettent d'affecter à une variable la valeur donnée par la fonction mathématique suivant la valeur de l'argument de cette dernière.

Exemple : ABS (-9)
signifie valeur absolue
et -9 est l'argument.

L'argument peut être également une expression ou une fonction.
TAN(X) signifie : donne la valeur de tangente dont l'argument est X.

Pour toutes les fonctions trigonomètriques l'argument doit être exprimé en Radian

$$1 \text{ radian} = \frac{180^\circ}{3.14159}$$

$$\text{Exemple : 45 degrés} = \frac{3.14159 \times 45}{180} = 0{,}7854 \text{ Radian}$$

Tapez : `PRINT=TAN (0,7854)` puis RET
1
OK

Vous trouverez à l'Annexe III comment calculer certaines formules mathématiques avec votre VG5000.

7. Fonctions alphanumériques

Ce sont des fonctions qui effectuent des travaux sur les chaînes de caractères. Elles permettent la concaténation, c'est-à-dire l'association de chaînes de caractères :

Chanter + as donne Chanteras
Chanter + ont donne Chanteront

ou encore permettent l'extraction de mots ou de lettres à l'intérieur d'une chaîne de caractères.

Exemple : `PRINT "CHANTER" + "ONT"`
CHANTERONT

Vous trouverez toutes ces instructions expliquées dans le chapitre Instructions BASIC.

14

La ponctuation

Comme dans tous langages, BASIC utilise en plus des mots, des signes de ponctuation. Voici les signes utilisés dans VG5000 et leur utilisation.

Point virgule :
Après une instruction PRINT, permet d'afficher les caractères suivants juste après le dernier caractère affiché.

```
10 PRINT "BASIC";
20 PRINT "VG5000";
30 PRINT "MICROSOFT";
RUN
BASICVG5000MICROSOFT
```

Virgule :
La virgule sépare entre eux les différents paramètres de certaines instructions.
Ex. : TXT, J, K.
Dans une instruction PRINT, elle permet de décomposer l'écran en trois zones verticales de 13 caractères chacune. L'affichage se fait après une virgule au début de la zone suivante.
Dans le programme ci-dessus - changez-les ; des lignes 10 et 20 par des virgules et lancer le programme.

```
BASIC         VG5000        MICROSOFT
```

Deux points :
Séparent dans une même ligne des instructions différentes. Avec VG5000 il est possible de placer plusieurs instructions sur une même *ligne informatique* (dans la limite de 127 caractères).

```
10 INIT5:CURSORX5:CURSORY10:PRINT"BASIC","VG5000","MICROSOFT"
RUN
BASIC         VG5000        MICROSOFT
```

Guillemets "....."
Ils sont très importants car ils déterminent si les données sont des valeurs numériques ou des chaînes de caractères.
35 + 7 est très différent de "35 + 7" ou même de "35" + "7" considérés comme chaînes de caractères. Ils sont affichés tels quels par l'ordinateur.
Ils sont aussi utilisés pour exprimer des nombres en valeur hexadécimale (voir l'instruction &).

Parenthèses (.....)
A la suite d'une variable, indiquent qu'il s'agit d'un indice A (x).
A la suite d'une instruction, indiquent une expression numérique.
A la suite d'une fonction, indiquent l'argument de cette fonction.
A l'intérieur d'une expression arithmétique elles forcent la priorité du calcul.

15

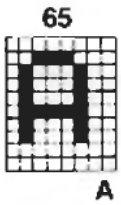
Mode texte - Mode graphique

Pour l'affichage VG5000 dispose des deux jeux de caractères qui sont en mémoire.

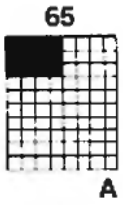
Le premier jeu est un jeu de caractères alphanumériques comprenant toutes les lettres majuscules, minuscules et accentuées, les chiffres et les caractères de ponctuation (Annexe 4). Ce jeu de caractères est mis en service par l'instruction TX* qui définit la couleur, la taille, la stabilité des caractères affichés sur l'écran.

Le deuxième jeu comprend des caractères graphiques (Annexe 5) qui sont mis en service pour l'instruction GR* qui définit la couleur des caractères, la couleur de fond, la stabilité.

Chaque caractère, texte ou graphique, est codé de 0 à 127 suivant le standard ASCII (American Standard Communication Information Interchange) qui code les caractères sur 8 bits (soit 256 possibilités).



Caractère texte



Caractère graphique

Les caractères texte peuvent être affichés de 2 manières : soit directement au clavier, soit avec l'instruction CHR$ qui utilise leur valeur ASCII. Reportez-vous à l'annexe 4.
`PRINT "A"` donne sur l'écran A (voir annexe 4).
`PRINT CHR$(65)` donne également A.

Tous les caractères non disponibles sur le clavier devront être appelés de cette manière.
En l'absence d'indication TX ou GR c'est-à-dire dès la mise en service du VG5000 l'initialisation est faite automatiquement en TX0, 0, 0 (lettre noire).

Exemple : `TX 5,0,0:PRINT"A";CHR$(66);CHR$(67)`

En mode graphique il est impératif d'initialiser le mode en utilisant l'instruction GR. Comme les caractères ne sont pas accessibles au clavier il faut utiliser, pour les afficher, l'instruction CHR$
GR1,4,0 : PRINT CHR$ (65) donne un caractère en rouge sur fond bleu.

Pour afficher plusieurs caractères graphiques les uns à côté des autres, il faudra afficher CHR$ plusieurs fois de suite en les séparant par des points virgules.

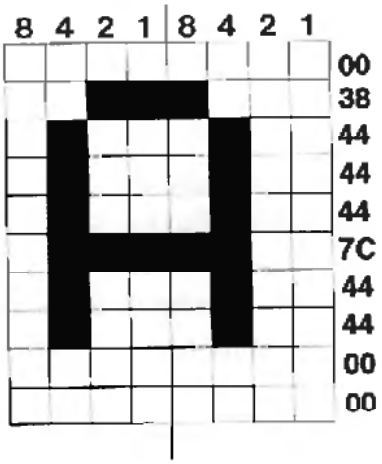
Si vous vous souvenez que CHR$ (65) représente A en mode texte, essayez de taper
`GR1,4,0 : PRINT"A"` et regardez le résultat.
L'ordinateur reste en mode graphique et interprète A comme CHR$ (65).
Cette façon d'afficher des caractères graphiques est plus facile.
C'est pourquoi pour chaque caractère graphique nous avons indiqué son code ASCII au-dessus et en-dessous à droite, la lettre ou le signe disponible au clavier qui lui correspond.

16

Jeux de caractères définissables par l'utilisateur

En plus des 2 jeux de caractères que nous venons de voir, VG5000 permet à l'utilisateur de créer ses propres caractères.

Un maximum de 96 caractères en mode texte et 96 caractères en mode graphique peuvent ainsi être définis, soit 192 caractères.



Chaque caractère est placé dans une matrice de 8 x 10.
Chaque case à l'intérieur de la matrice peut être "occupée" ou "noircie", ce qui permet de dessiner un caractère qui pourra prendre des attributs soit du mode texte, soit du mode graphique.
Pour passer en mode texte ou en mode graphique les caractères ainsi définis, on utilise l'instruction ET* en mode texte et EG* en mode graphique pour indiquer qu'il s'agit de caractères spéciaux (par rapport à TX et GR qui ne concernent que les caractères en mémoire prédéfinis dans la ROM).

Pour définir un caractère, on considère la matrice ligne par ligne (il y a 10 lignes).

Chaque ligne est décomposée en 2 fois 4 cases.
Dans chaque groupe de 4 cases, on donne un poids différent à chaque case.

Poids 8 4 2 1 8 4 2 1

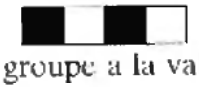
4	3	2	1	4	3	2	1

4 cases 4 cases
1 ligne

La case n° 1 a le poids 1 (2^0)
La case n° 2 a le poids 2 (2^1)
La case n° 3 a le poids 4 (2^2)
La case n° 4 a le poids 8 (2^3)

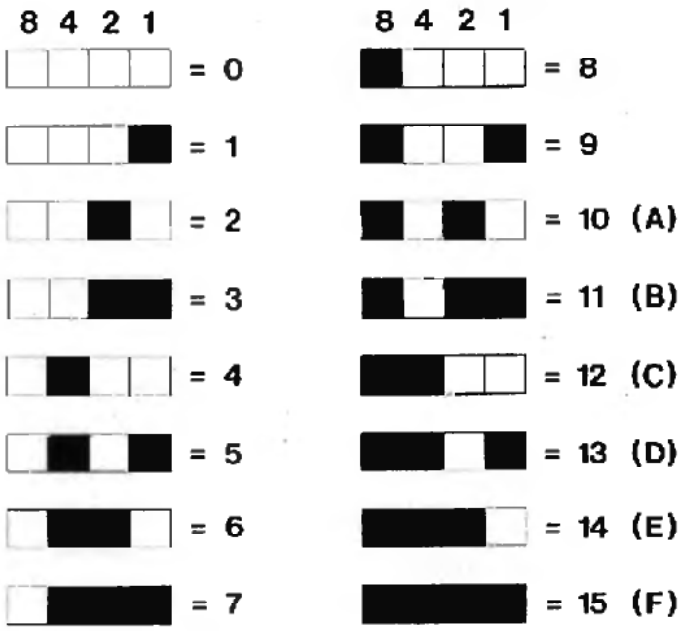
Lorsqu'une case est occupée ou noircie, on lui donne la valeur 1 x par son poids ; si elle est inoccupée, on lui donne la valeur 0 qui, multipliée par son poids, donne toujours 0, puis on additionne les valeurs de toutes les cases occupées d'un groupe de 4 cases, qui prend ainsi la valeur 0 si aucune des 4 cases n'est occupée ou 15 si toutes les cases sont occupées.

8 2



ce groupe a la valeur
10 ou A

* *(voir ces instructions)*



Les valeurs de 0 à 15 sont représentées par des lettres de A à F (code hexadécimal).

Pour définir une ligne qui contient 8 cases, 2 valeurs hexadécimales seront nécessaires :
la valeur minimum sera 00 la valeur maximum sera FF.

La valeur de cette ligne est A6.

Pour définir un caractère, il faudra donner la valeur des 10 lignes, c'est-à-dire 20 chiffres, et les mettre en mémoire.

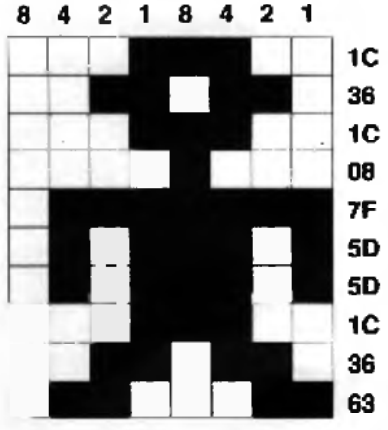
Pour cela, on a 2 instructions BASIC :

SETET s'il s'agit d'un caractère texte
SETEG s'il s'agit d'un caractère graphique

(SET signifie point ; ET : exécution texte et EG : exécution graphique).

Pour distinguer plusieurs caractères les uns des autres, on donne un numéro à SETET ou SETEG.
Ces numéros vont de 032 à 127 (96 caractères) ; c'est l'utilisateur qui choisit arbitrairement le numéro (Il est conseillé de choisir, pour les raisons que nous avons exposées plus haut - page 29 - des chiffres qui correspondent aux caractères du clavier).

Robot



Le caractère ci-contre aura la valeur

SETET 065,"1C361C087F5D5D1C3663"

Il pourra avoir tous les attributs des caractères texte : double hauteur, double largeur, etc.

Comment réaliser un dessin sur l'écran

Comme nous l'avons vu, l'écran est divisé en 25 lignes de 40 caractères ; par ailleurs, nous avons à notre disposition 2 instructions :

CURSORX et CURSORY

qui permettent de déplacer le curseur en X et en Y :
horizontalement (ligne) X pourra aller de 0 à 39
verticalement (colonne) Y pourra aller de 0 à 24.

Pour chaque zone caractère de l'écran, il sera possible de définir X et Y et d'afficher le caractère voulu aux coordonnées ainsi définies.

Il faudra répéter cette opération autant de fois qu'il y aura de caractères différents sur l'écran.

Pour vous aider à créer vos propres dessins, vous trouverez en annexe une grille "ECRAN" composée de cases représentant les positions que pourront prendre les caractères sur l'écran (avant de l'utiliser, nous vous conseillons d'en faire des photocopies).

Lorsque votre dessin est réalisé sur cette feuille, il vous faut définir tous les caractères différents dont vous avez besoin.

Là encore, pour vous aider, vous trouverez en annexe une grille "CARACTERE" qui vous permettra, en dessinant dans chaque grille le caractère voulu, de calculer facilement sa valeur.

Exemple de programme qui utilise le caractère Robot.

```
10 INIT 0
20 SETET 065, "3C24663C18FF18183C66"
30 ET4,3,0
40 CURSORY 09:CURSORX 17 :PRINT "AAAA"
50 CURSORY 10:CURSORX 17 :PRINT "AAAA"
60 FOR I=0 TO 6
70 CURSORY 10:CURSORX 9+I
80 ETI,0,0:PRINT "A";
90 CURSORX 28-I
100 ETI,0,0:PRINT "A"
110 NEXT I
120 END
```

17

Liste des messages d'erreur et d'information

Lorsque votre ordinateur constate une erreur dans le programme qui l'empêche de fonctionner normalement, il affiche un message d'erreur destiné à faciliter la recherche de ladite erreur.

Voici la liste des messages :

Appel de fonction incorrecte

Un paramètre hors des limites est transmis à une fonction mathématique ou chaîne. Un message d'erreur FC peut aussi être produit par :

1. un exposant anormalement petit ou anormalement grand
2. un argument négatif ou zéro avec LOG
3. un argument négatif pour SQR
4. une mantisse négative avec un exposant non entier
5. un appel d'une fonction USR pour laquelle l'adresse de départ n'a pas été donnée.
6. un argument incorrect pour AND, CALL, CSAVE, LEFT$, MID$, NOT, PEEK, POKE, TAB, SPS, STRING, etc.

Chaîne trop longue

Une expression chaîne est trop longue

Dépassement de capacité

Le résultat d'un calcul est trop grand pour être représenté en format numérique BASIC-80. Si un dépassement de capacité négatif se produit, le résultat est zéro et l'exécution continue sans erreur.

Division par zéro

Le programme a rencontré une division par zéro dans une expression, ou l'opération d'élévation à une puissance donne 0 élevé à une puissance négative. L'infini machine (10^{38} ou $1{,}710^{38}$) avec le signe du numérateur est fourni comme résultat de le division, ou l'infini machine positif est fourni comme résultat de l'élévation à la puissance et l'exécution continue.

Données épuisées

Une instruction READ est exécutée alors qu'il ne reste pas dans le programme d'instructions DATA avec des données non lues.

Espace-chaîne épuisé

Les variables chaîne dépassent le montant alloué à l'espace chaîne. Allouez davantage d'espace chaîne avec CLEAR, ou diminuez la taille et le nombre de chaînes.

Erreur de syntaxe

Le programme a rencontré une ligne qui contient une séquence incorrecte de caractères (par exemple des parenthèses sans correspondance, une commande ou une instruction mal orthographiée, une ponctuation incorrecte, etc.).

Fonction utilisateur non définie

Une fonction est appelée avant que la définition de fonction (instruction DEF) n'ait été donnée.

Formule chaîne trop complexe

Une expression chaîne est trop complexe. L'expression doit être fragmentée en expressions plus petites

Impossible de continuer

Vous avez tenté de continuer un programme qui :

1. s'est arrêté par suite d'erreur,
2. a été modifié pendant une interruption d'exécution.
3. n'existe pas
4. a été arrêté par les commandes CTRL + ⏏

Incorrect en direct

Une instruction non valable en mode direct est entrée comme commande en mode direct.

Indice hors des limites

Il est fait référence à un élément de tableau soit avec un exposant hors des dimensions du tableau soit avec un nombre incorrect d'exposants.

