

**Commodore  
plus/4**

# Guide d'utilisation





---

---

# **COMMODORE PLUS/4 MANUEL DE L'UTILISATEUR**

---

---

---

---

# TABLE DES MATIERES

<b>INTRODUCTION</b>	1
<b>CHAPITRE 1</b> Installation	7
<b>CHAPITRE 2</b> Utilisation du clavier et de l'écran	22
<b>CHAPITRE 3</b> Utilisation de Logiciel	33
<b>CHAPITRE 4</b> Démarrage	42
<b>CHAPITRE 5</b> Chiffres et calcul	58
<b>CHAPITRE 6</b> Introduction au BASIC	68
<b>CHAPITRE 7</b> Couleurs et graphiques	83
<b>CHAPITRE 8</b> Musique et sons	104
<b>ENCYCLOPEDIE DU COMMODORE PLUS/4</b>	113
<b>1</b> Encyclopédie du BASIC 3.5	115
Commandes	118
Instructions	130
Fonctions	158
Variables et opérateurs	164
<b>2</b> BASIC 3.5 Abréviations	169
<b>3</b> Programmes de conversion	172
<b>4</b> Messages d'erreur	174
<b>5</b> TEDMON	184
<b>6</b> Codes d'affichage écran	193
<b>7</b> Codes ASCII et CHR\$	196
<b>8</b> Cartes mémoire écran et couleur	199
<b>9</b> Carte des registres mémoire	201
<b>10</b> Fonctions mathématiques dérivées	203
<b>11</b> Table des notes de musique	204
<b>12</b> Programmes à essayer	206
<b>13</b> Guide RS-232	209
<b>14</b> Bibliographie	215
<b>INDEX</b>	216

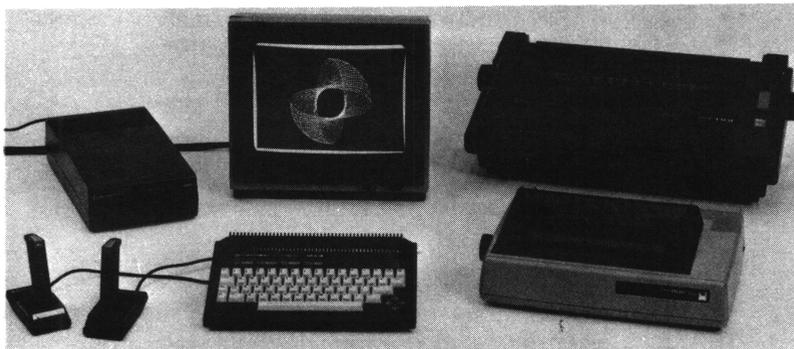
---

---

---

---

## INTRODUCTION



Vous avez fait une sage acquisition... le Plus/4 est le premier ordinateur familial. Ce manuel va vous aider à utiliser votre Plus/4 professionnelles. Bien sûr, il peut toujours être utilisé comme un ordinateur familial. Ce manuel va vous aider à utiliser votre Plus/4 dans ses applications élémentaires. Vous apprendrez à :

- installer votre Plus/4
- utiliser les touches du clavier
- accéder aux différents types de logiciels Commodore
- utiliser les capacités mathématiques, graphiques, sonores et de programmation de votre Plus/4.

L'autre manuel livré avec votre ordinateur (Le manuel des Logiciels résidents du Plus/4) vous explique comment utiliser le traitement de texte, le tableur électronique, la gestion de fichiers et les graphiques. Si votre principal intérêt réside dans ces applications professionnelles et si vous ne pouvez attendre pour les utiliser, nous vous recommandons de lire au moins le chapitre 1 de ce manuel (Installation) avant de passer au Manuel des logiciels résidents.

---

---

## SPECIFICATIONS DU PLUS/4

---

- 64K RAM (60K disponibles pour la programmation en BASIC)
  - Clavier de style machine à écrire
  - Logiciels résidents
  - Possibilité de fenêtres à l'écran
  - Touche HELP
  - 8 touches de fonction pré-programmées et reprogrammables
  - 4 touches curseur indépendantes
  - Utilisation de la plupart des périphériques du COMMODORE 64 et du VIC-20
  - 121 couleurs (16 couleurs primaires, 8 niveaux de brillance)
  - Plus de 75 commandes BASIC
  - Haute résolution graphique
  - Lignes textes dans le mode graphique haute résolution
  - Clavier avec touches graphiques
  - Touches de contrôle des couleurs
  - Définition de l'écran: 320 X 200 Pixels
  - Caractères en vidéo-inversé et clignotants
  - Générateur de sons à deux tons
  - Moniteur résident pour le langage machine (17 commandes)
- 

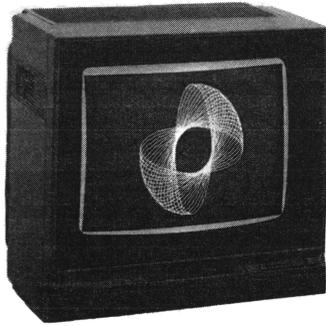
## CREATION D'UN SYSTEME INFORMATIQUE COMPLET



Ordinateur: Commodore Plus/4

---

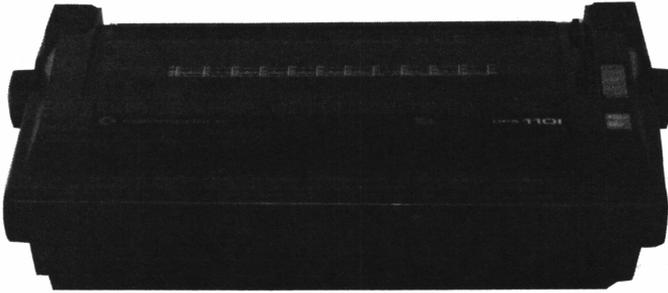
---



Ecran: Moniteur couleur Commodore 1702 ou 1802/1803



Mémoire: Lecteur de cassette Commodore 1531 (magnétophone) ou lecteur de disquettes Commodore



Imprimante: Imprimante Commodore



Contrôleurs: Joysticks Commodore PLUS/4

---

---

## RECOMMANDATIONS

Jusqu'à maintenant vous avez assez lu et vous voulez commencer. Voici ce que nous vous recommandons de faire maintenant:

- Envoyez-nous votre carte de garantie
- Abonnez-vous aux magazines Commodore pour être informé en permanence sur les nouveautés concernant votre ordinateur

Lisez ce manuel et essayez les exercices. Lisez le manuel des logiciels résidents et habituez-vous à l'ensemble de ces quatre logiciels. Allez souvent chez les revendeurs Commodore pour vous tenir informé sur les nouveaux développements de logiciels, les livres et les périphériques. Apprenez, programmez, créez des fichiers, des graphiques, écrivez, calculez, jouez... profitez au maximum de votre Commodore Plus/4.

---

---

# **CHAPITRE 1**

## **INSTALLATION**

- 
- **Déballez votre Commodore Plus/4**
- 
- **Apprenez à connaître les interrupteurs et les connecteurs**
- 
- **Installez votre Commodore Plus/4**
- 
- **Tableau de dépannage**
- 
- **Périphériques**
-

---

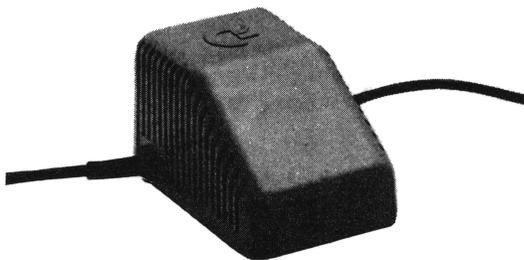
---

## DEBALLEZ VOTRE COMMODORE PLUS/4

Maintenant que vous avez ouvert votre boîte contenant votre nouveau Plus/4 et que vous avez trouvé ce manuel, la première chose que vous devriez faire est de vérifier que vous avez bien tous les éléments qui sont sur cette liste. Vous devez avoir :



### 1. Votre Commodore Plus/4



- ### 2. La boîte d'alimentation,
- une extrémité se branche à une prise électrique et l'autre à droite sur le panneau arrière de l'ordinateur.



3. Le câble RF

Il relie le téléviseur à la sortie jack du côté gauche du Plus/4. Vous n'avez pas besoin de ce câble si vous connectez votre Plus/4 à un moniteur.

4. Le manuel de l'utilisateur

5. Autre documentation  
Carte de garantie

6. Le manuel des logiciels résidents du Plus/4

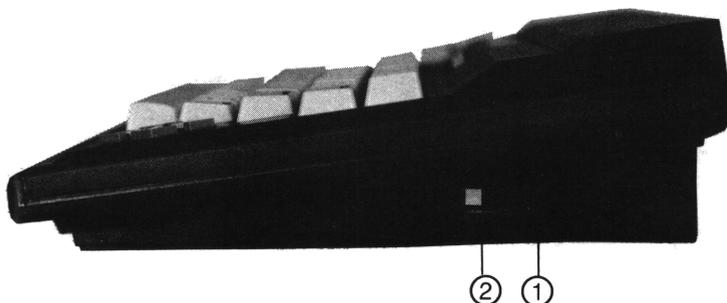
Si vous ne trouvez pas tous ces éléments dans la boîte, demandez à votre revendeur Commodore de vous les fournir.

Avant de connecter quoi que ce soit, vous devriez regarder les dessins suivants de votre ordinateur. Ces dessins montrent toutes les sorties de manière à ce que vous installiez votre système informatique rapidement et facilement.

---

# APPRENEZ A CONNAITRE LES INTERRUPTEURS ET LES CONNECTEURS

## Le côté droit de votre PLUS/4



---

### 1 L'interrupteur On/Off

Votre Plus/4 doit être en position OFF lorsque vous mettez ou retirez des cartouches ou un périphérique tel que imprimante ou lecteur de disquettes. Il y a une lampe témoin située au bas du clavier à gauche, qui s'allume quand l'interrupteur est sur ON.

---

### 2 La touche Reset (Mise à zéro)

Il y a deux manières d'utiliser la touche RESET :

1. Vous pouvez utiliser la touche RESET pour remettre en marche votre ordinateur comme si vous veniez de le brancher. Pressez juste une fois la touche RESET. Attention: quand vous appuyez sur la touche RESET, vous perdez le programme BASIC qui est en mémoire à ce moment là.\*

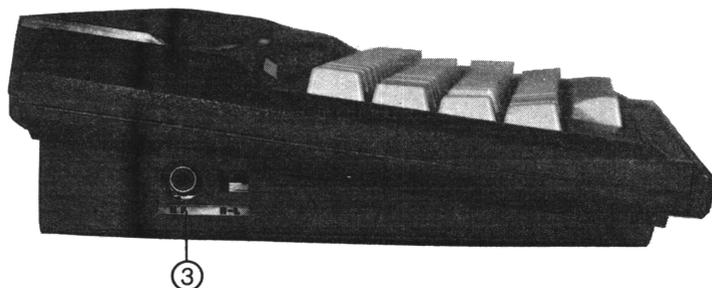
2. Si vous voulez remettre en marche votre Plus/4 et si vous voulez garder votre programme BASIC, maintenez un doigt sur la touche RUN/STOP et appuyez sur la touche RESET. Quand vous faites cela, votre Plus/4 rentre dans le moniteur langage machine. Tapez X et appuyez sur la touche RETURN pour retourner au BASIC. Votre programme reste intact dans la mémoire du Plus/4. Tapez seulement LIST pour visualiser le programme à l'écran.

\*Quand vous appuyez sur RESET, le Plus/4 génère automatiquement la commande NEW, qui remet l'écran à blanc. Ceci peut être modifié. Voir le Guide de référence du programmeur Plus/4 pour les informations concernant la commande UNNEW si vous pressez la touche RESET par accident.

---

---

## Le côté gauche de votre PLUS/4



Le connecteur du côté gauche du Plus/4 est utilisé pour la connection TV. Il n'est pas utilisé si vous connectez votre Plus/4 à un moniteur.

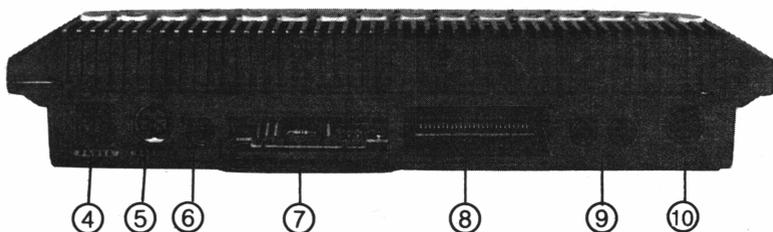
---

### **3** La prise Jack RF

C'est là où vous branchez une extrémité du câble RF (le câble noir). Vous branchez l'extrémité la plus longue dans cette prise et l'autre extrémité dans le téléviseur.

---

## L'arrière de votre ordinateur



---

---

Les prises à l'arrière de votre ordinateur permettent de connecter des accessoires variés à votre Plus/4. Chaque connecteur est différent. Soyez sûr de connecter chaque accessoire à la bonne prise.

---

#### **4 La prise d'alimentation**

L'extrémité finale du câble de la boîte d'alimentation se branche là. Branchez l'autre extrémité à une prise électrique.

---

#### **5 Le bus série**

Vous pouvez brancher un lecteur de disquette ou une imprimante dans ce connecteur. Si vous voulez connecter les deux, branchez d'abord le lecteur de disquette dans ce connecteur, puis branchez le câble de l'imprimante à l'arrière du lecteur de disquette.

---

#### **6 Le connecteur du lecteur de cassettes**

Vous branchez le lecteur de cassette Commodore 1531 à ce connecteur.

---

#### **7 Le connecteur RS-232**

Des accessoires tels qu'un modem ou un adaptateur RS-232 se branchent là. Un adaptateur RS-232 donne la possibilité de relier des accessoires qui ne sont pas prévus pour les connecteurs de l'équipement standard Commodore.

---

#### **8 Le connecteur d'extension mémoire**

Les cartouches de logiciel Plus/4 se branchent là. Avant de mettre ou d'enlever des cartouches, soyez sûr que l'interrupteur du Plus/4 est sur OFF.

---

#### **9 Joy 1 et Joy 2: les connecteurs de jeux**

Vous pouvez brancher les manettes de jeux dans ces connecteurs. Le Plus/4 utilise des manettes de jeux spéciales disponibles auprès des revendeurs Commodore.

---

#### **10 Le connecteur vidéo**

C'est là que vous branchez le câble qui relie le moniteur à votre Plus/4. Bien que ce soit un connecteur 8 broches, vous pouvez aussi brancher un connecteur 5 broches. Le moniteur couleur Commodore est livré avec un connecteur 8 broches pour être utilisé avec le Plus/4.

---

---

## **INSTALLEZ VOTRE PLUS/4**

- Pour installer votre Plus/4, vous aurez besoin d'au moins deux prises de courant: une pour votre Plus/4 et une pour votre téléviseur ou votre moniteur.
  - Si vous utilisez un lecteur de disquettes et une imprimante, vous aurez besoin d'autres prises de courant.
  - Votre Plus/4 doit être placé à une distance confortable de votre téléviseur.
  - Soyez sûr que votre ordinateur est sur OFF avant de commencer l'installation. Vérifiez que la lampe témoin est éteinte.
-

---

---

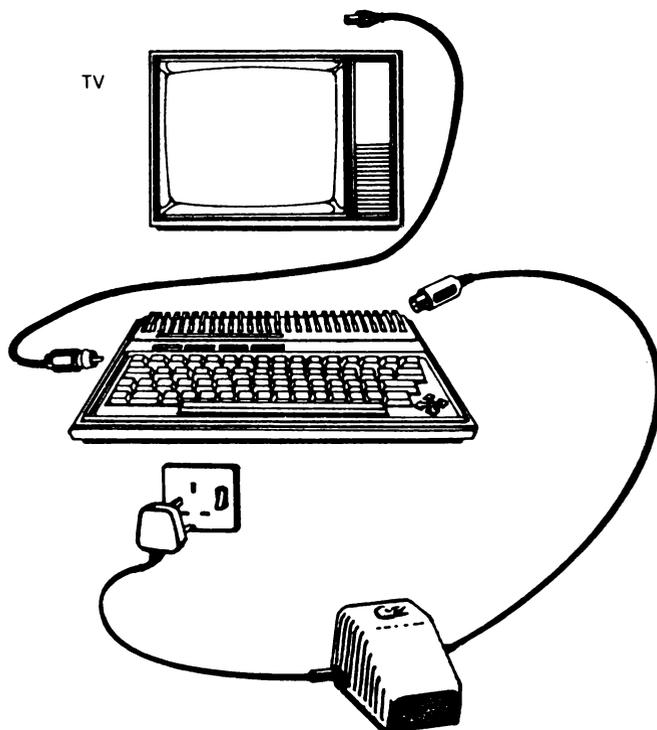
## CONNEXION DE VOTRE COMMODORE PLUS/4 A UN TELEVISEUR

### 1. Téléviseur « PAL »\*

A l'aide du câble RF joint, connectez la prise RF (N° 3) sur la partie gauche de votre Commodore Plus/4 avec l'entrée antenne UHF de votre téléviseur. Fixez votre téléviseur sur le canal voulu.

Au cas où votre téléviseur possède une prise péritelvision, branchez un câble Din/Péritel (disponible chez votre distributeur Commodore) entre le connecteur vidéo (N° 10) à l'arrière de votre Commodore Plus/4 et la prise Péritel de votre téléviseur.

\* Norme de décodage couleur européenne



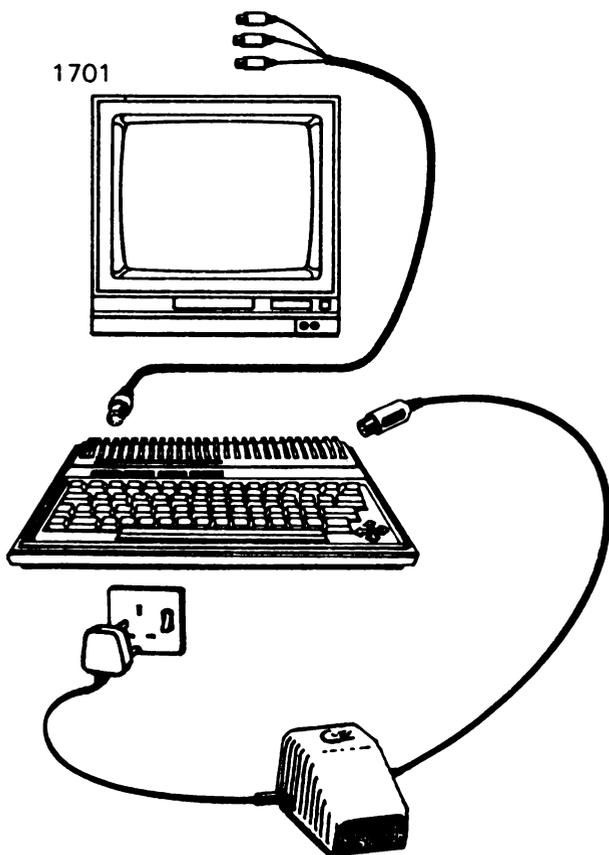
---

---

Certains téléviseurs disposent d'une entrée vidéo. Il vous est possible d'utiliser votre téléviseur comme un moniteur à l'aide d'un câble disponible chez votre revendeur Commodore.

## 2. Téléviseur « SECAM »\*

La connexion de votre Commodore Plus/4 à un téléviseur Secam est possible à l'aide de nos convertisseurs (disponibles chez votre revendeur Commodore) PAL/Péritélévision et UHF Secam Antenne.



---

---

## Etape finale

1. Fixez le câble de la boîte d'alimentation à votre Plus/4. Branchez la prise carrée du câble dans l'entrée POWER à l'arrière de votre ordinateur; branchez l'autre extrémité dans la prise de courant.
2. Si vous utilisez un téléviseur, choisissez un canal inutilisé. Si vous utilisez un moniteur couleur Commodore, utilisez les prises Jack arrière, vérifiez que l'interrupteur BACK/FRONT est sur BACK.
3. Allumez votre ordinateur. (L'interrupteur est sur le côté droit de votre Plus/4)

Si tout se passe bien le message suivant apparaît à l'écran:

**COMMODORE BASIC 3.5 60671 BYTES FREE**

**READY.**



Le curseur clignotant sous le message READY vous indique que le Plus/4 attend que vous commenciez à taper. La couleur du fond est blanche, les lettres sont noires, l'écran est entouré de bleu ciel.

4. Aidez-vous des tableaux de dépannage si vous avez un problème. Vous pouvez régler votre TV pour obtenir une meilleure image.

---

---

## TABLEAU DE DEPANNAGE

Symptôme	Cause	Remède
Lampe témoin éteinte	L'ordinateur n'est pas sous tension	Vérifiez que l'interrupteur est sur ON
	Le câble d'alimentation est débranché	Vérifiez la prise d'alimentation
	La prise n'est pas branchée sur le courant	Vérifiez la connection électrique
	Fusible défectueux dans l'ordinateur	Faites remplacer les fusibles par un revendeur agréé
Pas d'image	TV sur le mauvais canal	Sélectionnez un autre canal à l'aide du réglage fin
	Branchement incorrect	Branchez l'ordinateur à l'entrée UHF du téléviseur
	Câble vidéo non branché	Vérifiez la connection du câble à la TV
	TV éteinte	Allumez la TV
Image brouillée avec cartouche insérée	Cartouche pas bien insérée	Réinsérez la cartouche après avoir éteint ordinateur
Image sans couleur	TV mal réglée	Régalez la TV
	TV mal connectée	Vérifiez les connections
	TV aux normes Secam	Utilisez un convertisseur
Image OK et pas de son	Volume TV trop bas	Augmenter le son TV
	TV mal réglée	Régalez la TV

---

---

---

---

**IMPORTANT:** Certains postes de TV ne peuvent pas représenter la totalité de l'écran du Plus/4. Les colonnes les plus à droite et les plus à gauche de l'écran du Plus/4 seront coupées. Nous vous recommandons d'utiliser une TV différente ou un moniteur tel que les Commodore 1702.

Si ce n'est pas possible, vous pouvez régler ce problème en appuyant sur la touche ESCape, puis sur la touche R. Cela réduit la taille de la représentation de l'écran pour le visualiser en entier. Vous devez répéter cette opération chaque fois que vous allumerez votre Plus/4.

---

---

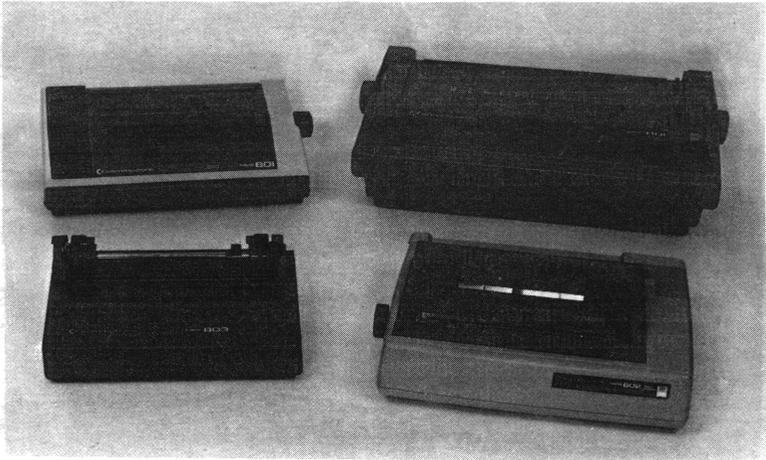
## PERIPHERIQUES

Les périphériques sont des accessoires que vous ajoutez à votre système Plus/4. Ces accessoires sont disponibles chez votre revendeur Commodore et vous permettent d'utiliser le Plus/4 à son maximum. Les périphériques offrent au système Plus/4 la possibilité d'enregistrer et de sauver des données, d'imprimer (en noir et blanc ou en couleur), d'utiliser des logiciels sur disquette ou sur cassette et d'accéder à des informations et des services disponibles par télécommunication.

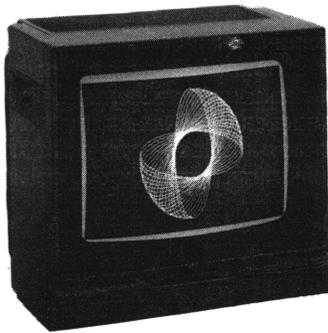


Pour sauver ou rappeler des programmes, vous aurez besoin d'un périphérique stockant les données. Les données peuvent être enregistrées et retrouvées sur des disquettes ou des cassettes. Pour utiliser les disquettes, vous aurez besoin d'un lecteur de disquettes rapide et efficace, compatible avec le Plus/4. (Commodore 1541). Pour l'utilisation de cassettes, le lecteur de cassette Commodore 1531 est particulièrement adapté.





Quand vous utilisez un logiciel de traitement de texte ou un logiciel graphique sur le Plus/4, une imprimante reproduira sur papier ce qu'il y a à l'écran. Il y a plusieurs modèles d'imprimantes Commodore compatibles avec le Plus/4: la MPS-801, la MPS-802, la MPS-803 (avec entraînement par picot) et la DPS-1101 (qualité lettre). Chaque imprimante est spécialisée dans un type d'impression. Demandez à votre revendeur celle qui correspond à vos besoins.



Un moniteur permet d'obtenir des images encore plus nettes, notamment avec l'un des moniteurs Commodore. On peut par ailleurs se brancher sur des réseaux proposant des informations, des logiciels, des nouvelles, les cours de la bourse par exemple par le biais d'un téléphone couplé à un modem (modulateur/démodulateur) : toutes les possibilités de télécommunications auxquelles peut rêver un informaticien sont alors possibles...

---

---

# CHAPITRE 2

## UTILISATION DU CLAVIER ET DE L'ECRAN

---

- Le clavier
  - Touches spéciales
  - Touches de fonction programmables
  - La touche HELP
-

---

---

## LE CLAVIER



La plupart des touches sur le clavier du Plus/4 sont identiques aux touches d'une machine à écrire, mais chaque touche peut faire plus qu'une touche de machine à écrire. Dans cette section, vous apprendrez comment utiliser les touches spéciales comme la touche **C** et les touches flèche du curseur. Cette section vous montre les caractéristiques supplémentaires de chaque touche et comment utiliser les symboles graphiques figurant sur la face avant de beaucoup de touches.

Pendant que nous vous décrivons le clavier du Plus/4, vous devrez trouver ces touches et les utiliser.

---

---

## LES TOUCHES SPECIALES

Vous devez appuyer sur la touche [ ] à la fin de chaque ligne d'instructions que vous entrez au clavier de votre Plus/4. Vous devez considérer cette touche comme une touche "ENTREE" parce que la touche [ ] entre réellement les informations et les instructions dans l'ordinateur.

Cette touche fonctionne comme la touche majuscule d'une machine à écrire. Votre Plus/4 a deux touches [ ] et une touche [ ], qui fonctionne comme le blocage majuscule d'une machine à écrire.

En appuyant la touche [ ], vous obtenez le symbole graphique figurant sur la face avant droite des touches graphiques quand vous êtes en mode majuscule/graphique.



Votre Plus/4 est automatiquement en mode majuscule/graphique quand vous l'allumez. Dans ce mode, toutes les lettres sont en majuscule sans avoir à appuyer sur la touche [ ].

Quand vous appuyez sur la touche [ ] et sur une touche lettre, vous obtenez des majuscules quand vous êtes dans le mode texte minuscule/majuscule (comme avec une machine à écrire). Dans ce mode, les lettres que vous tapez sont en minuscule, sauf si vous utilisez la touche [ ].

**NOTE:** Vous pouvez passer alternativement du mode majuscule/graphique au mode texte minuscule/majuscule en appuyant en même temps sur les touches [ ] et [G].

Appuyez sur cette touche pour arrêter un programme en cours sur le Plus/4. Si vous appuyez sur cette touche, quand un programme tourne sur le Plus/4, vous reprenez le contrôle de l'ordinateur

Quand vous appuyez simultanément sur les touches [ ] et [ ], le Plus/4 charge et exécute le premier programme de la disquette qui est dans le lecteur de disquettes.

---

---

## Les touches curseur



Il est très facile de déplacer le curseur rapidement sur l'écran dans toutes les directions. Il suffit d'appuyer sur la touche-flèche qui pointe dans la direction désirée. Comme toutes les touches du clavier du Plus/4, chaque touche est active tant qu'on a le doigt pressé dessus. Cette fonction de répétition automatique permet au curseur de se déplacer jusqu'à ce que la touche soit libérée.

**NOTE:** Vous pouvez déplacer le curseur sur les lettres et les chiffres à l'écran sans les affecter.

Vous pouvez insérer ou supprimer des lettres et des chiffres sur la ligne que vous êtes en train de taper, en appuyant sur cette touche. Quand vous appuyez sur cette touche seule, le caractère immédiatement à gauche du curseur disparaît et le curseur se positionne à la place de ce caractère. Vous pouvez utiliser la touche curseur pour retourner au milieu d'une ligne et appuyer sur [ ] pour effacer un caractère. Quand vous faites cela, la lettre à gauche du curseur est effacée et le reste des caractères de la ligne se déplace d'un espace vers la gauche.

Vous pouvez créer un espace pour insérer un caractère, en utilisant les touches [ ] et [ ]. L'espace est créé à droite du curseur; le curseur lui-même ne bouge pas. Quand vous insérez un espace au milieu d'une ligne, le reste de cette ligne se déplace vers la droite.

La touche [ ] fait gagner beaucoup de temps quand vous voulez modifier ce que vous avez tapé.

Cette touche a trois fonctions: HOME, CLEAR, CLEAR WINDOW. Quand vous appuyez sur cette touche, le curseur se déplace immédiatement dans le coin supérieur gauche de l'écran. C'est la position HOME. Le reste de l'écran ne bouge pas. Si vous appuyez simultanément sur les touches [ ] et [ ], non seulement le curseur revient en position HOME mais l'écran se vide. Il ne reste que le curseur clignotant dans le coin supérieur gauche de l'écran. Si vous appuyez sur cette touche deux fois de suite,

---

---

toutes les fenêtres que vous avez créées sont supprimées. Les fenêtres sont des zones de travail que vous créez sur une partie de l'écran; vous aurez de plus amples informations plus loin.

### **Ctrl**

Cette touche fonctionne toujours avec une autre touche. La touche **Ctrl** s'utilise comme la touche **Shift** : vous devez garder votre doigt appuyé dessus pendant que vous appuyez sur l'autre touche.

1. Comme le paragraphe sur les touches couleur l'explique, la touche **Ctrl** plus une touche couleur permet de choisir la couleur des caractères à l'écran.
2. En appuyant sur les touches **Ctrl** et S, le programme fait une pause pendant l'impression et la liste à l'écran (appuyez sur une touche quelconque pour faire repartir le programme).
3. La touche **Ctrl** est aussi utilisée avec les touches **Page Up** et **Page Down**.

De plus, certains logiciels utilisent la touche **Ctrl** comme une touche de fonction.

### **C**

Comme la touche **Shift**, la touche **C** fonctionne avec d'autres touches. Cette touche a quatre fonctions:

1. Utilisée avec la touche **Shift**, la touche **C** permet de passer du mode majuscule/graphique au mode texte minuscule/majuscule.
2. Dans les deux modes, la touche **C** fonctionne comme la touche SHIFT et permet d'utiliser les symboles graphiques situés sur la face avant gauche de chaque touche. Gardez un doigt sur la touche **C** pendant que vous appuyez sur une touche graphique.



3. Si vous voulez choisir la couleur des caractères parmi les huit couleurs que vous avez sur le bas de la face avant des touches couleur, appuyez sur la touche **C** et sur la touche couleur choisie.
4. Si vous voulez ralentir le défilement de l'écran, appuyez sur la touche **C**. La vitesse de défilement de l'écran ralentira considérablement. Quand vous relâchez cette touche, le défilement reviendra à sa vitesse normale.

---

---

## Les touches couleur



Vous pouvez choisir la couleur des caractères et des symboles graphiques de l'écran parmi les 16 couleurs offertes par votre Plus/4. C'est facile à faire:

- Si vous choisissez une des 8 couleurs située sur le haut de la face avant des touches couleur (BLK pour black-noir), maintenez le doigt sur la touche **REVERSE** et appuyez sur la touche couleur choisie.
- Si vous choisissez une des 8 couleurs située sur le bas de la face avant des touches couleur (ORNG pour orange), maintenez le doigt sur la touche **C** et appuyez sur la touche couleur choisie.

Exercez-vous à changer de couleur pour être sûr d'avoir compris le fonctionnement de ces touches. Vous remarquerez qu'après avoir changé de couleur, les caractères que vous taperez ensuite auront la couleur choisie.

**REVERSE** **REVERSE**

Le Plus/4 vous permet d'imprimer des caractères en vidéo inverse. Autrement dit, si vous utilisez des caractères noirs sur fond jaune, vous pouvez utiliser la touche REVERSE pour obtenir des caractères jaunes sur fond noir.

Voici ce que vous devez faire pour obtenir des caractères en vidéo inversé : appuyez sur les touches **CONTROL** et **REVERSE**. Maintenant tout ce que vous taperez sera en vidéo inversé jusqu'à ce que vous appuyez sur les touches **CONTROL** et **REVERSE**, sur la touche **RETURN** ou sur les touches **STOP** et O. Cela vous permet de revenir en situation normale.

**FLASH** **FLASH**

Vous pouvez faire clignoter les caractères à l'écran de manière continue. Appuyez sur les touches **CONTROL** et **FLASH** pour faire clignoter tout ce que vous taperez. Appuyez sur les touches **CONTROL** et **FLASH**, sur la touche **RETURN** ou sur la touche **STOP** pour revenir en affichage normal.

---

---

## LES TOUCHES GRAPHIQUES

Comme nous l'avons déjà dit, quand vous allumez le Plus/4, vous êtes en mode majuscule/graphique. Dans ce mode, vous pouvez taper l'ensemble des 60 graphiques qui se trouvent sur la face avant de la plupart des touches, ainsi que toutes les lettres majuscules sans utiliser la touche . La touche  vous permet d'avoir accès aux graphiques dans ce mode, au lieu des lettres majuscules.