Ligne non définie

Une référence de ligne dans une instruction GOTO, GOSUB, IF...GOTO, ON...GOTO, ON...GOSUB concerne une ligne qui n'existe pas.

NEXT sans FOR

Une variable d'une instruction NEXT ne correspond pas à une variable d'une instruction FOR exécutée précédemment et sans correspondance.

Non reconnue

L'erreur n'est pas reconnaissable.

Dernière information ignorée

Apparaît lorsqu'on demande d'afficher une donnée qui n'a pas été définie. Exemple : READ I, J nécessite deux variables alors qu'une seule a été définie précédemment.

Recommencez au début

Ce message survient à la suite d'une instruction INPUT à laquelle on répond par une valeur numérique alors que l'ordinateur attend une chaîne de caractères ou inversement.

Arrêt en ligne xx

Ce message apparaît après une instruction STOP pour indiquer où est arrêté le programme ou [↑] + [STOP RUN] a été frappé. On peut continuer l'exécution du programme en tapant la commande CONT.

Trouvé : "nom du programme"

S'affiche sur l'écran lors d'un chargement à partir d'une cassette lorsque l'ordinateur a reconnu le programme demandé.

Passé : "nom du programme"

Indique, lors d'un chargement, que l'ordinateur a rencontré un programme, qu'il le passe puisqu'il ne s'agit pas du programme demandé.

Annulé - Repositionner la bande

Un arrêt du magnétophone ou une manipulation du clavier a eu lieu pendant le chargement. Rembobiner la bande et relancer le chargement.

Mauvais fichier

Un défaut est survenu pendant le chargement ; le programme ne peut être exécuté.

Fichier non correspondant

Pendant la vérification d'un fichier sur cassette, il est trouvé une différence avec le contenu de la mémoire. Le fichier sur cassette est le bon fichier.

Imprimante pas prête

L'imprimante n'envoie pas le signal READY. Vérifier la connexion et appuyer sur n'importe quelle touche autre que [PRT] pour continuer (ce message n'est valable que si on utilise l'extension VG5216 et une imprimante).

Pause...

Une cassette comportant un texte ASCII a été chargé. Pendant qu'il est interprêté par le BASIC de VG5000, ce qui demande plusieurs secondes, ce message est affiché. Quand l'interprétation est terminée, Ok! est affiché.

Périphérique non connecté

Le périphérique n'est pas prêt (non allumé, non branché, etc.).

Opérande manquant

Une expression contient un opérateur qui n'est pas suivi d'un opérande, ou une valeur manque dans le calcul d'une expression.
Exemple : B = 5 + [RET]

Opérande mal adapté

Un nom de variable chaîne reçoit une valeur numérique ou inversement ; une fonction qui attend un argument numérique reçoit un argument chaîne ou inversement.

Retour sans GOSUB

Le programme a rencontré une instruction RETURN sans qu'il y ait d'instruction GOSUB correspondante.

Sortie de mémoire

Un programme est trop important, a trop de boucles FOR ou de GOSUB, trop de variables, ou des expressions trop compliquées.

Tableau redimensionné

Deux instructions DIM sont données pour le même tableau, ou une instruction DIM est donnée pour un tableau après que la dimension par défaut de 10 ait été établie pour ce tableau.

18

COMMANDES - Instructions et fonctions BASIC 80 par microsoft

Ce chapitre décrit toutes les commandes et instructions BASIC du VG5000

Étudiez-les en détail ; de leur bonne connaissance dépendra le déroulement correct de vos programmes.

Si vous ne comprenez pas bien comment fonctionne une instruction ou l'utilité d'un paramètre, n'hésitez pas à modifier votre programme et regarder ce qui se passe.

Certains exemples ne comportent pas de numéro de ligne : cela indique que l'instruction peut être utilisée telle quelle en mode direct.

Chaque description se présente de la façon suivante :

Syntaxe Indique la syntaxe correcte de l'instruction. Voir ci-dessous les règles de notation de la syntaxe

But Indique le but ou l'action de l'instruction

Remarques Décrit en détail le mode d'utilisation de l'instruction

Exemples Donne des exemples de programmes complets ou partiels qui illustrent l'utilisation de l'instruction

Notation s'appliquant à la syntaxe Les règles suivantes s'appliquent à toutes les syntaxes d'instruction ou de commande

1. Les éléments en lettres capitales doivent être entrés tels quels.
2. Les éléments en lettres minuscules doivent être définis par l'utilisateur.
3. Les éléments soulignés sont facultatifs.
4. Tous les signes de ponctuation (virgules, parenthèses, points-virgules, traits d'union, signes "égal") doivent être intégralement respectés.
5. Les éléments suivis de 3 points de suspension ... peuvent être répétés un nombre de fois indéterminé (dans la limite de la longueur de la ligne : 127 caractères).

***ATTENTION :** Dans les exemples de programmes il est très important de ne pas confondre le zéro avec la lettre O.*

Les arguments des fonctions sont toujours entre parenthèses. Dans les formats donnés aux fonctions de ce chapitre, les arguments ont été abrégés de la façon suivante :

X et Y	représentent les expressions ou constantes numériques
I et J	représentent des expressions ou constantes entières
X$ et Y$	représentent des expressions-chaîne de caractères.

Important : Certaines valeurs en particulier les adresses mémoires se présentent sous forme &"7080" ; cela indique que le chiffre donné ici - 7080 - est exprimé en code HEXADECIMAL (Voir Fonction &"...." page suivante).

1 &"....."

(Hexadécimal)

Syntaxe & "NNNN"

But Indiquer à l'ordinateur que le nombre NNNN qui se trouve entre guillemets est exprimé en HEXADECIMAL

Remarques Le code HEXADECIMAL correspond à la base de numération 16.
Cette numération utilise les mêmes chiffres que la base DECIMALE (10) de 0 à 9, mais utilise des lettres pour les nombres de 10 à 15.

DECIMAL	HEXA	DECIMAL	HEXA
0	0	8	8
1	1	9	9
2	2	10	A
3	3	11	B
4	4	12	C
5	5	13	D
6	6	14	E
7	7	15	F

La valeur maximale de N est F, soit 15 en DECIMAL.

& "FF" = 255 en DECIMAL
& "FFF" = 4095 en DECIMAL
& "FFFF" = 65535 en DECIMAL.

Exemple

```
PRINT & "FFF"     puis RET
4095
PRINT & "DF3"     puis RET
3571
```

Note VG5000 répond toujours en décimal
Pour les adresses la valeur positive la plus grande acceptée est &"7FFF", la valeur négative la plus petite est – &"8000"

---

2 ABS

(absolue)

Syntaxe ABS(X)

But Donner la valeur absolue de l'expression X.

Exemple

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

3 ACTION

(Bouton Action)

Syntaxe ACTION (I)

But Donner une valeur suivant la position du bouton Action ou de la touche ESPACE

Remarques Le paramètre I dépend de la manette utilisée :

I = 0 pour la commande de droite
I = 1 pour la commande gauche
I = 2 pour la touche ESPACE

Si le bouton Action de la manette considérée est enfoncé, la valeur donnée par Action (I) est 1, sinon zéro.

Exemple

```
10 A=ACTION(0)
20 IF A=0 GOTO 10
30 PRINT A;
40 GOTO 10
RUN
```

Tant que le bouton Action de la manette de gauche est enfoncé, on affiche des 1.
Dès qu'on relâche le bouton, on arrête l'affichage.

Cette instruction n'est utilisable avec les valeurs 0 et 1 que si l'on possède les manettes VU0001 et l'interface manette VG5200 ou l'extension VG 5216.

---

4 AND

(Et)

Syntaxe AND

But Effectuer la fonction logique ET.

Remarques Si un évènement "ET" un autre arrivent simultanément, alors faire telle action.

Exemple

```
10    INPUT X
20    INPUT Y
30    IF X>10 AND Y<5 THEN 60
40    PRINT "GAGNE"
50    END
60    PRINT "PERDU"
70    GOTO 10
RUN
```

? Entrez un nombre pour X [RET]
? Entrez un nombre pour Y [RET]
(Si vous avez perdu, vous pouvez recommencer)
(Pour arrêter le programme, faire [↑] + [STOP RUN]).

5 ASC

(ASCII)

Syntaxe ASC(X$)

But Donner la valeur numérique correspondant au code ASCII du premier caractère de la chaîne X$ (voir les codes de caractères à l'annexe 4).
Si X$ est nul, un message d'erreur "Appel de fonction incorrect" est affiché.

Exemple

```
10    X$ = "TEST"
20    PRINT ASC(X$)
RUN
84    84 est la valeur ASCII de T
OK
```

---

6 ATN

(Arc Tangente)

Syntaxe ATN(X)

But Donner une valeur de Arc Tangente X.

Remarques Le résultat est compris entre $-\pi/2$ et $+\pi/2$
X doit être exprimé en Radian.

Exemple

```
10    X = 3
20    PRINT ATN(X)
RUN
1.24905
```

7 AUTO

(Automatique)

Syntaxe AUTO I, J

But Numéroter automatiquement les lignes d'un programme.
Dès qu'une ligne est validée (touche RET), le curseur revient au début de la ligne suivante qui est automatiquement numérotée.

Remarques I est le numéro de la première ligne qui sera numérotée après que la commande AUTO ait été exécutée.
J est le pas de numérotation des lignes suivantes.
A défaut de I et J, ceux-ci prennent la valeur 10.
Une ligne erronée peut être corrigée sans sortir du mode Auto.
Si un autre N° de ligne doit être utilisé, il suffit de changer le N° de ligne donné par AUTO, la numérotation continuera à partir de ce nouveau N°.
Pour sortir du Mode Auto, placer le curseur juste à droite du chiffre des unités du n° de ligne et taper [RET]

Exemple

```
AUTO 1000,100
1000 PRINT "AUTO"
1100■
```

└ emplacement du curseur pour sortir du mode Auto.

---

8 CALL

(Appeler)

Syntaxe CALL I

But Appeler un programme en mémoire écrit en langage machine.

Remarques I doit être une expression entière. Elle représente l'adresse à laquelle on passe le contrôle au microprocesseur.
Pour sortir du programme en code machine et revenir en BASIC, le programme en langage machine doit se terminer par l'instruction &"C9".

Exemple :

```
CALL 0
```
(Transfèrera le contrôle du programme dans la ROM BASIC à l'adresse 0 qui correspond à RESET)

```
CALL & "7080"+10
```
(Transfert du contrôle à l'adresse &"7080" en hexadécimal, + 10 en code décimal, soit 28810

I doit être compris entre (-&"8000") et + &"7FFF".

9 CHR$

(Caractère)

Syntaxe CHR$ (I)

But Transformer un code ASCII en caractère ou commande pour la machine.

Exemple

```
PRINT CHR$ (66)
B                    (B a le code caractère 66)
OK !
```

---

10 CLEAR

(Effacer, Réserver)

Syntaxe CLEAR I , J

But Remettre toutes les variables numériques à 0 et toutes les variables chaînes à la valeur nulle et, en option, définir le nombre d'espaces chaînes et la fin de la mémoire disponibles pour les programmes en BASIC.

Remarques I définit la totalité de l'espace occupé par les espaces chaînes.
Ce doit être une expression entière. La valeur par défaut est 50 octets.
I est compris entre −32768 et 32767.
J est un emplacement mémoire qui, s'il est spécifié, définit la plus haute position mise à la disposition de BASIC-80.
J doit être inférieur à 32767 pour le VG5000 standard 16 K RAM et à 42864 avec l'extension mémoire 16 K.
CLEAR ne doit pas être utilisé dans un sous programme.

Exemple

```
10   CLEAR 2000, 24576
```

Réserve le nombre d'espaces chaînes à 2000 octets et définit l'emplacement mémoire le plus haut que BASIC-80 peut utiliser : 24576.

11 CLOAD

(Charger)

Syntaxe

CLOAD "nom de programme" , I
CLOAD "nom de programme" nom de variable chaîne
CLOAD* "nom de programme" nom de tableau
CLOADA "nom de programme" , I
CLOAD? "nom de programme"

But

Charger un programme à partir d'une cassette.

Remarques

"nom de programme" est le nom indiqué par le programmeur lorsque le programme a été sauvegardé (voir CSAVE).
"nom de programme" peut avoir de 1 à 6 caractères. Les mêmes lettres doivent être utilisées entre CSAVE et CLOAD en respectant majuscules et minuscules.
Si le "nom de programme" est omis, le premier programme rencontré sur la cassette est chargé.
I est un numéro de ligne à partir duquel le programme s'exécute après le chargement.
I est facultatif.
Si un programme est présent dans la mémoire du VG5000, CLOAD exécute une commande NEW avant de charger le programme demandé.
Si le programme est en code machine, les données seront stockées en mémoire aux mêmes adresses que lors de la sauvegarde (CSAVEM).

Si les données constituent l'image écran, celles-ci seront stockées dans la topographie mémoire d'écran.
Si le programme contient une chaîne de caractères, les données sont chargées dans la variable-chaîne appelée "nom de variable-chaîne" spécifiée lors de la sauvegarde (CSAVE) des données. Assurez-vous que la variable a été réservée avant son chargement.
*CLOAD** charge les données dans le tableau appelé "nom de tableau" spécifié lors de la sauvegarde (CSAVE) du tableau. Le tableau peut être numérique ou alphanumérique ; assurez-vous que ce dernier a été dimensionné avant son chargement (voir l'instruction DIM).
CLOADA permet d'ajouter un autre programme après le programme qui se trouve en mémoire. L'instruction NEW n'est pas exécutée et les 2 programmes peuvent être chargés l'un derrière l'autre. Il est aussi possible d'exécuter l'instruction RENUM pour en faire un seul et même programme (sous réserve de ne pas avoir 2 N° de ligne identiques).
CLOAD? vérifie par comparaison que le programme sur bande est identique au programme en mémoire. S'ils sont identiques, BASIC affiche "OK" ; s'ils ne sont pas identiques, BASIC affiche "mauvais fichier".

Exemple

```
CLOAD "PROGA", 500
```

Charge le programme "PROGA" et commence son exécution au numéro de ligne 500.

12 CONT

(Continuer)

Syntaxe CONT

But Continuer l'exécution du programme après mise en œuvre de la touche STOP ou d'une instruction STOP.

Remarques L'exécution reprend à l'endroit où l'interruption s'est produite.
CONT est en principe utilisé conjointement à STOP pour la mise au point d'un programme. Lorsque l'exécution est arrêtée, les valeurs intermédiaires peuvent être examinées et modifiées par des instructions en mode direct. L'exécution peut être relancée avec CONT ou avec GOTO en mode direct qui reprend l'exécution à un numéro de ligne spécifié.

Exemple

```
10 PRINT "BONJOUR"
20 STOP
30 PRINT "BONNE NUIT"
RUN
BONJOUR
ARRET EN 20      (Tapez CONT puis [RET])
BONNE NUIT
OK!
```

---

13 COS

(Cosinus)

Syntaxe COS(X)

But Donner le cosinus de X.
X doit être exprimé en Radian

Exemple

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

14 CSAVE

(Sauvegarde d'un programme)

Syntaxe
CSAVE (v) "nom de programme", I
CSAVEM (v) "nom de programme", S , T , I
CSAVEL
CSAVES (v) "nom de programme"
CSAVE* (v) "nom de programme" nom de tableau
CSAVEX (v) "nom de programme" nom de variable chaîne

But
Sauvegarder un programme sur cassette.

Remarques
CSAVE sauvegarde sur cassette le programme se trouvant en mémoire.
"Nom de programme" est une expression-chaîne choisie par l'utilisateur pour son programme.
I est le numéro de ligne à partir duquel le programme s'exécutera après le chargement.
I est facultatif.
v indique la vitesse de transmission ; il peut prendre la valeur 1 ou 2.
1 effectue la sauvegarde à la vitesse de 1200 bauds.
2 effectue la sauvegarde à la vitesse de 2400 bauds.
v est facultatif.
Si v est omis, la vitesse est automatiquement 1200 bauds.
CSAVEX sauvegarde la variable-chaîne spécifiée sur cassette.
*CSAVE** sauvegarde le tableau spécifié sur cassette. Le tableau peut être numérique ou alphanumérique. Les éléments d'un tableau multidimentionné sont sauvegardés avec l'indice changeant le plus rapidement, le plus à gauche.
CSAVEM sauvegarde le contenu de la mémoire en code machine.
S représente l'adresse du premier octet à stocker sur la bande (en décimal) et T représente le nombre d'octets à stocker. L'adresse de départ doit être comprise entre −32768 et 32767.
CSAVES sauvegarde sur la cassette l'image sur l'écran.
CSAVEL saute la bande d'amorce de la bande magnétique.