Il y a deux symboles graphiques sur chaque touche graphique:

- Pour obtenir le symbole graphique de droite, maintenez le doigt sur la touche  et appuyez sur la touche choisie.
- Pour obtenir le symbole graphique de gauche, maintenez le doigt sur la touche  et appuyez sur la touche choisie.

Vous pouvez créer des dessins et des diagrammes en juxtaposant des graphiques côte à côte ou l'un sur l'autre. Essayez d'utiliser les touches graphiques pour voir comment elles fonctionnent. Le chapitre 7 donne plus d'explications sur les graphiques.

Vous pouvez passer du mode majuscule/graphique au mode texte minuscule/majuscule en appuyant en même temps sur les touches  et . Dans ces deux modes, tapez les instructions BASIC sans utiliser la touche .

Dans le mode texte, vous pouvez taper les lettres en majuscule et en minuscule comme sur une machine à écrire. (Vous devez appuyer sur la touche  pour obtenir des majuscules). Vous pouvez aussi utiliser les caractères graphiques situés sur la face avant gauche des touches graphiques comme dans le mode graphique: maintenez un doigt sur la touche  pendant que vous appuyez sur la touche choisie. Les graphiques situés sur la face avant gauche des touches sont parfaites pour créer des diagrammes, des graphiques et des tableaux de gestion.



La touche  vous permet d'utiliser plusieurs fonctions de l'éditeur d'écran, y compris les fonctions de gestion des fenêtres du Plus/4. Les fenêtres sont des zones de l'écran (que vous définissez) qui peuvent être utilisées comme des espaces de travail sans affecter le reste de l'écran. La touche  permet de gérer plusieurs fonctions de l'éditeur d'écran, et d'autres utilisations ordinaires, telles que l'insertion, la suppression et le défilement de l'écran.

---

---

La touche **Insert** est généralement utilisée avec les touches standards de l'alphabet. Pour activer une fonction, appuyez sur la touche **Insert** et ensuite sur une des touches suivantes:

- A Mode d'insertion automatique
- B Définition du coin inférieur droit de la fenêtre (à l'endroit du curseur)
- C Désactivation du mode d'insertion automatique
- D Suppression de la ligne active
- I Insertion d'une ligne
- J Déplacement du curseur au début de la ligne active
- K Déplacement du curseur en fin de ligne active
- L Activation du défilement de l'écran
- M Désactivation du défilement de l'écran
- N Retour à la taille normale de l'écran
- O Désactivation des modes insertion, guillemets, vidéo inversé et clignotant
- P Effacement de tout ce qu'il y a sur la ligne jusqu'au curseur
- Q Effacement de tout ce qu'il y a sur la ligne à partir du curseur
- R Réduction de la taille de l'écran
- T Définition du coin supérieur gauche de la fenêtre
- V Défilement de l'écran vers le haut
- W Défilement de l'écran vers le bas
- X Désactivation de la fonction ESCape

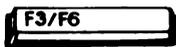
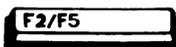
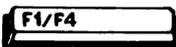
### **Symboles spéciaux**

Le clavier du Plus/4 contient aussi des symboles spéciaux qu'on ne trouve pas sur les machines à écrire, ni même sur la plupart des ordinateurs. Ces symboles spéciaux comprennent le signe de la Livre anglaise (£), pi, les signes supérieurs et inférieurs (><), les crochets ([ ]) et la flèche vers le haut. Ces symboles spéciaux sont utilisés pour la programmation du Plus/4.

---

---

## LES TOUCHES DE FONCTION PROGRAMMABLES



Les quatre touches de fonction en haut du clavier ont des fonctions spéciales qui vous permettent de gagner du temps en exécutant des tâches répétitives en ne frappant qu'une seule touche.

Vous pouvez visualiser ce que font ces touches en tapant KEY au clavier et en appuyant sur la touche RETURN.

L'écran affiche:

### KEY

KEY 1, "SYS 1525:3 — PLUS — 1"

KEY 2, "DLOAD" + CHR\$(34)

KEY 3, "DIRECTORY" + CHR\$(13)

KEY 4, "SCNCLR" + CHR\$(13)

KEY 5, "DSAVE" + CHR\$(34)

KEY 6, "RUN" + CHR\$(13)

KEY 7, "LIST" + CHR\$(13)

KEY 8, "HELP" + CHR\$(13)

Voici ce que chaque touche fait exactement:

La touche F1 est définie pour activer l'ensemble des logiciels intégrés.

F2 imprime "DLOAD" à l'écran. Vous n'avez plus qu'à entrer le nom du programme à charger à partir de la disquette et à appuyer sur [ ] au lieu de taper DLOAD soi-même.

F3 donne le répertoire des fichiers de la disquette qui est dans le lecteur de disquettes.

F4 efface l'écran (même en mode graphique)

F5 imprime "DSAVE" à l'écran. Vous n'avez plus qu'à entrer le nom du programme à sauvegarder sur disquette et à appuyer sur [ ] .

F6 lance le programme en mémoire

---

F7 donne la liste du programme en mémoire

F8 (touche ) souligne les erreurs dans les instructions du programme en les faisant clignoter

Pour utiliser une de ces fonctions, vous n'avez qu'à appuyer sur la touche de fonction appropriée. Vous devez utiliser la touche  pour obtenir les fonctions 4,5,6 et 7.

Vous pouvez redéfinir ces touches pour qu'elles effectuent les fonctions dont vous avez besoin. La redéfinition est simple, vous utiliserez la commande KEY. Vous pouvez redéfinir ces touches à partir d'un programme BASIC, ou bien les modifier à tout moment en mode direct. (Les nouvelles définitions sont effacées quand vous éteignez votre Plus/4.) Vous pouvez redéfinir autant de touches que vous voulez autant de fois que vous le voulez.

---

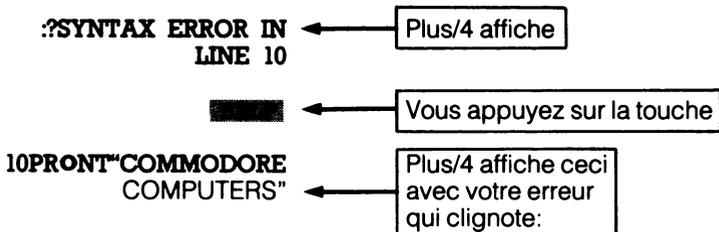
---

## LA TOUCHE HELP

**HELP**

Si vous faites une erreur de programmation, le Plus/4 affiche un message d'erreur. Ces messages d'erreur sont expliqués plus en détail dans la section 4 de l'encyclopédie du Plus/4 qui se trouve dans la seconde partie du manuel.

Vous pouvez obtenir une aide supplémentaire pour vos erreurs en utilisant la touche . Après apparition du message d'erreur, appuyez sur la touche  pour localiser votre erreur. Quand vous appuyez sur la touche , la ligne où se trouve l'erreur apparaît sur l'écran avec l'instruction erronée qui clignote. Par exemple:



---

---

# **CHAPITRE 3**

## **UTILISATION DU LOGICIEL**

---

- Introduction

---
- Logiciel résident du Plus/4

---
- Cartouches

---
- Cassettes

---
- Disquettes

---

---

---

## INTRODUCTION

La famille de logiciels disponible pour votre Plus/4 s'agrandit rapidement. Votre revendeur peut vous mettre au courant des nouveaux produits et vous informer à propos des caractéristiques des logiciels disponibles à ce jour.

Votre Commodore Plus/4 peut utiliser des logiciels sur cartouches, cassettes et disquettes, disponibles chez votre revendeur Commodore. Vous n'avez qu'à les charger dans votre Plus/4. Vous pouvez aussi créer et sauvegarder vos propres programmes sur cassettes ou sur disquettes.

## LOGICIEL RESIDENT DU PLUS/4

Votre Plus/4 est équipé d'un ensemble de logiciels intégrés. Ce sont des programmes résidents en mémoire, que l'on peut activer en appuyant sur la touche de fonction appropriée. Votre logiciel résident en mémoire permet à votre ordinateur de faire du traitement de texte, de la gestion de fichiers, des calculs sur tableur électronique et des graphiques. Le logiciel résident est prêt à être utilisé chaque fois que votre ordinateur est allumé. Quand vous branchez votre ordinateur, un message d'écran vous dit quels sont les logiciels disponibles et quelle touche de fonction vous devez presser pour rendre ces logiciels actifs. Vous pouvez aussi utiliser la commande KEY pour connaître la définition des touches de fonction. Si une touche de fonction permet d'accéder au logiciel résident, la définition de F 1 sera : SYSXXXXX : nom du logiciel. Appuyer. seulement la touche de fonction 1 et ensuite sur la touche **RESIDENT** pour activer le programme.

---

---

## CARTOUCHES

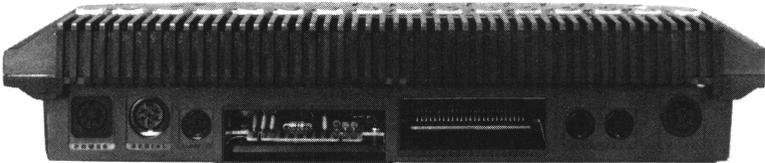
### Chargement des cartouches

Commodore offre un assortiment complet de logiciels sur cartouche pour votre Plus/4. Il y a des programmes familiaux, éducatifs et professionnels, ainsi que des jeux passionnants disponibles sur cartouche. Suivez les instructions suivantes pour utiliser les cartouches.

INSTRUCTION 1      Eteignez votre Plus/4.

**IMPORTANT :** Vous devez **éteindre** votre ordinateur avant **d'insérer ou de retirer** les cartouches. Si vous ne faites pas cela, vous pouvez **endommager** la **cartouche** et **l'ordinateur**.

INSTRUCTION 2      Tenez la cartouche avec l'étiquette vers le haut, et insérez la cartouche fermement dans l'emplacement cartouche à l'arrière de votre ordinateur.



INSTRUCTION 3      Allumez votre Plus/4

INSTRUCTION 4      Utilisez le jeu ou le programme selon les instructions du logiciel. Un programme sur cartouche commence immédiatement, alors qu'il faut appuyer sur une touche de fonction pour faire commencer les autres logiciels.

---

# CASSETTES

## Chargement des cassettes

De nombreux produits logiciels pour le Plus/4 sont disponibles sur cassettes. Ces cassettes sont identiques aux cassettes de musique que vous utilisez pour votre magnétophone ou votre appareil stéréo. Les cassettes sont utilisées avec le lecteur de cassette, disponible chez votre revendeur Commodore.

Vous pouvez aussi utiliser le lecteur de cassette pour sauvegarder les programmes que vous écrivez vous-même. Le paragraphe suivant explique comment sauvegarder des programmes sur cassette.

Les étapes pour charger une cassette sont les mêmes si vous utiliser un logiciel enregistré ou un programme que vous avez sauvegardé.

ETAPE 1 Insérez la cassette dans le lecteur et fermez la porte.

ETAPE 2 Rembobinez la cassette jusqu'au début en appuyant sur la touche REWIND du lecteur.

ETAPE 3 Quand la cassette est rembobinée au début, tapez LOAD et appuyez sur **LOAD**. L'ordinateur répond par le message suivant:

**PRESS PLAY ON TAPE**

ETAPE 4 Appuyez sur la touche PLAY du lecteur. L'écran se vide lorsque le lecteur se met en marche. Quand le programme est trouvé, le message à l'écran est le suivant:

**FOUND**"nom du programme"

ETAPE 5 Appuyez sur la touche Commodore pour charger le programme qui a été trouvé. S'il y a plus d'un programme sur la cassette, et que le programme trouvé n'est pas celui que vous désirez, appuyez sur la barre d'espace pour continuer la recherche.

Une fois que le programme a été chargé, le mot READY apparaît. Si vous voulez arrêter le chargement avant qu'il ne soit terminé, appuyez sur RUN/STOP. Après le chargement du logiciel, tapez RUN pour faire commencer

---

---

le programme. Vous pouvez aussi obtenir la liste du programme ou la modifier, si c'est un programme en BASIC.

NOTE: Pour charger un programme spécifique de la cassette, utilisez l'expression LOAD"nom du programme".

## Sauvegarde des programmes sur cassette

Quand vous écrivez un programme et que vous voulez le sauvegarder, suivez les étapes suivantes:

ETAPE 1 Tapez:

**SAVE**"nom du programme"

Vous donnez le nom que vous voulez au programme, mais il ne doit pas dépasser 16 lettres et/ou chiffres de long.

ETAPE 2 Appuyez sur la touche . L'ordinateur donne ce message:

**PRESS RECORD AND PLAY ON TAPE**

ETAPE 3

Appuyez sur les touches RECORD et PLAY de votre lecteur de cassette. L'écran se vide de son contenu. Quand votre programme est sauvegardé, le message READY s'inscrit à l'écran.

Exemples des commandes SAVE pour un lecteur de cassettes :

**SAVE"MONJOB"** ←

**SAVE"3TEST"** ←

Les noms entre "" sont les noms spécifiques des programmes sauvegardés.

NOTE: Quand vous sauvegardez un programme sur cassette, faites attention à la position de la bande magnétique dans la cassette. En particulier, faites attention à ne pas sauvegarder un programme au tout début de la bande, car beaucoup de bandes ont des repères magnétiques, qui n'enregistrent aucune information. Dans ce cas, une partie du programme ne serait pas sauvegardée.

Quand vous chargez ou sauvegardez un programme, si vous décidez d'arrêter avant que l'opération ne soit terminée, vous devez presser sur la touche RUN/STOP. Ensuite, appuyez sur la touche STOP du lecteur de cassette.

---

# DISQUETTES

## Chargement des disquettes



Les disquettes sont rapides et sont faciles à utiliser. Il faut manipuler les disquettes et le lecteur de disquettes avec soin. Suivez les étapes suivantes pour charger une disquette:

ETAPE 1 Soyez sûr que votre lecteur de disquettes est branché.



ETAPE 2 Insérez la disquette dans le lecteur. La face de la disquette avec l'étiquette doit toujours être sur le dessus. Insérez la disquette dans l'ouverture du lecteur de manière à ce que le côté de la disquette avec l'étiquette rentre le dernier. Cherchez l'encoche qui se trouve sur le côté de la disquette (elle peut être recouverte d'une petite étiquette). Cette encoche doit se trouver à gauche quand vous insérez la disquette, si vous êtes en face du lecteur. Vérifiez que la disquette est enfoncée en entier.

ETAPE 3 Fermez la porte de protection du lecteur de disquettes quand la disquette est insérée.

ETAPE 4 Tapez:

**DLOAD**"nom du programme"

(nom spécifique  
du programme  
à charger)

Pour gagner du temps, vous pouvez appuyer sur la touche de fonction KEY 2 et vous ne tapez que le nom du programme et le deuxième guillemet.

ETAPE 5 Appuyez sur **RETURN**. La disquette tourne et l'écran affiche :

---

---

**SEARCHING FOR nom du programme**

**LOADING**

**READY**



ETAPE 6 Votre logiciel est prêt à être utilisé. Maintenant tapez RUN et appuyez sur RETURN pour exécuter le programme.

Si la lumière rouge du lecteur de disquettes clignote encore à la fin de l'instruction DLOAD, quelque chose ne va pas. Tapez:

?DS\$ (et appuyez sur **RETURN**)

pour trouver ce qui ne va pas.

Exemples de commandes DLOAD:

<b>DLOAD**</b>	charge le premier programme de la disquette.
<b>DLOAD"FICHIER"</b>	charge un programme appelé FICHIER.
<b>DLOAD"SET**"</b>	charge le premier programme de la disquette qui commence par les lettres SET.

**DLOAD"\$"** charge le répertoire, c'est une liste de tous les programmes qui se trouvent sur la disquette dans le lecteur.

## **Formatage d'une disquette**

Le formatage prépare une disquette vierge à l'utilisation. Toute disquette vierge doit être formatée avant usage, en utilisant la commande HEADER.

**ATTENTION:** Ne formatez pas une disquette qui a déjà été utilisée, à moins que vous ne vouliez effacer toutes les données qui se trouvent sur cette disquette. Formater efface tout ce qu'il y a sur une disquette.

---

---

La syntaxe de la commande HEADER est :

**HEADER** "nom de la disquette",U<sup>°</sup>périphérique<sup>°</sup>,I<sup>°</sup>,Dlecteur<sup>°</sup>

---

- Le nom que vous utilisez est le nom de la disquette. Donnez un nom à la disquette de 16 caractères au plus.
- Le périphérique<sup>°</sup> spécifie quel périphérique est utilisé par votre ordinateur (lecteur de disquettes ou lecteur de cassettes) et c'est généralement le numéro 8.
- Le I est le numéro d'identification de la disquette, il est composé de la lettre I suivie de deux lettres et/ou chiffres, comme, I21,IR5, etc... Donnez à vos disquettes le numéro que vous voulez, mais vous devez donner à chaque disquette un numéro différent pour éviter les erreurs.
- Si vous avez un double lecteur de disquettes, ajoutez D0 ou D1 pour identifier le numéro de lecteur. Si vous avez un lecteur de disquettes simple, vous devez utiliser D0.

### **ARE YOU SURE?**

Dès que vous appuyez sur RETURN après avoir taper la commande HEADER, le Plus/4 vous demande ARE YOU SURE? C'est votre dernière chance pour changer d'avis.

Pour formater la disquette, écrivez YES ou Y et appuyez sur . Si vous décidez de ne pas formater la disquette, écrivez NO ou N et appuyez sur .

Voici quelques exemples de commandes HEADER:

**HEADER"LETTRES",U8,I07,D0**  
**HEADER"FINANCES",U8,IS3,D0**

Maintenant que vous savez comment formater une disquette, vous êtes prêt à écrire et à sauvegarder des programmes sur votre Plus/4. La première section de l'encyclopédie du Plus/4 vous donnera plus d'informations sur la commande HEADER.

## **Sauvegarde de programmes sur disquettes**

Quand vous voulez réutiliser un programme que vous avez écrit, soyez sûr de le sauvegarder avant de charger un autre programme ou d'éteindre le Plus/4. Si vous ne sauvegardez pas, vous perdez le programme.

---

---

Lorsque vous modifiez un programme sauvegardé, vous devez le sauvegarder à nouveau si vous voulez garder la nouvelle version.

Quand vous sauvegardez à nouveau un programme, vous remplacez l'ancienne version par la nouvelle. Si vous voulez garder les deux, l'ancienne et la modifiée, vous devez donner à la nouvelle version un nom différent quand vous la sauvegardez.

Suivez ces étapes pour sauvegarder un programme sur disquette:

ETAPE 1 Tapez **DSAVE** "nom du programme".

ETAPE 2 Appuyez sur **ENTER**. L'ordinateur affiche ce message à l'écran quand le programme est sauvegardé :

**SAVING "nom du programme"**

**OK**

**READY.**



Exemple:

**DSAVE "MONPROG5"**

Le nom du programme peut avoir jusqu'à 16 caractères de long.

Si la lumière rouge du lecteur de disquettes clignote encore à la fin de la commande DSAVE, quelque chose ne va pas. Tapez:

**?DSS** (et appuyez sur **ENTER**)

pour savoir ce qui ne va pas.

## LA COMMANDE DIRECTORY

Quand vous sauvegardez un programme sur disquette, l'ordinateur garde une liste de tous les fichiers sauvegardés sur la disquette. Vous pouvez afficher la liste comme une table des matières pour voir ce qu'il y a sur la disquette en utilisant la commande DIRECTORY:

Tapez : **DIRECTORY** puis appuyez sur **ENTER** (ou bien appuyez sur la touche de fonction F3)

Dès que vous avez appuyé sur **ENTER**, votre Plus/4 affiche tout ce qu'il y a sur la disquette.

Vous pouvez aussi ne faire afficher qu'une partie de la table des matières:

**DIRECTORY"MON\*\*"** puis appuyez sur **ENTER**

Ceci donne la liste de tous les fichiers qui commencent par les lettres MON.

---

---

# CHAPITRE 4

## DEMARRAGE

- 
- Les couleurs du clavier
- 
- Ecriture en couleur et en vidéo inverse
- 
- Quelques programmes simples
- 
- Correction des fautes de frappe
- 
- L'écran du Plus/4
- 
- Pour en savoir plus sur PRINT à l'écran
- 
- Les fenêtres de l'écran
-

---

---

## **INTRODUCTION**

Ce chapitre est fait pour vous faire connaître les caractéristiques et possibilités du Plus/4 et de vous montrer comment franchir les premières étapes de la programmation de votre ordinateur.

---

---

## LES COULEURS DU CLAVIER

Vous pouvez changer la couleur des caractères sur l'écran pour améliorer la lecture ou pour trouver une harmonie de couleur que vous aimez. Pour voir comment les caractères en couleur apparaissent sur l'écran, essayez ceci:

- ETAPE 1 Maintenez le doigt sur la touche 
- ETAPE 2 Appuyez sur la touche 6 pendant que vous maintenez le doigt sur . Le curseur devient vert.
- ETAPE 3 Enlevez le doigt de  et tapez quelques lettres. **Tout ce que vous taperez apparaîtra en vert maintenant.**

---

MAINTENEZ  ET UNE TOUCHE COULEUR	COULEUR RESULTANTE
1	NOIR
2	BLANC
3	ROUGE
4	TURQUOISE
5	POURPRE
6	VERT
7	BLEU
8	JAUNE

---

L'utilisation de  et d'une touche de 1 à 8 vous permet de choisir la couleur figurant sur le haut de la face avant de chaque touche couleur.

Maintenant, maintenez la touche  et en appuyant sur une des touches entre 1 et 8, le curseur prendra la couleur figurant sur le bas de la face avant de chaque touche couleur.

---

MAINTENEZ  ET UNE TOUCHE COULEUR	COULEUR RESULTANTE
1	ORANGE
2	BRUN
3	JAUNE-VERT
4	ROSE
5	BLEU-VERT
6	BLEU CLAIR
7	BLEU FONCE
8	VERT CLAIR

---

---

---

## ECRITURE EN COULEUR ET EN VIDEO INVERSE

Votre Plus/4 peut afficher des chiffres, des lettres et des symboles graphique dans les 16 couleurs différentes. Vous pouvez aussi afficher ces caractères en vidéo inverse, avec les couleurs des caractères et du fond inversées.

ETAPE 1 Videz l'écran de tous caractères en appuyant sur  et .

ETAPE 2 Maintenez la touche  avec le doigt et appuyez sur la touche .



ETAPE 3 Relâchez les touches et maintenez le doigt sur la barre d'espacement (la longue barre au bas du clavier).

ETAPE 4 Maintenez la barre d'espacement aussi longtemps que vous le désirez. Pendant que vous la maintenez, un trait de la même couleur que les lettres de l'écran s'étire. Si le trait arrive à la fin de la ligne, il continue sur la ligne suivante.

ETAPE 5 Relâchez la barre d'espacement (mais n'appuyez pas sur ).

ETAPE 6 Maintenez le doigt sur  et appuyez sur une des touches couleur (une couleur qui n'est pas sur l'écran). Aussitôt que vous faites cela, le curseur deviendra de la couleur que vous venez de choisir.

ETAPE 7 Maintenez le doigt sur la barre d'espacement à nouveau. Maintenant le Plus/4 tirera un trait de la nouvelle couleur. Continuez de changer de couleurs avec , C et les touches couleur. Puis appuyez sur la barre d'espacement pour tracer des traits de couleurs différentes.

ETAPE 8 Désactivez le mode vidéo inverse en maintenant avec le doigt  et en appuyant sur . En appuyant sur , on désactive aussi le mode vidéo inverse.



Essayez de taper quelques lettres en vidéo inverse. Maintenez juste le doigt sur CONTROL et appuyez sur RVS ON pour activer le mode

---

---

---

---

vidéo inverse et tapez ce que vous voulez. Les lettres en vidéo inverse font d'excellents titres. Vous pouvez aussi les utiliser pour souligner des mots et des chiffres importants. Essayez ceci:

**PRINT" R COMMODORE PLUS/4"**

Pour **R** appuyez sur **COMMODORE** et **RVS ON**.

Pour **PLUS/4** appuyez sur **COMMODORE** et **RVS OFF**.

Maintenant essayez la même ligne en remplaçant RVS ON et RVS OFF par FLASH ON et OFF.

Ces deux fonctions peuvent aussi être utilisées sur le Plus/4 comme élément d'une instruction d'un programme.

---

---

## QUELQUES PROGRAMMES SIMPLES DU PLUS/4

Tapez ce programme exactement comme il est imprimé ici. N'omettez pas les chiffres au début de chaque ligne, puisqu'ils sont les numéros de ligne qui indiquent l'ordre d'exécution du programme. Assurez-vous d'appuyer sur **ENTER** à la fin de chaque ligne que vous tapez.

10 PRINT"PLUS/4"

← La ligne 10 indique à l'ordinateur d'imprimer Plus/4 sur l'écran.

20 GOTO 10

← La ligne 20 indique à l'ordinateur de retourner à la ligne 10.

RUN

← RUN ordonne à l'ordinateur d'exécuter les deux lignes du programme.

Appuyez sur **ENTER** pour stopper le programme. Pourquoi votre Plus/4 a-t-il imprimé son nom autant de fois? GOTO dit à l'ordinateur de revenir à la ligne 10 et d'écrire Plus/4 encore et encore. Cette répétition est appelée une boucle.

Maintenant, tapez ceci:

NEW et **ENTER**

← Vous avez dit à l'ordinateur d'oublier le dernier programme et de se préparer pour un nouveau.

Le message suivant apparaît à l'écran:

READY

← Vous ne tapez pas ceci; c'est votre Plus/4 qui l'écrit pour vous dire qu'il est prêt pour un nouveau programme.

---

---

10 PRINT"PLUS/4"

Même ligne 10. PRINT indique au Plus/4 d'afficher ce qu'il y a entre "".

20 COLOR 0,12

La ligne 20 indique à l'ordinateur de changer la couleur de l'écran.

## RUN

Cette fois, il n'y a pas de boucle GOTO dans le programme, alors vos ordres ne sont exécutés qu'une fois.

## CORRECTION DES FAUTES DE FRAPPE

Si vous faites une faute de frappe, il y a plusieurs manières de les corriger:

1. Vous pouvez retaper une ligne quand vous le voulez, même après avoir exécuté le programme avec l'instruction RUN. Le Plus/4 remplace l'ancienne ligne par la nouvelle quand vous appuyez sur [REDACTED] pour entrer la nouvelle ligne. L'ancienne apparaît toujours à l'écran, mais le Plus/4 l'ignore. Quand vous avez deux instructions avec le même numéro de ligne, votre Plus/4 utilise la dernière ligne entrée. Par exemple, dans un programme court utilisant la commande COLOR pour changer la couleur de fond de l'écran, une erreur peut être faite :

10 COKOR 0,3  
20 PRINT"PLUS/4"

Erreur en ligne 10

Appuyez sur [REDACTED] pour aller à la ligne et retapez simplement la ligne 10 correctement:

10 COLOR 0,3 et [REDACTED]

Maintenant, la première ligne 10 est remplacée par la seconde ligne 10. Vous pouvez vérifier en tapant LIST, qui affiche une liste ligne par ligne de votre programme (image de la façon dont votre

---

---

programme est stocké dans votre ordinateur). Quand vous demandez la liste d'un programme, toutes les lignes apparaissent dans l'ordre correct et les lignes remplacées n'apparaissent pas.

**LIST** et [REDACTED]

On peut lire sur l'écran:

**10 COLOR 0,3**  
**20 PRINT"PLUS/4"**

Remplacer des lignes dans un programme est aussi une bonne manière de faire des expériences avec votre ordinateur. Quand vous remplacez une ligne, la nouvelle ligne peut être complètement différente de l'ancienne. Par exemple, au lieu de corriger la faute de frappe sur COLOR, vous pouvez taper ceci:

**10 PRINT"DIX EST SIX "** (avec un espace entre SIX et ")

Appuyez sur [REDACTED]

Maintenant tapez RUN et voyez ce qui se passe.

2. Vous pouvez effacer une ligne que vous ne voulez pas en tapant juste le numéro de la ligne et en appuyant sur [REDACTED]. L'ordinateur ignore la ligne même si elle apparaît toujours à l'écran. Tapez LIST pour obtenir la liste du programme pour être sûr que la ligne a bien été effacée du programme.

**10 PRINT"DIX EST SIX" [REDACTED]**

**20 PRINT"PLUS/4" [REDACTED]**

**10 [REDACTED]**

**LIST [REDACTED]**

**20 PRINT"PLUS/4"**

3. Vous pouvez modifier une ligne. Utilisez les touches du curseur pour le placer à l'endroit que vous voulez modifier dans la ligne. Maintenant retapez ce que vous voulez. Appuyez sur [REDACTED] quand vous avez fini.

---

---

NOTE : Quand vous travaillez sur des programmes avec des lignes numérotées, vous n'avez pas à être à la fin de la ligne pour appuyer sur **RETURN**. Votre Plus/4 mémorise la ligne entière même si vous appuyez sur RETURN au milieu de la ligne.

#### 10 PRINT"IL EST DEUX HEURES"

Si vous voulez changer le DEUX en CINQ, déplacez le curseur sur le D de DEUX.

#### 10 PRINT"IL EST|D)EUX HEURES"

Et maintenant tapez simplement CINQ au-dessus de DEUX et appuyez sur **RETURN**

#### 10 PRINT"IL EST CINQ HEURES"

---

NOTE: Quand vous tapez un " après une instruction PRINT, vous entrez dans le mode guillemets. Dans le mode guillemets, certaines touches ont des résultats différents. Par exemple, si vous appuyez sur la flèche vers le bas du curseur quand vous êtes en mode guillemets, le curseur ne bougera pas et vous verrez un Q en vidéo inversé s'imprimer sur l'écran. Quand vous exécutez l'instruction PRINT, le Q en vidéo inversé n'est pas écrit; mais le curseur se déplace vers le bas. En mode guillemets, l'ordinateur assume que tout ce que vous tapez, c'est ce que vous voulez voir ou faire plus tard quand vous exécuterez l'instruction PRINT.

---

4. Vous pouvez insérer des blancs dans un mot ou dans une ligne avec la touche **INST** (vous l'obtenez en maintenant avec le doigt **SHIFT** et en appuyant sur **INST/DEL**). Maintenez cette touche jusqu'à ce que vous ayez autant d'espaces que vous le voulez. (Notez que le curseur reste à la même place pendant que les espaces sont créés à droite). Puis tapez juste ce que vous voulez insérer.

#### 10 PRINT"CORE" RETURN

Pour changer ceci en COMMODORE-Plus/4, déplacez le curseur sur le C et appuyez sur **SHIFT** et **INST** jusqu'à ce que vous ayez suffisamment d'espaces créés. Ne vous souciez pas de compter les espaces. Vous pouvez juste deviner et puis insérer plus de blancs s'il n'y en a pas assez.

---

---

10 PRINT "C █ ORE"

↙ curseur

Maintenant ajoutez les autres lettres:

10 PRINT "COMMODORE" RETURN

5. Vous pouvez effacer des caractères et supprimer des blancs avec la touche **DEL** (vous l'obtenez en appuyant sur INST/DEL). Cette touche efface les caractères et les blancs tout de suite à gauche du curseur.

10 PRINT "MATIN D'ETE" RETURN

Vous pouvez changer ceci en SOIR D'ETE en déplaçant le curseur sur le A de MATIN, appuyez sur **INST/DEL** et tapez SOIR.

10 PRINT "M~~A~~TIN D'ETE"

-

Appuyez sur **INST/DEL** une fois

10 PRINT "ATIN D'ETE"

-

Tapez SOIR pour  
remplacer ATIN  
et appuyez sur **RETURN** .

## Un programme un peu plus long.

Maintenant que vous vous êtes exercé sur votre Plus/4, voici un programme à essayer qui sera un peu plus long à taper.

Premièrement, vider l'écran en maintenant le doigt sur **SHIFT** et en appuyant sur **CLR/HOME** . Ceci vide l'écran. Puis, vider la mémoire des programmes anciens en tapant NEW et en appuyant sur **RETURN** .

Tapez le programme exactement comme il est écrit. Assurez-vous de taper les numéros de ligne et tous les signes de ponctuation. Utilisez les messages pour corriger les erreurs si vous tapez quelque chose d'incorrect. N'oubliez pas d'appuyer sur **RETURN** à la fin de chaque ligne.

---

---

NOTE : Souvenez-vous que vous pouvez arrêter un programme en appuyant sur **RUN/STOP**

---

---

---

---

## NEW

10 COLOR 1,8

20 PRINT "QUELQUE CHOSE DE DROLE EST ARRIVE ";

30 COLOR 1,3

40 PRINT "EN CHEMIN VERS LE CLAVIER ";

50 COLOR 1,7

60 PRINT " ♥♥♥♥♥♥ "

70 GOTO 60

## RUN

Ligne 20, assurez de laisser un espace entre la dernière lettre et le " de fin.

Même commentaire pour la ligne 40.

Ligne 60, pour obtenir des cœurs, maintenez la touche  pendant que vous appuyez 6 fois sur le S.

Après avoir arrêté le programme (en appuyant sur ) , essayez de taper LIST. Le programme est affiché à l'écran, revoyez les messages pour corriger les fautes de frappe et essayez de changer ce programme pour lui faire dire quelque chose de plus intéressant.

CONSEIL: Voulez-vous ralentir ce programme sans l'arrêter?  
Maintenez simplement la touche 

## ECRAN TEXTE DU PLUS/4

Essayez de taper ce programme. (N'oubliez pas d'appuyer sur RETURN à la fin de chaque ligne)

## NEW

10 PRINT " ♥ ";(pour le cœur appuyez sur  et S)

---

---

---

---

20 GOTO 10

RUN

Maintenant votre écran se remplit de cœurs. Quand l'écran est entièrement couvert de cœurs, appuyez sur **ESC** pour arrêter le programme. Ce programme vous montre combien votre écran du Plus/4 est grand.

Maintenant tapez ce programme:

NEW

10 PRINT"♥" (pour le cœur en vidéo inversé, appuyez sur **ESC** et **ESC**)

20 FOR X=1 TO 40

30 PRINT"♥"; (pour le cœur appuyez sur **ESC** et s)

40 NEXT X

RUN

Lorsque vous exécutez ce programme, la première ligne de votre écran se remplit de cœurs. Il y a en tout 40 cœurs. Puisque la ligne est pleine, vous pouvez en déduire qu'il y a 40 positions le long de l'écran. Chaque position le long de l'écran est appelée une colonne.

Maintenant tapez ce programme:

NEW

10 PRINT"♥" (appuyez sur **ESC** et **ESC**)

20 FOR X=1 TO 25

30 PRINT"♦" (appuyez sur **ESC** et Z)

40 NEXT X

---

---

## RUN

Quand vous exécutez ce programme, la première colonne de votre écran se remplit de carreaux. Il y a 25 carreaux écrits, mais les trois premiers ont disparu en haut de l'écran parce que le mot READY entouré de deux lignes blanches apparaît toujours à la fin des programmes. Il y a, alors, 25 lignes. Un peu de logique vous permet de déduire que votre Plus/4 a 40 colonnes et 25 lignes. Le Plus/4 a 1000 positions différentes sur l'écran pour des lettres, chiffres, symboles graphiques, etc...

NOTE: Quelquefois vous taperez des lignes particulièrement longues sur votre Plus/4, comme celle-ci :

### 10 PRINT "J'AIME VOTRE TOUCHER SUR MON CLAVIER. VOUS VENEZ SOUVENT ICI?"

(C'est une longue ligne - avec plus de 50 caractères!)

Vous remarquerez que comme vous tapez ceci, vous n'avez pas assez de place sur la ligne. Mais continuez: le Plus/4 se déplace automatiquement sur la ligne suivante et continue à écrire jusqu'à ce que la ligne soit finie. Vous pouvez taper jusqu'à 80 caractères sur une ligne de programme (deux lignes entières).

Maintenant essayez d'exécuter ces lignes de programme. Le message s'inscrit en deux lignes. Si votre message est plus long qu'une ligne, le Plus/4 le laisse dépasser sur la ligne suivante. Le Plus/4 considère que le message est terminé quand vous appuyez sur **RETURN** et non quand vous arrivez à la fin d'une ligne. Vous vous habituerez à cela en vous exerçant sur votre Plus/4.

Maintenant tapez ce programme:

## NEW

```
10 PRINT " ♥ ";
```

```
20 GOTO 10
```

## RUN

Ligne 10, laissez un espace de chaque côté du cœur, pour le cœur tapez **SHIFT** et S.

Quand vous exécutez ce programme, vous pouvez voir qu'il est possible de dire exactement au Plus/4 où écrire quelque chose à l'écran.

---

---

## EN SAVOIR PLUS AU SUJET DE L'ECRITURE SUR ECRAN

Essayez ce programme:

```
NEW
10 PRINT "A", "B"
20 PRINT "A"; "B"
RUN
```

Voici ce que vous verrez à l'écran:

```
A           B ← (écriture de la ligne 10)
AB ← (écriture de la ligne 20)
```

Si les lignes 10 et 20 sont presque identiques, pourquoi y a-t-il une telle différence dans leur écriture à l'écran? La différence provient de la ponctuation entre les deux éléments à écrire.

Quand vous utilisez une virgule pour séparer les éléments dans une instruction PRINT, les éléments sont imprimés avec des espaces entre eux. Quand vous utilisez un point-virgule, les éléments sont écrits l'un à côté de l'autre.

Comme vous vous en souvenez, l'écran du Plus/4 a 40 colonnes. Ces colonnes sont divisées en quatre zones de 10 espaces, appelées PRINT ZONES. Quand vous utilisez une virgule pour séparer des éléments à écrire, le Plus/4 écrit le premier élément dans la première zone, le second élément dans la seconde zone, etc... Les virgules correspondent aux tabulations de machine à écrire.

PRINT ZONE 1										PRINT ZONE 2										PRINT ZONE 3										PRINT ZONE 4											
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40		
A										B																															

Si vous voulez écrire plus de quatre éléments séparés par des virgules, le Plus/4 ira automatiquement à la ligne. Par exemple:

```
PRINT "A", "B", "C", "D", "E", "F"
```

espace les lettres à l'écran comme suit:

		1	11	21	31	COLONNE
LIGNE 1	A	B	C	D		
2	E	F				

Quand vous utilisez des point-virgules pour séparer les éléments dans une instruction PRINT, le Plus/4 ignore les PRINT ZONES et écrit tous les éléments les uns à la suite des autres.

---

---

```
PRINT "A";"B";"C";"D";"E";"F"
```

donnera:

```
ABCDEF
```

Voici ce qui se passe quand le premier mot a une longueur de 12 caractères et est séparé du second par une virgule:

```
PRINT "ABCDEFGHIJKL","M"
```

donnera ceci:

```
ABCDEFGHIJKL      M
                zone 1      zone 3      zone 4
```

Maintenant videz votre écran et tapez ce programme:

```
NEW
```

```
10 PRINT 1,2
```

```
20 PRINT 1;2
```

```
RUN
```

```
1   2
1 2
```

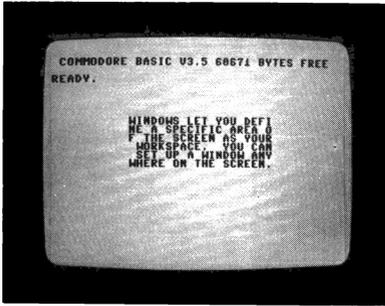
Ce programme vous montre deux choses:

1. Les chiffres n'ont pas besoin de guillemets dans l'instruction PRINT.
2. Les chiffres sont affichés avec un espace de chaque côté, ainsi si vous utilisez le point-virgule, les chiffres ne sont pas écrits les uns à la suite des autres comme les lettres, un espace est réservé pour le signe moins si nécessaire.

---

---

## FENETRES



Les fenêtres vous permettent de définir une partie de l'écran comme espace de travail. Tout ce que vous taperez (lignes que vous tapez, liste de programme, etc...) après avoir défini une fenêtre apparaîtra à l'intérieur de la fenêtre, sans modifier le reste de l'écran. Vous pouvez définir une fenêtre à tout endroit de l'écran.

Pour définir une fenêtre, suivez les étapes suivantes:

1. Déplacez le curseur à la position du coin supérieur gauche de la fenêtre.
2. Appuyez sur la touche ESCape et ensuite sur la touche T.
3. Déplacez le curseur à la position du coin inférieur droit de la fenêtre.
4. Appuyez sur la touche ESCape et ensuite sur la touche B. Votre fenêtre est maintenant définie.

Toutes les sorties écran se feront dans le rectangle ainsi défini. Pour supprimer la fenêtre, appuyez sur la touche HOME deux fois de suite. La fenêtre est maintenant supprimée et le curseur est positionné dans le coin supérieur gauche de l'écran.

---

---

# CHAPITRE 5

## CHIFFRES ET CALCULS

- 
- Chiffres et opérateurs BASIC
- 
- Calculs
- 
- Utilisation des variables
- 
- Mode direct
- 
- Fonctions numériques
- 
- Nombres aléatoires et autres fonctions
-

---

---

## CHIFFRES ET OPERATEURS BASIC

Vous pouvez utiliser votre Plus/4 comme une simple calculatrice. En plus des signes standards + et -, votre Plus/4 utilise le signe \* pour la multiplication et le signe / pour la division et les fractions. (Les ordinateurs utilisent le signe \* pour la multiplication au lieu du signe X, parce qu'ils ne peuvent pas faire la différence entre la lettre X et le symbole X). Vous pouvez utiliser ces opérateurs et les chiffres en mode direct (sans numéro de ligne) ou en mode programme. Il ne faut pas mettre les chiffres ou les opérateurs entre guillemets pour les opérations.

<b>OPERATEURS BASIC MATHÉMATIQUES</b>		<b>OPERATEURS BASIC LOGIQUES</b>	
Addition	+	Plus grand que	>
Soustraction	-	Plus petit que	<
Division et fraction	/	Egal	=
Multiplication	*	Plus grand que ou égal	=>
Puissance	↑	Plus petit que ou égal	<=
		Différent de	<> ou ><

NOTE: Votre Plus/4 n'accepte pas de virgule dans les nombres. Par exemple, vous devez taper 1094.01 au lieu de 1094,01. Si vous insérez une virgule dans un nombre, le Plus/4 considère que vous lui donnez deux nombres (séparés par une virgule), ainsi votre Plus/4 lira 1094 et 01 au lieu de 1094.01.

## FRACTIONS ET DECIMALES

Vous pouvez écrire une fraction comme ceci : .5 ou 1/2 (votre Plus/4 calcule automatiquement la division).

Si vous donnez une fraction dans une instruction PRINT, vous obtiendrez un nombre entier ou un nombre décimal. Par exemple :

```
PRINT 139/493+5 5.28194726
```

Voici un exemple utilisant le nombre pi(3.14...) qui représente le rapport de la circonférence d'un cercle à son diamètre. Utilisez cette valeur en appuyant sur la touche pi.

```
PRINT pi/374 8.39998036E-03
```

---

---

## NOTATION SCIENTIFIQUE

Que veut dire votre Plus/4 par l'expression E-03?

Votre Plus/4 peut afficher des nombres décimaux de – 999 999 999 à 999 999 999 dans la numérotation standard. Les nombres plus grands ou plus petits (avec plus de 9 chiffres) sont automatiquement affichés en notation scientifique. Vous pouvez entrer des nombres sous cette forme et votre Plus/4 les lira sans problème (avec certainement moins de problèmes que vous). La notation scientifique est souvent utile, puisqu'elle permet d'afficher des grands nombres avec moins de chiffres.

Voici comment le nombre 198 505 478 est écrit en notation scientifique:

1.98505478E+8

Il n'y a qu'un seul chiffre à gauche du point décimal, le chiffre après la lettre E désigne le nombre de chiffres qui sont après la virgule.

Pour un nombre plus petit que un avec plusieurs décimales, le second nombre après la lettre E est en négatif, pour indiquer que le point décimal a été déplacé vers la droite.

Par exemple:

.0003359 = 3.359E – 4

Autres exemples:

20 = 2E + 1    le point décimal est déplacé d'un chiffre vers la  
105000 = 1.05E + 5    le point décimal est déplacé de 5 chiffres vers l  
.0666 = 6.66E – 2    le point décimal est déplacé de 2 chiffres vers l

---

---

## CALCULS

Pour calculer, tapez PRINT et une expression mathématique.  
Souvenez-vous de ne pas mettre de guillemets.  
Tapez ce programme:

**NEW**

**10 PRINT 1+2, 2-1**

**20 PRINT 2\*2, 4/2**

**RUN**

**3        1**

**4        2**

(utilisez le / de la touche?)

Pour la première fois, l'instruction PRINT n'affiche pas exactement ce que vous avez tapé. Ainsi, votre Plus/4 résoud des opérations et affiche les réponses. Vous n'avez qu'à utiliser la commande PRINT pour calculer en ne mettant pas de guillemets. Maintenant essayez ceci:

**NEW**

**10 PRINT "2001/2010"**

**20 PRINT 2\*3**

**RUN**

**2001/2010**

**6**

(un espace est réservé pour le signe)

Puisque l'expression mathématique de la ligne 10 est entre guillemets, votre Plus/4 l'affiche comme un texte: exactement comme on l'a écrit entre les guillemets. L'expression n'est pas calculée et aucun espace n'est réservé pour le signe.

Maintenant déplacez le curseur sur la ligne 10 et modifiez la ligne comme suit:

**10 PRINT "2\*3 + 1 =";2\*3+1**

(n'oubliez pas le point-virgule)

**RUN**

**2\*3+1= 7**

(l'espace à gauche du 7 est réservé pour le signe)

**6**

(la réponse à la ligne 20 reste la même)

Si vous voulez afficher à la fois le problème et la solution, vous n'avez qu'à les taper deux fois: une fois entre les guillemets et une fois sans, comme suit:

**10 PRINT "2+2=";2+2**

---

---

---

---

## MODE DIRECT

Vous pouvez toujours insérer un calcul dans un programme, ou obtenir une réponse immédiate en tapant PRINT suivi de l'expression à calculer sans numéro de ligne, puis appuyez sur RETURN, comme suit:

```
PRINT 3-6
-3
PRINT 24/(6+2)
3
```

Quand vous n'indiquez pas de numéro de ligne avant une instruction BASIC, vous n'avez pas besoin de taper RUN, pour que l'ordinateur exécute l'instruction; vous êtes en mode immédiat ou direct. Si vous indiquez un numéro de ligne, cela signifie que l'instruction qui suit ce numéro fait partie d'un programme BASIC; vous êtes en mode programme. Ces deux modes sont acceptables par l'ordinateur.

Vous pouvez aussi inclure du texte entre guillemets et des expressions mathématiques à résoudre dans une seule instruction PRINT en mode direct.

```
PRINT "2 A LA PUISSANCE 3 EGALE";2↑3
```

(le ↑ symbolise la puissance, pour l'obtenir tapez SHIFT et 0)

```
2 A LA PUISSANCE 3 EGALE 8
```

## PRIORITE DES OPERATEURS

Le second exemple du dernier paragraphe vous montre que vous pouvez effectuer plus d'un calcul sur une seule ligne. Essayez ceci :

```
PRINT 200+50/5
```

Est-ce la réponse que vous attendiez ?

Essayez ceci :

```
PRINT (200+50)/5
```

Votre Plus/4 effectue toujours les calculs dans un certain ordre. Les calculs sont effectués de la gauche vers la droite; dans le cadre de cette règle générale, certaines opérations sont prioritaires. L'ordre dans lequel votre Plus/4 effectue les opérations, s'appelle la priorité des opérateurs.

---

---

PREMIEREMENT: votre Plus/4 s'occupe des nombres négatifs (pas les soustractions, juste les nombres négatifs).

DEUXIEMENT: votre Plus/4 résoud les expressions contenant des puissances.

TROISIEMENT: votre Plus/4 résoud toutes les multiplications et les divisions, de la gauche vers la droite.

QUATRIEMENT: votre Plus/4 résoud les additions et les soustractions de la gauche vers la droite.

<p>NOTE: Votre Plus/4 résoud en premier les expressions entre parenthèses. Vous pouvez aussi ajouter des parenthèses à l'intérieur de parenthèses: <math>36 * (12 + (A/3))</math>. Le contenu des parenthèses les plus à l'intérieur est calculé en premier.</p>
--

Pour plus de clarté, il vaut mieux mettre les nombres négatifs entre parenthèses. Par exemple, si vous voulez multiplier 45 par  $-5$ , tapez :  $45 * (-5)$ . Votre Plus/4 peut comprendre avec ou sans parenthèses.

---

---

## UTILISATION DES VARIABLES

L'exemple 36\* (12 + (A/3)) démontre une des caractéristiques les plus puissantes d'un ordinateur. Quand vous utilisez une lettre au lieu d'un nombre dans une expression mathématique, vous utilisez une variable. Une variable représente une valeur :

10 A = 3

20 PRINT "TOTAL: "; A \* 4

Si vous lancez ce programme, le résultat à l'écran :

TOTAL : 12

Il y a trois types de variables que vous pouvez utiliser :

---

TYPE	SYMBOLE	DESCRIPTION	EXEMPLES	VALEURS TYPE
Virgule flottante		réel (décimal) ou nombres entiers	X, AB, T4	23.5, 12, 1.3E + 2
Entier	%	nombres entiers	X%, A1%	15, 102, 3
Chaîne de Caractères	\$	lettres, nombres et tous les autres caractères entre guillemets	X\$, MS\$	"TOTAL:", "DAY1", "CBM"

---

Chaque fois que vous voulez qu'une variable soit entière, le symbole de cette variable doit inclure le signe %. Une variable contenant du texte doit se terminer par le signe \$. Si la variable ne possède pas un de ces symboles (%,\$), votre Plus/4 la considère comme une variable type virgule flottante (un nombre réel). Les variables entières sont un sous-ensemble des variables de type virgule flottante; ce sont des nombres réels sans chiffres après la virgule.

Utilisez toujours le bon type de variable. Si vous essayez d'assigner une chaîne de caractères à une variable entière, votre programme ne marchera pas. Le programme suivant vous montre quelles variables peuvent être utilisées dans une situation donnée. Vous verrez ce qui se passe en utilisant différents types de données:

10 REM PROGRAMME AVEC DES DONNEES NUMERIQUES

20 PRINT "ENTREZ UN NOMBRE"

30 INPUT X%

40 PRINT "BRAVO!"

50 PRINT "VOTRE NOMBRE EST "; X%

(c'est la variable)

Essayez d'entrer des valeurs et regardez ce qui se passe :

UN CINQUIEME

.043

10

---

---

---

---

## FONCTIONS NUMERIQUES

Des fonctions numériques sont inclus dans le BASIC 3.5 de votre Plus/4, ce sont les fonctions mathématiques que l'on retrouve sur la plupart des calculettes scientifiques (telles que sinus, cos, tg, etc...)

La plupart de ces fonctions peuvent être utilisées en tapant le nom de la fonction suivi de nombres entre parenthèses, comme ceci :

### FUNCTION(X)

Exemple : pour trouver le sinus d'une variable, vous tapez :

**PRINT SIN(X)**

avec X comme variable à entrer.

Vous pouvez aussi inclure une de ces fonctions dans une ligne de programme, comme le montre l'exemple suivant :

```
10 FOR X=1 TO 5  
20 PRINT "LA RACINE CARREE DE";X;"EST";SQR(X)  
30 NEXT X
```

Vous avez une liste complète de ces fonctions numériques dans l'encyclopédie du BASIC 3.5. Quelques fonctions plus complexes sont expliquées dans le paragraphe suivant.

## NOMBRES ALEATOIRES ET AUTRES FONCTIONS

Choisir un nombre aléatoire, c'est prendre 10 morceaux de papier, d'écrire un nombre de 1 à 10 sur chaque morceau, de mettre ces morceaux dans un chapeau et d'en tirer un au hasard. Le nombre choisi est un nombre aléatoire. Le morceau de papier est remis dans le chapeau et on effectue un second tirage. Chaque fois que l'on tire un morceau de papier on le remet dans le chapeau, on garde ainsi à chaque fois les mêmes possibilités. Quand un nombre est sélectionné, il n'y a aucun moyen de connaître le suivant, mais vous savez que c'est un nombre entre 1 et 10. C'est la base des nombres aléatoires.

Les nombres aléatoires sont utiles à la programmation, en donnant des éléments de chance ou de hasard. Les nombres aléatoires ont généralement des bornes, c'est-à-dire des limites supérieures et inférieures. Dans l'exemple du chapeau, les bornes sont le 1 et le 10. La limite inférieure est le 1, la limite supérieure est le 10, ce qui signifie que tout nombre entre 1 et 10 a la même chance d'être sélectionné.

---

---

Examinons comment votre Plus/4 génère des nombres aléatoires et ce qu'on peut faire avec. Le programme suivant génère cinq nombres aléatoires.

```
10 FOR X=1 TO 5:PRINT RND(X):NEXT X
```

Ces nombres aléatoires sont tous plutôt compliqués, avec plusieurs chiffres après la virgule... mais l'usage des nombres aléatoires nécessite des nombres entiers. Vous pouvez transformer ces nombres en nombres entiers (sans chiffres après la virgule) en utilisant la fonction `INTeger`. Cette dernière supprime tous les chiffres à droite de la virgule. L'exemple suivant vous permet de générer un nombre aléatoire dans les limites que vous choisissez. Vous pourrez utiliser cette formule chaque fois que vous voudrez utiliser une variable ou un nombre dans votre programme.

**INT**(taille\***RND**(1))borne inférieure

La commande `INT` indique à l'ordinateur de supprimer tous les chiffres après la virgule et de ne vous donner que la partie entière, comme 1, 45 ou 320 à la place des nombres 1.223, 45.667 ou 320.59. Les nombres entiers sont plus faciles à utiliser pour les nombres aléatoires.

**Borne inférieure** : correspond au plus petit nombre qui peut être choisi par l'ordinateur.

**Taille** : c'est la quantité de nombres dans le groupe.

Par exemple, si vous voulez choisir un nombre aléatoire entre 1 et 5, la borne inférieure est 1 et la taille est 5. Si vous voulez choisir un nombre aléatoire entre 15 et 20, la borne inférieure est 15 et la taille est 6, parce que vous avez le choix entre 6 nombres. Si vous choisissez un nombre entre 2 et 100, la borne inférieure est 2 et la taille est 99. Maintenant essayons un programme.

```
10 PRINT INT(5*RND(1))+1
```

Tapez `RUN` et appuyez sur . Lancez le programme plusieurs fois, et à chaque fois vous obtiendrez un nombre aléatoire entre 1 et 5. Maintenant nous allons afficher 15 nombres aléatoires avec une borne inférieure 1 et une taille 5... Notez que ces 15 nombres choisis sont sélectionnés au hasard entre 1 et 5.

---

```
10 FOR X=1 TO 15 (boucle 15 fois)
20 PRINT INT(5*RND(1))+1
20 NEXT X
```

(sélectionne des nombres aléatoires)

Tapez RUN et appuyez sur RETURN.

Une manière efficace d'utiliser cette formule est de la transformer en une **fonction définie par l'utilisateur**. Ce type de fonction est très utile pour les calculs mathématiques et est facile à utiliser sur votre Plus/4. Les fonctions définies par l'utilisateur vous permettent de programmer une formule et laissent votre Plus/4 donner des valeurs à calculer. Ceci peut être utilisé dans des buts différents. La section 10 de l'encyclopédie contient une liste des fonctions mathématiques dérivées que vous pouvez utiliser comme des fonctions définies par l'utilisateur.

Voici une instruction utilisant la fonction définie par l'utilisateur pour générer des nombres aléatoires:

```
10 DEF FNR(X)=INT(X*RND(1))+1
```

Cela nous donne un nombre aléatoire compris entre 1 et X. FNR est le nom de la fonction définie par l'instruction.

Exemple utilisant cette fonction:

```
10 DEF FNR(X)=INT(X*RND(1))+1
20 DO
30 COLOR 1,FNR(16),5:REM DONNE UNE COULEUR ENTRE 1 ET 16
40 PRINT"LA RECHERCHE CONTINUE"
50 LOOP
```

L'utilisation des fonctions définies par l'utilisateur permet de gagner de l'espace mémoire, si vous utilisez plus d'une fois la fonction, et rend votre programme plus facile à lire et à comprendre.

---

---

# **CHAPITRE 6**

## **INITIATION A LA PROGRAMMATION BASIC**

- 
- Introduction
  - Modes programme
  - Instructions Entrée/Sortie (E/S)
  - Instructions de contrôle et boucles
  - Instructions conditionnelles
  - Sous – programmes
  - REMarque
-

---

---

## INTRODUCTION

Jusqu'à maintenant vous avez ou n'avez pas compris exactement ce qui se passait dans les programmes qui vous montraient les possibilités de votre Plus/4. Ce chapitre va vous expliquer les commandes BASIC qui ont été utilisées dans ces programmes. Ce chapitre concerne seulement les instructions BASIC les plus utilisées, dont vous aurez besoin pour faire vos propres programmes. A la fin du chapitre, nous survolerons quelques techniques de programmation. Ce chapitre vous donnera une introduction à la programmation, mais ce n'est qu'une introduction. Pour apprendre vraiment à programmer, nous vous suggérons de choisir un bon livre de BASIC. (Voir la bibliographie de la section 14 de l'encyclopédie pour des suggestions). Il y a plusieurs versions de BASIC, chacune un peu différente. Votre Plus/4 est équipé d'une version avancée du langage BASIC, appelée Commodore BASIC 3.5.

---

---

## LES MODES PROGRAMME

Votre Plus/4 utilise les instructions et les commandes BASIC de deux manières: en mode direct et en mode indirect. On appelle aussi le mode direct mode immédiat et le mode indirect est aussi connu comme le mode programme.

Le mode direct ou immédiat, comme son nom l'indique, exécute de instructions et des commandes immédiatement (aussitôt que vous appuyez sur **RETURN** après avoir tapé une commande). Vous n'indiquez pas de numéro de ligne quand vous écrivez des commandes et des instructions en mode direct. Vous tapez simplement la commande ou l'instruction et vous appuyez sur **RETURN**. Ce mode est utilisé si vous voulez réaliser des calculs et obtenir un résultat immédiat. Des commandes telles que LIST, SAVE, LOAD, VERIFY et RUN sont généralement utilisées en mode direct. La plupart (mais pas toutes) des instructions BASIC fonctionnent en mode direct.

Le mode programme ou indirect, vous permet de regrouper une série d'instructions BASIC dans un jeu d'instructions qui seront exécutées dans l'ordre que vous avez donné. Chaque ligne du programme a un numéro qui indique à l'ordinateur le numéro d'ordre pour l'exécution du programme. Vous avez déjà vu plusieurs exemples du mode programme dans le chapitre 4. Vous devez vous souvenir que lorsque vous êtes en mode programme, vous devez appuyer sur **RETURN** pour entrer chaque ligne du programme en mémoire. Si vous n'appuyez pas sur **RETURN**, et que vous allez simplement à la ligne, la dernière ligne que vous venez de taper n'est pas entrée. Une fois que le programme est en mémoire, rien ne se passe jusqu'à ce que vous tapiez la commande RUN. La commande RUN dit au Plus/4 d'exécuter le programme en commençant par la ligne avec le numéro le plus petit.

Les lignes sont généralement numérotées de 10 en 10, parce que vous aurez souvent besoin d'insérer des lignes dans un programme à des endroits différents. Vous pourrez, si vous en avez besoin, ajouter neuf lignes nouvelles entre les lignes 10 et 20 d'un programme, par exemple. Cependant votre Plus/4 possède une commande BASIC RENUMBER, qui permet de changer les numéros de lignes. Cette commande vous permet donc d'ajouter des lignes et de remettre les lignes dans le bon ordre facilement.

---

## LES INSTRUCTIONS ENTREE/SORTIE

Les instructions ENTREE/SORTIE (E/S) sont utilisées dans les programmes pour communiquer avec la personne qui lance le programme. Avant de lancer le programme, si toutes les données à calculer sont disponibles, on n'utilise par une instruction ENTREE. Mais c'est souvent plus utile si l'ordinateur peut obtenir des données de la personne qui lance le programme (nous l'appellerons l'utilisateur du programme). Les programmes sont beaucoup plus souples si les données n'en font pas partie avant de les lancer.

Les instructions SORTIE peuvent être utilisées par l'ordinateur pour donner à l'utilisateur du programme les résultats des calculs.

Evidemment les instruction SORTIE sont nécessaires; il n'y aurait pas de sens à lancer un programme sans résultat.

Les programmeurs expérimentés utilisent aussi les instructions I/O pour communiquer avec les périphériques. Vous l'avez déjà fait vous-même, mais pas dans un programme, quand vous avez utilisé LOAD et SAVE avec votre Datassette ou votre lecteur de diquettes. LOAD est fondamentalement une instruction ENTREE puisque le Plus/4 obtient des données (votre programme) du Datassette ou du lecteur de disquettes. SAVE est une instruction SORTIE, car le Plus/4 envoie des données aux périphériques.

Dans cette introduction aux instructions I/O, nous nous limiterons aux instructions les plus courantes, celles que vous allez utiliser immédiatement: PRINT, INPUT, GETKEY et READ/DATA. PRINT est une instruction SORTIE, alors que les autres instructions sont de type ENTREE. (Toutes les instructions BASIC E/S peuvent être trouvées dans l'encyclopédie BASIC, à la fin de ce livre.)

---

Instruction : **PRINT**

Syntaxe : **PRINT** "texte entre guillemets" ou variables, chiffres, calculs, etc...

Vous avez déjà utilisé PRINT dans les programmes des chapitres précédents. D'après ces exemples et d'après la syntaxe, vous voyez que PRINT est une instruction très souple. Vous pouvez l'utiliser pour imprimer des messages, des dessins faits à partir de caractères graphiques, pour calculer, pour afficher la valeur d'une variable ou autres choses. Puisque l'instruction PRINT est très souvent utilisée, il vaut mieux apprendre à l'utiliser correctement.

### Utilisation n°1. Affichage de texte.

Supposons que dans votre programme, vous vouliez informer l'utilisateur du programme que le solde de son compte bancaire est négatif, ou que les lézards pourpres ne sont pas autorisés dans la

---

---

salle de contrôle. La manière la plus simple serait d'écrire votre instruction comme une chaîne de caractères. Les chaînes de caractères sont écrites exactement comme vous lez tapez. Elles sont entourées de guillemets (""). Par exemple:

**100 PRINT "VOUS ETES A DECOUVERT!"**

informerait l'utilisateur du programme que son compte bancaire est vide, alors que

**150 PRINT "VOUS NE POUVEZ PAS FAIRE ENTRER VOTRE AMI DANS LA SALLE DE CONTROLE"**

serait utilisé dans le second exemple.

Tout ce qui est entre guillemets est un "littéral", parce qu'il est imprimé exactement comme ce que vous avez tapé. Cela peut être: des mots, des lettres, des chiffres, des signes de ponctuations, etc...

Certaines touches, comme le curseur et les touches couleur, réagissent différemment quand elles sont utilisées dans des chaînes de caractères. Au lieu de changer la couleur ou de déplacer le curseur quand vous tapez la touche, un caractère est imprimé en vidéo inverse dans la chaîne. Quand le programme est lancé, le caractère est transformé en ce que vous avez tapé à l'origine. Cela vous permet de vider l'écran, de changer la couleur des caractères, de déplacer le curseur, tout cela dans votre programme. Par exemple, essayez ceci:

**10 PRINT " [REVERSE] [REVERSE] [REVERSE] 3 TESTING, [REVERSE]  
[REVERSE] 7 TESTING"**

Assurez vous d'appuyez simultanément sur les touches que vous utilisez avec SHIFT ou CONTROL. Les symboles en vidéo inversée sont des commandes pour l'ordinateur permettant de vider l'écran, de changer de couleur ou de déplacer le curseur.

**Utilisation n°2. Ecrire des nombres et des calculs.**

PRINT peut afficher les résultats de calculs faits dans une instruction PRINT (voir chapitre 5). Le Plus/4 calcule les opérations pour donner la réponse, puis l'affiche à l'écran. Par exemple:

**100 PRINT 58\*15,23,45+1000-45\*(4-3)**  
écrit  
**870 23 1000**

---

---

Ceci devient plus intéressant quand des variables sont aussi utilisées. Les données entrées par l'utilisateur peuvent être affichées et les calculs faits auparavant sont sauvegardés dans des variables et peuvent aussi être affichés ou même utilisés dans des calculs supplémentaires.

Exemples, tapez:

```
10 R=10*2:N=R-5
20 PRINT"R EST";R;"ET N EST";N
30 PRINT"MAIS 2 FOIS R =";R*2
40 PRINT"ET N MOINS 2 =";N-2
```

Normalement après chaque instruction PRINT, le curseur va automatiquement à la ligne. Vous pouvez empêcher ceci en mettant un point virgule (;) après l'instruction PRINT comme ceci :

```
200 PRINT"CES DEUX PARTIES DE PHRASES SERONT";
210 PRINT"IMPRIMEES SUR LA MEME LIGNE"
```

---

Instruction : **INPUT**

Syntaxe : **INPUT**"message facultatif";variable à entrer

L'instruction INPUT vous permet d'obtenir des données au clavier venant des utilisateurs du programme et de les utiliser dans le programme. Le message facultatif vous permet de dire à l'utilisateur exactement ce que vous lui demandez, le message est affiché quand l'instruction INPUT est exécutée, avec un point d'interrogation. A ce moment, le Plus/4 attend que l'utilisateur tape une réponse, et qu'il appuie sur la touche . La donnée entrée par l'utilisateur est mise dans une variable. Vous pouvez obtenir de l'utilisateur soit une chaîne de caractères en utilisant une variable chaîne de caractère (A\$, par exemple), soit un nombre en utilisant une variable numérique. L'instruction INPUT peut seulement être utilisée en mode programme.

Exemples :

Tapez:

```
10 PRINT"QUEL EST VOTRE NOM?";
20 INPUT A$
30 PRINT"ENCHANTE DE FAIRE VOTRE CONNAISSANCE";A$;" "
40 INPUT"QUEL AGE AVEZ-VOUS?".AG
50 PRINT AG;"EST PLUS AGE QUE MOI."
RUN
```

---

---

Instruction : **GETKEY**

**Syntaxe** : **GETKEY** variable à entrer

GETKEY est une autre manière d'entrer des variables pendant l'exécution du programme. L'instruction GETKEY n'accepte qu'une touche à la fois. Quelque soit la touche frappée, elle est affectée à la chaîne de caractères que vous avez spécifiée dans l'instruction GET (A\$, par exemple). GETKEY est utile parce qu'elle vous permet d'entrer des données d'un seul caractère à chaque fois sans avoir à appuyer sur [ ] après chaque caractère. L'instruction GETKEY peut seulement être utilisée dans un programme.

Exemple de GETKEY dans un programme :

```
1000 PRINT"CHOISISSEZ A, B, C, D, E OU F"  
1010 GETKEY A$
```

---

Instruction : **READ/DATA**

**Syntaxe** : **READ** variable à entrer  
**DATA** élément de donnée à lire

Les instructions READ/DATA sont utilisées pour affecter facilement des valeurs à des variables. Vous pouvez considérer l'instruction READ comme une instruction INPUT qui demande les données au Plus/4 et non pas à l'utilisateur. Quand le Plus/4 exécute une instruction READ, il recherche la prochaine instruction DATA et affecte ses données aux variables de l'instruction READ.

L'instruction est toujours utilisée avec l'instruction DATA. L'instruction DATA est juste une ligne de données (noms et nombres) dans un programme. L'instruction READ est utilisée pour affecter ces valeurs aux variables. (Pour chaque variable de l'instruction READ, votre Plus/4 "lit" une valeur dans la ligne DATA.) Une instruction DATA ne peut pas être exécutée et peut être placée partout dans le programme. Assurez-vous que le type de variable de l'instruction READ est le même que le type des données de l'instruction DATA (des variables nombre pour des nombres, des variables texte pour du texte). Sinon, le message d'erreur suivant apparaît: TYPE MISMATCH ERROR.