Exemple
Voir chapitre " Sauvegarde d'un programme sur cassette ".

---

15 CURSORX

(Curseur X)

Syntaxe
CURSORX I

But
Envoyer le curseur en un point spécifié de la ligne sur laquelle il se trouve.

Remarques
I est un numéro de colonne et doit être une expression entière comprise entre 0 et 39.
(En Mode Texte, il est préférable d'avoir I>=1 car pour I=0 le fond devient noir).

Exemple
```
CURSORX 25 : PRINT "A"
```
A s'inscrit à la position 25 de la ligne en cours.

16 CURSORY

(Curseur Y)

Syntaxe CURSORY I

But Fait passer le curseur au numéro de ligne spécifié.

Remarques I est un numéro de ligne, et doit être une expression entière comprise entre 0 et 24.

Exemple
```
CURSORY 10 : CURSORX 19 : PRINT "A"
```
A s'inscrit sur la ligne 10 à la position 19 de la ligne.

---

17 DATA

(Donnée)

Syntaxe DATA liste de constantes

But Stocker des constantes numériques et alphanumériques.
Celles-ci pourront être relues par les instructions READ du programme (Voir READ).

Remarques Les instructions DATA ne sont pas exécutables et peuvent être placées n'importe où dans le programme. Une instruction DATA peut contenir autant de constantes qu'une ligne peut en contenir et un nombre quelconque d'instructions DATA peut être utilisé dans un programme.

Les instructions READ accèdent aux instructions DATA chronologiquement (par numéro de ligne) et les données qui y sont contenues sont considérées comme une liste continue d'éléments, sans tenir compte du nombre d'éléments par ligne ou de la position des lignes dans le programme.

La liste de constantes peut contenir des constantes numériques. (Aucune expression numérique n'est autorisée dans la liste). Les constantes chaînes des instructions DATA doivent se trouver entre guillemets seulement si elles contiennent des virgules, des signes "deux points" ou des espaces significatifs à gauche ou à droite. Sinon, les guillemets ne sont pas nécessaires.

Le type de variable (numérique ou chaîne) donné dans l'instruction READ doit concorder avec la constante correspondante de l'instruction DATA.

Les instructions DATA peuvent être relues à partir du début avec l'instruction RESTORE, ou partiellement avec RESTORE numéro de ligne.

Exemple
```
10  DATA AZ,56,"TY",YU,IO,OP,44,DF,
20  READ A$,B,C$,D$,E$,F$,G
30  PRINT A$;B;C$;D$;E$;F$;G
40  PRINT B+G
RUN
AZ 56TYYUIOOP 44
 100
```

18 DEF FN

(Définition de fonction)

Syntaxe DEF FN nom (paramètre) = expression numérique

But Définir sa propre fonction pour l'utiliser dans un programme.

Remarques DEF FN attribue à une variable - Nom - une fonction qui doit être déterminée par l'utilisateur.
Exemple : AB (Z) = (X + 3)/(X – 1)
AB est le nom de la fonction (X + 3)/(X – 1).
Si Z le paramètre n'est pas indiqué, cette fonction sera exécutée avec la valeur de la variable X contenue dans le programme au moment de l'exécution.
Si le paramètre Z est défini - AB (5) par exemple - la fonction sera calculée avec la valeur 5 pour X.
Paramètre et variable situés dans l'expression numérique sont internes à la fonction DEF FN ; ils n'affectent pas les autres paramètres ou variables de même nom que pourrait contenir le programme.
DEF FN doit être exécutée avant que la fonction qu'elle définit puisse être appelée. Si une fonction est appelée avant d'avoir été définie, un message d'erreur "Fonction non définie" se produit.
DEF FN ne peut contenir que des fonctions numériques.
Les fonctions chaîne sont interdites.
DEF FN peut contenir plusieurs variables mais un seul paramètre.
DEF FN n'est pas utilisable en mode direct.

Exemple

```
10  X=3 : Z=9
20  DEFFN AB(X)=(X+3)/(Z-1)
30  K=FNAB(5)
40  PRINT"X=";X
50  PRINT"K=";K
RUN
X= 3
K= 1
```

---

19 DELIM

(Délimiter)

Syntaxe DELIM I,J,K

But Encadrer et faire ressortir un texte, le souligner.

Remarques I change la couleur des délimiteurs eux-mêmes.
La forme du délimiteur correspond à celle du curseur (code CHR$(127)).
J change la couleur de fond entre les délimiteurs.
I et J sont des expressions entières.

0 = noir	4 = bleu
1 = rouge	5 = violet
2 = vert	6 = turquoise
3 = jaune	7 = blanc

K doit être une expression entière :

0 = normal	1 = soulignage

Cette instruction ne peut être utilisée qu'en mode texte.
Les caractères de texte de la position DELIM à la fin de la ligne reçoivent les caractéristiques spécifiées par DELIM.

Exemple

```
10  INITO
20  FOR I=1 TO 7
30  CURSORX 13:CURSORY 5+I
40  DELIM I,3,1:TXI,0,0
50  PRINT "DELIM";I;3;1;" ";:DELIM I,0,0
60  NEXT I
```

---

20 DIM

(Dimensionner)

Syntaxe DIM liste des variables indicées (nb d'éléments maximum)

But Spécifier le nombre maximal d'indices de variables de tableau et réserver la mémoire en conséquence.

Remarques Si un nom de variable de tableau est utilisé sans une instruction DIM, la valeur maximale de ses indices est fixée à 10.
Si un indice supérieur au maximum spécifié est utilisé, un message d'erreur "Indice hors de limites" se produit.
La valeur minimale d'un indice est 0.
L'instruction DIM donne à tous les éléments des tableaux spécifiés une valeur initiale de 0.
Le nombre d'éléments d'une variable peut être plus petit que le nombre d'éléments réservés dans DIM.

Exemple

```
10    DIMA(16)
20    DATA 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16
30    FOR I=0 TO 15:READA(I)
40    PRINT A(I);
50    NEXT I
60    PRINT "A(0)=";A(0)
RUN
1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16
A(0)= 1
```

21 DISPLAY

(Afficher)

Syntaxe DISPLAY I

But Afficher tous les caractères sur l'écran avec une vitesse déterminée.

Remarques Le paramètre I détermine la vitesse d'affichage sur l'écran.
Pour I = 1 , les caractères apparaissent dans les 20 milli-secondes ;
Pour I = 10 , en 200 milli-secondes, etc.
La valeur maximum pour I est 255.
I est facultatif,
DISPLAY 20 est pris par défaut.
Pour la visualisation d'objet rapide, il est recommandé d'utiliser des valeurs de I comprises entre 1 et 8.
Avec la fonction PLAY on utilisera DISPLAY 1 si des objets animés sont accompagnés par de la musique.

Exemple

```
10 DATA AZ,56," TY",BU,IO,SP,44,DF,
20 READ A$,B ,C$,D$,E$,F$,G
30 PRINT A$;B;C$;D$;E$;F$;G
40 DISPLAY 150: PRINT B+G
RUN
AZ 56 TYBUIOSP 44
 100
```

---

22 EG

(Mode Graphique Spécial)

Syntaxe EG I,J,K

But Permettre d'initialiser le mode graphique spécial.

Remarques I modifie la couleur du caractère défini par l'utilisateur.

0 = noir	4 = bleu
1 = rouge	5 = violet
2 = vert	6 = turquoise
3 = jaune	7 = blanc

J modifie la couleur de fond du caractère, ainsi que la couleur de fond de tout le reste de la ligne (du curseur à la fin).
Ce doit être une expression entière :

0 = noir	4 = bleu
1 = rouge	5 = violet
2 = vert	6 = turquoise
3 = jaune	7 = blanc

K dit être une expression entière :

0 = stable	1 = clignotement

Exemple

```
10   EG 1,4,0
20   CURSORX 10 : CURSORY 10
30   PRINT CHR$(71)
40   SETEG 71,"00112244888844221100"
```

---

23 END

(Fin)

Syntaxe END

But Arrêter l'exécution d'un programme et revenir au niveau des commandes.

Remarques Les instructions END peuvent être placées dans le programme pour arrêter l'exécution. Contrairement à l'instruction STOP, END n'entraîne pas l'affichage d'un message "Arrêt en XX".
Une instruction END à la fin d'un programme est facultative.
Après une instruction END, BASIC-80 vous redonne toujours les commandes.

24 ET

(Mode Texte Spécial)

Syntaxe ET I,J,K

But Permettre d'initialiser le mode texte spécial et de définir les attributs des caractères.

Remarques I modifie la couleur des caractères :

0 = noir	4 = bleu
1 = rouge	5 = violet
2 = vert	6 = turquoise
3 = jaune	7 = blanc

J change l'état des caractères prédéfinis :

0 = normal	4 = inversion vidéo
1 = double hauteur	5 = inversion vidéo et double hauteur
2 = double largeur	6 = inversion vidéo et double largeur
3 = double hauteur et double largeur	7 = inversion vidéo et double hauteur/ largeur

K doit être une expression entière :

0 = stable	1 = clignotement

La couleur de fond des caractères de texte prédéfinie est la couleur de fond de la dernière instruction GR, DELIM ou EG.

Exemple

```
10    ET 4,7,0
20    PRINT CHR$(71);CHR$(71)
30    PRINT CHR$(71);CHR$(71)
40    SETET(71),"00112244888844221100"
```

---

25 EXP (Exponentielle)

Syntaxe EXP(X)

But Donner e élevé à la puissance "x". "x" doit être inférieur ou égal à 87.3365. Si EXP est > 87.3365, le message d'erreur "Dépassement de capacité" est affiché (overflow).

Exemple

```
10    X=5
20    PRINT EXP(X-1)
RUN
54.5982
OK !
```

26 FRE

(Libre)

Syntaxe FRE (0)

But Donner le nombre d'octets encore disponibles dans la mémoire par BASIC-80 au moment ou la question est posée.

Exemple

```
PRINT FRE(0)
13758  (si aucun programme n'a été rentré dans la mémoire)
OK!
```

---

27 FOR ... NEXT .. STEP

(Pour...Prochain..Pas)

Syntaxe FOR variable = x TO y STEP z
Pour... à ..Pas de
NEXT variable
Prochain
où x, y et z sont des constantes ou des expressions numériques.
Nota : Ne laisser qu'un seul espace après l'affichage de x. Si on utilise la numérotation hexadécimale ne pas laisser d'espace après &"....".

But Permettre d'exécuter une série d'instructions un certain nombre de fois : c'est ce qu'on appelle une boucle.

Remarques "Variable" joue un rôle de compteur. La première expression numérique x est la valeur initiale du compteur. Les lignes de programme qui suivent l'instruction FOR sont exécutées alors jusqu'à ce que l'instruction NEXT soit rencontrée.
Le compteur est incrémenté* de la valeur spécifiée par STEP.
Le programme vérifie si la valeur du compteur est supérieure à la valeur finale (y). Si ce n'est pas le cas, BASIC-80 revient à l'instruction suivant l'instruction FOR et le processus recommence.
Si elle est supérieure, l'exécution se poursuit avec l'instruction qui suit l'instruction NEXT. Il s'agit d'une boucle FOR ... NEXT.
Si STEP n'est pas spécifié, l'incrément est de 1.
Si STEP est négatif, la valeur finale du compteur est définie comme étant inférieure à la valeur initiale. Le compteur est décrémenté* à chaque passage dans la boucle, et la boucle est exécutée jusqu'à ce que la valeur du compteur soit inférieure à la valeur finale.
Le corps de la boucle est sauté si la valeur initiale de la boucle multipliée par le signe de STEP dépasse la valeur finale multipliée par le signe de STEP.

Boucles imbriquées

Les boucles FOR ... NEXT peuvent être imbriquées.
Autrement dit, une boucle FOR ... NEXT peut être située à l'intérieur d'une autre boucle FOR ... NEXT. Lorsque les boucles sont imbriquées, chacune doit avoir son propre nom de variable comme compteur.
L'instruction NEXT de la boucle intérieure doit apparaitre avant celle de la boucle extérieure. Si les boucles imbriquées se terminent toutes au même point, une seule instruction NEXT suffit pour toutes les boucles, à condition se spécifier le nom des variables (NEXT K, J, I).
Si une instruction NEXT est rencontrée avant l'instruction FOR correspondante, un message d'erreur "NEXT sans FOR" apparait et l'exécution est terminée.

*Nota : En informatique, incrémenter signifie augmenter et décrémenter, diminuer (en général d'une unité).

Exemple 1

```
10  K=10
20  FOR I=1 TO K STEP 2
30  PRINT I;
40  K=K+10
50  PRINT K
60  NEXT I
RUN
1    20
3    30
5    40
7    50
9    60
OK!
```

Exemple 2

```
10  J=0
20  FOR I=1 TO J
30  PRINT I
40  NEXT I
```

Dans cet exemple, la boucle n'est pas exécutée car sa valeur initiale dépasse sa valeur finale.

Exemple 3

```
10 FOR I=1 TO 3
20 PRINT :PRINT "I=";I
30 FOR J=1 TO 3
40 PRINT "J=";J,
50 FOR K=1 TO 2
60 PRINT "K=";K,
70 NEXT K,J,I
RUN
```

Boucles imbriquées notez les positions respectives de I, J et K dans NEXT.

28 GOSUB...RETURN

(Aller à un sous-programme ... Revenir)

Syntaxe GOSUB numéro de ligne

RETURN

But Aller à un sous-programme et en revenir.

Remarques "Numéro de ligne" correspond à la première ligne du sous-programme.
Un sous-programme peut être appelé un nombre de fois illimité dans un programme, ou être appelé à partir d'un autre sous-programme. La place disponible en mémoire est la seule limite à cette imbrication de sous-programmes.

La ou les instructions RETURN d'un sous-programme provoquent un retour à l'instruction qui suit l'instruction GOSUB qui les a appelées.
Un sous-programme peut contenir plus d'une instruction RETURN, s'il est nécessaire de revenir en des points différents du sous-programme.

Les sous-programmes peuvent se touver n'importe où dans le programme, mais il est recommandé de bien les distinguer du programme principal. Pour empêcher toute intrusion involontaire dans un sous-programme, il faut le faire précéder d'une instruction STOP, END ou GOTO qui permette au programme de le contourner.

Nota : Pour augmenter la vitesse d'exécution, il est préférable de placer les sous-programmes au début du programme.

Exemple

```
10  GOSUB 40
20  PRINT "RETOUR DE SOUS-PROGRAMME"
30  END
40  PRINT "SOUS-PROGRAMME ";
50  PRINT "EN ";
60  PRINT "COURS"
70  RETURN
RUN
SOUS-PROGRAMME EN COURS
RETOUR DE SOUS-PROGRAMME
OK
```

29 GOTO

Syntaxe GOTO numéro de ligne

But Aller inconditionnellement hors de la séquence normale du programme vers un numéro de ligne spécifié.

Remarques Si "numéro de ligne" correspond à une instruction exécutable, cette instruction et les suivantes sont exécutées.
Si c'est une instruction non exécutable, l'exécution reprend à la première instruction exécutable rencontrée après le numéro de ligne.

Exemple

```
10  READ R
20  PRINT "R=";R
30  A=3.14*R ↑ 2
40  PRINT "SURFACE =";A
50  GOTO 10
60  DATA 5,7,12
RUN
R= 5            SURFACE = 78.5
R= 7            SURFACE = 153.86
R= 12           SURFACE = 452.16
DONNEES EPUISEES EN 10
OK!
```

(Après le 3e retour par GOTO 10, READ ne peut plus lire de données puisqu'elles ont déjà été lues).