Exemple:

```
10 READ A$,B$,C$,D$,E$  
20 PRINT A$:PRINT B$:PRINT C$  
30 PRINT D$:PRINT E$
```

---

---

**40 DATA GROUCHO, HARPO, CHICO**  
**50 DATA ZEPPPO, GUMMO**

L'ordinateur affiche :

GROUCHO  
HARPO  
CHICO  
ZEPPPO  
GUMMO

---

---

## INSTRUCTIONS DE CONTROLE ET BOUCLES

Ce serait ennuyeux si votre ordinateur ne pouvait exécuter un programme que ligne après ligne. L'ordinateur ne pourrait que commencer à la première ligne, passer à la seconde et ainsi de suite jusqu'à la dernière ligne. Cela conduirait à de très longs programmes; si vous vouliez faire deux fois la même chose (écrire Bonjour!), vous devriez alors recopier la même ligne de programme. Ce court exemple (écrire Bonjour!) n'est pas très significatif, mais avec des instructions plus compliquées cela deviendrait fastidieux. C'est pourquoi, l'ordinateur a des instructions de contrôle. Ces instructions indiquent à l'ordinateur d'abandonner l'ordre normal des lignes du programme et d'aller directement à la séquence voulue. Le Plus/4 a plusieurs types d'instructions de contrôle: absolues (GOTO) qui transfèrent toujours le contrôle à une autre ligne; itératives (FOR/NEXT) qui transfèrent le contrôle un nombre donné de fois; et pour les amateurs de programmation structurée, les boucles (DO/LOOP).

---

Instruction : **GOTO**

Syntaxe : **GOTO** ligne n°

GOTO indique à l'ordinateur d'aller immédiatement à la ligne spécifiée dans l'instruction. Par exemple, si sur la ligne n°20 il y a GOTO 40, votre Plus/4 ira directement à la ligne 40, ignorant toutes les instructions se trouvant entre ces deux lignes.

Exemple d'utilisation :

Tapez :

```
10 PRINT"UN TIENS VAUT MIEUX QUE DEUX TU L'AURAS"  
20 GOTO 10
```

L'ordinateur imprimera le message de la ligne 10 indéfiniment jusqu'à ce que vous appuyiez sur la touche STOP, comme suit :

```
UN TIENS VAUT MIEUX QUE DEUX TU L'AURAS  
UN TIENS VAUT MIEUX QUE DEUX TU L'AURAS  
UN TIENS VAUT MIEUX QUE DEUX TU L'AURAS
```

```
BREAK IN 10  
READY.
```

(quand vous appuyez  
sur la touche )

L'impression continuera pour toujours. Chaque fois que votre Plus/4 arrivera en ligne 20, il reviendra en ligne 10. C'est ce qu'on appelle

---

---

une boucle infinie en langage informatique. Généralement vous voulez répéter une instruction un certain nombre de fois ou jusqu'à ce qu'un évènement se produise. C'est pourquoi les instructions FOR/NEXT et DO/LOOP sont disponibles en BASIC.

GOTO peut aussi être utilisé en mode direct. GOTO n° de ligne démarrera le programme à partir de la ligne spécifiée, en conservant les valeurs des variables (alors que l'instruction RUN met les variables à zéro).

---

Instruction : **FOR/NEXT**

Syntaxe : **FOR** variable = valeur de début **TO** valeur de fin  
quelques instructions BASIC  
**NEXT** variable

L'instruction FOR/NEXT crée une boucle qui sera répétée un certain nombre de fois. Les instructions entre le FOR et le NEXT correspondant, sont répétées dans la boucle. La variable de l'instruction FOR fonctionne comme un compteur. Elle est initialisée avec votre valeur de début. Ensuite, les lignes de programme suivant FOR sont exécutées jusqu'à ce que l'ordinateur rencontre l'instruction NEXT correspondante. L'instruction NEXT indique à l'ordinateur d'ajouter 1 au compteur. Si le compteur est inférieur ou égal à la valeur de fin, l'ordinateur revient à la ligne qui se trouve après l'instruction FOR. Sinon votre Plus/4 continue sur l'instruction qui suit NEXT.

Exemple d'utilisation d'une boucle FOR/NEXT:

```
10 PRINT"COMPTEUR"  
20 FOR J=1 TO 10  
30 PRINT"NOUS AVONS":J  
40 NEXT J  
50 PRINT"NOUS AVONS COMPTE JUSQU'A":J
```

Vous pouvez aussi définir une valeur d'itération dans l'instruction FOR. Au lieu d'ajouter 1 à la variable du compteur, votre Plus/4, ajoutera la valeur d'itération. Si vous utilisez un pas de 5 avec l'instruction FOR M=10 TO 30, votre compteur prendra les valeurs 10,15,20,25,30 après chaque boucle. La commande STEP vous permet aussi d'utiliser des pas négatifs (en utilisant une valeur d'itération négative).

---

Autre exemple avec un pas négatif:

```
10 PRINT"COMPTE A REBOURS"  
20 FOR J= 10 TO 0 STEP -1  
30 PRINT"NOUS SOMMES A":J  
40 NEXT J  
50 PRINT"NOUS SOMMES ARRIVES A":J
```

---

Instruction : **DO UNTIL/WHILE...LOOP UNTIL/WHILE**

Syntaxe : **DO UNTIL** (conditions) **WHILE** (conditions)

quelques instructions BASIC

**[EXIT]**

**LOOP UNTIL** (conditions) **WHILE** (conditions)

La combinaison d'instructions DO/LOOP est une autre façon de créer une boucle. Cette combinaison est très puissante et souple. La méthode DO/LOOP pour les boucles est une commande utilisée dans les langages à programmation structurée. Dans ce chapitre, nous n'allons en voir que quelques possibilités.

Si vous voulez créer une boucle infinie, commencez un programme avec DO et finissez le avec LOOP, comme suit:

```
100 DO:PRINT"ESSAI"  
110 LOOP
```

Appuyez sur la touche **STOP** pour arrêter le programme.

Il est plus utile de combiner DO/LOOP avec UNTIL. La boucle continuera jusqu'à ce que la condition UNTIL soit satisfaite.

```
100 DO:INPUT"AIMEZ-VOUS VOTRE ORDINATEUR";A$  
110 LOOP UNTIL A$="OUT"  
120 PRINT"MERCI"
```

Vous pouvez consulter l'encyclopédie BASIC à la fin du livre pour de plus amples informations.

---

---

## INSTRUCTIONS CONDITIONNELLES OU DECISIONNELLES

Les instructions conditionnelles sont utilisées pour faire des choix. Une des possibilités les plus puissantes d'un ordinateur est de faire des choix à partir d'évènements. Une des instructions conditionnelles offerte par votre Plus/4 est IF/THEN.

---

Instruction : **IF/THEN**

Syntaxes : **IF** (conditions) **THEN** instruction (simplement si la condition est vraie)

Concrètement, l'instruction IF/THEN fonctionne comme suit :

IF(si condition est vraie) THEN (alors exécution instructions)

En fait vous avez toujours su comment fonctionnait cette instruction, combien de fois n'avez vous pas entendu cette fameuse phrase :

Si tu manges ta soupe, ALORS tu auras du dessert. Cela peut vous paraître simpliste, mais c'est le fond de cette instruction.

Si la condition de l'instruction IF est vraie, tout ce qui suit THEN est exécuté.

Exemple :

```
10 INPUT"QUELLE EST LA DIXIEME LETTRE DE L'ALPHABET";A$
20 IF A$="J"THEN PRINT"OK":GOTO 100
30 INPUT"EST-CE UN A";X$
40 IF X$="A" THEN 60
50 PRINT"FAUX, ESSAYEZ ENCORE":GOTO 30
60 PRINT"TAPEZ UN B"
70 GETKEY A$: IF A$="B" THEN PRINT "OK"
100 PRINT"C'EST FINI POUR CETTE FOIS"
```

A la ligne 40, vous avez THEN 60. Cela signifie THEN GOTO 60, mais puisque la combinaison THEN GOTO est souvent utilisée, BASIC vous permet d'oublier le GOTO. ELSE est une option de l'instruction IF/THEN, celle-ci dirige votre ordinateur vers une action spécifique dans le cas où la condition du IF n'est pas satisfaite. Par exemple: IF B>5 THEN 40 ELSE GOTO 10. L'encyclopédie BASIC vous donne de plus amples informations sur IF/THEN/ELSE.

---

---

## SOUS PROGRAMMES

Si vous avez une suite d'instructions qui se répètent plus d'une fois dans votre programme, vous avez deux choix: vous pouvez réécrire cette routine, ou vous pouvez créer un sous programme. Un sous programme, est une série d'instructions qui peut être utilisée à tout moment de votre programme. Quand le sous programme est terminée, le programme revient automatiquement sur l'instruction suivant l'appel de la sous programme.

---

Instruction : **GOSUB/RETURN**

Syntaxe : **GOSUB** ligne n°

GOSUB est utilisé pour appeler un sous programme. Comme avec GOTO, le contrôle est transféré à la ligne spécifiée. Cependant, à la différence de GOTO, le Plus/4 garde en mémoire le numéro de ligne du GOSUB. Quand on arrive sur le RETURN, le contrôle revient sur l'instruction suivant l'appel GOSUB.

Exemple:

```
5 T=0: FOR J=1 TO 99
10 PRINT"DONNEZ-MOI UN NOMBRE ENTRE 1 ET 10"
20 INPUT N
30 IF N<1 THEN GOSUB 100: GOTO20
40 IF N>10 THEN GOSUB 100:GOTO 20
50 T=T+N
60 NEXT J
70 PRINT"LE TOTAL EST" J
80 END
100 PRINT"CE NOMBRE EST EN DEHORS DES LIMITES"
105 PRINT"TAPEZ UN NOMBRE ENTRE 1 ET 10"
110 RETURN
```

Si l'ordinateur arrive sur RETURN sans être passé par un GOSUB, vous aurez l'erreur suivante: RETURN WITHOUT GOSUB ERROR. Vous devez faire attention de ne jamais entrer dans un sous programme sans avoir utilisé d'instructions GOTO ou GOSUB. La meilleure méthode est de regrouper toutes les routines GOSUB et GOTO ensemble en fin de programme et de les séparer du programme normal par une instruction END.

---

Instruction : **REM**

Syntaxe : **REM** message

---

---

---

---

REM est utilisé pour insérer dans votre programme des commentaires ou des REMarques. Cette instruction n'est pas considérée comme une partie du programme;

C'est un message que vous ne pouvez voir que dans la liste de votre programme. Souvent, si vous ne faites pas de commentaires, vous n'arriverez pas à comprendre ce que fait le programme 6 mois après l'avoir écrit. Vous pouvez utiliser REM pour donner des indications qui vous permettront d'expliquer des opérations ou d'ajouter des informations à vos messages.

Exemple:

```
1560 E=R/I*9:REM SURFACE DU TERRAIN  
100 INPUT A,B:REM A EST LA LONGUEUR EN METRE ET B LA  
LARGEUR
```

---

---

## SOMMAIRE

Comme nous l'avons indiqué dans l'introduction, ceci ne peut pas être un manuel complet sur le BASIC. Nous n'avons parlé que de quelques instructions BASIC. Toutes les commandes BASIC de votre Plus/4 sont dans l'encyclopédie BASIC, avec syntaxes, descriptions et exemples. N'ayez pas peur de faire des essais. Si vous voulez apprendre plus sérieusement le BASIC, nous vous donnons dans la section 14 de l'encyclopédie une liste de livres sur la programmation BASIC. Quand on commence à programmer, il est difficile de s'arrêter !

---

---

# CHAPITRE 7

## COULEURS ET GRAPHIQUES

- 
- Caractères graphiques
- 
- Animation
- 
- Contrôle des couleurs
- 
- Graphiques haute résolution
- 
- Points, traits et titres
- 
- Carrés, cercles, polygones et peintures
- 
- Graphiques multi-couleur
-

---

---

## CARACTERES GRAPHIQUES

Chaque touche lettre contient deux caractères graphiques différents, ainsi que @, —, \* et £. Pour obtenir ces caractères graphiques, vous devez maintenir le doigt sur la touche  ou la touche  pendant que vous appuyez sur le caractère graphique désiré.

Si votre PPlus/4 est en mode majuscule/graphique, maintenez le doigt sur  et appuyez sur la touche lettre, vous obtiendrez le caractère graphique figurant sur la face avant droite de la touche. Ces caractères comprennent les couleurs des cartes à jouer, un cercle vide, un cercle plein et un ensemble de traits et de courbes, vous permettant de dessiner à l'écran.

Voici des exemples pour vous aider à utiliser ces caractères :

---

### Exercice 1 : grand cercle

- 1: Appuyez sur la touche .
- 2: Appuyez sur les touches U et I.
- 3: Appuyez sur .
- 4: Appuyez sur les touches J et K.
- 5: Appuyez sur .

---

### Exercice 2 : serpent

- 1: Appuyez sur la touche .
- 2: Appuyez sur les touches U, I, U, I, U, I.
- 3: Appuyez sur .
- 4: Appuyez sur les touches K, J, K, J, K, J.
- 5: Appuyez sur .

---

### Exercice 3 : ligne brisée

- 1: Appuyez sur la touche .
- 2: Appuyez sur les touches E, D, C, \*, F, R.
- 3: Appuyez sur .

---

### Exercice 4 : deux croix

- 1: Appuyez sur la touche .
  - 2: Appuyez sur les touches M, espace, N, espace, -.
  - 3: Appuyez sur .
  - 4: Appuyez sur les touches espace, V, espace, \*, +, \*.
  - 5: Appuyez sur .
  - 6: Appuyez sur les touches N, espace, M, espace, -.
  - 7: Appuyez sur .
- 
-

---

---

Quand vous avez fini, n'oubliez pas de relâcher la touche

■.

Ne vous êtes-vous pas demandé pourquoi le message d'erreur: SYNTAX ERROR, n'est pas apparu à l'écran? Pourtant il y a des caractères sur la ligne que l'ordinateur ne peut pas comprendre.

La raison en est que le Plus/4 ne fait pas attention à ce que vous avez tapé quand la touche ■ est enfoncée et que vous tapez sur ■. Si vous tapez sur ■ sans la touche ■, l'ordinateur essaiera de comprendre ce que vous avez voulu dessiner.

Jusqu'à maintenant, nous n'avons pas parlé des caractères graphiques situés sur la face avant gauche des touches. Ces caractères fonctionnent comme ceux situés sur la face avant droite, sauf que vous devez appuyer sur la touche  au lieu de la touche ■. Il n'y a pas de verrouillage de la touche  vous devez donc garder le doigt appuyé dessus.

Vous pouvez obtenir l'ensemble de ces caractères graphiques dans les deux modes, majuscule/graphique et texte minuscule/majuscule.

La face avant gauche des touches graphiques comprend des traits et des angles utilisés pour dessiner des diagrammes et des tableaux. Par exemple, voici comment souligner un mot:

Déplacez le curseur sur la ligne qui est sous le mot que vous voulez souligner. Maintenez le doigt sur la touche  et appuyez sur la touche T, ce qui souligne le mot.

---

### Exercice 5 : demi-barre

- Etape 1 : Maintenez la touche  enfoncée durant tout l'exercice.
- 2 : Appuyez sur les touches D, I, I, F.
- 3 : Appuyez sur ■.

---

### Exercice 6 : coin

- Etape 1 : Maintenez la touche  enfoncée et tapez sur T, Y, U.
- 2 : Appuyez sur les touches ■ et 9.
- 3 : Maintenez la touche  enfoncée et tapez sur O, P, @ et espace.
- 4 : Appuyez sur ■.
- 
-

---

---

### Exercice 7 : fenêtre

- Etape 1 : Maintenez la touche  enfoncée jusqu'à l'étape 4.  
2 : Appuyez sur les touches A, R, S et  .  
3 : Appuyez sur la touche Q.  
4 : Vous pouvez relâcher la touche  .  
5 : Maintenez enfoncée la touche  et tapez +.  
6 : Relâchez la touche  et réenfoncez la touche  que vous maintiendrez jusqu'à la fin.  
7 : Appuyez sur les touches W et  .  
8 : Appuyez sur les touches Z, E, X et  .
- 

Le propos de ces exercices est de vous montrer comment les symboles graphiques du Plus/4 peuvent être manipulés pour créer différentes formes ou dessins. Ce n'est qu'une partie de ce que vous pouvez faire. Maintenant que vous avez une bonne idée de ce qui se passe avec l'utilisation des symboles graphiques, vous pouvez expérimenter de nouveaux types graphiques et voir comment cela se passe.

---

---

## ANIMATION

Le cinéma n'est qu'une suite d'images fixes. Chaque image est un peu différente des précédentes. Le projecteur montre chaque image très peu de temps et passe à la suivante. La scène devient animée.

L'animation avec un ordinateur fonctionne de la même manière. Un film est animé à la vitesse de 30 images par seconde. Le changement doit être suffisamment rapide pour tromper l'œil. Dans un programme, vous devez donc dessiner une image, attendre une fraction de seconde et passer à une nouvelle image.

Pour créer un programme qui dessine, vous utilisez l'instruction PRINT avec des caractères graphiques. La plus simple animation consiste à afficher en alternance deux caractères, pour donner un effet de mouvement.

Le programme suivant simule le mouvement d'une balle.

Tapez NEW et [ ] avant d'entrer chaque nouveau programme. Assurez-vous de bien appuyer sur [ ] à la fin de chaque ligne.

```
10 PRINT" [ ] [ ] Q"  
20 FOR L=1 TO 100  
30 NEXT L  
40 PRINT" [ ] [ ] W"  
50 FOR L=1 TO 100  
60 NEXT L  
70 GOTO 10
```

(tapez les touches [ ] et lettres en même temps)

Tapez RUN et [ ] .

---

Pour obtenir un effet plus intéressant, vous pouvez construire un petit dessin à partir de plusieurs caractères graphiques. Ensuite vous ne changez qu'une partie de ces caractères, cela vous donnera un effet de mouvement pour une partie du dessin, comme le montre le programme suivant:

---

---

---

---

IMPORTANT: Chaque fois qu'on fait référence à [ ] et [C] vous devez appuyer en même temps sur ces touches et sur celles qui les suivent dans le programme.

```
10 PRINT " [ ] [ ] M [ ] W [ ] N"  
20 PRINT [C] + [ ]  
30 PRINT " [ ] N [ ] M"  
40 FOR L=1 TO 100: NEXT L  
50 PRINT " [ ] [ ] W [ ]  
60 PRINT " [C] T [C] + [C] T"  
70 PRINT [ ] [C] G [C] G"  
80 FOR L=1 TO 100:NEXT L  
90 GOTO 10
```

Tapez RUN et appuyez sur [ ] .

Dans les deux exemples que vous venons de voir, nous n'avons travaillé que sur une partie de l'écran. La prochaine étape est de déplacer la figure animée sur l'écran. La fonction TAB vous aide à déplacer les objets à partir du côté gauche. Le programme suivant représente un serpent rampant sur l'écran.

Assurez-vous d'appuyer en même temps sur la touche [ ] et le caractère qui suit.

```
5 FOR A=0 TO 30  
10 PRINT " [ ] [ ] "  
20 PRINT TAB(A) " [ ] U [ ] I [ ] U [ ] I"  
30 PRINT TAB(A) " [ ] K [ ] J [ ] K [ ] J"  
40 FOR L=1 TO 100: NEXT L  
50 PRINT " [ ] [ ] "  
60 PRINT TAB(A+1) " [ ] I [ ] U [ ] I [ ] U"  
70 PRINT TAB(A+1) " [ ] J [ ] K [ ] J [ ] K"  
80 FOR L=1 TO 100:NEXT L  
90 NEXT A
```

Quand on utilise des caractères comme le cercle ( [ ] Q), vous pouvez jouer à des jeux vidéo. Pour déplacer la balle, effacez la balle et remplacez-la dans une nouvelle position comme dans le programme suivant.

---

---

```
10 PRINT "  "
20 PRINT " Q ";
30 FOR L=1 TO 50: NEXT L
40 GOTO 20
```

(ligne 20, la touche curseur gauche fait apparaître un caractère graphique de contrôle avant le deuxième guillemet)

Tapez RUN et appuyez sur **STOP** . Appuyez sur STOP quand vous voulez arrêter le mouvement de la balle.

---

---

## CONTROLE DES COULEURS

Chaque partie de l'écran peut être d'une couleur différente. La bordure peut être d'une couleur, le fond d'une autre et les caractères encore d'une autre. Vous savez déjà comment changer la couleur des caractères en utilisant le clavier. La commande COLOR permet de changer la couleur des autres parties de l'écran.

Rendez rouge le bord de l'écran en tapant la commande COLOR4,3 et en appuyant sur **RETOUR**. Le chiffre 4 correspond au bord de l'écran et 3 est le numéro de la couleur rouge (le même chiffre que sur la touche rouge).

Maintenant tapez COLOR 0,7 et appuyez sur **RETOUR**. Le fond de l'écran devient bleu. Le chiffre 0 correspond au fond et le 7 à la couleur bleu (la même chiffre que sur la touche).

Le premier chiffre après la commande COLOR correspond à la partie de l'écran que vous voulez changer. Le 0 pour le fond, le 1 pour les caractères et le 4 pour la bordure. Vous connaîtrez les autres parties de l'écran (2 et 3) dans le paragraphe des graphiques colorés à la fin du chapitre.

### Code des zones de l'écran

---

Zone N°	NOM
0	Fond
1	Caractères
2	Multi-couleur 1
3	Multi-couleur 2
4	Bordure

---

Chaque couleur a aussi un niveau de brillance modifiable. Vous pouvez ajouter un chiffre de 0 (plus sombre) à 7 (plus brillant) après le numéro de couleur pour faire varier la couleur. Taper COLOR 4,3,0 et appuyez sur **RETOUR**. Le bord devient rouge-foncé. Tapez COLOR 4,3,7 et le bord devient rouge-brillant.

---

---

## Code des couleurs

N°	Couleur	N°	Couleur
1	Noir	9	Orange
2	Blanc	10	Brun
3	Rouge	11	Jaune vert
4	Turquoise	12	Rose
5	Pourpre	13	Bleu vert
6	Vert	14	Bleu clair
7	Bleu	15	Bleu foncé
8	Jaune	16	Vert clair

En résumé la commande couleur a la syntaxe suivante :

**COLOR** zone de l'écran, couleur, brillance.

Voici un programme rapide vous montrant les couleurs du Plus/4 :

Tapez NEW et appuyez sur la touche **CONTROL** **AVIRON** . N'oubliez pas d'appuyer sur **CONTROL** **AVIRON** à chaque ligne.

```
10 COLOR 0,7,7
20 FOR M=0 TO 7
30 FOR N=1 TO 2
40 FOR L=1 TO 16
50 PRINT " CONTROL AVIRON ";
60 READ A
70 COLOR 1,A,M
80 PRINT " ESPACE ESPACE ";
90 NEXT L
100 PRINT
110 RESTORE
120 NEXT N,M
130 COLOR 1,2,4
200 DATA 7,14,4,13,6,16,11,8,10,9,3,12,5,15,2,1
```

Maintenant tapez RUN et appuyez sur **CONTROL** **AVIRON** . Vous voyez un écran bleu-brillant avec les 15 couleurs et chacune dans leurs 8 niveaux de brillance.

NOTE: Comme tous les termes BASIC vus dans ce chapitre, COLOR est aussi bien une commande qu'une instruction. La distinction n'est pas importante pour COLOR ou les autres commandes graphiques, puisqu'elle est basée sur son utilisation en mode direct ou mode programme.

---

---

## GRAPHIQUE HAUTE RESOLUTION

L'écran de votre Plus/4 contient 25 lignes de 40 caractères chacune ou au total 1000 positions de caractères sur l'écran. Chaque caractère est formé de points, avec 8 lignes de 8 points chacune. Votre écran a au total 320 points sur chaque ligne et 200 lignes de points ou bien 64 000 points au total. Votre Plus/4 contrôle chaque point.

En utilisant les caractères graphiques standards, vous avez un contrôle limité sur chaque point. Vous pouvez utiliser les 256 éléments de la police de caractères du Plus/4, ce qui vous permet de créer beaucoup de dessins. Mais imaginez ce que vous pourriez créer en contrôlant chaque point!

Les possibilités graphiques haute résolution du Plus/4 vous permettent de le faire. Vous pouvez utiliser des commandes qui vous laissent dessiner ou effacer des points, des traits, des cercles ou d'autres formes.

Il y a une limite aux possibilités graphiques du Plus/4. Celui-ci ne peut utiliser que deux couleurs dans les 8x8 positions du point dans le caractère. C'est à dire, chaque espace de 8x8 points sur l'écran n'a que deux couleurs (caractère et fond). Vous pouvez utiliser des couleurs différentes pour chaque caractère, mais seulement deux couleurs dans chaque position. Un autre mode graphique que nous expliquerons plus tard, vous permet d'utiliser quatre couleurs différentes par position de caractère au détriment de la haute résolution.

Le programme suivant utilise des possibilités graphiques haute résolution du Plus/4, en particulier la commande GRAPHIC. Commencez par taper NEW et appuyez sur . Après avoir tapé chaque ligne, appuyez sur . Une fois le programme tapé, écrivez RUN et appuyez sur  comme d'habitude.

```
10 COLOR 0,1
20 GRAPHIC 1,1
30 FOR L=2 TO 16
40 COLOR 1,L,2
50 DRAW 1,0,L*12 TO 319,L*12
60 DRAW 1,L*18,0 TO L*18,199
70 NEXT L
80 FOR L=1 TO 5000:NEXT L
90 COLOR 1,2,3
100 GRAPHIC 0
```

Remarquez que les couleurs changent près des intersections.

---

---

---

---

Pour passer du mode graphique normal (mode texte) au mode haute résolution, tapez la commande GRAPHIC2,1 et appuyez sur [REDACTED]. Qu'est-ce qui arrive? L'écran devient blanc et le curseur réapparaît au bas de l'écran. Le Plus/4 divise l'écran en deux parties: le haut pour le graphique et les 5 lignes du bas pour le texte. Si vous ne voulez pas des 5 lignes du bas, vous pouvez utiliser la commande GRAPHIC1,1, mais vous ne pourrez pas voir les commandes que vous taperez.

Vous pouvez revenir en mode texte et vice versa, avec la commande GRAPHIC. La commande GRAPHIC 0 vous fait passer en mode texte, alors que GRAPHIC 2 vous fait passer en mode haute résolution sans effacer l'écran. Si vous ajoutez,1 à la commande, l'écran se vide.

Généralement les commandes GRAPHIC sont de la forme suivante :

**GRAPHIC** mode, effacement (la dernière partie est facultative)

---

CODE MODE	EFFET
0	Texte
1	Haute résolution
2	Haute résolution + texte
3	Multi-couleur
4	Multi-couleur + texte

---

---

CODE EFFACEMENT	EFFET
0	Ecran non effacé
1	Ecran effacé

---

Il y a un autre moyen d'effacer l'écran haute résolution. La commande SCNCLR vide l'écran sans changer le mode graphique.

Dès que vous utilisez le mode haute résolution, l'ordinateur réserve 10 K de mémoire pour l'écran. Cette mémoire est prise sur l'espace mémoire réservé au programme BASIC. Quand vous quittez le mode graphique, vous pouvez récupérer cet espace mémoire avec la commande GRAPHIC CLR.

---

---

## POINTS, TRAIT ET TITRES

Tapez les commandes GRAPHIC2,1:DRAW 1,0,0 et appuyez sur **RETURN**. Regardez attentivement le coin supérieur gauche de l'écran. Le Plus/4 y a dessiné un point noir.

Dans la commande DRAW, le premier chiffre est toujours 1 (couleur du caractère) ou 0 (couleur du fond). Les autres chiffres représentent la position du point en ligne et colonne. Si vous voulez mettre un point, colonne 17, ligne 20 tapez DRAW 1,17,20. Pour effacer ce point, tapez DRAW 0,17,20.

La commande DRAW permet aussi de tracer une trait entre deux points. Il suffit d'ajouter TO et les coordonnées de cet autre point, comme suit: DRAW 1,1,1 TO 100,100. Ceci trace un trait du point 1,1 au point 100,100.

Si vous êtes habitué à tracer des graphes en mathématiques, vous serez surpris à la première utilisation de votre ordinateur. Le systèmes de coordonnées du Plus/4 est différent de ce que vous avez l'habitude d'utiliser. En maths, le point 0,0 se trouverait au centre ou au coin inférieur gauche de l'écran, mais pour l'ordinateur c'est le coin supérieur gauche. Vous vous habituerez vite à ce système.

Dès que vous avez mis un point à l'écran, vous pouvez tracer un trait de celui-ci à n'importe quel point de l'écran, par exemple: DRAW 1 TO 150,50. Cela tracera un trait du dernier point créé au point colonne 150 ligne 50. Si votre programme utilise souvent la commande DRAW TO, vous pouvez placer le premier point à l'écran en utilisant la commande LOCATE, comme suit : LOCATE 100,100.

La commande DRAW peut avoir les formes suivantes:

COMMANDE	RESULTAT
<b>DRAW</b> source couleur, colonne, ligne	POINT
<b>DRAW</b> source couleur, col.,ligne TO col., ligne	TRAIT
<b>DRAW</b> source couleur TO colonne, ligne	TRAIT DEPUIS LE DERNIER POINT

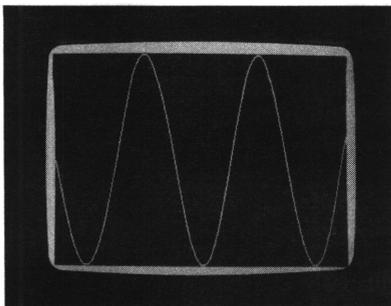
La source couleur est 0 pour la couleur des caractères et 1 pour la couleur du fond.

Pour effacer des points ou des traits à l'écran, utilisez la commande DRAW suivie du chiffre 0. Si vous créez un point avec DRAW 1,1,1 vous pouvez l'effacer avec DRAW 0,1,1. Un trait créé avec DRAW 1,1,1 TO 100,100 est effacé avec DRAW 0,1,1 TO 100,100.

---

Le programme suivant dessine une courbe avec la fonction sinus. Tapez NEW et appuyez sur **RETURN**. Assurez-vous d'appuyer sur **RETURN** à la fin de chaque ligne et de taper RUN à la fin du programme.

```
10 COLOR 0,1
20 COLOR 1,2
30 GRAPHIC 1,1
40 LOCATE 0,100
50 FOR X=1 TO 319
60 Y = INT(100 + 99 * SIN(X/20))
70 DRAW 1 TO X,Y
80 NEXT X
90 FOR L=1 TO 5000
100 NEXT L
110 GRAPHIC 0
```



Ne tapez pas NEW après avoir exécuté le programme. Pour avoir un graphique différent, changez la ligne 70 en:

```
70 DRAW 1,X,Y
```

Ce programme trace la même courbe en utilisant des points à la place des traits.

## La commande CHAR

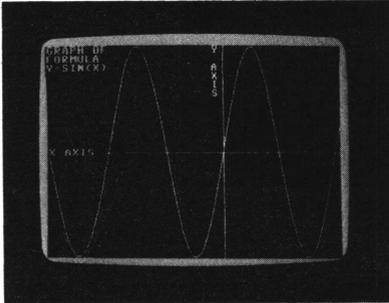
Les graphiques sont plus utiles et compréhensibles si vous leur donnez des titres. Vous pouvez utiliser la commande CHAR pour insérer du texte dans du graphique haute résolution. Par exemple, l'instruction CHAR 1,0,5,"SALUT" met le mot SALUT ligne 6 sur la gauche de l'écran. Le premier chiffre après la commande CHAR est soit 1 (pour écrire) ou 0 (pour effacer). les deux autre chiffres sont les numéros de ligne et de colonne où le texte doit se trouver.

---

---

Laissez ces deux programmes en mémoire: ne tapez pas NEW.  
Ajoutez les lignes suivantes:

```
81 CHAR 1,0,0,"GRAPH DE":CHAR 1,0,1,"FORMULE"  
82 CHAR 1,0,2,"Y=SIN(X)"  
83 DRAW 1,0,100 TO 319,100:DRAW 1,189,0 TO 189,199  
84 CHAR 1,0,12,"X-AXIS":CHAR 1,22,0,"Y"  
85 CHAR 1,22,2,"A":CHAR 1,22,3,"X"  
86 CHAR 1,22,4,"T":CHAR 1,22,5,"S"
```



---

---

## CARRÉS, CERCLES, POLYGONES ET PEINTURES

En utilisant la commande DRAW, vous pouvez dessiner avec des points et des traits. Pour dessiner un carré, vous pouvez utiliser la commande DRAW 1,0,0 TO 100,0 TO 100,100 TO 0,100 TO 0,0 (traçant le carré en joignant les quatres points principaux) ou vous pouvez utiliser la commande BOX.

### LA COMMANDE BOX

## TRACER DES RECTANGLES

Votre Plus/4 a une commande pour faciliter le dessin de carrés et d'autres formes rectangulaires. La commande BOX vous laisse choisir les points de deux coins opposés du carré. Pour réaliser le même carré que précédemment, utilisez BOX 1,0,0,100,100. Le chiffre 1 signifie à nouveau que vous voulez dessiner et non effacer. Les quatre chiffres suivants sont les coordonnées des coins opposés du carré, (0,0) coin supérieur gauche (100,100) près du milieu de l'écran.

La commande BOX peut tracer un rectangle en changeant les coins. Vous pouvez même faire tourner le rectangle en spécifiant un angle (en degré) après la dernière coordonnée, comme suit: BOX 1,50,50,100,100,45. Le rectangle a tourné de 45° dans le sens des aiguilles d'une montre.

Si vous voulez dessiner un rectangle plein au lieu des traits seuls, vous ajoutez, 1 après l'angle. Pour obtenir un rectangle plein au milieu de l'écran, tapez: BOX 1,100,50,220,150,,1. Notez que vous avez besoin d'une virgule supplémentaire pour l'angle, même si vous ne voulez pas faire tourner le rectangle.

La commande BOX a les formes suivantes:

COMMANDE	EFFET
BOX 1,col1,ligne1,col2,ligne2	traits
BOX 1,col1,ligne1,col2,ligne2,angle	rotation
BOX 1,col1,ligne1,col2,ligne2,,plein	plein
BOX 0,col1,ligne1,col2,ligne2,angle,plein	effacer

Voici deux programmes utilisant la commande BOX.