30 GR

(Graphique)

Syntaxe GR I,J,K

But Initialiser le mode graphique.

Remarques I change la couleur du caractère.
Ce doit être une expression entière :

0 = noir	4 = bleu
1 = rouge	5 = violet
2 = vert	6 = turquoise
3 = jaune	7 = blanc

J change la couleur du fond sur lequel se trouve le caractère depuis la position du curseur jusqu'à la fin de la ligne.
Ce doit être une expression entière :

0 = noir	4 = bleu
1 = rouge	5 = violet
2 = vert	6 = turquoise
3 = jaune	7 = blanc

K doit être une expression entière :

0 = stable	1 = clignotement

Si I, J, K sont omis, GR prend la valeur GR 7,0,0

Exemple

```
10   CURSORX 5: CURSORY 10
20   GR 1, 4, 0 : PRINT "AZERTYUIOP"
```

A la ligne 10, la couleur de fond sera bleue à partir de la position 5 jusqu'à 39. Chaque caractère à partir de la position 5 de cette ligne sera un caractère graphique rouge stable.

31 IF ... THEN

(Si... Alors)

IF ... GOTO

(Si... Aller à)

Syntaxe IF expression THEN instruction(s) ou numéro de ligne

Syntaxe IF expression GOTO numéro de ligne

But Prendre une décision concernant le déroulement du programme, d'après le résultat retourné par une expression.

Remarques Si le résultat de "expression" n'est pas zéro, la clause THEN ou GOTO est exécutée.

THEN peut être suivi par un numéro de ligne pour le branchement, ou par une ou plusieurs instructions à exécuter.

GOTO est toujours suivi d'un numéro de ligne.

Si le résultat de "expression" est zéro, la clause THEN ou GOTO est ignorée. L'exécution continue la ligne exécutable suivante.

Dans "Expression" éviter les valeurs hexadecimales.

Exemple

```
10  INPUT I
100 IF (I<20) AND (I>10) GOTO 300
110 IF (I<5) OR (I>25) GOTO 500
120 PRINT "HORS LIMITES"
200 END
300 PRINT "GAGNE"
400 END
500 PRINT "PERDU"
```

Dans cet ensemble, un test détermine si I est supérieur à 10 et inférieur à 20.
Si I est dans ces limites, un branchement a lieu à la ligne 300.
Si I n'est pas dans ces limites, l'exécution continue à la ligne 110.
Le test de la ligne 110 détermine si I est inférieur à 5 ou supérieur à 25.
Dans ce cas, l'exécution passe à la ligne 500.

Important : VG5000 accepte sur une même ligne l'instruction
IF ... Then ... : GOTO

Dans ce cas si la condition n'est pas respectée, la clause (THEN) n'est pas exécuté *et le branchement (GOTO) est ignoré :* le programme se poursuit à la ligne suivante.

32 INIT

(Initialiser)

Syntaxe INIT I,J

But Initialiser la couleur de l'écran.

Remarques L'écran est effacé et prend la couleur de fond désirée.
I est le numéro de couleur de fond.
J est le numéro de la couleur du bord.
I et J doivent être des expressions entières :

0 = noir	4 = bleu
1 = rouge	5 = violet
2 = vert	6 = turquoise
3 = jaune	7 = blanc

Le curseur est positionné en ligne 1, position 1, en commande.
L'instruction INIT prend automatiquement les instructions :
TX0,0,0:SCROLL:DISPLAY 20
Si vous appuyez sur [CTRL] + [⏏] , l'instruction INIT6,6 est exécutée.

Exemple :

```
10   INIT0,0
RUN
```

donne un écran noir
(Pour revenir à l'état initial INIT6,6, faites [CTRL] + [⏏])

33 INPUT

(Entrer)

Syntaxe INPUT "chaîne de caractères" ; liste de variables

But Permettre l'entrée des données à partir du clavier pendant l'exécution du programme.

Remarques Lorsqu'une instruction INPUT est rencontrée, l'exécution du programme s'arrête et un point d'interrogation s'affiche pour indiquer que le programme attend des données. Si "chaîne de caractères" est inclus, la chaîne précède le point d'interrogation. Les données requises sont alors entrées sur le terminal.

Les données entrées sont affectées à la ou aux variables données dans la "liste de variables". Le nombre d'éléments de données fourni doit être le même que le nombre de variables de la liste. Les éléments de donnée sont séparés par les virgules.

Les noms de variables de la liste peuvent être des noms de variables numériques ou alphanumériques (y compris des variables indicées). Le type de chaque élément de données entrées doit correspondre au type spécifié dans le nom de la variable. (Les chaînes entrées dans une instruction INPUT n'ont pas à être entre guillemets).

Si vous répondez à INPUT avec trop ou trop peu d'éléments, ou avec un type de valeur incorrect (numérique au lieu de chaîne, etc.), vous obtenez le message "?". Recommencez au début. Aucune valeur entrée n'est prise en compte tant qu'une réponse acceptable n'est pas donnée.

INPUT est interdit en mode direct.

Pour donner à une variable numérique la valeur zéro, entrez zéro.
Pour donner à une variable chaîne la valeur nulle, entrez un point d'interrogation.

Un "?" n'est pas permis dans "chaîne de caractères".

Entrée de plusieurs variables : il est possible d'entrer plusieurs variables les unes à la suite des autres en les séparant par des virgules et en les validant toutes ensembles à la fin (voir programme Chapitre 12 ligne 60).

Exemple 1

```
10 INPUT X
20 PRINT X" AU CARRE DONNE" X ↑ 2
30 END
RUN
? 5      (Le 5 est frappé par l'utilisateur en réponse au point d'interrogation)
5 AU CARRE DONNE 25
OK!
```

Exemple 2

```
10 PI=3.14
20 INPUT "QUEL EST LE RAYON";R
30 A=PI*R ↑ 2
40 PRINT "LA SURFACE DU CERCLE EST";A
50 PRINT
60 GOTO 20
OK
RUN
QUEL EST LE RAYON?  7.4(L'UTILISATEUR TAPE 7.4)
LA SURFACE DU CERCLE EST DE 171.946
```

34 INT

(Entière)

Syntaxe INT(X)

But Donner la partie entière de X.

Exemple

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

---

35 KEY (0)

(Touche)

Syntaxe KEY (0)

But Donner la valeur ASCII représentant le caractère de la touche enfoncée.

Remarques Pour les lettres de A à Z, c'est la valeur ASCII de la lettre minuscule qui est donnée.
Pour obtenir la valeur ASCII de la lettre majuscule, la touche [↑] blocage en minuscule est inopérante ; par contre, les touches [↑] et [CTRL] restent actives.

Exemple 1

```
10 PLAY"T10 O1"
20 IF KEY(0)= 113 THEN PLAY "A"
30 IF KEY(0)= 115 THEN PLAY "B"
40 IF KEY(0)= 100 THEN PLAY "C"
50 IF KEY(0)= 102 THEN PLAY "D"
60 IF KEY(0)= 103 THEN PLAY "E"
70 IF KEY(0)= 104 THEN PLAY "F"
80 IF KEY(0)= 106 THEN PLAY "G"
90 IF KEY(0)= 107 THEN PLAY "O2 A"
100 IF KEY(0)= 108 THEN PLAY "O2 B"
110 IF KEY(0)= 109 THEN PLAY "O2 C"
120 GOTO 10
```

Les touches Q, S, D, F, G, H, J, K, L, M permettent de jouer les notes LA, SI, DO, RE, MI, FA, SOL, LA, SI, DO.

Exemple 2

```
10 IF KEY(0)=0 OR KEY(0)=13  GOTO 10
20 A=KEY(0)
30 PRINT CHR$(A);"=";KEY(0),
40 FOR I=1 TO 100:NEXT I
50 GOTO 10
RUN      (Taper n'importe quelle touche alphanumérique)
```

ATTENTION : `0` = zéro - `O` = lettre

36 LEFT$

(Gauche)

Syntaxe LEFT$ (XS,I)

But Donner une chaîne comprenant les I caractères le plus à gauche de X$.
I doit être compris entre 0 et 255.
Si I est supérieur à LEN(X$), la totalité de la chaîne (X$) sera retournée.
Si I = 0, la chaîne nulle (longueur zéro) sera retournée.

Exemple

```
10    A$="BASIC-80"
20    B$=LEFT$(A$,5)
30    PRINT B$
RUN
BASIC
```

---

37 LEN

(Longueur)

Syntaxe LEN(X$)

But Indiquer le nombre de caractères d'une chaîne. Les caractères non imprimables et les espaces sont comptés.

Exemple

```
10    X$="PORTLAND, OREGON"
20    PRINT LEN(X$)
RUN
16
OK
```

---

38 LET

(Soit)

Syntaxe LET variable = expression

But Affecter la valeur d'une expression à une variable.

Remarques Notez que le mot LET est facultatif. Autrement dit, le signe égal est suffisant pour affecter une expression à un nom de variable.

Exemples

```
110   LET D=12
120   LET E=12 ↑ 2
130   LET F=12 ↑ 4
140   LET S=D+E+F
150   PRINT S
```

ou

```
110   D=12
120   E=12 ↑ 2
130   F=12 ↑ 4
140   S=D+E+F
150   PRINT S
RUN
 20 892
```

39 LIST

(Lister)

Syntaxe LIST numéro de ligne de départ, numéro de ligne d'arrivée

But Lister tout ou partie du programme en mémoire sur le terminal.

Remarques BASIC-80 revient toujours au niveau des commandes après l'exécution de LIST.

Si "numéro de ligne" est omis, le programme est listé à partir du numéro de ligne le plus bas. Le listage est interrompu soit par la fin de la liste du programme, soit par la touche ↑ + STOP RUN .

Si "numéro de ligne" est inclus, BASIC-80 liste le programme à partir de cette ligne.

Si "numéro de ligne d'arrivée" est donné, BASIC-80 listera le programme entre le numéro de ligne de départ et le numéro de ligne d'arrivée.

Pour interrompre momentanément le listage du programme, appuyez sur STOP RUN

Chaque fois que vous frapperez sur STOP RUN . BASIC-80 listera une ligne supplémentaire du programme. Pour relancer le listage, il suffit d'appuyer sur une touche quelconque.

Exemple

```
LIST                (liste le programme actuellement en mémoire)
LIST 500,550        (Liste toutes les lignes du programme de 500 à 550 incluse)
LIST 200            (Liste toutes les lignes depuis 200 jusqu'à la fin)
LIST 200-200        (Ne liste que la ligne 200)
LIST-200            (Liste depuis le début jusqu'à la ligne 200).
```

---

40 LLIST

(Lister sur papier)

Syntaxe LLIST numéro de ligne de départ, numéro de ligne d'arrivée.

But Imprimer sur papier à l'aide d'une imprimante la liste d'un programme.

Remarques Cette instruction est similaire à l'instruction LIST (voir cette instruction) qui sert à donner la liste d'un programme sur l'écran.

Cette instruction n'est utilisable que si l'on possède l'interface VG5216 et une imprimante VW0010 ou VW0020 (en préparation).

41 LOAD

(Charger)

Syntaxe LOAD aaaa, bbbb

But Permet de charger des programmes enregistrés en code ASCII.

Remarques Cette commande permet de charger un programme en BASICODE.

aaaa Représente la 1re ligne à charger

bbbb Représente la dernière ligne à charger

Si la mémoire contient déjà un programme, le nouveau programme se mélange à lui, les lignes ayant les mêmes numéros du 1er programme sont effacées et remplacées par celle du nouveau.

Les nº de ligne différents s'insèrent entre les lignes déjà existantes.

Pour reconstruire les lignes du nouveau programme une partie de la mémoire de programme est utilisée temporairement.

De ce fait il peut arriver qu'un message d'erreur "Sortie de mémoire" apparaisse si le programme est très important ; il faut dans ce cas charger le programme en plusieurs fois en faisant "LOAD" à partir de la dernière ligne acceptée.

Si un programme peut être chargé de cette manière, on exécute un chargement aussi loin que possible puis on le sauvegarde par un "CSAVE", c'est-à-dire normalement ; on efface ensuite la mémoire on charge la partie suivante que l'on sauvegarde également par un nouveau CSAVE.

La 1re partie sera chargée par CLOAD puis la 2e partie par CLOADA, on a alors un programme complet que l'on traite normalement.

Lorsque le programme est important, il peut y avoir un temps assez long entre l'arrêt du magnétophone et l'accessibilité au clavier, un message "PAUSE" est alors affiché indiquant que l'interprétation est en cours. Attendre OK! (cela peut demander plusieurs secondes).

---

42 LPOS(X)

(Positionner sur imprimante)

Syntaxe LPOS(X)

But Donner la position de la tête d'imprimante après le dernier caractère imprimé.

Remarque Cette instruction est similaire à l'instruction POS(X) (excepté Pos (255)). La valeur de X peut être comprise entre 0 et le nb maximum de colonne de l'imprimante.

43 LPRINT

(Imprimer sur imprimante)

Syntaxe LPRINT liste d'expression

But Imprimer sur papier à l'aide d'une imprimante.

Remarques LPRINT est similaire à l'instruction PRINT (voir cette instruction). Lorsqu'on désire imprimer sur papier les résultats d'un programme, il suffit de remplacer dans la liste du programme tous les PRINT par LPRINT.
Cette instruction n'est utilisable que si l'on possède l'interface VG5216 et une imprimante VW0010 ou VW0020.

---

44 LOG

(Logarithme)

Syntaxe LOG(X)

But Donner le logarithme naturel de X. X doit être supérieur à zéro.

Exemple

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

---

45 MID$

(Milieu)

Syntaxe MID$ (X$,I,J)

But Donner une chaîne d'une longueur de J caractères de X$ à partir du I^e^ caractère.

Remarques I et J doivent être compris entre 0 et 255.
Si J est omis ou s'il y a moins de J caractères à droite du I^e^ caractère, tous les caractères à droite du I^e^ caractère sont retournés.
Si I > LEN(X$), MID$ retourne une chaîne nulle.

Exemple

```
10    A$="BON"
20    B$="JOUR APRES-MIDI SOIR"
30    PRINT A$;MID$(B$,5,11)
RUN
BON APRES-MIDI
OK
```

46 NEW

(Neuf)

Syntaxe NEW

But Supprimer le programme actuellement en mémoire et effacer toutes les variables.

Remarques NEW est entré au niveau des commandes pour effacer la mémoire avant d'entrer un nouveau programme. BASIC-80 revient toujours au niveau des commandes après l'exécution de NEW.

Exemple NEW puis RET

---

47 NOT

(non Logique)

Syntaxe NOT X

But Donne le complément à 1 de la valeur entière d'un nombre ou d'une expression

---

48 ON ... GOSUB

(Sur -résultat d'expression- ... Aller au sous-programme)

et ON ... GOTO

(Sur -résultat d'expression- ... Aller à telle ligne)

Syntaxe ON expression GOTO liste de numéros de ligne
ON expression GOSUB liste de numéros de ligne

But Brancher à un ou plusieurs numéros de ligne spécifiés, selon la valeur donnée par l'expression.

Remarques La valeur de "expression" détermine le numéro de ligne qui sera utilisé pour le branchement. Par exemple, si la valeur est 3, le troisième numéro de ligne de la liste sera la destination du branchement. (Si la valeur n'est pas entière, la partie décimale sera arrondie).

Dans l'instruction ON ... GOSUB, chaque numéro de ligne de la liste doit être le premier de ligne d'un sous-programme.

Si la valeur de "expression" est négative ou supérieure à 255, un message d'erreur "Appel de fonction incorrecte" est affiché.

Si "expression" = 0, GOTO ou GOSUB est ignoré et le programme continue à la ligne suivante.

Exemple

```
10 INPUT L
20 ON L GOSUB 40,50,60
30 PRINT"L EST EGAL A 0 OU SUPERIEUR A 3":END
40 PRINT "L=1":END
50 PRINT "L=2":END
60 PRINT "L=3":END
```

---

49 OR

(Ou)

Syntaxe OR

But Effectuer l'opération logique OU.