N'oubliez pas de taper NEW et d'appuyer sur **ENTER** avant d'entrer chaque programme et de taper **STOP** à la fin de chaque ligne.

```
10 COLOR 0,1
20 COLOR 1,2
30 GRAPHIC 2,1
40 A=RND(1)*20+10
```

---

```
50 FOR L=0 TO 359 STEP A
60 BOX 1,100,30,220,130,L
70 NEXT L
80 FOR L=1 TO 2000: NEXT L
90 GRAPHIC 0,1
```

```
5 TRAP 60
10 GRAPHIC 2,1
20 DEF FNA(X)=INT(RND(1)*X)
30 COLOR 1,FNA(15)+1
40 BOX 1,FNA(320), FNA(160),FNA(320),FNA(160),,1
50 GOTO 30
60 COLOR 1,2,3: GRAPHIC 0
```

Appuyez sur  et tapez RUN après avoir terminé. Maintenez la touche  pour terminer l'exécution du programme.

Le second programme dessine des carrés de couleurs différentes partout dans l'écran. Vous remarquerez que certaines parties de l'écran bougent et d'autres non. Nous vous en avons expliqué la raison au début de ce chapitre.

## TRACER DES CERCLES

Votre Plus/4 a aussi des commandes pour tracer des cercles. Comme avec la commande BOX, on peut faire varier la forme du cercle pour en faire un ovale (ou ellipse) et nous pouvons faire tourner l'ovale. Nous pouvons aussi ne dessiner qu'une partie de ce cercle (un arc).

Cette commande trace un cercle au centre de l'écran: CIRCLE 1, 160,100,50. Elle demande au Plus/4 de tracer un cercle dont le centre est sur la ligne 160 et sur la colonne 100, avec un rayon de 50. Cela donne en fait un ovale, puisque les points à l'écran sont plus hauts que larges. Pour le transformer en un vrai cercle, vous devez ajouter un nombre séparé pour indiquer que la hauteur est différente de la largeur, comme suit:  
CIRCLE 1,160,100,50,42.

Le Plus/4 peut aussi dessiner un carré, un triangle ou tout autre polygone en utilisant la commande CIRCLE. Il faut indiquer à l'ordinateur le nombre de degrés entre chaque point sur le cercle, comme ceci: CIRCLE 1,160,100,50,42,,,120. Cette commande dessine un triangle, puisque chaque côté est à 120° de l'autre. (Omettre les valeurs entre virgules dans une commande graphique indiquera à l'ordinateur de prendre les valeurs par défaut à la place des nombres manquants.) Une formule simple pour trouver la valeur de l'angle d'un polygone à n côtés est  $360/n$ .

---

---

Voici un programme rapide pour tracer des polygones:

```
10 GRAPHIC 2,1
20 INPUT"COMBIEN DE COTES";A
30 IF A<2 OU A>100 THEN PRINT "SOYEZ RAISONNABLE":
   GOTO 20
40 CIRCLE 1,160,80,40,33,,,,,360/A
50 GOTO 20
```

Vous pouvez choisir de ne dessiner qu'un arc de cercle au lieu du cercle entier. La commande CIRCLE accepte les angles de début et de fin en degré. La commande CIRCLE 1,160,100,50,42,90,180 n'affichera que l'arc inférieur droit du cercle.

Pour faire tourner un ovale, ajouter l'angle de rotation dans le sens des aiguilles d'une montre à la fin de la commande, comme par exemple CIRCLE 1,160,100,100,20,,,30.

La commande CIRCLE peut prendre les formes suivantes:

COMMANDE	EFFET
<b>CIRCLE</b> 1,col centre,ligne centre,rayon	ovale
<b>CIRCLE</b> 1,col c.,ligne c.,largeur,hauteur	cercle/ovale
<b>CIRCLE</b> 1,col c.,ligne c.,lar.,haut.,début,fin	arc
<b>CIRCLE</b> 1,col c.,ligne c.,lar.,haut.,,,	angle
<b>CIRCLE</b> 1,col c.,ligne c.,lar.,haut.,,,,angle pt	polygone

NOTE: Quand il y a des virgules sans nombres dans une commande comme CIRCLE ou BOX, le Plus/4 considère ces virgules comme l'entrée de valeurs par défaut pour les paramètres manquants de la commande. CIRCLE,160,40,100,100 est lu par l'ordinateur comme CIRCLE 1,160..., lisant la valeur par défaut 1 pour la source de la couleur.

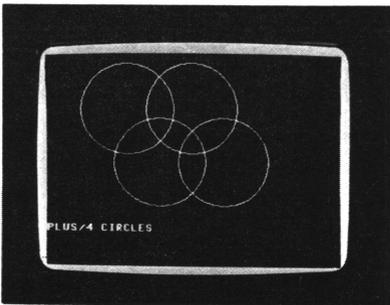
Le prochain programme utilise la commande CIRCLE pour créer un effet intéressant. Tapez NEW puis appuyez sur  pour effacer le dernier programme de la mémoire avant de taper ce nouveau programme.

---

```
10 COLOR 0,1
20 COLOR 1,2
30 GRAPHIC 1,1
40 A=RND(1)*20+10
50 FOR L=0 TO 359 STEP A
60 CIRCLE 1,160,100,80,40,,,L
70 NEXT L
80 FOR L=1 TO 2000:NEXT L
80 GRAPHIC 0,1
```

Voici un programme qui utilise la commande CIRCLE pour créer un dessin simple.

```
NEW
10 COLOR 0,1
20 COLOR 1,2
30 GRAPHIC 2,1
40 FOR L=1 TO 4
50 Y=50
60 IF L=2 OR L=4 THEN Y=100
70 X=L*35 + 50
80 CIRCLE 1,X,Y,50
90 NEXT L
100 PRINT "PLUS/4 CIRCLES"
```



---

---

## LA COMMANDE PAINT

La commande PAINT colore n'importe quelle surface dont les limites sont des traits à l'écran. S'il n'y a aucun trait de tracé, l'écran se remplit en entier. La commande BOX peut être utilisée pour remplir des carrés et des rectangles en couleur. La commande PAINT peut colorer des formes irrégulières et autres surface de l'écran, qui ne peuvent pas être colorées avec une autre commande.

Dans le dernier exercice, nous avons créé un dessin avec 4 anneaux imbriqués. En ajoutant quelques instructions PAINT, nous pouvons colorer les surfaces définies par les anneaux.

Ajoutez ces lignes au programme précédent:

```
110 FOR L=0 TO 1
120 PAINT 1,120+35*L,75
130 NEXT L
```

---

## LES GRAPHIQUES MULTI-COULEURS

La haute résolution graphique du Plus/4 vous permet de définir un point de l'écran, mais nous avons vu que la possibilité d'y associer une couleur est limitée. La plupart des programmes haute résolution ne peuvent utiliser qu'une ou deux couleurs.

Afin de pouvoir avoir plus de couleurs différentes, votre Plus/4 a un mode graphique intermédiaire appelé mode graphique multi-couleurs. Dans ce mode, vous pouvez contrôler deux fois moins de points sur chaque ligne qu'en mode haute résolution, car chaque point est deux fois plus grand. Vous avez 160 points sur chaque ligne et vous avez toujours 200 lignes. Vous échangez la possibilité des multi-couleurs contre une moins bonne résolution de l'écran.

Avant de commencer l'utilisation du mode graphique multi-couleur, revoyez la commande GRAPHIC au début du chapitre. Vous verrez que l'écran multi-couleur sans texte est GRAPHIC 3 et que l'écran multi-couleur avec 5 lignes de texte est GRAPHIC 4.

Maintenant regardez le tableau des commandes COLOR. Il y a deux zones de l'écran que nous n'avons pas encore utilisées, 2 et 3. Ces zones ont deux couleurs supplémentaires. Vous pouvez utiliser une des trois couleurs (1 la couleur des caractères, 2 une couleur supplémentaire et 3 une autre couleur supplémentaire). Ces couleurs ne se mélangent pas comme c'était le cas pour les couleurs haute-résolution dans les programmes précédents de ce chapitre.

Ces deux programmes utilisent des graphiques multi-couleurs, le premier avec des anneaux et le second montrant des effets "néons".

```
10 COLOR 0,1
20 GRAPHIC 4,1
30 FOR L=1 TO 4
40 Q=L:IF Q>3 THEN Q=Q-3
50 COLOR Q,L+1
60 Y=50
70 IF L=2 OR L=4 THEN Y=100
80 X=L*18+25
90 CIRCLE Q,X,Y,25,42
100 NEXT L
```

---

---

Tapez NEW puis appuyez sur [REDACTED]. N'oubliez pas d'appuyer sur [REDACTED] à la fin de chaque ligne. Tapez RUN et appuyez sur [REDACTED] à la fin du programme.

```
10 COLOR 0,1
20 GRAPHIC 3,1
30 COLOR 3,1
40 TRAP 200
50 DRAW 3,10,10 TO 10,100: DRAW 3,10,55 TO 30,55
60 DRAW 3,30,10 TO 30,100: DRAW 3,50,10 TO 80,10
70 DRAW 3,65,10 TO 65,100: DRAW 3,50,100 TO 80,100
80 FOR L=0 TO 7
90 COLOR 3,2,L
100 FOR M=1 TO 100: NEXT M
110 NEXT L
120 COLOR 3,1
130 FOR M=1 TO 100: NEXT M
140 GOTO 80
200 GRAPHIC 0:COLOR 1,2,7
```

La zone couleur 3, la seconde zone multi-couleur, a une possibilité spéciale que n'ont pas les autres. Dès que vous avez dessiné sur l'écran en utilisant la zone 3, vous pouvez changer la couleur de cette zone partout où elle apparaît à l'écran en utilisant la commande COLOR. Si vous utilisez COLOR 3,5 et que vous dessinez, vos graphiques seront en pourpre. Si vous changez avec COLOR 3,6 toutes les zone pourpres passeront en vert. Cela ne marche pas avec les autres zones.

Le guide de référence du programmeur du Plus/4 contient plus d'informations sur le mode graphique.

---

---

# CHAPITRE 8

## MUSIQUE ET SONS

- 
- Introduction
- 
- Commande VOLUME
- 
- Commande SOUND
- 
- Effets sonores
- 
- Faire de la musique
- 
- Le Plus/4 musical
-

---

---

## INTRODUCTION

Voici un petit programme pour faire de la musique avec votre Plus/4. Tapez le programme exactement comme ci-dessous et n'oubliez pas d'appuyer sur **STOP** à la fin de chaque ligne. A la fin du programme tapez RUN et appuyez sur **STOP**. Quand le programme vous demande d'entrer un nombre, tapez un nombre entre 0 et 1023 et appuyez sur **STOP**. Pour arrêter le programme, entrez la valeur 0.

```
10 VOL 8
20 DO
30 INPUT X
35 IF X>1023 OR X<0 THEN PRINT"ENTRE 0 ET 1023 S.V.P.":GOTO30
40 SOUND 1,X,10
50 LOOP UNTIL X=0
```

Appuyez sur la touche **STOP** pour arrêter le programme.

Voici comment jouer une simple note sur votre Plus/4. Tapez NEW et appuyez sur **STOP** pour mettre à zéro la mémoire du Plus/4.

D'abord : tapez NEW et appuyez sur **STOP**  
tapez VOL 8 et appuyez sur **STOP**

Puis : tapez SOUND 1,266,60 et appuyez sur **STOP**

Vous devriez entendre une note jouée pendant à peu près une seconde. Si vous n'entendez rien, augmentez le volume de votre TV ou de votre moniteur et essayez une nouvelle fois.

Ceux sont les deux seules commandes que vous avez besoin de connaître pour jouer de la musique sur le Plus/4. Voyons comment ces commandes fonctionnent.

## LA COMMANDE VOLUME

La commande VOL contrôle le VOLUME des notes que joue votre Plus/4. Pensez aux trois premières lettres du mot "volume" pour vous souvenir de la commande VOL. Le nombre qui vient après VOL règle le volume. Pensez à la commande VOL comme à un bouton de volume sur votre Plus/4. Quand le bouton est à zéro, le volume est éteint et vous n'entendez rien. Quand le bouton est à 7, le volume est au maximum et votre Plus/4 jouera aussi fort qu'il le peut.

Essayez de nouveau le premier exemple et mettez un nombre différent après la commande VOL. Plus le nombre est petit, plus la note est faible.

---

---

## LA COMMANDE SOUND

La commande SOUND donne au Plus/4 tout ce dont il a besoin pour générer la note que vous voulez jouer. La commande SOUND est suivie de 3 nombres définissant la note.

**SOUND** voix, valeur de la note, durée.

Le premier nombre de la commande SOUND fait référence à la voix. Le nombre pour la voix peut être 1, 2 et 3. Le son du Plus/4 est produit par deux "voix" différentes, 1 pour la première voix et 2 pour la seconde. L'option 3 fait référence à la possibilité de la voix 2 de produire un son ou un bruit.

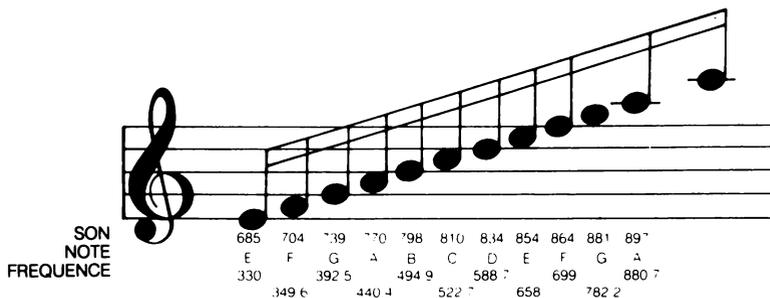
Voix 1. Cette voix ne joue que des sons. Sélectionnez cette voix avec 1 après la commande SOUND.

Voix 2. Cette voix est comme la voix 1, mais elle peut être utilisée pour produire des sons ou des bruits. Tapez un 2 après la commande SOUND pour utiliser cette voix pour les sons, ou un 3 pour utiliser cette voix pour des bruits, pour produire des effets sonores tels que l'orage et la pluie.

Le second nombre après la commande SOUND est la valeur de la note (fréquence). Cela peut être un nombre entre 0 et 1023. Ce nombre indique à votre Plus/4 la hauteur de la note à jouer. Plus le nombre est grand, plus la note est haute. Les valeurs les plus hautes (aux environs de 1023) ne sont pas audibles à l'oreille humaine.

NOTE: En voix 3 le bruit est "nul" entre 600 et 940. Vous pouvez utiliser des valeurs du registre en dehors de cet intervalle pour créer des bruitages.

Voici une portée qui montre toutes les notes d'une gamme, avec les valeurs des notes. Il y a un tableau complet des notes du Plus/4 dans la section 11 du manuel du Plus/4.



---

---

Essayez le programme suivant sur votre Plus/4 :

NEW

10 VOL 7 (réglage du volume)

20 X=0

30 DO

40 SOUND 1,X,5 (joue la note)

50 X=X+5

60 LOOP UNTIL X=1020

70 VOL 0 (éteint le volume)

80 END

**Tapez RUN et appuyez sur [ ]** . Ce programme vous donne une idée de la gamme musicale du Plus/4.

Le troisième nombre après la commande SOUND contrôle la durée d'une note. Cela indique au Plus/4 combien de temps la note doit être jouée. Ce nombre doit être compris entre 0 et 65535. Il met en route un compteur, qui s'incrémente chaque soixantième de seconde. Une durée de 60 tient la note une seconde. Plus le nombre est grand, plus la note est longue. En fait, si vous utilisez 65535, la note dure plus de 16 minutes. Pour arrêter le son, utilisez une durée 0, ce qui ne produit aucun son.

---

## EFFET SONORE MUSICAL

Des effets sonores peuvent être créés, en utilisant soit des sons musicaux ou des bruits. Dans un programme BASIC simple, on peut utiliser une combinaison de commandes SOUND pour produire des bruitages inhabituels et intéressants. Par exemple, la boucle FOR/NEXT/STEP peut être utilisée pour créer des effets sonores. Ce programme utilise une boucle FOR/NEXT avec un pas négatif, pour descendre d'un grand nombre vers un petit.

```
10 VOL 8 (met le VOLume à 8)
20 FOR S= 1000 TO 700 STEP -25 (boucle avec pas négatif)
30 SOUND 1,S,1
40 NEXT S
```

Tapez RUN et appuyez sur la touche  pour entendre l'effet sonore. La ligne 20 permet d'obtenir des valeurs décroissantes entre 1000 et 700 en ne sélectionnant qu'un nombre tous les 25. La ligne 30 indique au Plus/4 de ne jouer la note qu'un instant en mettant la durée à 1, ce qui représente 1/60ème de seconde. Vous pouvez renouveler l'expérience en modifiant les valeurs des variables.

---

---

## BRUITAGE

Si vous utilisez la valeur 3 de la voix de la commande SOUND, vous produirez des bruits. Vous l'utilisez pour créer des bruitages plutôt que des sons. Le programme suivant utilise la voix numéro 3 pour imiter le bruit d'un orage.

```
10 VOL 2 (met le volume)
20 R=INT(RND(0)*10)+1 (génère un nombre entre 1 et 10)
30 FOR X=1 TO R
40 SOUND 3,600+30*X,10
50 NEXT X
60 FOR X=R TO 1 STEP-1
70 SOUND 3,600+30*X,10
80 NEXT X
90 T= INT(RND(0)*100)+30
100 SOUND 3,600,T
110 GOTO20
```

Les lignes 30 et 60 mettent en place des boucles pour les valeurs (fréquences) des notes, l'une est croissante et l'autre décroissante. Ces boucles sont basées sur le nombre aléatoire de la ligne 20. Il est important d'avoir des variations dans la hauteur du son, comme dans les orages avec des vents de forces différentes. En ligne 40 et 70, les commandes SOUND créent les bruitages. En ligne 90 et 100, un délai aléatoire est créé pour donner l'illusion d'un orage avec des écarts entre les roulements de tonnerre. Le programme choisit un nombre aléatoire qui est utilisé pour une nouvelle commande SOUND. Cette nouvelle commande reste à la même hauteur et donne un fond sonore constant qui sert de contrepoint aux rafales de vent.

La création de bruitages est très motivante, en essayant de capter les vrais éléments du son que vous voulez créer. Pour obtenir de bons bruitages, il faut faire de multiples expériences.

---

---

## FAIRE DE LA MUSIQUE

Maintenant que vous avez eu un aperçu de la manière dont on peut créer des effets sonores sur votre Plus/4, faisons de la musique. Voici quelques programmes.

Le premier programme utilise les touches 1 à 8 comme celles d'un piano. Tapez le programme et RUN à la fin.

```
5 SCNCLR                (vide l'écran)
10 FOR X=1 TO 8: READ N(X):NEXT X
20 VOL 7
30 DO
40 GETA$:IF A$="" THEN 40
50 A=ASC(A$): IF A<49 OR A>56 THEN 90
60 N=A-48
70 SOUND 1,N(N),5
80 COLOR 0,N,3
90 LOOP UNTIL A=32
100 VOL 0: COLOR4,2,7
110 DATA 169,262,345,383,453,516,571,596
```

Appuyez sur les touches de 1 à 8 pour jouer des notes. Le bord de l'écran change même de couleur pour chaque note. Quand vous avez fini de jouer, appuyer sur la barre d'espacement pour arrêter le programme.

Voici les touches à appuyer pour avoir les notes d'une chansonnette.

### "TWINKLE, TWINKLE LITTLE STAR"

```
1  1  5  5  6  6  5
4  4  3  3  2  2  1
5  5  4  4  3  3  2
5  5  4  4  3  3  2
1  1  5  5  6  6  5
4  4  3  3  2  2  1
```



---

Ce programme joue une chanson en lisant une suite d'instructions DATA. Ces données vont par paires. Le premier nombre est la valeur de la note dans la commande SOUND et le deuxième est sa durée.

### **“ROW BOAT”**

```
10 VOL 8
20 DO
30 READ X,Y
40 SOUND 1,X,Y
45 FOR D=1 TO 550:NEXT (boucle de temporisation)
50 LOOP UNTIL X=0
60 END
100 DATA 169,45,169,45,169,30
110 DATA 262,15,345,45,345,30
120 DATA 262,15,345,30,383,15
130 DATA 453,60,596,45,453,45
140 DATA 345,45,169,45,453,30
150 DATA 383,15,345,30,262,15
160 DATA 169,60
200 DATA 0,0
```

Ce programme monte et descend la gamme à différentes vitesses et affiche des barres de couleur.

```
10 VOL 8
20 DO
30 D=INT(RND(0)*5) + 2:REM DUREE
40 S=INT(RND(0)*300)+700:REM DEBUT
50 R=INT(RND(0)*(1020-S)):REM BORNE
60 P=INT(RND(0)*30)+5:REM PAS
70 T=SGN(RND(1)-.5):IF T=0 THEN 70
80 FOR Z=S TO S+T*R STEP P*T
90 SOUND 1,Z,D
100 Y=(Z AND 15)+1:FOR X=1 TO D
110 PRINT CHR$(18);:COLOR 1, Y:PRINT"" (Laisser espace entre "")
120 NEXT X,Z
130 LOOP
```

Les instructions REM vous aident à vous souvenir de la signification des instructions.

---

---

## LA BOITE A MUSIQUE DE VOTRE PLUS/4

Le dernier programme est plus long. C'est la "BOITE A MUSIQUE DE VOTRE PLUS/4". Quand vous appuyez sur les touches de 1 à 9, la note est jouée et elle apparaît à sa places sur la portée.

```
5 GOSUB 1000
6 FOR X=1 TO 9:READ N(X):NEXT X
8 CHAR 1,8,1,"*LA BOITE A MUSIQUE*"
10 VOL7
20 DO
30 GET A$:IF A$=""THEN 30
35 A=ASC(A$):IF 1<49 OR A>57 THEN 50
36 N=A-48
40 SOUND 1,N(N),4
45 GSHAPE N$,150,8*(6+(9-N)),4
46 FOR Z=1 TO 50:NEXT Z
47 GSHAPE N$,150,8*(6+(9-N)),4
50 LOOP UNTIL A=32
55 VOL 0:GRAPHIC 0:SCNCLR
60 END
100 DATA 345,383,453,516,571,596,643,685,704
1000 GRAPHIC 1,1
1010 FOR Y=60 TO 124 STEP 16
1020 DRAW 1,100,Y TO 200,Y
1030 NEXT Y
1040 A$="FEDCBAGFE"
1050 FOR X=1 TO 9:C=13
1060 IF INT(X/2)=X/2 THEN C=14
1070 CHAR 1,C,X+6,MID$(A$,X,1),0
1075 CHAR 1,C+10,X+6,RIGHT$(STR$(10-X),1)
1080 NEXT X
1090 FOR X=1 TO 8:FOR Y=11 TO16:DRAW 1,X,Y:NEXT Y,X
1100 Y=1:X=8:DRAW1,8,16 TO X,Y
1110 SSHAPE N$,1,1,8,16
1120 GSHAPE N$,1,1,4
1130 RETURN
```

Comme vous pouvez le voir, ce n'est pas difficile d'écrire son propre programme musical. Les exemples de ce chapitre ne vous montrent qu'une partie des possibilités musicales de votre Plus/4. N'ayez pas peur d'essayer de nouveaux sons et bruitages pour créer votre propre chef d'œuvre.

---

---

# **ENCYCLO- PEDIE DU PLUS/4**

---

---

## **DE A à Z**

L'encyclopédie du Plus/4 contient des informations utiles pour le novice et l'expert en informatique. Quelques sections sont obligatoires pour les débutants, comme la section BASIC 3.5, qui explique et donne la liste de toutes les commandes, instructions BASIC avec leur syntaxe. D'autres sections seront utiles aux informaticiens dont les connaissances s'étendent au-delà du BASIC. TEDMON, le moniteur langage machine, donne des informations pour la programmation en langage machine sur le Plus/4. D'autres sections peuvent être utiles pour tous les utilisateurs de Plus/4: liste des messages d'erreur, exemples de programmes, tableau des notes de musique, etc... L'encyclopédie du Plus/4 est utile pour tous les programmeurs du Plus/4, débutants ou expérimentés.

---

---

## ENCYCLOPEDIE DU BASIC 3.5

Ce manuel vous a donné une introduction au langage BASIC pour vous donner un avant-goût de la programmation et du vocabulaire utilisé. Cette encyclopédie vous donne la liste complète des règles (syntaxe) du langage BASIC 3.5, avec une description rapide de chacune. Essayez ces commandes et souvenez-vous que vous ne pouvez pas endommager votre Plus/4 en entrant des programmes et qu'on apprend à programmer en programmant.

L'encyclopédie vous donne la syntaxe, une brève explication et des exemples des commandes et instructions BASIC 3.5. Si vous voulez apprendre BASIC, la section 14 vous donne une liste de livres sur le BASIC.

Les commandes et les instructions sont énumérées dans des sections différentes. A l'intérieur des sections, les commandes et instructions sont données en ordre alphabétique. Les commandes sont généralement utilisées en mode direct, alors que les instructions sont plus souvent utilisées dans des programmes. Dans la plupart des cas, les commandes peuvent être utilisées comme des instructions dans un programme si vous les faites précéder d'un numéro de ligne. De même, les instructions peuvent être utilisées comme des commandes en les écrivant en mode direct (c'est à dire sans numéro de lignes).

Les différentes opérations BASIC sont regroupées en sections suivant ces critères:

- **Commandes:** les commandes utilisées pour faire fonctionner les programmes, les mettre en forme, les sauvegarder et les effacer.
- **Instructions:** les instructions des programmes BASIC utilisées dans les lignes numérotées de programmes.
- **Fonctions:** les fonctions de chaînes de caractères, de chiffres et d'impression.
- **Variables et opérateurs:** les différents types de variables, noms de variable autorisés et les opérateurs arithmétiques et logiques.

Une explication plus complète des commandes du BASIC 3.5 est donnée dans le "Guide de référence du programmeur Plus/4" disponible chez votre revendeur Commodore.

---

---

## SYNTAXE DES COMMANDES ET INSTRUCTIONS

Les commandes et les instructions présentées dans cette section de l'encyclopédie sont régies par des conventions logiques qui ont été créées pour rendre la syntaxe la plus claire possible. Dans la plupart des cas, il y a des exemples pour illustrer l'utilisation des commandes. L'exemple suivant montre les conventions utilisées dans les commandes et instructions BASIC:

exemple : **LOAD**"*nom du programme*",**DO**,**U8**

LOAD est le mot clé

DO est l'argument et U8 l'argument supplémentaire (option)

La partie des commandes et instructions que vous devez taper exactement comme dans le manuel est en caractères gras, alors que le nom des commandes est en majuscules. Les mots que vous ne devez pas taper de la même manière, comme le nom d'un programme, sont en italique. Quand il y a des guillemets ("", souvent autour d'un nom de programme ou de fichier), vous devez les inclure à l'endroit approprié comme dans l'exemple de syntaxe.

- Les mots clés, aussi appelés mots réservés, sont imprimés en majuscules et en caractères gras: vous devez entrer ces mots clés exactement comme ils apparaissent. Cependant, beaucoup de mots clés ont des abréviations que vous pouvez aussi utiliser. (Voir la section 2)

Les mots clés sont des mots qui font partie du langage BASIC que votre ordinateur connaît. Les mots clés sont la partie centrale d'une commande ou d'une instruction. Ils indiquent à l'ordinateur quelle sorte d'action vous voulez qu'il exécute. Ces mots ne peuvent pas être utilisés comme noms de variables.

- Les arguments (aussi appelés paramètres) apparaissent en minuscules et en italique. Les arguments sont les parties d'une commande ou d'une instruction que vous choisissez: ils ajoutent une information spécifique aux mots clés sur la commande ou l'instruction. Par exemple, un mot clé donne l'ordre à l'ordinateur de charger un programme, alors qu'un argument spécifie à l'ordinateur quel programme il doit charger et un second argument spécifie quel lecteur de disquettes contient le programme. Les arguments sont des noms de fichiers, de variables, numéros de ligne, etc...
- Les crochets [] donnent des arguments facultatifs. Vous choisissez ou pas un des arguments de la liste, selon vos besoins.

- 
- 
- Les signes supérieurs et inférieurs <> indiquent que vous devez choisir un des arguments donnés.
  - Les barres verticales (/) séparent les éléments d'une liste d'arguments quand vos choix sont limités à ces arguments et que vous ne pouvez en utiliser aucun autre. Quand ces arguments sont entre crochets, votre choix est limité à ceux-ci, mais ils sont facultatifs.
  - Les ellipses..., une série de trois points, signifie qu'une option ou un argument peut être répété plus d'une fois.
  - Les guillemets " " entourent des chaînes de caractères, des noms de fichiers et d'autres expressions. Quand des arguments sont inclus entre guillemets dans une syntaxe, vous devez inclure ces guillemets dans votre commande ou instruction. Les guillemets ne sont pas utilisés pour décrire des syntaxes; ils sont une partie intégrale d'une commande ou d'une instruction.
  - Les parenthèses (). Quand les arguments sont entre parenthèses dans une syntaxe, vous devez inclure les parenthèses dans la commande ou l'instruction. Ces parenthèses ne sont pas des conventions; elles sont obligatoires dans la commande ou l'instruction.
  - Les variables font référence à un nom de variable BASIC autorisé, tel que X, A\$ ou T%.
  - Les expressions font référence à des expressions BASIC autorisées, telles que A+B+2 ou 5\*(X+3).

---

---

## LES COMMANDES BASIC

### AUTO

Syntaxe : **AUTO** [*nombre*]

Déclenche la commande de numérotation automatique des lignes qui facilite la programmation en affichant les numéros de ligne à votre place. Lorsque vous entrez une ligne de programme en appuyant sur **ENTER**, le prochain numéro de ligne s'affiche à l'écran avec le curseur au bon endroit. Le [*nombre*] fait référence au saut de numéro entre chaque ligne. AUTO sans argument annule l'auto-numérotation, comme la commande RUN. Cette instruction n'est utilisable qu'en mode direct.

Exemple :

**AUTO 10** (auto-numérotation avec un saut de 10)

**AUTO 50** (auto-numérotation avec un saut de 50)

**AUTO** (arrête l'auto-numérotation)

---

### BACKUP

Syntaxe : **BACKUP D***lecteur n°* **TO D***lecteur n°*  
[, **ON U***unité n°*]

Cette commande copie tous les fichiers d'une disquette sur une autre dans les systèmes à double lecteurs de disquettes. Vous pouvez utiliser une disquette vierge qui n'a jamais été formatée par la commande HEADER, parce que la commande BACKUP copie toutes les informations y compris le format. Vous devez toujours faire des BACKUP des fichiers importants, en cas de perte ou de détérioration de l'original.

Comme la commande BACKUP formate aussi la disquette, elle détruit toutes les informations de la disquette destination. Ainsi avant d'effectuer une copie sur une disquette, soyez sûr qu'elle ne contient rien d'important. Voir la commande COPY.

NOTE: Cette commande ne peut être utilisée que sur les doubles lecteurs de disquettes.
--

Exemple:

**BACKUP D0 TO D1** (copie tous les fichiers de la disquette du lecteur 0 sur la disquette du lecteur 1)

**BACKUP D0 TO D1, ON U9** (copie à partir du lecteur 0 vers le lecteur 1, sur le périphérique n°9)

---

---

---

---

## COLLECT

Syntaxe :

**COLLECT** [Dlecteur n°][,ON Uunité n°]

Cette commande est utilisée pour libérer l'espace occupé par des fichiers mal fermés et elle supprime les références à ces fichiers dans le répertoire.

Exemple:

**COLLECT D0**

---

## CONT (Continue)

Syntaxe :

**CONT**

Cette commande est utilisée pour reprendre l'exécution d'un programme qui a été arrêté soit en utilisant la touche STOP, soit en rencontrant dans le programme une instruction STOP ou une instruction END. Le programme redémarre exactement à l'endroit où il a été interrompu. CONT ne fonctionne pas si vous avez modifié ou ajouté des lignes au programme (ou même si vous avez juste déplacé le curseur dans le programme en utilisant la touche **■** sans rien changer), si le programme est arrêté à cause d'une erreur ou si vous avez provoqué une erreur avant de reprendre le programme. Dans ces cas, vous obtenez le message d'erreur: CAN'T CONTINUE ERROR.

---

## COPY

Syntaxe :

**COPY** [Dlecteur n°,]"*fichier source*"**TO** [Dlecteur n°,]"*fichier destination*"[,ON Uunité n°]

Cette commande copie un fichier d'une disquette d'un lecteur (le fichier source) vers une disquette de l'autre lecteur, dans le cas de doubles lecteurs. Elle copie aussi un fichier sur le même lecteur (avec un nom différent).

Exemples:

**COPY D0,"test" TO D1,"test prog"** (copie "test" du lecteur 0 au lecteur 1 sous le nom "test prog".)

**COPY D0,"truc" TO D1,"truc"** (copie "truc" de 0 en 1)

**COPY D0 TO D1** (copie tous les fichiers de 0 en 1)

**COPY "trav.prog" TO"copie"** (copie "trav.prog" sur la même disquette sous le nom "copie".)

---

---

---

---

## DELETE

Syntaxe :

**DELETE** [*première ligne n°*][*- dernière ligne °*]

Cette commande supprime des lignes BASIC. Elle ne fonctionne qu'en mode direct.

Exemples :

**DELETE 75** (supprime la ligne 75)

**DELETE 10 - 50** (supprime de la ligne 10 à 50 inclus)

**DELETE - 50** (supprime toutes les lignes jusqu'à la ligne 50)

**DELETE 75 -** (supprime toutes les lignes à partir de la ligne 75)

---

## DIRECTORY

Syntaxe :

**DIRECTORY** [*Dlecteur n°*][*, Uunité n°*][*, "nom de fichier"*]

Cette commande affiche le répertoire de la disquette sur l'écran. Utilisez **■** S pour interrompre le défilement (appuyer sur une autre touche pour continuer le défilement). Utilisez la touche C (la touche Commodore) pour ralentir le défilement. Cette commande ne peut pas être utilisée pour imprimer une image écran. Vous devez charger le répertoire de la disquette (effaçant le programme en mémoire) pour le faire.

Exemples :

**DIRECTORY** (donne le répertoire complet de la disquette)

**DIRECTORY D1,U9,"travail"** (donne le nom des fichiers nommés "travail" du lecteur 1 du périphérique 9 (périphérique 8 par défaut))

**DIRECTORY "ab"** (donne la liste de tous les fichiers commençant par "ab".)

**DIRECTORY D0,"fichier?.bak"** (le ? remplace n'importe quel caractère à cette position: fichier1.bak, fichier2.bak.)

---

---

NOTE: Pour imprimer le répertoire du lecteur 1, périphérique 8, utilisez le programme suivant :

```
LOAD"$0",8  
OPEN4,4:CMD4:LIST  
PRINT#4:CLOSE4
```

---

## **DLOAD**

Syntaxe :

**DLOAD**"*nom du fichier*" [, **D**lecteur n°] [, **U**unité n°]

Cette commande charge un programme de la disquette dans la mémoire de l'ordinateur (utilisez LOAD pour charger un programme à partir du lecteur de cassette). Vous devez indiquer un nom de programme.

Exemples :

**DLOAD "banque"** (cherche sur la disquette le programme "banque" et le charge)

**DLOAD (A\$)** (charge un programme dont le nom est la variable A\$. Vous obtenez une erreur si A\$ est vide.)

La commande DLOAD peut être utilisée dans un programme BASIC pour trouver et exécuter un autre programme du disque. C'est ce qu'on appelle chaînage.

---

---

## DSAVE

Syntaxe :

**DSAVE** "*nom de fichier*" [,**D**lecteur no][,**U**unité no]

Cette commande sauvegarde un programme sur disquette (utilisez la commande SAVE pour sauvegarder sur le Datassette). Vous devez indiquer un nom de programme.

Exemples :

**DSAVE**"**banque**" (sauvegarde le programme "banque" sur disquette)

**DSAVE(A\$)** (sauvegarde sur disquette le programme dont le nom est la valeur de A\$)

**DSAVE**"**prog3**",**D0,U9** (sauvegarde le programme "prog3" sur le lecteur de disquette 0 du périphérique 9)

---

---

## HEADER

Syntaxe :

**HEADER** "*nom de disquette*" [,**i.d.no**],**D**lecteur n° [,**ON U**unité no]

Avant d'utiliser une disquette vierge, vous devez la formater avec la commande HEADER. Si vous voulez effacer complètement une disquette pour la réutiliser, vous pouvez utiliser la commande HEADER. Cette commande divise la disquette en secteurs appelés blocs et elle crée sur la disquette une table des matières, qui est appelée répertoire ou catalogue. le nom de disquette peut être n'importe quelle chaîne de caractères de 16 caractères au maximum. Le i.d.no (numéro d'identification de la disquette) comprend 2 caractères au choix. Donnez à chaque disquette un numéro d'i.d. différent. Faites attention quand vous formatez une disquette parce que la commande efface toutes les données enregistrées. Si vous n'indiquez pas l'i.d.no, vous aurez un formatage rapide, l'ancien numéro d'i.d. restera sur la disquette, mais vous ne pouvez utiliser le formatage rapide que si la disquette a déjà été formatée car la commande HEADER rapide ne fait qu'effacer le répertoire de la disquette au lieu de la formater.

Exemples :

**HEADER** "**madisquette**",**I23,D1**

**HEADER** "**enreg**",**I45,D1,U8**

---

---

## HELP

Syntaxe :

**HELP**

La commande HELP est utile quand vous avez fait une erreur dans votre programme. Si vous tapez HELP, la ligne, où l'erreur se trouve, est affichée avec la partie de la ligne contenant l'erreur qui clignote.

---

## KEY

Syntaxe :

**KEY** [*touche no, chaîne de caractère*]

L'ordinateur Plus/4 a huit touches de fonction: 4 sans la touche SHIFT et 4 avec la touche SHIFT. Le Plus/4 vous permet de définir la fonction de chaque touche.

La commande KEY sans paramètres affiche la liste des touches de fonction et leurs définitions correspondantes. Les données que vous associez à une touche sont affichées quand vous appuyez sur la touche. La longueur maximum pour l'ensemble des définitions est de 128 caractères. Une commande (ou une série de commandes) peut être associée à une touche. Par exemple :

**KEY 7,"GRAPHIC0"+CHR\$(13)+"LIST"+CHR\$(13)**

indiquera à l'ordinateur de sélectionner le mode texte et de donner la liste de votre programme, quand la touche "F7" est pressée. Le CHR\$(13) est le code ASCII de la touche . Utilisez CHR\$(34) pour avoir des guillemets dans la chaîne de caractères de la commande KEY.

Les touches peuvent être redéfinies à l'intérieur d'un programme. Par exemple:

**10 KEY 2,"test"+CHR\$(34):KEY 3,"non"  
10 FOR I=1 TO 8:KEY I,CHR\$(I+132):NEXT**

(définit les touches de fonction telles qu'elles sont définies sur le Commodore 64 et le VIC 20)

Pour redéfinir les touches de fonctions avec leurs valeurs par défaut, il suffit d'éteindre et de réallumer le Plus/4 ou d'appuyer sur le bouton RESET.

---

---

---

---

## LIST

Syntaxe :

**LIST** [*première ligne*] [-*dernière ligne*]

La commande LIST vous permet de visualiser les lignes du programme BASIC que vous avez tapées ou que vous avez chargées en mémoire du Plus/4. Quand la commande est utilisée seule (sans nombre derrière), le Plus/4 affiche la liste complète de votre programme. Vous pouvez ralentir le défilement avec la touche C, l'arrêter momentanément avec les touches **STOP** et S (le défilement reprend en tapant n'importe quelle autre touche) ou le stopper avec la touche **STOP**. Si vous indiquez un numéro de ligne, le Plus/4 affiche cette ligne. Si vous tapez LIST avec deux numéros séparés par un tiret, le Plus/4 vous montre toutes les lignes comprises entre ces deux numéros. Si vous tapez LIST suivi d'un numéro et d'un tiret, vous aurez toutes les lignes comprises entre ce numéro et la fin du programme. Et enfin, si vous tapez LIST, suivi d'un tiret et d'un numéro, vous aurez toutes les lignes du début jusqu'à la ligne dont vous avez indiqué le numéro. En utilisant toutes ces possibilités, vous pouvez examiner toutes les parties d'un programme et amener facilement des lignes à l'écran pour les modifier.

Exemples :

**LIST** (liste entière du programme)

**LIST 100-** (liste de la ligne 100 jusqu'à la dernière ligne)

**LIST 10** (affiche seulement la ligne 10)

**LIST -100** (liste de la première ligne à la ligne 100)

**LIST 10-200** (liste de la ligne 10 jusqu'à la ligne 200)

---

---

## LOAD

Syntaxe :

**LOAD** [*"nom de fichier"*][,*unité no*][,*translatable*]

Cette commande est utilisée pour charger en mémoire un programme du lecteur-enregistreur de cassettes ou d'une disquette. Si vous tapez seulement LOAD et appuyez sur la touche **STOP**, l'écran du Plus/4 se vide. Appuyez sur la touche PLAY du lecteur-enregistreur de cassettes et

le Plus/4 se met à la recherche des programmes sur la cassette. Dès qu'il en trouve un, le Plus/4 affiche FOUND"nom de fichier". Vous

---

---

pouvez appuyer sur la touche **C** pour charger le programme. Dès que le programme est chargé (LOADING), vous pouvez utiliser les commandes **RUN**, **LIST** ou modifier le programme.

Vous pouvez aussi taper la commande **LOAD** suivi d'un nom de programme, qui est le plus souvent une chaîne de caractères entre guillemets ("nom de programme"). Le nom peut être suivi d'une virgule (après les guillemets) et d'un chiffre ( ou une variable numérique), qui servent à déterminer le périphérique où le programme est enregistré (disquette ou cassette). Si vous n'indiquez pas de chiffres, le Plus/4 prend le périphérique 1 par défaut, qui est le lecteur-enregistreur de cassettes.

L'autre périphérique normalement utilisé avec la commande **LOAD** est le lecteur de disquettes qui a le numéro 8.

Exemples :

**LOAD** (charge le premier programme de la cassette)

**LOAD"hello"** (cherche sur la cassette le programme nommé "hello" et le charge s'il le trouve)

**LOAD A\$** (cherche le programme et charge le programme qui a la valeur de A\$)

**LOAD "hello",8** (cherche et charge le programme nommé "hello" sur la disquette)

La commande **LOAD** peut être utilisée dans un programme **BASIC** pour charger et exécuter le prochain programme de la cassette. c'est ce qu'on appelle le chaînage.

L'argument translatable détermine l'endroit de la mémoire où le programme sera chargé. Le nombre 0 indiquera au Plus/4 de charger le programme en début de la zone mémoire réservée aux programmes **BASIC**. Le nombre 1 indiquera de charger le programme à la même adresse mémoire où était le programme quand il a été sauvegardé. La valeur par défaut de l'argument translatable est 0. On utilise généralement cet argument pour charger des programmes écrits en langage machine.

---

---

## NEW

Syntaxe :

**NEW**

Cette commande efface le programme en mémoire et met à zéro toutes les variables. Sauf si vous aviez sauvegardé le programme, il est perdu et vous êtes obligé de le retaper. Faites attention quand vous utilisez cette commande. La commande NEW peut aussi être utilisée dans un programme BASIC. Quand le Plus/4 arrive sur cette ligne, le programme est effacé et tout s'arrête. Ce n'est pas spécialement recommandé en utilisation normale.

---

## RENAME

Syntaxe :

**RENAME** [,Dlecteur n°]"ancien nom"**TO**"nouveau nom"[,Uunité n°]

Cette commande est utilisée pour changer le nom d'un fichier sur disquette.

Exemple :

**RENAME**"test"**TO**"test final"

(change le nom du fichier "test" en "test final".)

---

## RENUMBER

Syntaxe :

**RENUMBER** [*nouvelle ligne de départ*[, *saut*[,*ancienne ligne de départ*]]]

Cette commande renumérote les lignes du programme en mémoire. Après la renumérotation, le programme commencera par la nouvelle ligne de départ que vous définissez. Si vous ne définissez pas de nouvelle ligne de départ, la valeur par défaut est 10.

Le saut définit l'écart entre chaque numéro de ligne, la valeur par défaut est 10.

L'ancienne ligne de départ est la ligne à partir de laquelle le programme sera renuméroté. Cela vous permet de renuméroter une partie de votre programme. La valeur par défaut correspond à la première ligne du programme.

Cette commande ne peut être exécutée qu'en mode direct.

---

Exemples :

**RENUMBER 20,20,1** (renumérote de 20 en 20 à partir de la ligne 1, qui devient la ligne 20)

**RENUMBER,,65** (renumérote de 10 en 10 à partir de la ligne 65, qui devient la ligne 10)

---

## **RUN**

Syntaxe :

**RUN** [*ligne n°*]

Une fois que le programme est tapé ou chargé en mémoire, la commande RUN fait démarrer l'exécution du programme. Cette commande met à zéro toutes les variables avant de commencer à exécuter le programme. S'il n'y a pas de numéro de ligne après RUN, l'ordinateur démarre sur le numéro de ligne le plus bas. S'il y a un numéro de ligne, le programme commence à partir de la ligne spécifiée. RUN peut être utilisé dans un programme.

Exemples :

**RUN** (démarre le programme avec la première ligne)

**RUN 100** (démarre l'exécution avec la ligne 100)

---

## **SAVE**

Syntaxe :

**SAVE** [*"nom de fichier"*],[*unité n°*],[*repère EOT*]]

**Cette commande mémorise le programme en mémoire sur cassette ou disquette. Si vous tapez seulement SAVE et la touche **STOP**, le programme est sauvegardé sur cassette. L'ordinateur n'a aucun moyen de savoir si un programme est déjà enregistré sur la cassette à cet endroit, aussi vérifiez vos cassettes. Si vous tapez SAVE suivi d'un nom entre guillemets ou d'une variable chaîne de caractères, le Plus/4 sauvegarde le programme sous ce nom, pour qu'il soit plus facilement trouvé et chargé la prochaine fois. Si vous voulez indiquer un numéro de périphérique, vous devez mettre après le nom de fichier une virgule suivie d'un numéro ou d'une variable numérique. Le périphérique numéro 1 est le lecteur-enregistreur de cassettes et le 8 le lecteur de disquettes. Après le numéro de périphérique, il peut y avoir une virgule suivie d'un 1 ou d'un 0. Si c'est un 2, le Plus/4 place un repère de fin de cassette (END OF TAPE, EOT) à la fin du programme.**

---

---

---

Si vous essayez de charger un programme et que le Plus/4 trouve un de ces repères, vous obtiendrez le message d'erreur FILE NOT FOUND ERROR (fichier non trouvé).

Exemples :

**SAVE** (sauvegarde le programme sur cassette sans nom)  
**SAVE "hello"** (sauvegarde sur cassette sous le nom "hello")  
**SAVE A\$** (sauvegarde sur cassette sous le nom de la variable A\$)  
**SAVE "hello",8** (sauvegarde sur disquette sous le nom "hello")  
**SAVE "hello", 1,2** (sauvegarde sur cassette sous le nom "hello" et place un repère EOT après le programme.)

---

## SCRATCH

Syntaxe :

**SCRATCH** "*nom du fichier*" [,**D**lecteur n°][,**U**unité n°]

Cette commande efface un fichier du répertoire de la disquette. Par mesure de précaution, le Plus/4 affiche à l'écran: ARE YOU SURE? (êtes-vous sûr?) avant d'exécuter la commande. Vous tapez Y (oui) pour confirmer l'effacement et N (non) pour annuler la commande. Utilisez cette commande pour effacer des fichiers inutiles et pour récupérer de la place sur la disquette.

Exemples :

**SCRATCH "ma copie", D1** (efface le fichier "ma copie" sur la disquette du lecteur 1)

---

## VERIFY

Syntaxe :

**VERIFY** "*nom du fichier*"[,**u**unité n°][,**t**ranslatable]

Cette commande incite le Plus/4 à contrôler le programme sur la disquette ou la cassette par rapport à celui qui est en mémoire. Ceci vous permet de vérifier que le programme que vous venez de sauvegarder est bien enregistré. Dans le cas où votre support est mauvais ou que quelque chose ne marche pas, le Plus/4 vous l'indique. Cette commande est aussi utilisée pour positionner la cassette de manière à ce que le Plus/4 puisse écrire après le dernier programme enregistré sur cette cassette.

---

---

---

---

Vous n'avez qu'à indiquer au Plus/4 de vérifier (VERIFY) le dernier programme de la cassette. Un message d'erreur s'affichera à l'écran, indiquant que le programme en mémoire est différent du programme sur cassette (ce que vous savez déjà, car ce programme n'est pas en mémoire). Maintenant la cassette est bien positionnée et vous pouvez sauvegarder le prochain programme sans risque d'écraser un ancien programme.

La commande VERIFY sans autres indications après la commande incite le Plus/4 à contrôler le programme suivant sur la cassette par rapport au programme présent en mémoire, quelque soit son nom. VERIFY suivi d'un nom de programme ou d'une variable chaîne de caractères recherche le programme concerné et ensuite le contrôle. VERIFY suivi d'un nom, d'une virgule et d'un numéro, contrôle le programme sur le périphérique ayant ce numéro (1 pour le Datassette, 8 pour le lecteur de disquettes). L'argument translatable est le même que celui de la commande LOAD.

Exemples :

**VERIFY** (contrôle le prochain programme sur la cassette)

**VERIFY "hello"** (cherche sur cassette et contrôle le programme "hello")

**VERIFY "hello",8,1** (cherche sur disquette et contrôle le programme "hello")

---

---

## INSTRUCTIONS BASIC

### BOX

Syntaxe :

**BOX** [*source couleur* n°],*a1,b1,a2,b2*[[*,angle*][*,coulore*]]

*source couleur* : de 0 à 3; valeur par défaut 1, couleur du fond.

*a1,b1* : coordonnées du premier coin.

*a2,b2* : coordonnées du coin opposé, par défaut PC.

*angle* : rotation dans le sens des aiguilles d'une montre en degrés, par défaut 0 d°.

*coulore* : colore le dessin (BOX) (0: hors fonction, 1: en fonction), valeur par défaut 0.

Cette instruction vous permet de dessiner un rectangle de toute taille, n'importe où à l'écran. Pour garder les valeurs par défaut, mettre une virgule sans donner de valeur. Le point de rotation est au centre du rectangle. Le curseur point (Pixel Cursor, PC) est positionné aux coordonnées *a2,b2* à la fin de l'instruction BOX.

Exemples :

**BOX 1,10,10,60,60** (trace un rectangle)

**BOX,10,10,60,60,45,1** (trace un losange plein)

**BOX,30,90,,45,1** (trace un polygone plein)

---

### CHAR

Syntaxe :

**CHAR** [*source couleur* n°],*x,y* [*,chaîne de caractères*][*,vidéo inversé*]

Autre syntaxe :

**CHAR** [*source couleur* n°],*x,y* [,"*chaîne de caractères*"] [*,vidéo inversé*]

*source couleur* : de 0 à 3.

*x* : n° de colonne caractère (0 à 39).

*y* : n° de ligne caractère (0 à 24).

*chaîne* : chaîne de caractères à imprimer.

*vidéo inversé* : 0, hors fonction et 1, en fonction.

---

Du texte (chaînes de caractères alpha-numériques) peut être affiché à l'écran à un endroit donné avec l'instruction CHAR. La définition du caractère est lue directement dans la ROM définissant les caractères du Plus/4. Vous indiquez les coordonnées x et y de la position de départ et la chaîne de caractères que vous voulez afficher. La source couleur et la vidéo inverse sont en option.

La chaîne de caractères s'affiche sur la ligne suivante si elle dépasse les limites de l'écran. Quand vous êtes en mode texte, l'instruction CHAR est équivalente à la commande PRINT, avec vidéo inverse, curseurs, clignotants, etc... Ces fonctions de contrôle à l'intérieur de la chaîne de caractères ne fonctionnent pas quand l'instruction CHAR est utilisée en mode graphique.

NOTE: Dans les modes multi-couleur, pour afficher un caractère en multi-couleur 2, il faut mettre la couleur source à 0 et la vidéo inverse à 1. Pour afficher un caractère en multi-couleur 1, il faut mettre la couleur source à 0 et la vidéo inverse à 0.

---

## CIRCLE

Syntaxe :

**CIRCLE** [*sc*],[*a,b*],[*xr*],[*yr*],[*da*],[*fa*],[*angle*],[*inc*]]]

*sc* : source couleur (0 à 3)

*a,b* : coordonnées du centre, par défaut le curseur point (PC)

*xr* : rayon x

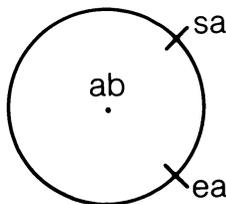
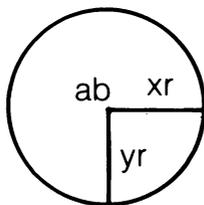
*yr* : rayon y, par défaut *xr*

*da* : angle de début d'arc, par défaut 0

*fa* : angle de fin d'arc, par défaut 360

*angle* : rotation dans le sens des aiguilles d'une montre en degrés, par défaut 0°.

*inc* : nombre de degrés entre les segments, par défaut 2.



---

---

L'instruction CIRCLE vous permet de dessiner un cercle, une ellipse, un arc, un triangle ou un octogone. Le point final de la figure est sur la circonférence et est défini par l'angle de fin d'arc. Toute rotation se fait par rapport au centre. Définir le rayon Y égal au rayon X ne tracera pas un cercle, parce que les échelles des coordonnées de X et de Y sont différentes. Les arcs sont dessinés en commençant par l'angle de début d'arc et en finissant par l'angle de fin d'arc dans le sens des aiguilles d'une montre. L'incrément contrôle la définition de la figure, plus l'incrément est petit, plus la forme est ronde.

Exemples :

**CIRCLE,160,100,65,10** (dessine une ellipse)

**CIRCLE,160,100,65,50** (dessine un cercle)

**CIRCLE,60,40,20,18,,,,,45** (dessine un octogone)

**CIRCLE,260,40,20,,,,,90** (dessine un losange)

**CIRCLE,60,140,20,18,,,,,120** (dessine un triangle)

---

## CLOSE

Syntaxe :

**CLOSE** *fichier* n°

Cette instruction achève et ferme tous les fichiers utilisés par des instructions OPEN. Le numéro suivant CLOSE est le numéro du fichier à fermer.

Exemple :

**CLOSE 2** (ferme le fichier logique 2)

---

## CLR

Syntaxe :

**CLR**

Cette instruction met à zéro toutes les variables sans affecter le programme. Cette instruction s'exécute automatiquement avec les commandes RUN et NEW ou à la saisie d'une ligne BASIC.

---

## CMD

Syntaxe :

**CMD** *fichier* n°[,*liste*]

CMD envoie ce qui s'affiche normalement à l'écran (ex: PRINT, LIST, mais non POKE sur l'écran) vers un autre périphérique. Celui-ci peut

---

---

---

---

être une imprimante ou un fichier sur disquette ou cassette. Le périphérique ou le fichier doivent être d'abord ouverts. La commande CMD doit être suivie d'un numéro ou d'une variable numérique se référant à un fichier.

Exemples :

**OPEN 1,4** (ouvre le périphérique n°4: l'imprimante)

**CMD 1** (toutes les sorties écran vont sur l'imprimante)

**LIST** (la liste est imprimée, mais n'apparaît pas à l'écran, même le mot READY)

**PRINT #1** (renvoie les sorties sur écran)

**CLOSE 1** (ferme le fichier)

---

## COLOR

Syntaxe :

**COLOR** *source* n°, *couleur* n°[,*brillance* n°]

Cette instruction donne la couleur d'une des cinq sources suivantes:

---

Numéro	Source
0	Fond
1	Caractère
2	Multi-couleur 1
3	Multi-couleur 2
4	Cadre

---

Vous pouvez utiliser des numéros de couleur compris entre 1 et 16 (noir, blanc, etc...). En option, vous pouvez ajouter la brillance qui varie de 0 à 7, avec 0 le moins brillant et 7 le plus brillant, la valeur par défaut est 7. La brillance vous permet de sélectionner ces huit niveaux d'intensité pour toutes les couleurs, excepté le noir.

---

## DATA

Syntaxe :

**DATA** *suite de constantes séparées par des virgules*

Cette instruction est suivie d'une liste d'éléments à utiliser par les instructions READ. Ces éléments peuvent être des valeurs numériques ou des chaînes de caractères et ils sont séparés par des virgules. Les chaînes de caractères n'ont pas besoin d'être entre

---

---

---

---

guillemets, à moins qu'elles ne contiennent des espaces, des doubles points et des virgules. Si deux virgules se suivent, la valeur sera un 0 pour une variable numérique ou une chaîne vide pour une variable chaîne. Voyez aussi l'instruction RESTORE qui permet au Plus/4 de relire les données.

Exemple :

**DATA 100,200,FRED,"hello, Maman",,3,14,ABC123**

---

## **DEF FN**

Syntaxe :

**DEF FN** *nom(variable)=expression*

### **(DEFinit une FonctioN)**

Cette commande vous permet de définir un calcul complexe comme étant une fonction. Dans le cas où une formule longue est utilisée plusieurs fois dans un programme, cette commande permet de gagner de la place. Le nom que vous donnez à la fonction commence par FN suivi par un nom de variable numérique. Vous devez d'abord définir la fonction en utilisant l'instruction DEF suivie par le nom de votre fonction, suivi par une variable numérique entre parenthèses. Ensuite vous avez un signe = suivi de la formule que vous voulez définir. Vous pouvez alors "appeler" la formule en substituant un nombre à la variable comme suit :

**10 DEF FNA(X)=12\*(34.75-X/.3)+X**

**20 PRINT FNA(7)**

(Le chiffre 7 est inséré à la place de X dans la formule)

---

## **DIM**

Syntaxe :

**DIM** *variable(indices)[,variable(indice)]...*

Avant d'utiliser un tableau de variables, le programme doit d'abord exécuter l'instruction DIM pour le DIMensionner (sauf si le tableau n'a pas plus de 11 éléments). L'instruction DIM est suivie du nom du tableau, qui peut être n'importe quel nom de variable. Ensuite, entre parenthèses, vous donnez le nombre (ou une variable numérique) d'éléments dans chaque dimension. Un tableau de plus d'un élément est appelé matrice. Vous pouvez utiliser autant de

---

---

---

---

dimensions que vous voulez, mais gardez à l'esprit que la liste complète des variables que vous êtes en train de créer prend de la place en mémoire et il est facile de ne plus avoir d'espace mémoire si vous ne faites pas attention.

Pour vous représenter le nombre de variables créées par chaque instruction DIM, multipliez le nombre total d'éléments dans chaque dimension du tableau. (Chaque tableau commence avec un élément 0.)

NOTE: Les tableaux de nombres entiers prennent 2/5ème de la place des tableaux de nombres réels (virgule flottante).

Exemple :

**10 DIM A\$(40), B7(15), CC%(4,4,4)**  
(41 éléments, 16 éléments, 125 éléments)

Vous pouvez donner une dimension à plusieurs tableaux dans une instruction DIM en séparant les tableaux par des virgules. Si le programme exécute plus d'une fois la même instruction DIM, vous aurez le message d'erreur: REDIM'D ARRAY ERROR. Dans la pratique, il vaut mieux placer l'instruction DIM au début du programme.

---

## **DO/LOOP/WHILE/UNTIL/EXIT**

Syntaxe :

**DO[UNTIL** *argument logique* **WHILE** *argument logique*]*instructions*  
**[EXIT]**

**LOOP[UNTIL** *argument logique* **WHILE** *argument logique*]

Exécute les instructions entre DO et LOOP. Si, ni UNTIL ni WHILE ne modifient DO ou LOOP, l'exécution des instructions présentes continue indéfiniment. S'il y a EXIT dans une boucle, l'exécution est transférée à la première instruction suivant LOOP. Les boucles DO peuvent être imbriquées les unes dans les autres, suivant les règles des boucles FOR-NEXT. Si le paramètre UNTIL est utilisé, le programme continuera de boucler jusqu'à ce que l'argument logique soit satisfait (devienne vrai). Le paramètre WHILE est l'opposé de l'argument UNTIL: le programme continue de boucler aussi longtemps que l'argument logique est vrai. Un exemple d'argument logique est: A=1 ou G>=65.

---

---

---

---

Exemple :

```
DO UNTIL X=0 OR X=1
:
LOOP
DO WHILE A$=" ":GET A$:LOOP
```

---

## **DRAW**

Syntaxe :

**DRAW** source couleur n°[,a1,b1[TOa2,b2...]]

Avec cette commande vous pouvez dessiner des points individuels, des lignes et des formes. Vous donnez une source couleur (0-3), commençant en a1,b1 et finissant en a2,b2.

Exemples :

**DRAW 1,100,50** (pas de point final défini, valeur par défaut a1,b1, pour dessiner un point)

**DRAW,10,10 TO 100,60** (pour dessiner une ligne)

**DRAW, TO 25,30** (pour dessiner une ligne)

**DRAW,10,10 TO 10,60 TO 100,60 TO 10,10** (pour dessiner une forme)

---

## **END**

Syntaxe :

**END**

Quand le programme rencontre une instruction END, le programme s'arrête immédiatement. Vous pouvez utiliser la commande CONT pour redémarrer le programme à l'instruction suivant END.

---

## **FOR...TO...STEP**

Syntaxe :

**FOR** variable=valeur de départ **TO** valeur de fin [**STEP** incrément]

Cette instruction agit avec l'instruction NEXT pour répéter une section du programme un nombre déterminé de fois. Vous pouvez donner au Plus/4 une valeur de fin élevée pour créer une pause dans

---

---

---

---

le programme de quelques secondes, au cas où vous ayez besoin de compter quelque chose ou si une opération doit être répétée un certain nombre de fois (imprimer par exemple).

La variable de boucle est incrémentée ou décrétementée pendant la boucle FOR/NEXT. Les valeurs de départ et de fin sont les limites de la valeur de la variable de boucle.

La logique de l'instruction FOR est la suivante: premièrement, la variable de boucle est mise à la valeur de départ. Quand le programme rencontre la commande NEXT, il ajoute la valeur de l'incrément (STEP ou PAS, valeur par défaut 1) à la valeur de la variable de boucle et vérifie que cette valeur n'est pas supérieure à la valeur de fin. Si la valeur n'est pas supérieure, la prochaine ligne exécutée est l'instruction qui suit FOR. Si la valeur de boucle est supérieure à la valeur de fin de comptage, alors la prochaine instruction exécutée est celle qui suit NEXT. Voyez aussi l'instruction NEXT.

Exemple :

```
10 FOR L=1 TO 10
20 PRINT L
30 NEXT L
40 PRINT "J'AI FINI! L="L
```

Ce programme imprime les nombres de 1 jusqu'à 10 sur l'écran, suivis par le message: J'AI FINI! L=11.

La fin de la valeur de la boucle peut être suivie du mot STEP et d'un autre nombre ou variable. Dans ce cas, la valeur suivant STEP est ajoutée chaque fois à la place de 1. Ceci vous permet de compter en arrière ou de compter par fractions.

Vous pouvez mettre des boucles à l'intérieur d'autres boucles, appelées des boucles imbriquées. Vous devez faire attention à ce que la dernière boucle commencée soit la première terminée.

Exemple :

```
10 FOR L=1 TO 100
20 FOR A=5 TO 11 STEP 2 (cette boucle est imbriquée)
30 NEXT A
40 NEXT L
```

---

---

## GET

Syntaxe :

**GET** *liste de variables*

L'instruction GET vous permet d'obtenir des données à partir du clavier, un caractère à la fois. Lorsque GET est exécutée, le caractère qui est frappé est assigné à la variable. Si aucun caractère n'est frappé, un caractère zéro (blanc) est assigné et le programme continue. On n'a pas besoin d'appuyer sur la touche **RETURN** et la touche **RETURN** peut être assignée à la variable avec GET.

GET est suivi d'un nom de variable, généralement une variable-chaîne. Si une variable numérique a été utilisée et si une touche non numérique est pressée, le programme s'arrêtera avec un message d'erreur. L'instruction peut être placée dans une boucle, pour contrôler un résultat nul (blanc), cette boucle se poursuit jusqu'à ce qu'une touche soit pressée. L'instruction GETKEY peut aussi être utilisée dans ce cas. Cette commande ne peut être exécutée que dans un programme.

Exemple :

```
10 GET A$:IF A$<>"A" THEN 10
```

(cette ligne attend que la touche A soit pressée pour continuer.)

---

## GETKEY

Syntaxe :

**GETKEY** *liste de variables*

L'instruction GETKEY est presque identique à GET. Contrairement à GET, GETKEY attend que l'utilisateur tape un caractère sur le clavier. Ceci permet de l'utiliser facilement pour attendre qu'un caractère soit tapé.

Cette commande ne peut être utilisée qu'à l'intérieur d'un programme.

Exemple :

```
10 GETKEY A$ (cette ligne attend qu'un caractère soit tapé, appuyer sur n'importe quelle touche fera continuer le programme.)
```

---

## GET#

Syntaxe :

**GET#** *fichier n°, liste de variables*

GET# est utilisée avec un périphérique ou un fichier ouvert (OPENed) auparavant, pour lire un caractère à la fois à partir de ce périphérique ou de ce fichier. Elle fonctionne de la même manière

---

---

---

---

que GET. Cette commande ne peut être exécutée que dans un programme.

Exemple :  
**GET# 1,A\$**

---

## **GOSUB**

Syntaxe :  
**GOSUB** *ligne #*

Cette instruction est similaire à GOTO, mis à part que l'ordinateur mémorise la ligne de programme exécutée en dernier avant GOSUB. Lorsqu'une ligne comportant une instruction RETURN est rencontrée, le programme revient à l'instruction suivant immédiatement GOSUB. La partie de programme appelée par GOSUB est un sous-programme. Ceci est utile lorsqu'il existe dans votre programme une routine utilisée dans plusieurs parties du programme. Au lieu de retaper constamment la routine, vous créez un sous-programme et vous faites appel à lui avec GOSUB à partir de différentes parties du programme. Voyez aussi l'instruction RETURN.

Exemple :  
**20 GOSUB 800**  
: (indique d'aller au sous-programme en ligne 800  
et de l'exécuter)  
:  
**800 PRINT" SALUT":RETURN**

---

## **GOTO ou GO TO**

Syntaxe :  
**GOTO** *ligne n°*

Lorsqu'une instruction GOTO est exécutée, la ligne suivante à exécuter sera celle comportant le numéro de ligne suivant GOTO. Si vous l'utilisez en mode direct, GOTO vous permet de commencer l'exécution du programme au numéro de ligne donné sans mettre à zéro les variables.

Exemple :  
**10 PRINT"COMMODORE"**  
**20 GOTO 10**  
(le GOTO en ligne 20 fait répéter la ligne 10 jusqu'à ce que vous appuyez sur **ENTER**)

---

---

---

---

# GRAPHIC

Syntaxe :

**GRAPHIC**<mode[,effacement]/CLR>

MODE	DESCRIPTION
0	texte
1	graphique haute résolution
2	graphique haute résolution plus texte
3	graphique multi-couleur
4	graphique multi-couleur plus texte.

La commande GRAPHIC 1-4 alloue un espace mémoire de 10 K et le début de l'espace réservé au BASIC est mis au dessus de la zone réservée à la haute résolution. Cette zone reste allouée, même si l'utilisateur revient en mode texte (GRAPHIC 0). Si le second argument de l'instruction est 1, l'écran se vide. La commande GRAPHIC CLR annule l'espace mémoire de 10K réservé et le rend à nouveau disponible au BASIC.