Remarques Faire une action déterminée si un évènement OU un autre arrive.

Exemple

```
5     PRINT "Choisissez 2 nombres X et Y compris entre 0 et 10"
10    INPUT X
20    INPUT Y
30    IF X<=3 OR Y>=5 THEN 60
40    PRINT "PERDU"
50    GOTO 10
60    PRINT "GAGNE"
```

---

50 PAGE

(Page)

Syntaxe PAGE

But Figer l'écran

Remarques Après cette instruction, l'écran ne peut plus défiler vers le haut ou vers le bas. Pour faire défiler à nouveau, utiliser l'instruction SCROLL.

Exemple

```
10 INIT1,3:PAGE
20 FOR I=1 TO 200:PRINT I:NEXT I
30 GOTO 30
```

1) Remplacer GOTO 30 par END [RET]
et regardez
2) Ligne 10 supprimer PAGE [RET]
et regarder

51 PEEK

(Lire) *(terme purement informatique)*

Syntaxe PEEK(I)

But Lire l'octet à la position mémoire I.
I doit être compris entre (-&"8000") et (&"7FFF").

Exemple

```
10    A=PEEK(&"4010")
20    PRINT A
RUN
32
```

---

52 PLAY

(Jouer - de la musique)

Syntaxe PLAY "chaîne de caractères"

But Produire une mélodie définie par "chaîne de caractères".

Remarques "Chaîne de caractères" se compose de notes et de paramètres définissant la valeur de la note, la durée de la note, l'octave, le temps, les dièses, les bémols, les pauses.
Suivant la notation anglo-saxonne, les notes sont représentées par des lettres :

A = LA	D = RE	G = SOL
B = SI	E = MI	+ DIESE
C = DO	F = FA	− BEMOL

La pause est représentée par R.

Exemple : SI BEMOL = B−

La durée d'une note est donnée par le nombre qui suit la note.
Ce nombre est compris entre 0 et 99.

Exemple : G+25

La mélodie peut s'étendre sur 4 octaves. Un octave est représenté par O (lettre) suivi d'un chiffre de 1 à 4 qui définit la hauteur de l'octave.
1 définit l'octave le plus bas.

Exemple : O1

Le tempo est défini par T suivi d'un nombre compris entre 1 et 255.
1 est le tempo le plus rapide.

Exemple : T115

Une mélodie sera construite en définissant :

1 - le tempo
2 - l'octave
3 - les notes avec leurs caractéristiques #, b, durée.

Exemple de mélodie

```
5 CLEAR 214
10 A$="T20 02 CECEGGG32 03 ACBA 02 G48 R
20 B$="T20 02 CECEGGG32 03 A 02 GFED48 R
30 C$="T20 02 CECEGGG32 03 ACBAB48 R
40 D$="T30 03 ECBA 02 G 03 C 02 EG 03 A48 RB48 RC64.R
50 PLAY A$+B$+C$+D$
```

ATTENTION : 0 = zéro - O = lettre

Important : *Si la fonction PLAY est introduite dans un programme où un affichage variable est utilisé (boucle FOR ... NEXT), il est nécessaire de placer une instruction DISPLAY avant l'instruction pour que l'affichage se fasse dans l'ordre.*

---

53 POKE

(Ecrire en mémoire, terme purement Informatique)

Syntaxe POKE I, J

But Écrire un octet dans une position mémoire.

Remarques L'expression entière I est l'adresse de la position mémoire réceptrice (par POKE).
L'expression entière J est la donnée à insérer à l'adresse I.
J doit être compris entre 0 et 255.
I doit être compris entre –32768 et 32767 (-&"8000" et &"7FFF").
Noter que les données "Pokées" entre les adresses mémoires 0 et & "3FFF" incluse n'auront aucun effet puisqu'il s'agit de l'espace ROM. Il en est de même entre (-&"8000") et -&"1" avec VG5000 16 K RAM (voir carte mémoire Annexe 2).
La fonction complémentaire de POKE est PEEK.
L'argument de PEEK est une adresse à partir de laquelle on peut lire un octet.
POKE et PEEK sont utiles dans les cas suivants : stockage des données, chargement de sous-programmes en langage d'assemblage, échange d'arguments et de résultats avec des sous-programmes en langage d'assemblage.

Exemple

```
10    POKE 23040,255
```

54 POS

(Position)

Syntaxe POS(X)

But Donner la position en ligne du curseur. La position à gauche la plus éloignée est 0.
X est un argument fictif.

Exemple

```
10    PRINT "AZERTYU";
20    A=POS(X)
30    PRINT A
RUN
AZERTYU 8 (8 EST LA POSITION DU CURSEUR)
```

Nota : POS(255) donne la position verticale du curseur.

---

55 PRINT

(Afficher)

Syntaxe PRINT liste d'expressions

But Afficher des données sur l'écran.

Remarques Si "liste d'expressions" est omis, une ligne vierge est affichée.
Si "liste d'expressions" est inclus, les valeurs des expressions sont affichées sur le terminal.
Les expressions de la liste peuvent être des expressions numériques et/ou des expressions chaînes (les chaînes doivent se trouver entre guillemets).

Positions d'impression La position de chaque élément affiché est déterminée par la ponctuation qui sépare les éléments de la liste (Voir chapitre ponctuation).
Si l'impression continue au delà de la 40e colonne (la dernière position sur l'écran), un retour automatique s'effectue, les données continuent à être affichées à partir de la 1re position de la ligne suivante.
(Il faut noter que la position 0 n'est possible que si un délimiteur n'a pas été placé au départ de la ligne, par exemple en GR - mode graphique).
Un point d'interrogation peut remplacer le mot PRINT dans une instruction PRINT.

Exemple 1

```
10    X=5
20    PRINT X+5, X-5, X*(-5),
30    END
RUN
10          0          -25
OK
```

Dans cet exemple, les virgules de l'instruction PRINT entraînent l'impression de chaque valeur au début de la zone d'impression suivante.

Exemple 2

```
10 INPUT X
20 PRINT X"AU CARRE EST" X ↑ 2 "ET";
30 PRINT X"AU CUBE" X  ↑ 3
40 PRINT
50 GOTO 10
RUN
? 9
 9 AU CARRE EST 81 ET 9 AU CUBE 729

? 21
 21 AU CARRE EST 441 ET 21 AU CUBE 9261

?
```

Dans cet exemple, le point-virgule à la fin de la ligne 20 entraîne l'impression des deux instructions PRINT sur la même ligne, et la ligne 40 entraîne l'impression d'une ligne blanche avant le point d'interrogation suivant.

Exemple 3

```
10 FOR X=1 TO 4
20 J=J+5
30 K=K+10
40 ?J;K;"A";
50 NEXT X
RUN
 5 10A 10 20A 15 30A 20 40A
OK
```

Dans cet exemple, les points-virgules de l'instruction PRINT entraînent l'impression de chaque valeur immédiatement après la valeur précédente. Se rappeler que les nombres positifs sont précédés d'un espace. A la ligne 40, un point d'interrogation remplace le mot PRINT.

56 READ

(Lire)

Syntaxe READ liste de variables

But Lire des valeurs à partir d'une instruction DATA et les affecter à des variables (Voir DATA).

Remarques Une instruction READ doit toujours être utilisée conjointement à une instruction DATA. Les instructions READ affectent des variables aux valeurs de l'instruction DATA, une à une. Les variables de l'instruction READ peuvent être numériques ou de chaînes, et les valeurs lues doivent correspondre aux types de variables spécifiés.
S'il n'y a pas concordance, il en résultera "erreur de syntaxe".
Une seule instruction READ peut accéder à une ou plusieurs instructions DATA (l'accès se fera dans l'ordre). De même, plusieurs instructions READ peuvent accéder à une seule instruction DATA. Si le nombre de variables de "liste de variables" dépasse le nombre d'éléments contenus dans la ou les instructions DATA, un message "DATA épuisées en XX" apparait. (XX représentant le N° de ligne).
Si le nombre de variables spécifié dans READ est inférieur au nombre d'éléments contenus dans la ou les instructions DATA, les instructions READ suivantes commenceront à lire les données à partir du premier élément non lu.
S'il n'y a pas d'instructions READ suivantes, les données supplémentaires sont ignorées.
Pour relire les instructions DATA à partir du début, utilisez l'instruction RESTORE (Voir RESTORE).

Exemple 1

```
80 DATA 3.08,5.19,3.12,3.98,4.24
90 FOR I=1 TO 5
100 READ A(I)
110 PRINT"A(";I;")=";A(I)
120 NEXT I
```

Ce segment de programme affecte les valeurs des instructions DATA au tableau A. Après exécution, la valeur de A(1) sera de 3.08, etc

Exemple 2

```
10 PRINT "CITY", "STATE", "ZIP"
20 READ C$,S$,Z
30 DATA "DENVER",COLORADO, 80211
40 PRINT C$,S$,Z
RUN
CITY          STATE              ZIP
DENVER        COLORADO           80211
OK
```

Ce programme lit (READ) des données chaînes et des données numériques à partir de l'instruction DATA à la ligne 30.

57 REM

(Remarque)

Syntaxe REM remarque

But Permettre l'insertion de commentaires explicatifs dans un programme.

Remarques Les instructions REM ne sont pas exécutées mais apparaissent exactement comme elles ont été rentrées, lorsque le programme est listé.

Les instructions REM peuvent être la destination d'un branchement (à partir d'une instruction GOTO ou GOSUB) ; l'instruction à exécuter est alors la première instruction exécutable après l'instruction REM rencontrée.

Les instructions REM apparaissent en Rouge dans les programmes.

Exemple

```
140 REM CALCUL DE LA VALEUR MOYENNE
150 FOR I=1 TO 19
160 SOM=SOM+I
170 NEXT I
180 MOY=SOM/19
190 PRINT"MOY=";MOY
```

---

58 RENUM

(Renuméroter)

Syntaxe RENUM X,Y,Z

But Renuméroter les lignes d'un programme.

Remarques Lorsqu'on corrige un programme, il est souvent nécessaire d'ajouter des lignes entre des lignes déjà créées, ce qui donne un programme dont les lignes ne se suivent pas régulièrement (10 en 10 par exemple).

X indique le numéro de la ligne de départ de la renumérotation.

Y indique le numéro de ligne à partir de laquelle la renumérotation doit être effectuée.

Z donne le pas de la renumérotation.

La renumérotation se fait à partir d'une ligne jusqu'à la fin (X est toujours supérieur ou égal à Y).

X, Y, Z sont facultatifs. Par défaut la renumérotation se fait à partir de la ligne 10 jusqu'à la fin du programme avec un pas de 10.

Exemple

```
RENUM 16200,20,15
```

59 RESTORE

(Relire des DATA)

Syntaxe RESTORE numéro de ligne

But Permettre la relecture d'instructions DATA à partir d'un point spécifié.

Remarques Après l'exécution d'une instruction RESTORE, l'instruction READ suivante accède au premier élément de la première instruction DATA du programme. "Numéro de ligne".
Si un numéro de ligne est spécifié, l'instruction READ suivante accède au N° de ligne spécifié.

Exemple

```
10 READ A,B,C
20 READ D,E,F
30 DATA 1,2,3
40 DATA 4,5,6
50 RESTORE 40
70 READ G,H,I
80 PRINT "A=";A,"B=";B,"C=";C
90 PRINT "D=";D,"E=";E,"F=";F
100 PRINT "G=";G,"H=";H,"I=";I
RUN
A= 1          B= 2          C= 3
D= 4          E= 5          F= 6
G= 4          H= 5          I= 6
```

---

60 RIGHT$

(Droite)

Syntaxe RIGHT$ (X$,I)

But Donner les I caractères les plus à droite d'une chaîne XS.
I doit être compris entre 0 et 255.
Si I est supérieur à LEN(X$), la totalité de la chaîne (X$) sera retournée.
Si I = 0, la chaîne 0 (longueur 0) est retournée.

Exemple

```
10 A$="NICE COTE D'AZUR"
20 PRINT RIGHT$(A$,11)
RUN
COTE D'AZUR
OK!
```

61 RND

(Aleatoire)

Syntaxe RND(X)

But Donner un nombre aléatoire compris entre 0 et 1.
Si X <0, RND répète le dernier numéro généré.
Si X >0, RND génère le nombre aléatoire suivant dans la séquence.

Exemple

```
10 FOR I=1 TO 5
20 PRINT INT(RND(+1)*100);
30 NEXT I
RUN
 49 67 98 73 78
OK!
```

---

62 RUN

(Mettre en marche)

Syntaxe RUN numéro de ligne

But Exécuter le programme qui se trouve en mémoire.

Remarques "Numéro de ligne" est facultatif.
Si "numéro de ligne" est spécifié, l'exécution commence à partir de cette ligne.
Sinon, l'exécution commence au numéro de ligne le plus petit.
BASIC-80 revient toujours au niveau des commandes après l'exécution de RUN.

Exemple

```
RUN
```

63 SAVE

(Sauver)

Syntaxe SAVE (v) aaaa, bbbb

But Sauvegarder un programme BASIC en code ASCII.

Remarques Cette commande permet de faire un enregistrement compatible BASICODE.
(v) représente la vitesse d'enregistrement.
pour v = 1 la vitesse d'enregistrement est de 1200 BAUDS
pour v = 2 la vitesse est de 2400 BAUDS
aaaa représente la 1re ligne à enregistrer
bbbb représente la dernière ligne à enregistrer.
Il n'y a pas de nom de programme.

ATTENTION : *Cette commande efface le programme de la mémoire ; il est donc prudent de sauvegarder le programme BASIC par un CSAVE avant de faire un enregistrement avec "SAVE".*
Un message d'erreur "Sortie de Mémoire" peut arriver s'il n'y a pas assez de place en mémoire.
Dans ce cas recharger le programme par CLOAD puis sauvegarder le programme avec "SAVE" par parties.

---

64 SCREEN

(Écran)

Cette fonction n'est pas utilisée dans VG5000 mais ne donne pas de message d'erreur si elle apparaît dans un programme.
Permet la compatibilité avec des programmes écrits pour les Jeux G7400, Jet 741, Jopac équipés de l'extension Basic C7420.

---

65 SCROLL

(Dérouler)

Syntaxe SCROLL

But Faire défiler l'écran à nouveau après une instruction PAGE.

Remarques Après cette instruction, l'écran peut défiler vers le haut et vers le bas.
Pour figer l'écran, utilisez l'instruction PAGE.

Exemple Voir PAGE.

66 SETEG

(Point caractère graphique)

Syntaxe SETEG I, "JJJJJJJJJJJJJJJJJJ"

But Permettre de définir un caractère graphique spécial.

Remarques I est le code du caractère. Il doit être compris entre 032 et 127.
J est le code d'une demi-ligne ; une ligne est définie par JJ.
J doit être utilisé dans cette instruction vingt fois (10 lignes).
J doit être compris entre 0 et F (code hexadécimal).

Exemple

```
SETEG 088,"1C361C087F5D5D1C3663"
```

C'est la définition du caractère graphique présenté à la page 31
Ce caractère a le code 088.

---

67 SETET

(Point caractère texte)

Syntaxe SETET I,"JJJJJJJJJJJJJJJJJJ"

But Permettre dè définir un caractère de texte spécial.

Remarques I est le code du caractère. Il doit être compris entre 032 et 127.
J est le code de matrice d'une demi-ligne : JJ définissent une ligne.
J doit être utilisé dans cette instruction vingt fois (10 lignes).
J doit être compris entre 0 et F (code hexadecimal).

Exemple

```
10   SETET 65, "003844444447C44440000"
```

C'est la définition de la lettre A.
Cette lettre a le code de caractère 65.

68 SGN

(Signe)

Syntaxe SGN(X)

Action Si X > 0, SGN(X) donne + 1
Si X = 0, SGN(X) donne 0
Si X < 0, SGN(X) donne − 1

Exemple

```
10 INPUT X
20 ON SGN(X)+2 GOTO 40, 50, 60
30 PRINT "VG 5000"
40 PRINT "BASIC 80" :END
50 PRINT "MICROSOFT":END
60 PRINT "MADE IN FRANCE"
70 GOTO 10
RUN
? 9
MADE IN FRANCE
```

Dans cet exemple on ne passe jamais par la ligne 30.