---

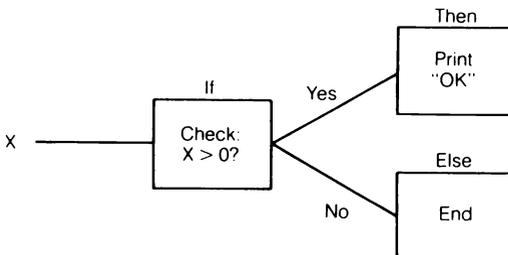
## IF...THEN [...:ELSE]

Syntaxe :

**IF** *expression* **THEN** *instruction* [**:ELSE** *instruction*]

IF...THEN permet au Plus/4 d'analyser une expression logique précédée par IF et de prendre une des deux voies d'action possibles. Si l'expression est vraie, l'instruction suivant THEN est alors exécutée. Ceci peut être une instruction BASIC. Si l'expression est fausse, le programme passe directement à la ligne suivante, à moins qu'il n'y ait ELSE. L'expression logique peut être une variable ou une formule, dans ce cas elle est considérée vraie si elle n'est pas nulle et fausse si elle est nulle. Dans la plupart des cas, il s'agit d'une expression impliquant des opérateurs relationnels (=,<,>,<=,>=,<>, AND, OR, NOT).

Si elle est présente, l'instruction ELSE doit être sur le même ligne que IF..THEN. Si l'instruction ELSE est présente, elle est exécutée quand



---

---

l'instruction THEN ne l'est pas. C'est à dire, ELSE est exécutée quand l'expression IF est fausse.

Exemple :

```
50 IFX>0 THEN PRINT "ok":ELSE END
```

(vérifie la valeur de X. Si X>0, l'instruction THEN est exécutée. Si X<=0, l'instruction ELSE est exécutée.)

---

## **INPUT**

Syntaxe :

```
INPUT[message;] liste de variables
```

L'instruction INPUT permet au programme d'obtenir des données de l'utilisateur en assignant ces données à une ou plusieurs variables. Le programme s'arrête, affiche un point d'interrogation (?) sur l'écran et attend que l'utilisateur tape la réponse et appuie sur la touche **ENTER**.

INPUT est suivi d'un nom de variable, ou d'une liste de noms de variables, séparés par des virgules. Un message peut être placé entre guillemets avant la liste des variables. Si ce message est présent, vous devez mettre un point-virgule (;) après le dernier guillemet. Si plus d'une variable doit être entrée (INPUT), elles doivent être séparées par des virgules lorsqu'elles sont entrées. Sinon, l'ordinateur demande les valeurs restantes en affichant deux points d'interrogation (??) à l'écran. Si vous appuyez sur **ENTER** sans entrer (INPUT) de valeur, la variable correspondante de l'instruction INPUT garde la valeur précédente. Cette instruction ne peut être exécutées que dans un programme.

Exemple :

```
10 INPUT"tapez un nombre SVP";A
```

```
20 INPUT"et votre nom";A$
```

```
30 INPUT B$
```

```
40 PRINT"vous ne saviez pas ce que je voulais!!!"
```

---

## **INPUT#**

Syntaxe :

```
INPUT# fichier n°, liste de variables
```

Cette instruction est similaire à INPUT, mais obtient les données d'un fichier ou d'un périphérique précédemment ouvert (OPENed). Aucun message n'est permis. Cette commande ne peut être utilisée qu'en mode programme.

Exemple :

```
INPUT#2, A$,C,D$
```

---

---

---

---

## LET

Syntaxe :  
**[LET]** *expression*

L'instruction LET n'est presque jamais utilisée dans des programmes parce qu'elle est optionnelle, mais cette instruction est le cœur de la programmation en BASIC. Quand vous voulez définir une variable ou donner une valeur, LET est utilisé. Le nom de la variable dans laquelle on met le résultat du calcul, est à gauche du signe égal et le nombre ou la formule à droite.

Exemple :  
10 **LET** A=5  
20 B=6  
30 C=A\*B+3  
40 D\$="hello"

---

## LOCATE

Syntaxe :  
**LOCATE** *coordonnée X, coordonnée Y*

La commande LOCATE vous permet de placer le curseur point (le curseur pixel PC) où vous le désirez sur l'écran. Le PC est la position du point de départ du prochain dessin. Contrairement au curseur texte, vous ne pouvez pas voir le PC, mais vous pouvez le déplacer avec la commande LOCATE. Par exemple:

### **LOCATE 160,100**

positionne le PC au centre de l'écran haute résolution. Vous ne voyez rien jusqu'à ce que vous dessiniez quelque chose. Vous pouvez obtenir la position du PC avec la fonction RDOT(0) pour la coordonnée X et RDOT(1) pour la coordonnée Y. La source couleur du point où se trouve le PC est connu avec la fonction RDOT(2) (dans toutes les commandes graphiques où l'option couleur est permise, vous pouvez choisir une valeur pour la source couleur de 0 à 3,

correspondant respectivement au fond, au caractères, à la multi-couleur 1 et à la multi-couleur 2.).

---

---

## MONITOR

Syntaxe :  
**MONITOR**

Cette commande vous fait passer du BASIC au moniteur langage machine résident. Le moniteur est utilisé pour développer, mettre au point et exécuter les programmes en langage machine, plus facilement qu'en BASIC. Voyez la section concernant les commandes du moniteur pour plus d'informations. (Quand vous êtes dans le moniteur, tapez X et appuyez sur            pour revenir en BASIC.)

---

## NEXT

**NEXT** [*variable*,...,*variable*]

L'instruction NEXT est toujours utilisée en liaison avec l'instruction FOR. Lorsque le programme atteint une instruction NEXT, il revient vers l'instruction FOR et contrôle la variable boucle (voyez l'instruction FOR pour plus de détails). Si la boucle est finie, le Plus/4 exécute l'instruction suivant NEXT. L'instruction NEXT peut être suivie d'un nom de variable ou d'une liste de noms de variable séparés par des virgules. S'il n'y a pas de noms mentionnés, NEXT fait référence à la dernière boucle démarrée. Si plusieurs variables sont indiquées, elles sont prises en compte de gauche à droite.

Exemple :

```
10 FOR L=1 TO 10:NEXT
20 FOR L=1 TO 10:NEXT L
30 FOR L=1 TO 10:FOR M=1 TO 10:NEXT M,L
```

---

---

## ON

Syntaxe :

**ON** *expression*<GOTO/GOSUB> *ligne n°1*[,*ligne n°2*,...]

Cette commande transforme les commandes GOTO et GOSUB en versions spéciales de l'instruction IF. ON est suivi d'une formule, puis GOSUB ou GOTO et d'une liste de numéros de ligne séparés par des virgules. Si le résultat de la formule est 1, la première ligne de la

---

---

---

---

liste est exécutée. Si le résultat est 2, la seconde ligne est exécutée et ainsi de suite. Si le résultat est zéro ou supérieur au nombre de lignes de la liste, la prochaine ligne exécutée sera celle immédiatement supérieure à l'instruction ON. Si le nombre est négatif, vous aurez le message d'erreur: ILLEGAL QUANTITY ERROR.

Exemple:

```
10 INPUT X:IF X<0 THEN 10
20 ON X GOTO 50,30,30,70
25 PRINT "TROUVE":GOTO 10
30 PRINT "TROP GRAND":GOTO 10
50 PRINT "TROP PETIT":GOTO 10
70 END
```

(si X=1, ON envoie sur le premier numéro de la liste, si X=2, sur le deuxième numéro (30))

---

## OPEN

Syntaxe :

**OPEN** *fichier n<sup>o</sup>,unité n<sup>q</sup>,adresse secondaire*],[*"nom de fichier, type,mode"*]

L'instruction OPEN permet au Plus/4 d'accéder à des périphériques, tels que le lecteur-enregistreur de cassettes, le lecteur de disquettes, l'imprimante ou même l'écran. OPEN est suivi d'un nombre (0-255) le numéro de fichier logique, qui est le nombre auquel toutes les instructions qui suivent se réfèrent. Il y a ensuite un second numéro appelé numéro de périphérique. Le numéro 0 correspond au clavier du Plus/4, 3 à l'écran, 1 au lecteur-enregistreur de cassettes, 4 à l'imprimante et 8 au lecteur de disquettes. Un 0 peut être inclus devant le numéro de périphérique (08 pour 8), ce qui est équivalent pour le Plus/4. Il est souvent intéressant d'utiliser le même numéro pour le fichier logique et le périphérique, parce qu'il est plus facile de s'en souvenir. Après le numéro de périphérique, vous pouvez avoir un troisième numéro, appelé adresse secondaire. Dans le cas du Datassette, ce numéro est 0 pour la lecture, 1 pour l'écriture et 2 pour l'écriture avec un repère de fin de fichier. Dans le cas du lecteur de disquettes, ce numéro fait référence au numéro de canal. Pour l'imprimante, l'adresse secondaire est utilisée pour indiquer le mode d'impression. Voyez le "manuel de référence du programmeur" pour plus d'information sur les adresses secondaires. Vous pouvez aussi avoir après le troisième numéro une chaîne de caractères, qui peut être une commande pour le lecteur de disquettes ou le nom d'un fichier. Le type et le mode ne font référence qu'aux fichiers sur disquettes (les types de fichier sont: prg, seq, rel et usr; les modes: READ et WRITE.).

---

---

Exemples :

10 **OPEN 3,3** (Ouvre l'écran comme un périphérique)

10 **OPEN 1,0** (Ouvre le clavier comme un périphérique)

20 **OPEN 1,1,0,"point"** (Ouvre sur la cassette un fichier en lecture dont le nom est "point")

**OPEN 4,4** (Ouvre un canal pour utiliser l'imprimante)

**OPEN 15,8,15** (Ouvre le canal de commande du lecteur de disquettes)

5 **OPEN 8,8,12,"test,SEQ,WRITE"**

(crée un fichier séquentiel en écriture sur disquette)

Voyez aussi: **CLOSE**, **CMD**, **GET#**, **INPUT#** et **PRINT#**, et les variables système **ST**, **DS** et **DS\$**.

---

## **PAINT**

Syntaxe :

**PAINT** [*source couleur*],[*a,b*],[*mode*]

Source couleur .....(0-3); par défaut 1, la couleur des caractères.  
a,b .....cordonnées de départ, la valeur par défaut est le PC.

Mode .....0=colore la surface définie avec la source couleur.  
1=colore la surface définie avec une source couleur différente du fond.

La commande **PAINT** vous permet de colorer une surface. Elle colore la surface du point de départ jusqu'à ce qu'elle rencontre une limite de même couleur (ou de tout autre couleur que celle du fond, ceci dépendant du mode choisi).

La position finale du PC est la position de départ (a,b).

NOTE : Si le point de départ est de la même couleur que celle que vous avez choisie (ou tout autre couleur que celle du fond lorsque vous êtes en mode 1), il n'y a aucun changement.
---

Exemple :

10 **CIRCLE, 160, 100, 65, 50** (trace un cercle)



20 **PAINT, 160, 100** (colore le cercle)



---

---

## POKE

Syntaxe :

**POKE** *adresse, valeur*

La commande POKE vous permet de changer une valeur dans la mémoire RAM du Plus/4 et de modifier les registres d'entrée/sortie du Plus/4. POKE est toujours suivi de deux nombres ou formules. Le premier nombre correspond à une adresse mémoire du Plus/4. C'est une valeur comprise entre 0 et 65535. Le second nombre a une valeur comprise entre 0 et 255, qui sera mise à l'adresse précédente, remplaçant la valeur auparavant mémorisée.

Exemple :

**10 POKE 28000,8** (met valeur 8 à l'adresse 28000)

**20 POKE 28\*1000,27** (met la valeur 27 en 28000)

NOTE: Voyez la fonction PEEK.

---

## PRINT

Syntaxe :

**PRINT** *liste*

L'instruction PRINT est la plus importante des instructions de sortie du BASIC. Bien que l'instruction PRINT soit la première apprise par la plupart des gens, il y a beaucoup de subtilités à connaître pour bien la maîtriser. PRINT peut être suivi de:

Chaîne de caractères entre guillemets ("texte")

Noms de variables (A, B, A\$, X\$)

Fonctions (SIN(23), ABS(33))

Signes de ponctuation (:,)

Les caractères entre guillemets sont imprimés exactement comme ils apparaissent. Les noms de variables utilisés avec l'instruction PRINT affichent la valeur qu'elles contiennent (nombres ou chaînes de caractères) de même pour les fonctions. Les signes de ponctuation sont utilisés pour aider le formatage des données à l'écran. Les virgules divisent l'écran en 4 colonnes de données, alors que les points-virgules ne laissent aucun espace. Ces signes peuvent être utilisés comme dernier symbole de la ligne, alors la prochaine instruction PRINT commencera l'impression à l'endroit d'arrêt précédent.

---

---

Exemples :

**RESULTAT**

```
10 PRINT "hello" hello
20 A$="ici":PRINT"hello,"A$ hello,ici
30 A=4:B=2:PRINT A+B 6
40 J=41:PRINT J::PRINT J-1 41 40
50 C=A+B:D=A-B:PRINT A;B;C,D 4 2 6 2
```

Voyez aussi les fonctions: POS(),SPC() et TAB()).

---

## **PRINT #**

Syntaxe :

**PRINT#** *fichier n°*, *liste*

Il y a peu de différences entre cette instruction et PRINT. PRINT# est suivi d'un nombre qui est le numéro de fichier logique se référant à un périphérique ou à un fichier précédemment ouvert (OPENed). Ce nombre est suivi par une virgule et par la liste des éléments à imprimer. La virgule et le point-virgule ont le même effet qu'avec l'instruction PRINT. Certains périphériques ne peuvent pas utiliser les fonctions TAB et SPC.

Exemple :

```
100 PRINT#1,"salut",A$,B$
```

---

## **PRINT USING**

Syntaxe :

**PRINT** [*fichier n°*] **USING** *format;liste;*

Cette instruction vous permet de définir un format de sortie pour les chaînes de caractères et les valeurs numériques, que ce soit pour l'écran, l'imprimante ou d'autres périphériques. Mettez entre guillemets le format souhaité. Ajoutez ensuite un point-virgule et la liste des éléments que vous voulez imprimer au format défini. La liste peut être composée de variables ou de constantes.

Exemple :

```
5 X=32:Y=100.23:A$="chat"
10 PRINT USING "$##.##":13.25,X,Y
20 PRINT USING "###>##";"cbm",A$
```

A l'exécution, vous avez : pour la 10

```
$13.25 $32.00 $*****
```

(les étoiles sont imprimées à la place de la valeur de Y, parce que Y

---

---

---

---

a 5 chiffres, ce qui n'est pas conforme au format.)

pour la 20

cbm chat

(laisse 3 espaces avant d'imprimer cbm comme défini dans le format)

CARACTERE	NUMERIQUE	CHAINE
Dièse (#)	X	X
Plus (+)	X	
Moins (-)	X	
Point décimal (.)	X	
Virgule (,)	X	
Dollar (\$)	X	
Quatre flèches (↑↑↑↑)	X	
Signe égal (=)		X
Signe supérieur (>)		X

Le dièse réserve une place dans le format de sortie pour un simple caractère. Si l'élément de donnée contient plus de caractères que vous n'avez de # dans votre format, vous aurez :

- Dans le cas de données numériques, le champs est rempli en entier d'astérisques (\*). Aucun nombre n'est imprimé.

Exemple :

**10 PRINT USING"#####", X**

Pour ces valeurs de X, vous aurez les formats affichés :

X=12.34 12

X=567.89 568

X=123456 \*\*\*\*\*

Dans le cas de données sous forme de chaîne de caractères, la donnée est tronquée à la limite du champs. Il y aura autant de caractères imprimés que de signes # dans le format. La chaîne est coupée à droite.

Les signes + et - peuvent être utilisés en première ou dernière position du format mais pas aux deux positions. Le signe + est imprimé si le nombre est positif, le signe - est imprimé si le nombre est négatif.

---

---

Si vous utilisez un signe - avec un nombre positif, un blanc s'affiche à la place du signe.

Si vous n'utilisez pas de signes + ou - dans votre format pour une donnée numérique, un signe - est imprimé avant le premier chiffre ou avant le symbole \$ si le nombre est négatif et aucun signe n'est imprimé si le nombre est positif. Cela signifie que vous pouvez mettre un caractère de plus si le nombre est positif. S'il y a trop de chiffres par rapport au format spécifié par les # et les signes +/-, vous aurez un débordement concrétisé par des astérisques (\*).

Le symbole point (.) définit la position du point décimal dans le nombre. Vous ne pouvez avoir qu'un seul point décimal dans le format. Si vous ne spécifiez pas de point décimal dans votre format, la valeur est arrondie à l'entier le plus proche et affichée sans chiffre après la virgule.

Quand vous spécifiez un point décimal, le nombre de chiffres précédant le point décimal (incluant le signe - si la valeur est négative) ne doit pas excéder le nombre de # avant le point décimal. Si vous avez trop de chiffres, vous aurez un débordement concrétisé par des \*.

Une virgule (,) réserve une place pour la virgule dans un format numérique. La position des virgules dans le format indique les emplacements où seront imprimées ces virgules. Seules les virgules à l'intérieur d'un nombre sont affichées. Les virgules à gauche du premier chiffre n'apparaissent pas. Un # au moins doit toujours précéder la première virgule d'un format.

Si vous indiquez des virgules dans votre format avec un nombre négatif, le signe - est imprimé en premier caractère même si c'est la position d'une virgule.

Exemples :

<b>FORMAT</b>	<b>EXPRESSION</b>	<b>RESULTAT</b>	<b>COMMENTAIRES</b>
##.#+	-.01	0.01-	Le 1er 0 est ajouté
##.#-	1	1.0	Le dernier 0 est ajouté
####	-100.5	-101	Arrondi à l'entier supérieur
####	-1000	****	Débordement: 4 chiffres et ne rentrent pas dans format
###.	10	10.	Point décimal ajouté
#\$##	1	\$1	Ajoute \$

Un symbole \$ indique qu'un signe \$ sera imprimé avec le nombre. Si vous voulez que le signe \$ flotte (toujours placé avant le nombre sans espace), vous devez mettre au moins un # avant le signe \$. Si vous spécifiez le signe \$ sans le faire précéder de #, le signe \$ est

---

---

imprimé à sa position exacte du format. Si vous indiquez des virgules, des signes + ou - dans le format avec un signe \$, votre programme affichera la virgule ou les signes + ou - avant le signe \$.

Le symbole 4 flèches (̂) est utilisé pour indiquer qu'un nombre doit être imprimé en format E+. Vous pouvez utiliser des # avec les 4 flèches pour déterminer la largeur du format. Le ^ peut être placé avant ou après les # dans le format.

Vous devez spécifier les 4 flèches (̂) quand vous voulez imprimer un nombre en format E- (notation scientifique). Si vous donnez plus d'une et moins de quatre flèches, vous aurez une erreur de syntaxe. Si vous donnez plus de quatre flèches, seules les quatre premières seront utilisées et la cinquième sera interprétée comme un symbole transparent.

Le signe = est utilisé pour centrer une chaîne de caractères dans un format. Vous indiquez la largeur du format avec le nombre de caractères (# et =). Si la chaîne de caractères est moins longue que le format spécifié, celle-ci est centrée. Si la chaîne de caractères est plus longue que le format spécifié; les caractères de droite sont coupés et celle-ci correspond alors au format. Le signe > est utilisé pour justifier à droite une chaîne de caractères. Vous indiquez la largeur du format avec les # et >. Si la chaîne de caractères est moins longue que le format, celle-ci est justifiée à droite. Si elle est plus longue, les caractères de droite sont coupés et celle-ci correspond au format.

---

## **PUDEF**

Syntaxe :

**PUDEF** "1 à 4 caractères"

La commande PUDEF vous permet de redéfinir les quatre symboles utilisés dans l'instruction PRINT USING. Vous pouvez remplacer les blancs, virgules, points décimaux et \$ par n'importe quel autre caractère en plaçant ce nouveau caractère en position correcte dans PUDEF.

Position 1 : le caractère transparent. La valeur par défaut est le blanc. Placez-y un nouveau caractère quand vous voulez remplacer le blanc.

Position 2 : la virgule.

Position 3 : le point décimal.

Position 4 : le signe \$.

---

---

---

---

Exemples :

10 PUDEF"\*" (imprime \* à la place des blancs)

20 PUDEF"@" (imprime @ à la place des virgules)

30 PUDEF",." (imprime un point à la place des virgules et une virgule à la place des points décimaux)

40 PUDEF",.F" (imprime F à la place de \$, un point à la place des virgules et une virgule à la place des point décimaux)

---

## READ

Syntaxe :

**READ** *liste de variables*

READ est utilisé pour assigner les informations provenant des instructions DATA à des variables, de telle manière que les informations soient utilisées. La liste de variables de l'instruction READ peut contenir des chaînes de caractères et des nombres. Evitez à l'instruction READ de lire une chaîne de caractères si elle attend un nombre, ce qui produirait une erreur.

Exemple :

**READ A\$,G\$**

---

## REM

Syntaxe :

**REM** message

L'instruction REM (REMarque) est une explication pour celui qui lit une liste du programme. Cette instruction peut expliquer une partie du programme ou donner des informations supplémentaires. Les instructions REM n'affectent en aucune manière le fonctionnement du programme, mais elles en ralentissent l'exécution, parce qu'elles augmentent la taille du programme. REM peut être suivi de n'importe quel caractère, mais l'utilisation de caractères graphiques donne d'étranges résultats.

Exemple :

10 NEXT X:REM CETTE LIGNE EST INUTILE

---

---

## RESTORE

Syntaxe :

**RESTORE** [*ligne n°*]

Lorsque cette instruction est exécutée dans un programme, le pointeur de l'instruction DATA est réinitialisé sur le premier élément de la liste. Ceci vous permet de relire (re-READ) les informations. Si un numéro de ligne suit RESTORE, le pointeur se place sur la ligne spécifiée. Sinon, le pointeur se place sur la première ligne du programme comportant une instruction DATA.

Exemple :

**RESTORE 200**

---

## RESUME

Syntaxe :

**RESUME** [*ligne n°NEXT*]

Cette instruction est utilisée pour reprendre l'exécution d'un programme après détection d'une erreur (TRAP). RESUME sans argument reprend l'exécution du programme sur la ligne où il y a eu l'erreur. RESUME NEXT reprend l'exécution du programme sur la ligne suivant l'erreur; RESUME suivi du numéro de ligne reprend l'exécution du programme sur la ligne spécifiée.

---

## RETURN

Syntaxe :

**RETURN**

Cette instruction est toujours utilisée avec GOSUB. Lorsque le programme rencontre une instruction RETURN, il passe immédiatement à l'instruction suivant la dernière commande GOSUB exécutée. Si aucune commande GOSUB n'a été générée auparavant, un message d'erreur apparaît: RETURN WITHOUT GOSUB ERROR et le programme s'arrête.

---

## SCALE

Syntaxe :

**SCALE**<1/0>

La commande SCALE peut changer en modes haute résolution et multi-couleur l'échelle des points de l'écran. Exemple :

---

---

## SCALE 1

met en fonction ce processus. Les coordonnées peuvent être échelonnées de 0 à 1023 en X et Y par rapport aux valeurs normales, qui sont :

Mode multi-couleur ..... X= 0 à 159 Y= 0 à 199

Mode haute résolution ..... X= 0 à 319 Y= 0 à 199

Cette commande est mise hors fonction avec SCALE 0.

---

## SCNCLR

Syntaxe :

**SCNCLR**

Vide l'écran, que ce soit l'écran graphique, l'écran texte ou les deux.

---

## SOUND

Syntaxe :

**SOUND** voix n°, fréquence, durée

Cette instruction produit un son utilisant une des trois voix avec une fréquence de 0 à 123 et une durée allant de 0 à 65353 60ème de seconde.

V	VOIX
---	------

- 
- |   |                           |
|---|---------------------------|
| 1 | Voix 1 (son)              |
| 2 | Voix 2 (son)              |
| 3 | Voix 2 (silence et bruit) |

Si un son pour une voix donnée est envoyé, et que le son précédent pour la même voix est encore en fonction, BASIC attend la fin du son précédent pour démarrer le nouveau. L'instruction SOUND avec une durée de zéro, est un cas spécial. Il oblige BASIC à arrêter immédiatement le son en train d'être émis par cette voix, sans tenir compte du temps restant imparti à l'instruction SOUND précédente. Voir la table des notes de musique (section 11) pour les valeurs des fréquences correspondant aux notes réelles.

Exemple :

**SOUND 2, 800, 360**

(joue une note utilisant la Voix 2 avec une fréquence de 800 pour une minute)

---

---

## SSHAPE/GSHAPE

Les commandes SSHAPE et GSHAPE sont utilisées pour sauvegarder et restituer des zones rectangulaires d'écran multi-couleur ou haute résolution avec des variables chaînes de caractères. La commande pour sauvegarder une zone est:

Syntaxe :

**SSHAPE** *variable*, *a1*, *b1* [,*a2*,*b2*]

*variable* : nom de la chaîne de caractères pour sauvegarder les données.

*a1*,*b1* : coordonnées du premier coin.

*a2*,*b2* : coordonnées du point opposé (par défaut le PC).

Parce que les chaînes sont limitées à 255 caractères en BASIC, la taille de la zone que vous voulez sauvegarder est limitée. Vous pouvez calculer la taille de la chaîne de caractères avec les formules suivantes:

$L(mcm) = \text{INT}((\text{ABS}(a1-a2)+1)/4 + .99) * (\text{ABS}(b1-b2)+1) + 4$

$L(h.r) = \text{INT}((\text{ABS}(a1-a2)+1)/8 + .99) * (\text{ABS}(b1-b2)+1) + 4$

(*mcm*)=mode multi-couleur;(h.r)=haute résolution

La forme est sauvegardée ligne par ligne. Les quatre derniers bytes de la chaîne contiennent les longueurs de colonne et de ligne moins un (par exemple:  $\text{ABS}(a1-a2)$ ) en format poids lourd/poids faible (si vous êtes en mode SCALE, vous divisez les longueurs par 3.2 (X) et par 5.12 (Y)).

La commande pour restituer une forme sauvegardée est la suivante :

Syntaxe :

**GSHAPE** *variable* [, [*a*,*b*] [,*mode*]]

*variable* : chaîne de caractères contenant la forme.

*a*,*b* : coordonnées du point supérieur gauche à partir duquel sera dessiné la forme (par défaut: PC).

*mode* : mode de placement:

0 placement identique à l'ancien

1 placement en inverse

2 placement avec une comparaison logique OR

3 placement avec une comparaison logique AND

4 placement avec une comparaison logique XOR

---

---

Exemples :

**SSHape** "bateau",0,0

(sauvegarde la forme de la zone écran à partie du coin supérieur gauche jusqu'au curseur, sous le nom "bateau".)

**GSHape** "bateau",,,1

(affiche la forme "bateau" inversée avec le coin supérieur gauche positionné à l'endroit du curseur.)

---

## STOP

Syntaxe :

**STOP**

Cette instruction arrête l'exécution du programme. Le message **BREAK IN LINE n°** s'affiche, le numéro correspond au numéro de la ligne contenant **STOP**. Le programme peut être redémarré en utilisant la commande **CONT**. **STOP** est normalement utilisé dans la mise au point d'un programme.

---

## SYS

Syntaxe :

**SYS** *adresse*

L'instruction **SYS** est suivie d'un nombre décimal ou d'une variable numérique comprise entre 0 et 65535. Le programme commencera alors l'exécution du programme en langage machine commençant à partir de cet emplacement mémoire. Cette commande est similaire à la fonction **USR**, mais ne permet pas le passage de paramètres. Voyez le "Guide de référence du programmeur du Plus/4" pour plus d'informations sur le langage machine.

---

## TRAP

Syntaxe :

**TRAP** [*ligne n°*]

Quand elle est en fonction, l'instruction **TRAP** détourne toutes les routines d'erreur (y compris la touche **STOP**) excepté "UNDEF'D STATEMENT ERROR". Dès la détection d'erreur, un repère est mis en place et l'exécution est transférée au numéro de ligne indiqué dans **TRAP**. Le numéro de la ligne dans laquelle il y avait l'erreur peut être consulté en utilisant la variable système **EL**. Le numéro du message d'erreur est contenu dans la variable système **ER**. La

---

---

fonction ERR\$(ER) vous donne le message d'erreur correspondant au numéro ER.

NOTE: Une erreur dans la routine TRAP ne peut pas être détournée par l'instruction TRAP. L'instruction RESUME peut être utilisée pour reprendre l'exécution du programme. TRAP sans numéro de ligne met l'instruction hors fonction.

---

## TRON

Syntaxe :  
**TRON**

L'instruction TRON est utilisée pour mettre au point les programmes. Elle met en fonction le mode trace. Quand vous êtes en mode trace, pour chaque instruction exécutée, le numéro de ligne de cette instruction est imprimé à l'écran.

---

## TROFF

Syntaxe :  
**TROFF**

Cette instruction met hors fonction le mode trace.

---

## VOL

Syntaxe :  
**VOL** *niveau de volume*

Cette commande permet de modifier le VOLUME de la commande SOUND. La valeur de cette commande varie de 0 à 8, où 8 est le volume maximum et 0 le volume minimum. VOL: influe sur les deux voix.

---

## WAIT

Syntaxe :  
**WAIT** *adresse,valeur 1[,valeur 2]*

L'instruction WAIT est utilisée pour arrêter le programme jusqu'à ce que le contenu d'un emplacement mémoire se modifie d'une façon spécifique. L'adresse doit être comprise entre 0 et 65535. Valeur 1 et valeur 2 sont comprises entre 0 et 255.

Le contenu de l'emplacement mémoire est tout d'abord comparé logiquement avec la valeur 2 (si présente) avec un OU exclusif et ensuite comparé logiquement avec la valeur 2 avec un AND. Si le résultat est zéro, le programme continue les comparaisons. Quand le résultat est différent de zéro, le programme continue sur l'instruction suivante.

---

---

## POUR EN SAVOIR PLUS SUR LES INSTRUCTIONS GRAPHIQUES

Il y a quelques concepts qui s'appliquent à toutes les instructions graphiques. Nous commençons par le concept de curseur point (Pixel Cursor, PC). Le PC est similaire au curseur du mode texte. C'est la position où le prochain point va apparaître. Contrairement au curseur texte, le PC est invisible. Toutes les commandes de dessin (DRAW) utilisent le PC. De plus, la commande LOCATE vous permet de repositionner le PC sans dessiner.

Chaque fois que vous utilisez les coordonnées X,Y dans une commande de dessin (DRAW), vous pouvez utiliser à la place des coordonnées relatives. Les coordonnées relatives sont basées sur la valeur du PC. Pour utiliser les coordonnées relatives, placez juste un + ou un - devant chaque coordonnée. Le signe + avant la valeur X déplace le PC vers la droite et le signe - vers la gauche; de même, le signe - avant la valeur Y déplace le PC vers le haut et le signe + vers le bas. Par exemple:

**LOCATE +100,-25** (déplace le PC vers la droite de 100 Pixels et vers le haut de 25.)

**DRAW 1,+10,+10 TO 100,100** (dessine une ligne à partir du point situé 10 Pixels à droite et 10 Pixels en bas du PC jusqu'au point de coordonnées absolues 100,100.)

Vous pouvez aussi spécifier une distance et un angle relatif par rapport au PC, en séparant ces deux paramètres par un point-virgule.

Exemple :

**LOCATE 50;45** (déplace le PC d'une distance de 50 Pixels avec un angle de 45 degrés.)

---

---

## FONCTIONS

### FONCTIONS NUMERIQUES

Les fonctions numériques sont classifiées comme telles parce qu'elles donnent des résultats numériques. Les fonctions qu'elles exécutent vont du calcul de fonctions mathématiques à la définition d'un point sur l'écran. Les fonctions numériques ont la forme suivante :

#### **FONCTION**(argument)

où l'argument peut être une valeur numérique, une variable ou une chaîne de caractères.

---

#### **ABS(X)** (valeur absolue)

ABS fournit la valeur absolue du nombre, sans son signe (+ ou -). La réponse est toujours positive.

---

#### **ASC(X\$)**

Cette fonction fournit le code ASCII (nombre) du premier caractère de la chaîne X\$.

---

#### **ATN(X)** (arctangente)

Elle fournit l'angle dont la tangente est X, mesuré en radians.

---

#### **COS(X)** (cosinus)

Elle fournit la valeur du cosinus de X, où X est un nombre mesuré en radians.

---

#### **DEC**(chaîne de caractères hexadécimaux)

Elle fournit la valeur décimale d'une chaîne de caractères hexadécimaux (0<chaîne de caractères hexadécimaux<FFFF).

---

Exemple :

N=DEC("F4")

---

#### **EXP(X)**

Elle fournit la valeur de la constante mathématique e (2.71828183) élevée à la puissance de X.

---

#### **FNxx(X)**

Elle fournit la valeur de la fonction xx définie par l'utilisateur créée dans une instruction DEF FNxx.

---

#### **INSTR**(chaîne de caractère1,chaîne de caractère2,[position-début])

Elle fournit la position de la chaîne de caractères2 dans la chaîne de caractères1 à la [position-début] ou après elle. La position par défaut de la position-début est le début de la chaîne de caractères2. Si rien ne correspond, une valeur 0 est donnée.

---

---

---

---

Exemple :

PRINT INSTR("le chat dans le chapeau","chat")

le résultat est 4, parce que "chat" commence au 4ème caractère de la chaîne de caractères1.

---

### **INT(X)** (entier)

Elle fournit la valeur entière de X, c'est-à-dire avec toutes les positions décimales à droite du point décimal éliminées. Le résultat est toujours inférieur ou égal à X. Par conséquent, tous les nombres négatifs avec des positions décimales, deviennent des entiers inférieurs à leur valeur courante (par exemple, INT(-4.5) = -5).

Si la fonction INT doit être utilisée pour arrondir, sa forme est INT(X +/- .5).

Exemple :

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

Arrondit au centime supérieur.

---

### **JOY(n)**

Quand n=1 : position de la manette de jeux n°1.

Quand n=2 : position de la manette de jeux n°2.

Une valeur de 128 ou plus signifie que le bouton de mise à feu est abaissé. La direction est indiquée comme suit:

feu=128 +		haut		
		1		
	8		2	
gauche	7		3	droite
	6		4	
		5		
		bas		

Exemple :

JOY(2)=135

La manette de jeux n°2 tire vers la gauche.

---

### **LOG(X)** (logarithme)

Elle fournit le log naturel de X. Le log naturel est le log en base e (voir EXP(X)). Pour convertir en log base 10, diviser par LOG(10).