---

69 SIN

(Sinus)

Syntaxe SIN(X)

But Donner la valeur de Sinus(X)
COS(X)=SIN(X+3.14159/2)

Remarques X doit être exprimé en Radian.

Exemple

```
PRINT SIN(1.5)
.997495
OK
```

70 SOUND

(Son)

Syntaxe SOUND I,J,K.

But Emettre un son dont les caractéristiques sont données par les paramètres I, J, K.

Remarques La fréquence du son est donnée par le paramètre I. I doit être compris entre 0 et 255.
La durée du son est donnée par J qui doit également être compris entre 0 et 255.
K est un paramètre optionnel qui modifie le rapport cyclique de l'onde produite, la valeur par défaut est 0. K peut être compris entre 0 et 255.
I,J,K doivent être des valeurs entières.

Exemple

```
10    SOUND 255,12,4
20    SOUND 10,255
RUN
```

(et Ecouter)

---

71 SPC

(Espace)

Syntaxe SPC(I)

But Afficher I espaces sur l'écran ou sur imprimante. SPC ne peut être utilisé qu'avec des instructions PRINT ou LPRINT.
I doit être compris entre 0 et 255.

Exemple

```
PRINT "NOUS";SPC(14);"AUSSI"
NOUS              AUSSI
OK
```

---

72 SQR

(Racine carrée)

Syntaxe SQR(X)

But Donner la racine carrée de X. (X doit être > = 0).

Exemple

```
10    X=10
20    PRINT X;SQR(X)
RUN
10    3.16228
OK
```

73 STICKX - STICKY

(Joystick : manette)

Syntaxe STICKX(I)

But Donner la valeur "gauche" ou "droite" des commandes manuelles ou la valeur des touches [←] , [→] du clavier à droite.
La valeur donnée est 255 si la commande est poussée vers la gauche et de 1 si la commande est poussée vers la droite.
La valeur est 0 lorsque la commande est au repos.

Syntaxe STICKY(I)

But Donner la valeur "haut" ou "bas" des commandes manuelles ou la valeur [↑], [↓] des touches du clavier.
La valeur donnée est 255 si la commande manuelle est poussée vers le haut et 1 si la commande manuelle est poussée vers le bas.
La valeur est 0 lorsque la commande manuelle est au repos.

Remarques I = 0 pour la manette de droite
I = 1 pour la manette de gauche
I = 2 pour les touches [←] [→] [↑] [↓] du clavier.

Exemple

```
10 INIT2: CX=19:CY=12
20 TX4:DISPLAY 3
30 CURSORX CX:CURSORY CY:PRINT "O";
40 IF STICKX(2)=1 THENCX=CX+1
50 IF STICKX(2)=255 THENCX=CX-1
60 IF STICKY(2)=1 THENCY=CY+1
70 IF STICKY(2)=255 THENCY=CY-1
80 GOTO30
90 END
```

74 STOP

(Arrêt)

Syntaxe Stop

But Arrêter l'exécution du programme et revenir au niveau des commandes.

Remarques Les instructions STOP peuvent être utilisées n'importe où dans un programme pour arrêter l'exécution. Lorsqu'une instruction STOP est rencontrée, le message suivant s'imprime :

"ARRET en ligne xx"

BASIC-80 revient toujours au niveau des commandes après l'exécution de STOP.

La commande CONT permet de reprendre l'exécution.

Exemple

```
10    INPUT A, B, C
20    K=A ↑ 2*5.3:L=B ↑ 3/.26
30    STOP
40    M=C*K+100:PRINT M
RUN
? 1, 2, 3
ARRET EN 30
OK!
PRINT L
30.7692
OK!
CONT          puis [RET]
115.9
OK!
```

---

75 STORE

(Stocker)

Cette fonction n'est pas utilisée dans VG5000 mais ne donne pas de message d'erreur si elle apparaît dans un programme.

Permet la compatibilité avec des programmes écrits pour les Jeux G7400, Videopac 74 + Jet 741, Jopac équipés de l'extension BASIC C7420.

76 STR$

(String = chaîne de caractères)

Syntaxe STR$(X)

But Transformer une variable ou un nombre en chaîne de caractères.

Exemple

```
10 N= 1984
20 A$=STR$(N)
30 B$="JUIN"
40 PRINT B$+A$
RUN
JUIN 1984
```

---

77 TAB

(Tabulation)

Syntaxe TAB(I)

But Faire des espaces jusqu'à la position I sur l'écran ou sur imprimante.
Si la position d'impression en cours est déjà au-delà de la position I, TAB est sans effet.
TAB(1) est la position la plus à gauche.
I doit être compris entre 1 et 39.
TAB ne peut être utilisé que dans les instructions PRINT et LPRINT.

Exemple

```
PRINT "NOM";TAB(20);"MONTANT"
NOM                  MONTANT
OK
```

---

78 TAN

(Tangente)

Syntaxe TAN(X)

But Donner la valeur de tangente X.

Remarques X est exprimé en Radian.

Exemple

```
10 X=15
20 Y=2*TAN(X)/2
RUN
- 855994
OK
```

79 TX

(Mode texte)

Syntaxe TX I,J,K

But Initialiser le mode texte.

Remarques I modifier la couleur d'avant-plan. Ce doit être une expression entière :

0 = noir	4 = bleu
1 = rouge	5 = violet
2 = vert	6 = turquoise
3 = jaune	7 = blanc

J change l'état des caractères de texte. Ce doit être une expression entière :

0 = normal	4 = inversion vidéo
1 = double hauteur	5 = inversion vidéo et double hauteur
2 = double largeur	6 = inversion vidéo et double largeur
3 = double hauteur et double largeur	7 = inversion vidéo, double hauteur et double largeur

K doit être une expression entière :

0 = stable	1 = clignotement

La couleur de fond des caractères de texte est la couleur de fond de la dernière instruction GR ou DELIM.

Si I, J, K ne sont pas spécifiés, TX prend la vleur TX 7,0,0.

Exemple

```
10   TX 4, 0, 0
```

Chaque caractère sera un caractère de texte normal stable et bleu.

---

80 USR

(Utilisateur)

Syntaxe USR(X)

But Appeler un sous-programme en langage d'assemblage de l'utilisateur avec l'argument X.

---

81 VAL

(Valeur)

Syntaxe VAL(X$)

But Transformer la valeur numérique d'une chaîne de caractères en valeur numérique pure.

Remarques X$ doit commencer par +, − ou un chiffre.

Exemple

```
10 A$="1984"
20 X=VAL(A$)
30 B$=" EST UNE ANNEE BISSEXTILE"
40 PRINT A$+B$
50 PRINT X+6;SPC(2);" NE L'EST PAS"
RUN
1984  EST UNE ANNEE BISSEXTILE
1990  NE L'EST PAS
```

Annexe 1

Mots réservés

Les mots suivants ne peuvent être utilisés comme nom de variable :

ABS
ACTION
AND
ASC
ATN
AUTO

BRIGHT

CALL
CHR$
CLEAR
CLOAD
CONT
COS
CURSORX
CURSORY
CSAVE

DATA
DEF
DELIM
DISK
DIM
DISPLAY

EG
END
ET
EXP

FN
FOR
FOUND
FRE

GR
GOSUB
GOTO

IF
INIT
INPUT
INT

KEY

LEFT$
LEN
LET
LINE
LIST
LLIST
LOAD
LOG
LPEN
LPRINT
LPOS

MID$
MODEM

NEW
NEXT
NOT

ON
OR

PAGE
PEEK
PLAY
POKE
POS
PRINT

READ
REM
RENUM
RESTORE
RETURN
RIGHT$
RND
RUN

SAVE
SCREEN
SCROLL
SETEG
SETET
SGN
SIN
SOUND
SPC
SQR
STEP
STICKX
STICKY
STOP
STORE
STR$

TAB
TAN
THEN
TO
TX

USR

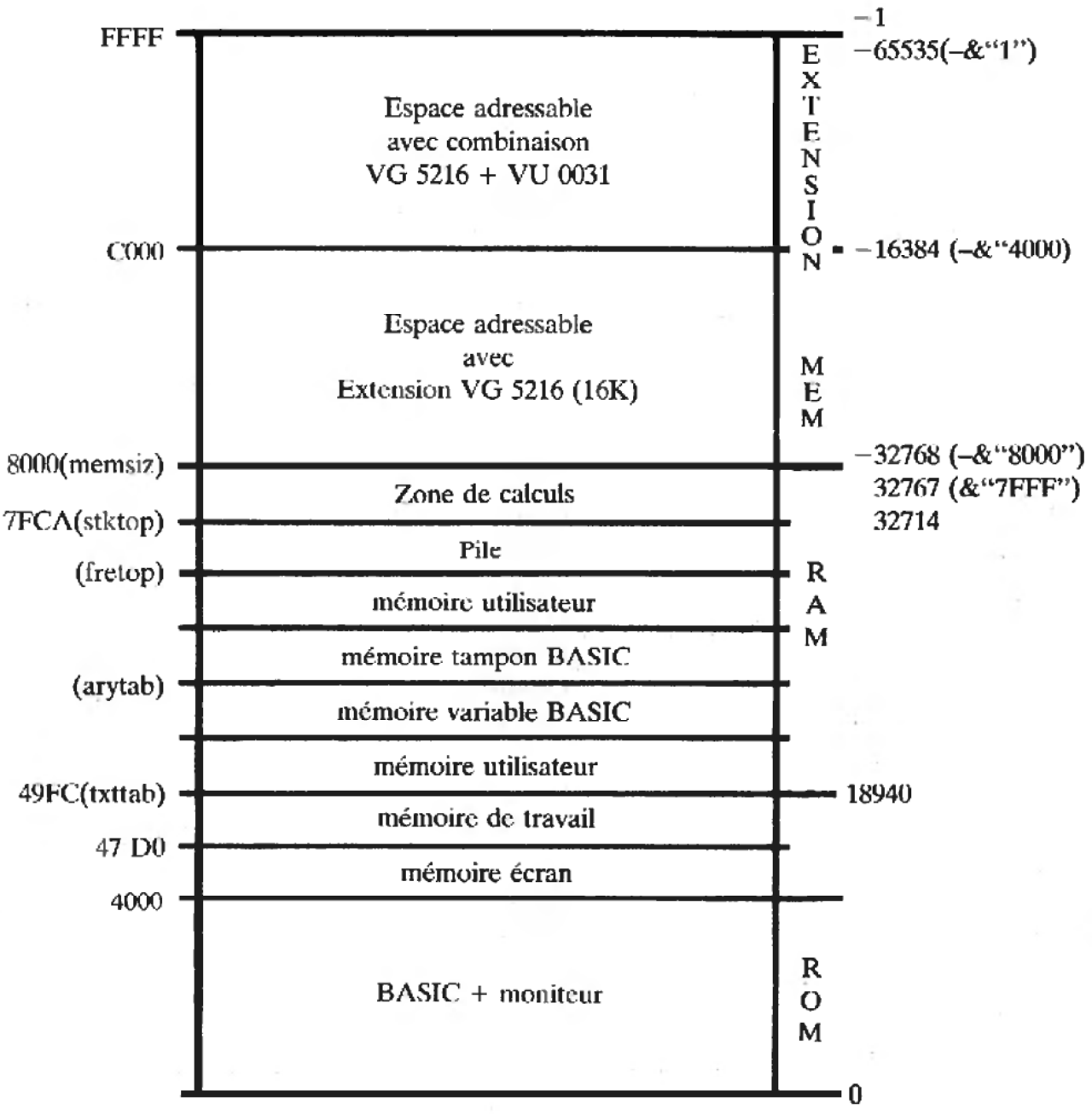
VAL

Les signes mathématiques et de ponctuations ne peuvent pas non plus être utilisés.

Annexe 2

Carte mémoire

Espace mémoire du Z 80



Annexe 3

Fonctions mathématiques

Un certain nombre de fonctions mathématiques qui ne sont pas propres à BASIC-80 peuvent être définies en BASIC-80 de la façon suivante :

FONCTIONS MATHEMATIQUES Définition BASIC-80

SECANTE =1/COS(X)

COSECANTE =1/SIN(X)

COTANGENTE =1/TAN(X)

SINUS INVERSE *=ATN(X/SQR(-X*X+1)

COSINUS INVERSE *=ATN (X/SQR(-X*X+1))+1.5708

SECANTE INVERSE *=ATN(X/SQR(X*X-1+SGN(SGN(X)-1)*1.5708

COSECANTE INVERSE *=ATN(X/SQR(X*X-1))+(SGN(X)-1)*1.5708

COTANGENTE INVERSE *=ATN(X)+1.5708

SINUS HYPERBOLIQUE =(EXP(X)-EXP(-X))/2

COSINUS HYPERBOLIQUE =EXP(X)+EXP(-X))/2

TANGENTE HYPERBOLIQUE =EXP(-X)/EXP(X)+EXP(-X))*2+1

SECANTE HYPERBOLIQUE =2/(EXP(X)+EXP(-X))

COSECANTE HYPERBOLIQUE =2/(EXP(X)-EXP(-X))

COTANGENTE HYPERBOLIQUE =EXP(-X)/(EXP(X)-EXP(-x))*2+1

COSINUS HYPERBOLIQUE INVERSE =LOG(X+SQR(X*X+1))

COSINUS HYPERBOLIQUE INVERSE =LOG(X+SQR(X*X-1)).

TANGENTE HYPERBOLIQUE INVERSE =LOG((1+X)/(1-X))/2

SECANTE HYPERBOLIQUE INVERSE =LOG((SQR(-X*X+1)+1)/X

COSECANTE HYPERBOLIQUE INVERSE =LOG((SGN(X)*SQR(X*X+1)+1)/X

COTANGENTE HYPERBOLIQUE INVERSE =LOG((X+1)/(X-1))/2

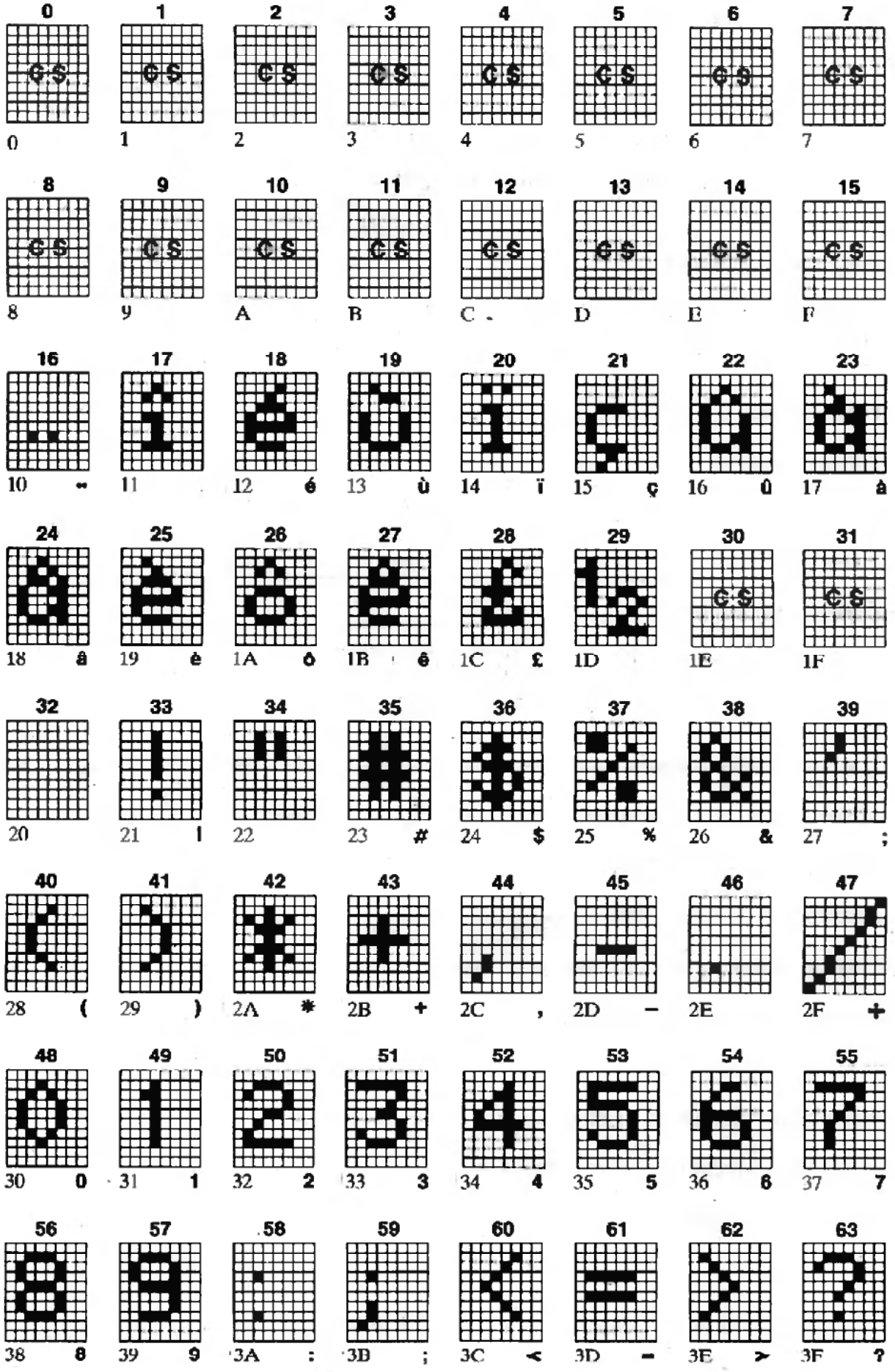
* *Pour les fonctions inverses, X représente la valeur du Sinus, Cosinus, etc.*

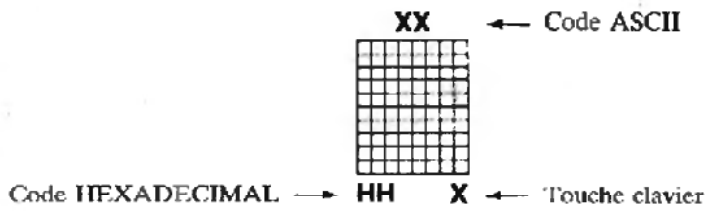
Annexe 4

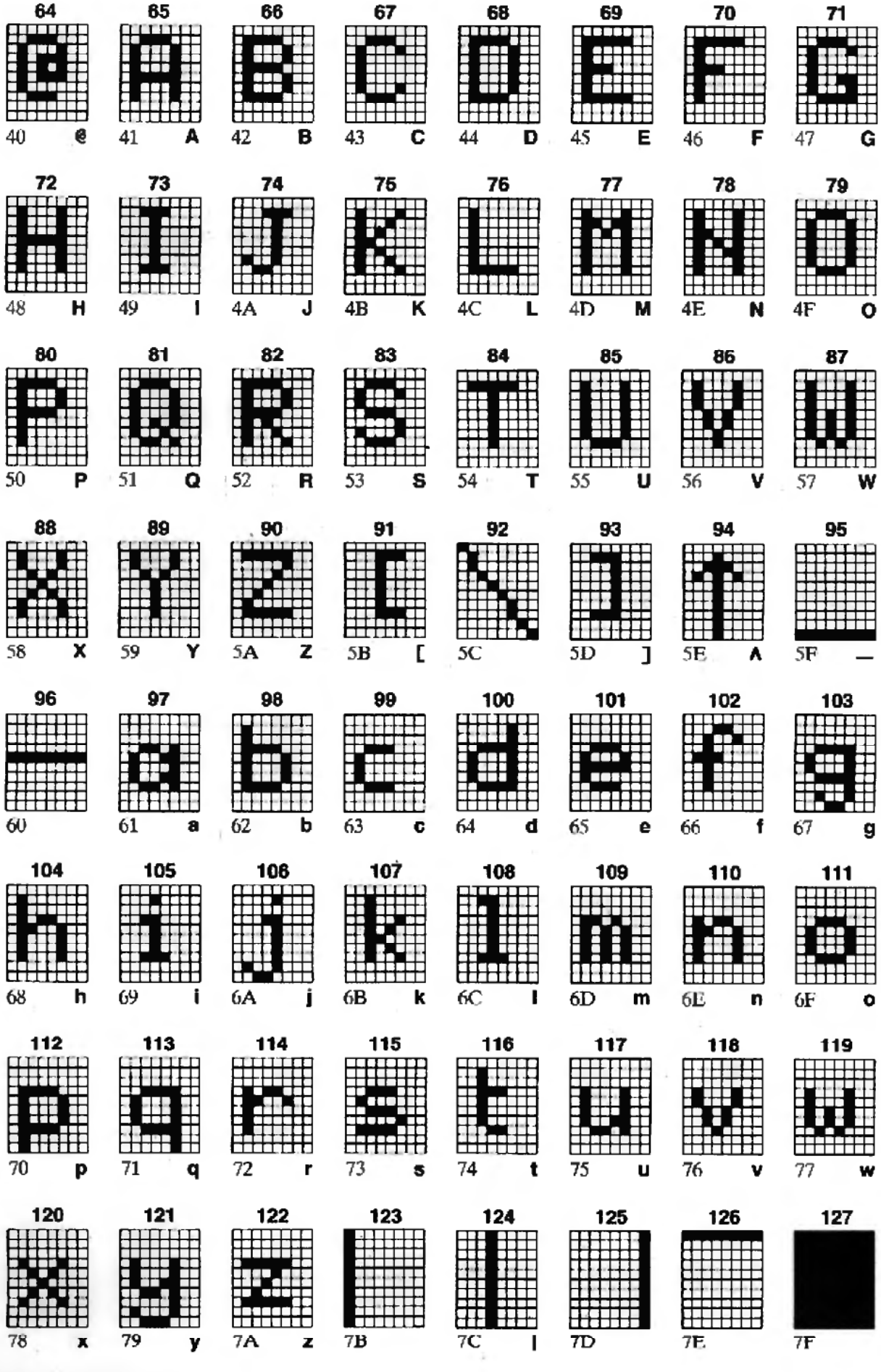
Codes des Caractères Textes

(ASCII - HEXADECIMAL et CLAVIER)

CS Code service (voir Annexe 6)



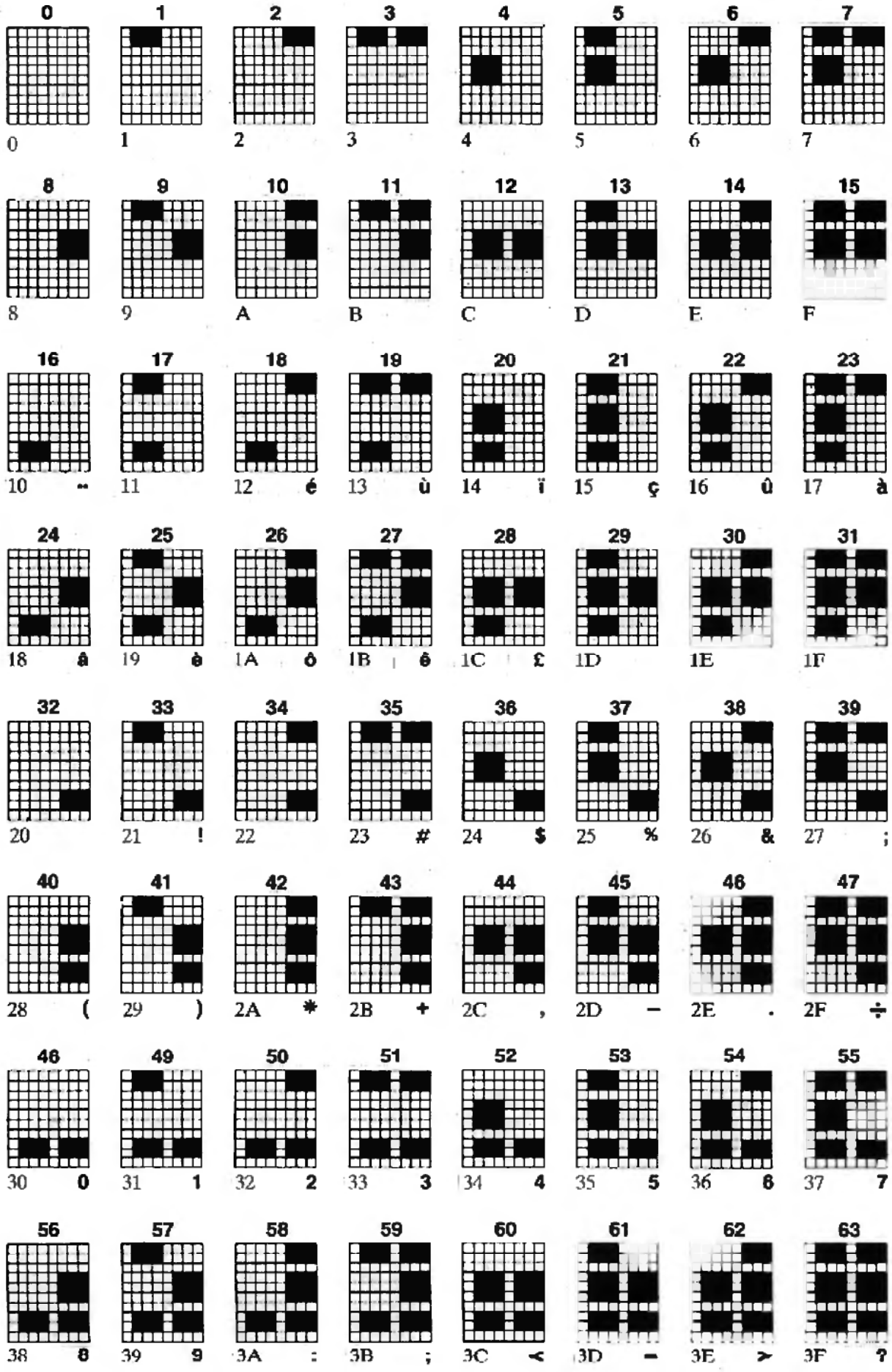

XX
Code ASCII
Code HEXADECIMAL
HH
X
Touche clavier

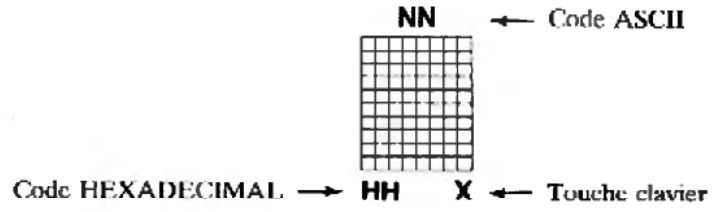

64 40 @
65 41 A
66 42 B
67 43 C
68 44 D
69 45 E
70 46 F
71 47 G
72 48 H
73 49 I
74 4A J
75 4B K
76 4C L
77 4D M
78 4E N
79 4F O
80 50 P
81 51 Q
82 52 R
83 53 S
84 54 T
85 55 U
86 56 V
87 57 W
88 58 X
89 59 Y
90 5A Z
91 5B [
92 5C
93 5D]
94 5E ^
95 5F _
96 60
97 61 a
98 62 b
99 63 c
100 64 d
101 65 e
102 66 f
103 67 g
104 68 h
105 69 i
106 6A j
107 6B k
108 6C l
109 6D m
110 6E n
111 6F o
112 70 p
113 71 q
114 72 r
115 73 s
116 74 t
117 75 u
118 76 v
119 77 w
120 78 x
121 79 y
122 7A z
123 7B
124 7C |
125 7D
126 7E
127 7F

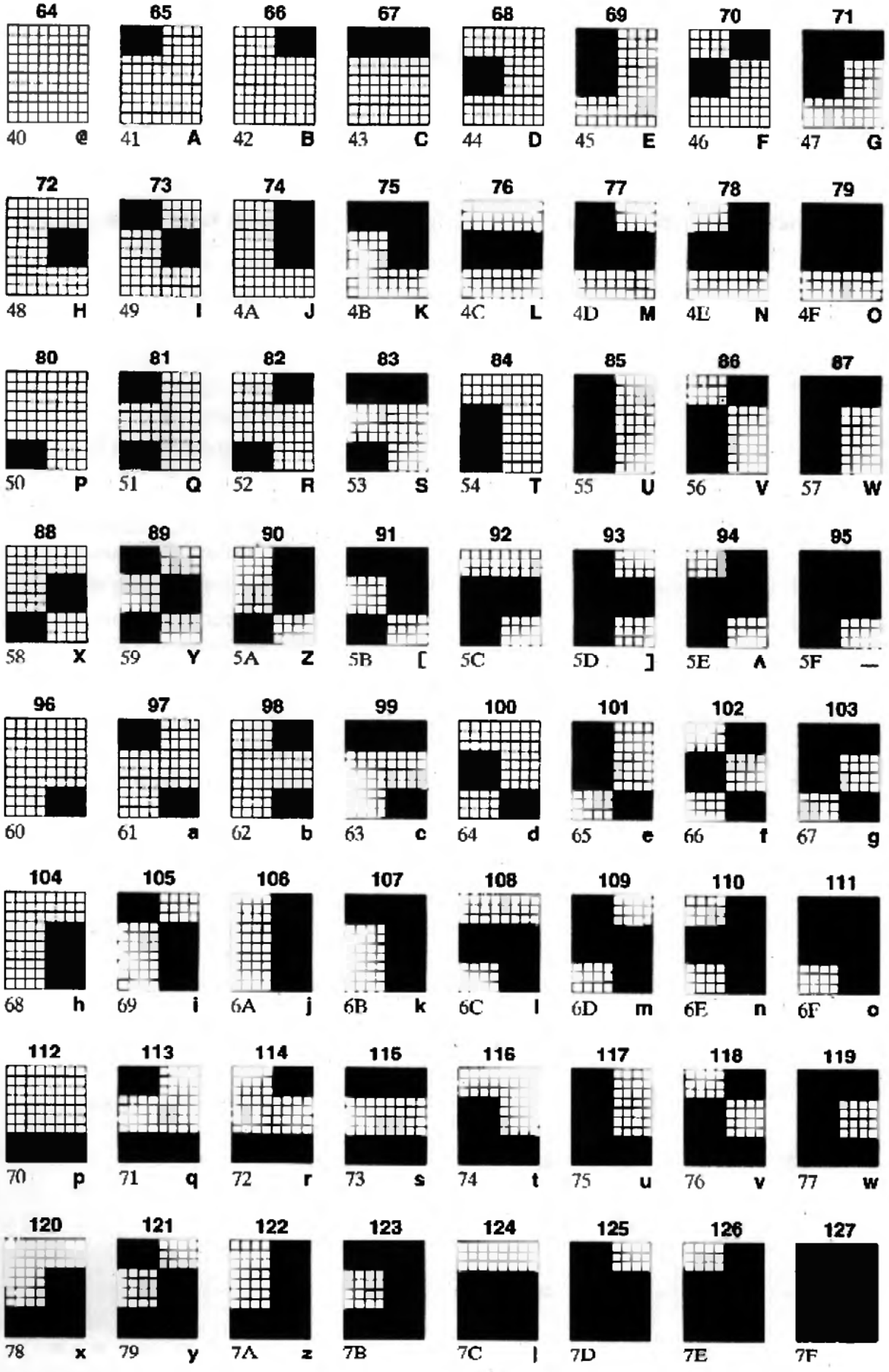
Annexe 5

Codes des Caractères Graphiques

(ASCII - HEXADECIMAL et CLAVIER)




NN
Code ASCII
Code HEXADECIMAL
HH
X
Touche clavier


64 40 @
65 41 A
66 42 B
67 43 C
68 44 D
69 45 E
70 46 F
71 47 G
72 48 H
73 49 I
74 4A J
75 4B K
76 4C L
77 4D M
78 4E N
79 4F O
80 50 P
81 51 Q
82 52 R
83 53 S
84 54 T
85 55 U
86 56 V
87 57 W
88 58 X
89 59 Y
90 5A Z
91 5B [
92 5C
93 5D]
94 5E ^
95 5F _
96 60
97 61 a
98 62 b
99 63 c
100 64 d
101 65 e
102 66 f
103 67 g
104 68 h
105 69 i
106 6A j
107 6B k
108 6C l
109 6D m
110 6E n
111 6F o
112 70 p
113 71 q
114 72 r
115 73 s
116 74 t
117 75 u
118 76 v
119 77 w
120 78 x
121 79 y
122 7A z
123 7B
124 7C |
125 7D
126 7E
127 7F

Annexe 6

VG5000

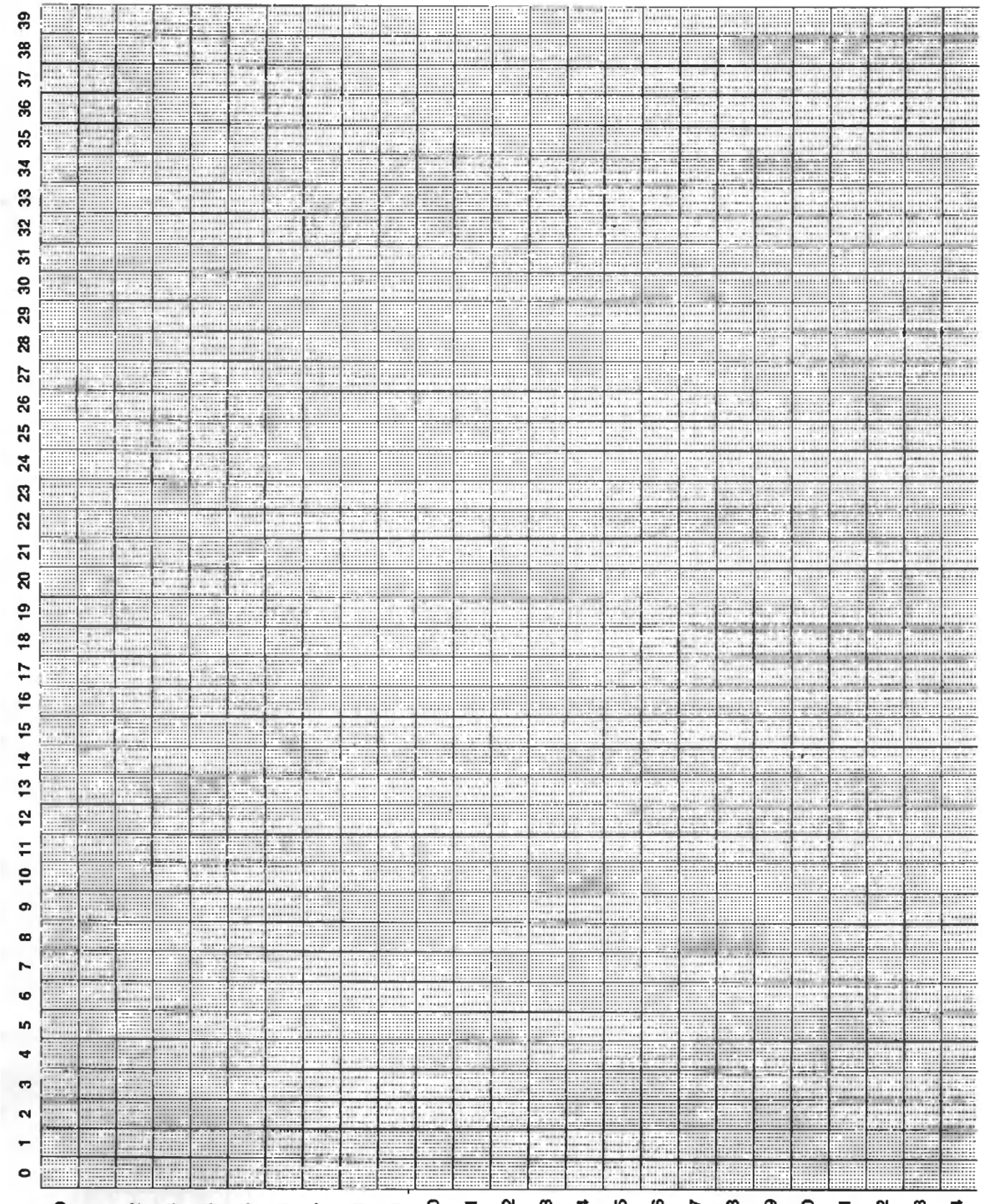
Code des fonctions de service

Ces codes de service peuvent être utilisés à l'intérieur d'un programme avec l'instruction CHR$ (...).
Certains codes ont le même effet que certaines touches clavier.

Touche clavier	Code CHR$ Mode texte	Mode graph.	En mode texte et graphique
-	0	128	= aucune action
-	1	129	= aucune action
EFFC	2	130	= effacement du caractère à gauche du curseur
RET	3	131	= retour chariot et interligne
EFFL	4	132	= effacement depuis le curseur jusqu'à la fin de la ligne
INSC	5	133	= insertion d'un caractère à la position du curseur
INSL	6	134	= insertion d'une ligne
→	7	135	= déplacement du curseur d'un caractère vers la droite
←	8	136	= déplacement du curseur d'un caractère vers la gauche
↑	9	137	= déplacement du curseur d'une ligne vers le haut
↓	10	138	= déplacement du curseur d'une ligne vers le bas
–	11	139	= pas d'action
	12	140	= retour du curseur au départ X=1,Y=0
RET	13	141	= retour chariot et interligne
	14	142	= émet un "bip" sonore
	15	143	= autorise la visualisation
	30	158	= effacement de l'écran sous le curseur
EFFE	31	159	= effacement de l'écran ; curseur en X=1,Y=0.

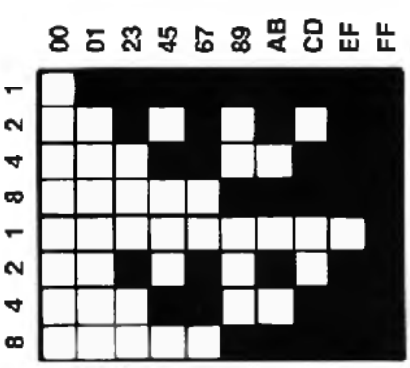
Annexe 7

Grille écran



Annexe 8

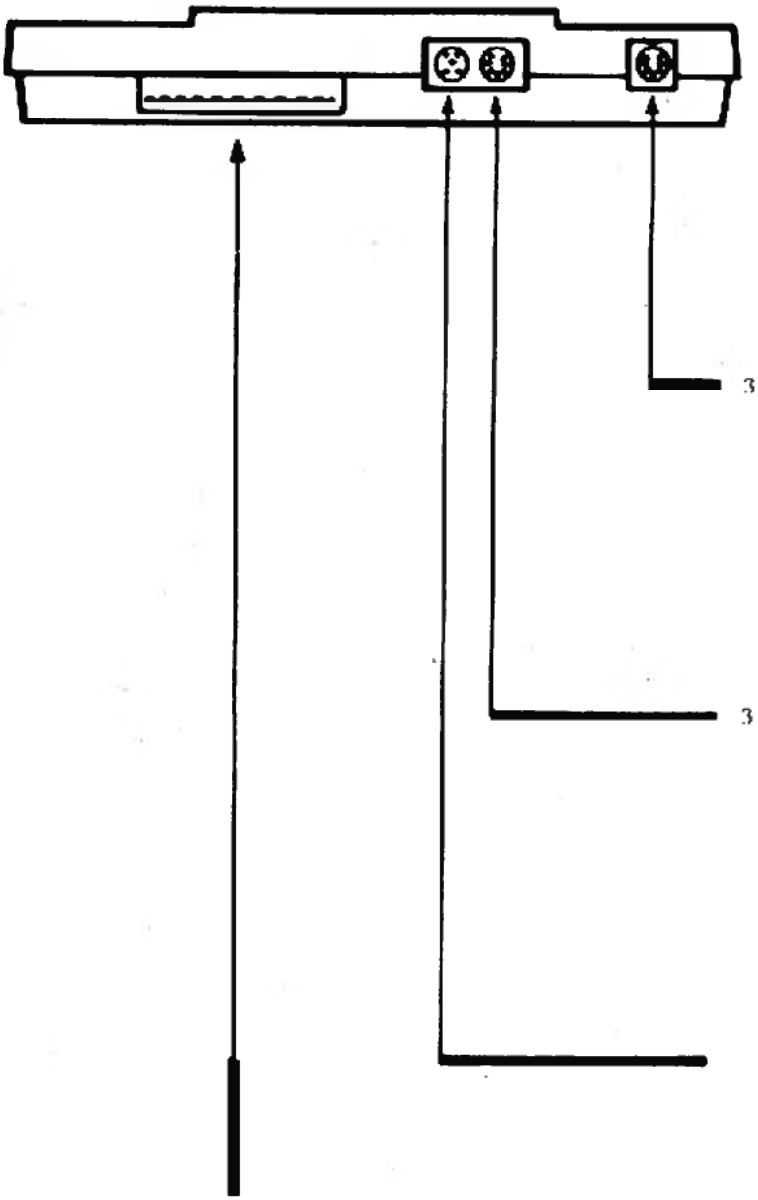
Grille caractères



Annexe 9

Schéma des Prises

(Vue AR du VG5000)



Prise magnétophone

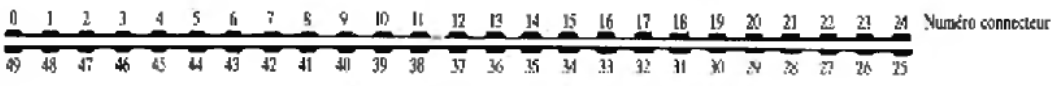
1 masse
2 masse
3 masse
4 rouge (micro)
5 blanc (casque ou ligne)
6 noir (télécommande)
7 masse
8 masse

Prise péritélévision

1 Commande rapide
2 Masse
3 Bleu
4 Sync.
5 Rouge
6 Commande lente
7 BF
8 Vert

Prise alimentation

1 NC
2 −12 V
3 + 5 V
4 +12 V
5 Masse



Connecteur bord de carte

0	$\overline{RFSH}$	7	Sound Ex	14	D_7	21	A4	28	A5	35	D_3	42	$\overline{INTEX}$	49	$\overline{WAITE}$
1	NC	8	$\overline{MI}$	15	D_0	22	A_2	29	A_7	36	D_5	43	$\overline{CE_1}$		
2	+5V	9	$\overline{WR}$	16	A_{14}	23	A_0	30	A_9	37	D_2	44	$\overline{CE_3}$		
3	+5V	10	$\overline{IORQ}$	17	A_{12}	24	−12V	31	A_{11}	38	$\overline{MREQ}$	45	$\overline{CSROMI}$		
4	$\overline{CE_4}$	11	$\overline{Reset}$	18	A_{10}	25	+12V	32	A_{13}	39	$\overline{RD}$	46	Masse		
5	$\overline{CE_2}$	12	D_6	19	A_8	26	A_1	33	A_{15}	40	$\overline{RESET}$	47	Masse		
6	$\overline{CE_0}$	13	D_4	20	A_6	27	A_3	34	D_1	41	Horl.	48	NC		

Annexe 10

Résumé des Instructions BASIC

	Syntaxe	Action
1	&"...."	Indiquer une valeur HEXADECIMALE
2	ABS(X)	Donner la valeur absolue de X
3	ACTION(I)	Donner la position du bouton ACTION
4	AND	Fonction logique ET
5	ASC(X$)	Donner la valeur ASCII d'un caractère
6	ATN(X)	Valeur de ARC Tangente X
7	AUTO i,j	Numérotation automatique des lignes
8	CALL i	Appel d'un programme en ASSEMBLEUR
9	CHR$(i)	Code caractère en ASCII
10	CLEAR i,j	Réserver un espace en mémoire
11	CLOAD	Charger un programme sur cassette
12	CONT	Continuer l'exécution d'un programme après STOP
13	COS(X)	Cosinus X
14	CSAVE"progra"	Sauvegarder un programme sur cassette
15	CURSORX	Positionner le curseur en X
16	CURSORY	Positionner le curseur en Y
17	DATA	Liste de données
18	DEFFN	Définir une fonction
19	DELIM I,J,K	Délimiter un texte (couleur, couleur, soulignement)
20	DIM A$(X)	Dimensionner un tableau
21	DISPLAY I	Régler la vitesse d'affichage
22	EG, I,J,K	Caractères spéciaux mode graphique (couleur, couleur fond, soulignement)
23	END	Indique la fin d'un programme
24	ET I,J,K	Caractères spéciaux mode texte (couleur, dimension, clignotement)
25	EXP(X)	Donne e à la puissance X
25	FRE	Donne le nombre d'octets restant libres
27	FOR...NEXT.STEP	Boucler, répéter une série d'instructions
28	GOSUB...RETURN	Aller à un sous-programme
29	GOTO NN	Aller à une ligne déterminée
30	GR I,J,K	Mode graphique (couleur, couleur fond, soulignement)
31	IF...THEN	Prendre une décision et effectuer une action
	IF...GOTO	Prendre une décision et aller à un numéro de ligne
32	INIT i,j	Initialiser les teintes de l'écran
33	INPUT	Entrer une valeur au clavier
34	INT(X)	Donner la partie entière d'un nombre décimal
35	KEY(0)	Valeur ASCII d'une touche
36	LEFT(X$,i)	Donner les I caractères les plus à gauche d'une chaîne
37	LEN(X$)	Donner la longueur d'une chaîne
38	LET	Affecter une valeur à une variable
39	LIST	Liste un programme sur l'écran
40	LLIST	Liste un programme sur imprimante
41	LOAD	Charge un programme en ASCII
42	LOG(X)	Donner le logarythme de X
43	LPOS(X)	Donner la position de la tête sur l'imprimante
44	LPRINT	Imprimer sur papier avec imprimante
45	MID$(X$,i,j)	Donner une chaîne de longueur j à partir de i caractères
46	NEW	Effacer la mémoire
47	NOT	Donner le complément à 1
48	ON...GOSUB	Sur résultat d'une expression aller à un sous-programme
	ON...GOTO	Sur résultat d'une expression aller à un numéro de ligne
49	OR	Fonction logique ou
50	PAGE	Empêcher l'affichage de défiler sur l'écran
51	PEEK(I)	Lire la valeur d'une case mémoire
52	PLAY(A$)	Jouer de la musique
53	POKE i,j	Écrire dans une case mémoire un octet
54	POS(X)	Donner la position du curseur sur une ligne
55	PRINT	Afficher sur l'écran
56	READ	Lire des valeurs inscrites dans DATA
57	REM	Remarques introduites dans un programme
58	RENUM	Renumérotation d'un programme
59	RESTORE	Permettre de relire des "DATA"
60	RIGHT(X$,i)	Donner les i caractères les plus à droite d'une chaîne
61	RND	Donner un nombre au hasard
62	RUN	Lancer un programme
63	SAVE	Sauver sur cassette un programme en ASCII
64	SCREEN	Non utilisée
65	SCROLL	Permettre de faire défiler l'écran après PAGE
66	SETHG I,"...."	Définir un caractère graphique spécial
67	SETETI,"..."	Définir un caractère texte spécial
68	SGN(X)	Donner une valeur suivant le signe de X
69	SIN(X)	Sinus de X
70	SOUND i,j,k	Créer un son
71	SPC(i)	Afficher des espaces
72	SQR(X)	Racine carrée de X
73	STICKX(i)	Valeur droite ou gauche d'une manette
	STICKY(i)	Valeur haut ou bas d'une manette
74	STOP	Arrêter le déroulement d'un programme
75	STORE	non utilisée
76	STR$(X)	Transformer un nombre en chaîne de caractères
77	TAB(i)	Faire des espaces sur l'écran
78	TAN(X)	Tangeante X
79	TX i,j,k	Initialise un mode texte (couleur, dimension, clignotement)
80	URS(o)	Appeler d'un sous-programme en ASSEMBLEUR
81	VAL(X$)	Transformer une valeur numérique d'une chaîne de caractères en valeur numérique pure.