---

---

---

---

**PEEK(X)**

Cette fonction donne les contenus de l'emplacement X de la mémoire, où X est compris entre 0 et 65535, en donnant un résultat compris entre 0 et 255. PEEK est fréquemment utilisé en liaison avec l'instruction POKE.

---

**RCLR(N)**

Elle fournit la couleur actuellement affectée à la source N ( $0 < N < 4$ ).

0=fond

1=caractère

2=multi-couleur 1

3=multi-couleur 2

4=cadre

---

**RDOT(N)**

Elle fournit la position du curseur point (PC) en X et Y.

N = 0 pour X

1 pour Y

2 pour la source couleur.

---

**RGR(X)**

Elle retourne la valeur du mode graphique (X est un argument factice).

---

**RLUM(N)**

Elle retourne le niveau de brillance affecté à la source N.

---

**RND(X)** (nombre aléatoire)

Cette fonction fournit un nombre aléatoire compris entre 0 et 1. Elle est utile dans les jeux, pour simuler un jet de dés et autres éléments de chance et dans les applications statistiques. Le premier nombre aléatoire doit être généré par la formule

RND(-TI), pour commencer avec un nombre différent à chaque fois.

Après quoi, X doit être 1 ou n'importe quel nombre positif (X représente l'origine ou la base du nombre aléatoire). Si X est nul, la fonction RND est réinitialisée à partir de la valeur de l'horloge interne, chaque fois que la fonction est utilisée. Si X est négatif, une séquence de nombres aléatoires est générée avec l'origine X. L'utilisation du même nombre négatif pour X entraîne la même séquence de nombres aléatoires. Une valeur positive de X vous donne un nombre aléatoire basé sur la séquence précédemment choisie.

Pour simuler un jet de dés, utiliser la formule  $\text{INT}(\text{RND}(1)*6+1)$ . Le nombre aléatoire compris entre 0 et 1 est multiplié par 6, ce qui donne un nombre compris entre 0 et 6 ( $>0$  et  $<6$ ). Ensuite, on ajoute 1 pour obtenir un nombre compris entre 1 et 7. La fonction INT élimine les chiffres après la virgule, donnant un chiffre de 1 et 6.

---

---

---

---

Pour simuler le jet de deux dés, ajouter deux des nombres obtenus avec la formule précédente.

Exemple :

<b>100 X=INT(RND(1)*6)+INT(RND(1)*6)+2</b>	Simule deux dés
<b>100 X=INT(RND(1)*1000)+1</b>	Nombre de 1 à 1000
<b>100 X=INT(RND(1)*150)+100</b>	Nombre de 100 à 249

---

**SGN(X)** (signe)

Cette fonction fournit le signe positif, négatif ou nul de X. Le résultat sera +1 s'il est positif, 0 s'il est nul et -1 s'il est négatif.

---

**SIN(X)** (sinus)

SIN(X) est la fonction trigonométrique sinus. Le résultat sera sinus(X) où X est un angle exprimé en radians.

---

**SQR(X)** (racine carrée)

Cette fonction fournit la racine carrée de X, où X est un nombre positif ou 0. Si X est négatif, le message suivant est affiché: ILLEGAL QUANTITY ERROR.

---

**TAN(X)** (tangente)

Elle fournit la tangente de X, où X est un angle en radians.

---

**USR(X)**

Lorsque cette fonction est utilisée, le programme saute vers un programme en langage machine dont le point de départ est contenu en adresse mémoire 1281 ou 1282. Le paramètre X est transmis au programme en langage machine par l'accumulateur en virgule flottante. Un autre nombre est retourné au programme BASIC par la variable de départ. C'est-à-dire, vous pouvez échanger des variables entre le BASIC et le langage machine. Voyez le "Guide de référence du programmeur du Plus/4 pour plus d'informations sur cette fonction et sur la programmation en langage machine.

---

**VAL(X\$)**

Cette fonction convertit une chaîne de caractères X\$ en un nombre, c'est la fonction inverse à STR\$. La chaîne de caractères est traitée de la gauche vers la droite, pour autant de caractères que le permet le format numérique. Si le Plus/4 trouve des caractères invalides, seule la partie de la chaîne jusqu'à ce caractère est convertie.

---

Exemple :

<b>10 X=VAL("123.456")</b>	(X=123.456)
<b>10 X=VAL("3E03")</b>	(X=3000)
<b>10 X=VAL("12A13B")</b>	(X=12)
<b>10 X=VAL("RIUO17")</b>	(X=0)
<b>10 X=VAL("-1.23.23.23")</b>	(X=-1.23)

---

---

---

---

## FONCTIONS DE CHAINES DE CARACTERES

Les fonctions de chaînes de caractères diffèrent des fonctions numériques parce qu'elles retournent des lettres, des graphiques ou des nombres dans une chaîne de caractères (définis par des guillemets) à la place de nombres.

---

### **CHR\$(X)**

Cette fonction retourne un caractère dont le code ASCII est X.

---

### **ERR\$(N)**

Cette fonction retourne une chaîne de caractères décrivant l'erreur numéro 1 (voir TRAP).

---

### **HEX\$(N)**

Cette fonction retourne une chaîne de caractères contenant la représentation hexadécimale de la valeur N ( $0 < N < 65535$ ).

---

### **LEFT\$(X\$,X)**

Elle fournit une chaîne de caractères contenant les X premiers caractères de la chaîne X\$.

---

### **LEN(X\$)**

Elle donne le nombre de caractères (y compris les espaces et autres symboles) de la chaîne X\$.

---

### **MID\$(X\$,S,X)**

Cette fonction donne une chaîne de X caractères, en partant de la position du Sème caractère de X\$. MID\$ peut aussi être utilisé à gauche d'une instruction d'affectation comme une pseudo variable aussi bien qu'une fonction. MID\$(variable, position de départ, longueur)=chaîne source.

Cette fonction modifie la chaîne de caractères, en plaçant une chaîne source à la position de départ pour la longueur spécifiée. La longueur par défaut est la longueur de la chaîne de caractères et une erreur sera signalée si la valeur de la position de départ plus la longueur de la chaîne source est plus grande que la longueur de la chaîne d'origine.

Exemple :

```
10 A$="la puce dans le chapeau":  
20 PRINT A$  
30 MID$(A$,4,4)="chat"  
40 PRINT A$
```

---

---

**RIGHT\$(X\$,X)**

Fournit les X caractères les plus à droite de X\$.

---

**STR\$(X)**

Elle fournit une chaîne de caractères qui est identique à la version affichée en X.

Exemple :

**A\$=STR\$(X)**

---

## **AUTRES FONCTIONS**

**FRE(X)**

Cette fonction fournit le nombre d'octets inutilisés disponibles dans la mémoire, quelle que soit la valeur de X.

---

**POS(X)**

Cette fonction fournit le numéro de la colonne (0-79) sur laquelle la prochaine instruction PRINT commencera sur l'écran. X peut avoir une valeur quelconque.

---

**SPC(X)**

Cette fonction est utilisée dans une instruction PRINT pour faire sauter X espaces en avant. X peut avoir toute valeur entre 0 et 255.

---

**PI**

Le symbole pi, quand il est utilisé dans une équation, a la valeur 3.14159265.

---

---

## VARIABLES ET OPERATEURS

Le Plus/4 utilise trois types de variables en BASIC. Il s'agit des variables numériques normales, numériques entières et de variables chaînes de caractères (alphanumériques).

### VARIABLES

Les **variables numériques** normales, aussi appelées **variables virgule flottante**, peuvent avoir toutes valeurs de 10 puissance -38 à 10 puissance +38, avec neuf chiffres significatifs. Quand un nombre a plus de 9 chiffres, comme N-10 ou N+10, l'ordinateur l'affiche en notation scientifique, avec le nombre normalisé à 1 chiffre et 8 décimales, suivi par la lettre E et la puissance de 10 par laquelle le nombre est multiplié. Par exemple, le nombre 12345678901 est représenté par 1.23456789E+10.

Les **variables entières** peuvent être utilisées pour de nombres entre +32767 et -32768 sans fraction. Une variable entière est un nombre comme 5, 10 ou -100. Des entiers prennent moins de place que des variables à virgule flottante, particulièrement quand elles sont utilisées dans un tableau.

Les **variables chaîne de caractères** sont celles qui sont utilisées pour des données caractères, qui peuvent contenir des nombres, des lettres et tout autre caractère que le Plus/4 peut générer. Un exemple d'une variable chaîne de caractères est "Plus/4".

### NOMS DE VARIABLES

Les noms de variables peuvent comprendre une seule lettre, une lettre suivie d'un nombre ou deux lettres. Les noms de variables peuvent avoir plus de 2 caractères, mais les deux premiers seulement sont significatifs.

Une variable entière est spécifiée par l'utilisation du signe pourcent (%) après le nom de la variable. Les variables chaînes de caractères possèdent le symbole \$ après leur nom.

Exemples:

Noms de variables numériques : A, A5, BZ.

Noms de variables entières : A%, A5%, BZ%.

Noms de variables chaînes de caractères : A\$, A5\$, BZ\$.

Les tableaux sont des listes de variables portant le même nom, utilisant des nombres supplémentaires pour spécifier un élément du tableau. Les tableaux sont définis en utilisant l'instruction DIM et peuvent contenir des variables à virgule flottante, entières ou chaînes de caractères. Le nom de la variable du tableau est suivi par un jeu de parenthèses () entourant le nombre de variables de la liste.

---

---

Exemples :

A(7), BZ%(11), A\$(87)

Des tableaux peuvent avoir plus d'une dimension. Un tableau à deux dimensions peut être considéré comme ayant des lignes et des colonnes, avec le premier nombre entre parenthèses identifiant la ligne et le second nombre la colonne (comme une grille sur une carte).

Exemples :

A(7,2), BZ%(2,3,4), Z\$(3,2)

## **NOMS DE VARIABLES RESERVES**

Il existe huit noms de variables qui sont réservés au Plus/4 et qui ne peuvent être utilisés pour aucun autre usage. Ces variables sont: DS, DS\$, ER, ERR\$, EL, ST, TI ou TI\$. Vous ne pouvez pas non plus utiliser des mots clés (KEYWORD) tels que TO et IF ou tout autres noms qui contiennent des mots clés, tels que SRUN, RNEW ou XLOAD comme noms de variables.

ST est une variable d'état qui est en rapport avec les opérations d'entrée/sortie (sauf les opérations normales écran/clavier). La valeur de ST dépend des résultats de la dernière opération d'entrée/sortie. Des explications plus détaillées sur ST sont données dans le "Guide de référence du programmeur du Plus/4", mais en général, si la valeur de ST est 0 l'opération est réussie.

TI et TI\$ sont des variables qui sont en rapport avec l'horloge en temps réel incorporée dans le Plus/4. L'horloge du système est remise à l'heure tous les 60èmes de seconde. Elle part à zéro quand le Plus/4 est branché et elle n'est réinitialisée que par la modification de la valeur de TI\$. La variable TI vous donne l'heure en 60èmes de secondes.

TI\$ est une chaîne de caractères qui lit la valeur de l'horloge en temps réel comme une horloge de 24 heures. Les deux premiers caractères contiennent le nombre des heures, les 3ème et 4ème caractères le nombre des minutes et les 5ème et 6ème caractères contiennent le nombre des secondes. Cette variable peut recevoir n'importe quelle valeur numérique et sera remise à l'heure comme une horloge de 24 heures.

Exemple :

TI\$="101530" règle l'horloge sur 10 heures 15 minutes 30 secondes.

---

---

La valeur de l'horloge est effacée lorsque l'ordinateur est arrêté. Elle repart à zéro lorsque le système est remis sous tension et est réinitialisée quand la valeur de l'horloge dépasse 235959 (23 heures, 59 minutes et 59 secondes).

La variable DS lit le canal de commande du lecteur de disquettes et fournit l'état du lecteur. Pour obtenir cette information en texte, faites afficher DS\$. Ces variables d'état sont utilisées après une opération sur disquette, comme DLOAD ou DSAVE, pour savoir pourquoi le voyant du lecteur de disquettes clignote en rouge.

Les variables ER, EL et ERR\$ sont utilisées dans les routines d'erreur. Elles ne sont généralement utiles qu'à l'intérieur d'un programme. ER retourne la dernière erreur rencontrée depuis l'exécution du programme. EL est la ligne où l'erreur est arrivée. ERR\$ est une fonction qui permet à votre programme d'imprimer un des messages d'erreur BASIC. PRINT ERR\$(ER) imprime le message d'erreur numéro ER.

---

---

## OPERATEURS BASIC

Les opérateurs arithmétiques comprennent les signes suivants:

---

- + addition
  - soustraction
  - \* multiplication
  - / division
  - ↑ élévation à une puissance
- 

Sur une ligne contenant plusieurs opérateurs, il existe un ordre déterminé selon lequel les opérations se déroulent toujours. Si plusieurs opérateurs sont utilisés en même temps, l'ordinateur fixe les priorités comme suit: premièrement l'élévation la puissance, ensuite multiplication et division et en dernier lieu addition et soustraction. Si deux opérations ont la même

priorité, les calculs seront alors exécutés de la gauche vers la droite. Vous pouvez modifier l'ordre des opérations en plaçant entre parenthèses le calcul à réaliser en premier lieu. Les opérations entre parenthèses ont lieu avant les autres opérations. Vous devez être sûr que vos équations ont le même nombre de parenthèses gauche que de parenthèses droite ou bien vous aurez un message d'erreur affiché pendant l'exécution de votre programme: SYNTAX ERROR.

Il existe également des opérateurs d'égalité et d'inégalité, appelés opérateurs relationnels. Les opérateurs arithmétiques ont la priorité sur les opérateurs relationnels.

---

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

Finalement, il existe trois opérateurs logiques, avec une priorité inférieure aux opérateurs arithmétiques et relationnels :

AND ET  
OR OU  
NOT NON

Ceux-ci sont très souvent utilisés pour associer les formules multiples dans les instructions IF...THEN. Quand ils sont utilisés avec des opérateurs arithmétiques, ils sont pris en compte en dernier (par exemple, après + et -).

---

---

Exemples :

**IF A=B AND C=D THEN 100** (nécessite que A=B & C=C soient vraies)

**IF A=B OR C=D THEN 100** (nécessite que A=B ou C=D soient vraie)

**A=5:B=4:PRINT A=B** (affiche une valeur 0)

**A=5:B=4:PRINT A>B** (affiche une valeur -1)

**PRINT 123 AND 15:PRINT 5 OR 7** (affiche 11 et 7)

---

---

## SECTION 2

### ABBREVIATIONS BASIC

---

MOT CLE	ABBREVIATION	TYPE
ABS	a SHIFT B	fonction numérique
ASC	a SHIFT S	fonction numérique
ATN	a SHIFT T	fonction numérique
AUTO	a SHIFT U	commande
BACKUP	b SHIFT A	commande
BOX	b SHIFT O	instruction
CHAR	ch SHIFT A	instruction
CHR\$	c SHIFT H	fonction chaîne de caractères
CIRCLE	c SHIFT I	instruction
CLOSE	cl SHIFT O	instruction
CLR	c SHIFT L	instruction
CMD	c SHIFT M	instruction
COLLECT	col SHIFT L	commande
COLOR	co SHIFT L	instruction
CONT	c SHIFT O	commande
COPY	co SHIFT P	commande
COS	aucune	fonction numérique
DATA	d SHIFT A	instruction
DEC	* aucune *	fonction numérique
DEF FN	d SHIFT E	instruction
DELETE	de SHIFT L	commande
DIM	d SHIFT I	instruction
DIRECTORY	di SHIFT R	commande
DLOAD	d SHIFT L	commande
DO	* aucune *	instruction
DRAW	d SHIFT R	instruction
DSAVE	d SHIFT S	commande
END	e SHIFT N	instruction
ERR\$	e SHIFT R	fonction chaîne de caractères
EXP	e SHIFT X	fonction numérique
FOR	f SHIFT O	instruction
FRE	f SHIFT R	fonction numérique
GET	g SHIFT E	instruction
GETKEY	getk SHIFT E	instruction
GET#	* aucune *	instruction
GOSUB	go SHIFT S	instruction
GOTO	g SHIFT O	instruction
GRAPHIC	g SHIFT R	instruction

---

---

MOT CLE	ABBREVIATION	TYPE
GSHAPE	g SHIFT S	instruction
HEADER	he SHIFT A	commande
HEX\$	h SHIFT E	fonction chaîne de caractères
IF...GOTO	* aucune *	instruction
IF...THEN...ELSE	* aucune *	instruction
INPUT	* aucune *	instruction
INPUT#	i SHIFT N	instruction
INSTR	in SHIFT S	fonction numérique
INT	* aucune *	fonction numérique
JOY	j SHIFT O	fonction numérique
KEY	k SHIFT E	commande
LEFT\$	le SHIFT F	fonction chaîne de caractères
LEN	* aucune *	fonction numérique
LET	l SHIFT E	instruction
LIST	l SHIFT l	commande
LOAD	l SHIFT O	commande
LOCATE	lo SHIFT C	instruction
LOG	* aucune *	fonction numérique
LOOP	lo SHIFT O	instruction
MID\$	m SHIFT l	fonction chaîne de caractères
MONITOR	m SHIFT O	instruction
NEW	* aucune *	commande
NEXT	n SHIFT E	instruction
ON...GOSUB	on...go SHIFT S	instruction
ON...GOTO	on...g SHIFT O	instruction
OPEN	o SHIFT P	instruction
PAINT	p SHIFT A	instruction
PEEK	p SHIFT E	fonction numérique
POKE	p SHIFT O	instruction
POS	* aucune *	fonction numérique
PRINT	?	instruction
PRINT#	p SHIFT R	instruction
PRINT USING	?us SHIFT l	instruction
PUDEF	p SHIFT U	instruction
RCLR	r SHIFT C	fonction numérique
RDOT	r SHIFT D	fonction numérique
READ	r SHIFT E	instruction
REM	* aucune *	instruction

MOT CLE	ABBREVIATION	TYPE
RENAME	re SHIFT N	commande
RENUMBER	ren SHIFT U	commande
RESTORE	re SHIFT S	instruction
RESUME	res SHIFT U	instruction
RETURN	re SHIFT T	instruction
RGR	r SHIFT G	fonction numérique
RIGHT\$	r SHIFT I	fonction chaîne de caractères
RLUM	r SHIFT L	fonction numérique
RND	r SHIFT N	fonction numérique
RUN	r SHIFT U	commande
SAVE	s SHIFT A	commande
SCALE	sc SHIFT A	instruction
SCNCLR	s SHIFT C	instruction
SCRATCH	sc SHIFT R	commande
SGN	s SHIFT G	fonction numérique
SIN	s SHIFT I	fonction numérique
SOUND	s SHIFT O	instruction
SPC(	s SHIFT P	fonction spéciale
SQR	s SHIFT Q	fonction numérique
SSHAPE	s SHIFT S	instruction
STatus	* aucun *	variable numérique réservée
STOP	s SHIFT T	instruction
STR\$	st SHIFT R	fonction chaîne de caractères
SYS	s SHIFT Y	instruction
TAB(	t SHIFT A	fonction spéciale
TAN	* aucun *	fonction numérique
TI	* aucun *	variable numérique réservée
TI\$	* aucun *	variable réservée chaîne
TRAP	t SHIFT R	instruction
TROFF	tro SHIFT F	instruction
TRON	tr SHIFT O	instruction
UNTIL	u SHIFT N	instruction
USR	u SHIFT S	fonction spéciale
VAL	* aucun *	fonction numérique
VERIFY	v SHIFT E	commande
VOL	v SHIFT O	instruction
WAIT	w SHIFT A	instruction
WHILE	w SHIFT H	instruction

---

---

## SECTION 3

### PROGRAMMES DE CONVERSION

#### Conversion de programmes BASIC standard en Commodore BASIC 3.5

Si vous avez un programme écrit dans un BASIC autre que le Commodore BASIC, vous devez faire quelques petits ajustements avant de lancer le programme sur votre Plus/4. Voici quelques indications pour rendre la conversion plus facile.

#### DIMENSION DES CHAINES DE CARACTERES

Vous devez enlever toutes instructions déclarant la longueur des chaînes de caractères. Une instruction, telle que DIM A\$(I,J), qui dimensionne une chaîne de caractères avec J éléments de longueur I, doit être converti avec l'instruction DIM A\$(J) du Commodore BASIC.

Certains BASIC utilisent une virgule ou un & pour concaténer les chaînes de caractères. Vous devez changer ces signes par le signe +, qui est l'opérateur de concaténation du Commodore BASIC 3.5.

Dans le Commodore BASIC, les fonctions MID\$, RIGHT\$ et LEFT\$ sont utilisées pour créer des sous-chaînes de caractères. Les formes telles que A\$(I) pour accéder au Ième caractère de A\$ ou A\$(I,J) pour avoir une sous-chaîne de A\$ de la position I à la position J, doivent être changées comme suit:

---

#### AUTRES BASIC

X\$ = A\$(I)

X\$ = A\$(I,J)

#### COMMODORE BASIC 3.5

X\$ = MID\$(A\$,I,1)

X\$ = MID\$(A\$,I,J)

---

### AFFECTATIONS MULTIPLES

Pour donner la valeur 0 à B et à C, certains BASIC permettent les instructions de la forme :

10 LET B=C=0

Commodore BASIC interpréterait le second signe = comme un opérateur logique et donnerait B=-1 si C=0. Il faut donc convertir cette instruction comme suit :

10 C=0:B=0

---

### INSTRUCTIONS MULTIPLES

Certains BASIC utilisent le signe / pour séparer des instructions multiples sur une ligne. Avec le BASIC 3.5, utilisez les deux point (: ) pour séparer les instructions.

---

---

---

---

## FONCTION MAT

Les programmes utilisant la fonction MAT disponible sur certains BASIC doivent être réécrits en utilisant la boucle FOR...NEXT pour fonctionner parfaitement.

---

## REPROGRAMMATION DES TOUCHES DE FONCTION

Vous pouvez reprogrammer les touches de fonction pour les rendre équivalentes aux touches de fonction du Commodore 64 et du VIC 20 (cela rendra plus facile les conversions des programmes faits sur ces machines).

Pour reprogrammer les touches, tapez la ligne suivante dans votre programme.

```
10 FOR I=1 TO 8:KEY I,CHR$(I+132):NEXT
```

Maintenant quand vous taperez sur une touche de fonction, la machine enverra un caractère transparent avec un code ASCII entre 133 et 140, comme le fait le Commodore 64. Pour vérifier cela dans un programme, vous pouvez utiliser la méthode suivante:

```
20 GETKEY A$:IF ASC(A$)=133 THEN PRINT "TOUCHE FONCTION 1  
APPUYEE":GOTO 20  
30 IF ASC(A$)>133 AND ASC(A$)<141 THEN PRINT "AUTRE  
TOUCHE DE FONCTION APPUYEE"  
40 GOTO 20
```

Après avoir fini votre programme, vous devez redéfinir les touches si vous voulez revenir à leur utilisation précédente. Vous pouvez faire cela avec une instruction dans un programme ou en réinitialisant votre Plus/4.

---

---

## SECTION 4

### MESSAGES D'ERREUR

Ces messages sont imprimés par BASIC. Vous pouvez aussi imprimer ces messages en utilisant la fonction ERR\$(*n*). Le numéro de l'erreur fait seulement référence à l'utilisation de cette fonction.

N°	ERREUR	
1	<b>TOO MANY FILES</b> trop de fichiers	On ne peut ouvrir que 10 fichiers en même temps.
2	<b>FILE OPEN</b> fichier ouvert	Le numéro de fichier est déjà utilisé.
3	<b>FILE NOT OPEN</b> fichier non ouvert	Le fichier dont le numéro est spécifié dans une instruction d'entrée/sortie doit être ouvert avant utilisation.
4	<b>FILE NOT FOUND</b> fichier non trouvé	Aucun fichier du nom spécifié n'existe sur la disquette ou bien un repère de fin de bande a été lu sur lecteur-enregistreur de cassettes.
5	<b>DEVICE NOT PRESENT</b> périphérique absent	Le périphérique d'entrée/sortie demandé n'est pas disponible.
6	<b>NOT INPUT FILE</b> fichier de sortie	Le fichier auquel vous voulez accéder n'a pas été ouvert en INPUT.
7	<b>NOT OUTPUT FILE</b> fichier d'entrée	Le fichier auquel vous voulez accéder n'a pas été ouvert en OUTPUT.
8	<b>MISSING FILE NAME</b> nom de fichier absent	Vous n'avez pas donné de nom de fichier après une commande disque.
9	<b>ILLEGAL DEVICE NUMBER</b> numéro périphérique faux	Vous avez essayé d'utiliser un périphérique de manière impropre (sauvegarder périphérique faux sur un écran)

---



---

10	<b>NEXT WITHOUT FOR</b> <b>NEXT sans FOR</b>	Les boucles sont mal imbriquées ou un nom de variable d'un NEXT ne correspond pas au nom de la variable du FOR.
11	<b>SYNTAX</b> syntaxe	Une instruction n'est pas reconnue par BASIC. Cela peut être un oubli ou une parenthèse de plus, un mot clé mal écrit.
12	<b>RETURN WITHOUT GOSUB</b> <b>RETURN sans GOSUB</b>	Une instruction RETURN a été rencontrée sans activation préalable d'une instruction GOSUB
13	<b>OUT OF DATA</b> plus de données	Une instruction READ a été exécutée sans qu'il ne reste de données à lire.
14	<b>ILLEGAL QUANTITY</b> quantité fausse	Un nombre utilisé comme argument d'une fonction ou d'une instruction est en dehors des limites autorisées.
15	<b>OVERFLOW</b> débordement	Le résultat d'une opération est plus grand que le plus grand nombre permis. (1.701411833E+38)
16	<b>OUT OF MEMORY</b> mémoire pleine	Il n'y a plus de mémoire disponible pour le programme et les variables ou il y a trop de DO, FOR ou GOSUB en train.
17	<b>UNDEF'D STATEMENT</b> instruction non définie	Vous faites appel à un numéro de ligne qui n'existe pas dans le programme.
18	<b>BAD SUBSCRIPT</b> mauvaise indice	Le programme essaye de se référer à un élément d'un tableau dont le n° est à l'extérieur des limites du DIM.

---



---

---

19	<b>REDIM'D ARRAY</b> tableau redimensionné	Un tableau ne peut être dimensionné qu'une fois. Si un tableau est utilisé avant qu'il ne soit dimensionné, un dimensionnement automatique est réalisé en fixant le nombre d'éléments à 10.
20	<b>DIVISION BY ZERO</b> division par zéro	La division par zéro n'est pas autorisée
21	<b>ILLEGAL DIRECT</b> illégal en mode direct	Les instructions INPUT et GET ne peuvent pas être utilisées en mode direct.
22	<b>TYPE MISMATCH</b> erreur de type	Cette erreur se présente lorsqu'un nombre est utilisé à la place d'une chaîne de caractères et vice versa.
23	<b>STRING TOO LONG</b> chaîne trop longue	Une chaîne de caractères ne peut contenir que 255 caractères.
24	<b>FILE DATA</b> données de fichier	Lecture de données erronées sur lecteur-enregistreur de cassettes.
25	<b>FORMULA TOO COMPLEX</b> formule trop complexe	L'expression de la chaîne évaluée doit être partagée au moins en deux parties pour que le système puisse la traiter.
26	<b>CAN'T CONTINUE</b> ne peut continuer	La commande CONT ne peut pas fonctionner soit parce que le programme n'a pas été lancé, soit parce qu'il y avait une erreur et qu'une ligne a été modifiée.
27	<b>UNDEF'D FUNCTION</b> fonction non définie	Référence à une fonction définie par l'utilisateur qui n'a jamais été définie.

---

---



---

28	<b>VERIFY</b> vérification	Le programme sur lecteur-enregistreur de cassettes ou sur disquette ne correspond pas à celui qui est en mémoire.
29	<b>LOAD</b> chargement	Problème avec le chargement du programme, réessayez.
30	<b>BREAK</b> arrêt	La touche STOP a été utilisée pour arrêter l'exécution du programme.
31	<b>CAN'T RESUME</b> ne peut reprendre	Une instruction RESUME est exécutée sans activation préalable de TRAP.
32	<b>LOOP NOT FOUND</b> LOOP non trouvé	Le programme a rencontré une instruction DO et ne peut pas trouver LOOP.
33	<b>LOOP WITHOUT DO</b> LOOP sans DO	Le programme a rencontré une instruction sans activation préalable de DO.
34	<b>DIRECT MODE ONLY</b> seulement en mode direct	Cette commande n'est permise qu'en mode direct, pas dans un programme.
35	<b>NO GRAPHICS AREA</b> pas de zone graphique	Une commande graphique a été rencontrée sans qu'une commande GRAPHIC n'ait été exécutée.
36	<b>BAD DISK</b> mauvaise disquette	La disquette ne peut pas être formatée soit parce que la disquette est détériorée, soit parce que la méthode de formatage rapide (sans i.d.) a été utilisée sur une disquette non formatée.

---



---

---

---

## DESCRIPTION DES MESSAGE D'ERREURS DU DOS

Ces messages sont donnés par les variables réservées DS et DS\$.

NOTE: Les numéros des messages d'erreur plus petits que 20 doivent être ignorés à l'exception de 01, qui donne le nombre de fichiers effacés avec la commande SCRATCH.

20	<b>READ ERROR</b> erreur lecture en-tête du bloc non trouvée	Le contrôleur disquette ne peut pas localiser l'en-tête du bloc de données désiré. Cela peut être dû à un numéro de secteur illégal ou à une détérioration de l'en-tête du bloc.
21	<b>READ ERROR</b> erreur lecture caractère non synchrone	Le contrôleur ne peut pas trouver le repère de synchronisation sur la piste désirée. Cela peut être dû à un mauvais alignement de la tête de lecture, à l'absence de disquette ou à une disquette non formatée ou mal mise. Cela peut aussi indiquer une panne matériel.
22	<b>READ ERROR</b> erreur lecture bloc de données absent	Le contrôleur ne peut pas lire un bloc de données qui a été mal écrit. Cette erreur, associée à une commande BLOCK, indique un accès à une piste ou secteur illégal.
23	<b>READ ERROR</b> erreur lecture erreur de somme de contrôle sur données	Ce message indique qu'il y a une erreur dans au moins un octet de données. Les données ont été lues en mémoire du DOS, mais la somme de de contrôle est fausse. Le message peut aussi révéler des problèmes de masse.

---



---

24	<b>READ ERROR</b> erreur lecture erreur de décodage d'un octet	Les données ou l'en-tête ont été lues en mémoire du DOS, mais une erreur matérielle a eu lieu, à cause d'une forme invalide dans l'octet de données. Ce message peut aussi révéler des problèmes de masse.
25	<b>WRITE ERROR</b> erreur écriture en vérification	Ce message est généré si le contrôleur détecte une non concordance entre la donnée écrite et la donnée en mémoire du DOS.
26	<b>WRITE PROTECT ON</b> protection écriture	Ce message est généré quand on a demandé au contrôleur d'écrire une donnée alors que la disquette est en protection écriture. Ceci est causé par une disquette dont l'encoche est recouverte d'une étiquette.
27	<b>READ ERROR</b> erreur lecture dans la somme de contrôle	Le contrôleur a détecté une erreur dans l'en-tête du bloc de données. Le bloc n'a pas été lu dans la mémoire du DOS. Ce message peut aussi révéler des problèmes de masse.
28	<b>WRITE ERROR</b> erreur écriture bloc de données trop long	Le contrôleur essaie de détecter le repère de synchronisation de la prochaine en-tête après l'écriture d'un bloc. Si ce repère n'apparaît pas après un délai déterminé, le message est généré. L'erreur est due à un mauvais format de disquette (les données continuent sur

---



---

---



---

		le prochain bloc) ou à une erreur matérielle.
29	<b>DISK ID MISMATCH</b> mauvais numéro de disquette	Le message est généré quand le contrôleur accède à une disquette qui n'a pas été initialisée. Le message peut aussi arriver si la disquette a une mauvaise en-tête.
30	<b>SYNTAX ERROR</b> erreur de syntaxe générale	Le DOS ne peut pas interpréter la commande envoyée sur le canal. C'est souvent dû à un numéro non autorisé de fichiers ou à une mauvaise forme de commande. Par exemple: deux noms de fichiers peuvent apparaître sur le côté gauche de la commande COPY.
31	<b>SYNTAX ERROR</b> erreur syntaxe commande invalide	Le DOS ne reconnaît pas la commande. La commande doit être mise en première position.
32	<b>SYNTAX ERROR</b> erreur syntaxe commande invalide	La commande envoyée est plus longue que 58 caractères.
33	<b>SYNTAX ERROR</b> erreur syntaxe nom de fichier invalide	La syntaxe est invalide dans la commande OPEN ou la commande SAVE.
34	<b>SYNTAX ERROR</b> erreur syntaxe pas de fichier donné	Le nom du fichier n'est pas dans la commande ou DOS ne le reconnaît pas comme tel. Généralement, un deux-point (:) a été oublié.
39	<b>SYNTAX ERROR</b> erreur syntaxe commande non valide	La commande envoyée au canal de commande (adresse secondaire 15) n'est pas reconnue par DOS.

---



---

---



---

50	<b>RECORD NOT PRESENT</b> enregistrement non présent	Lecture de disquette par INPUT# et GET# après le dernier enregistrement. Ceci arrive aussi avec un positionnement sur un enregistrement après la fin d'un fichier relatif. Si vous vouliez étendre le fichier avec un nouvel enregistrement (commande PRINT#), le message peut être ignoré. INPUT ou GET ne doivent pas être utilisés sans un nouveau positionnement.
51	<b>OVERFLOW IN RECORD</b> débordement d'enregistrement	L'instruction PRINT# dépasse les limites de l'enregistrement. L'information est tronquée. Puisque le retour chariot qui est le repère de fin d'enregistrement, est compris dans la taille de l'enregistrement. Ce message est généré si le nombre total des caractères de l'enregistrement (y compris le retour chariot) dépasse la longueur définie.
52	<b>FILE TOO LARGE</b> fichier trop grand	Les nombres d'enregistrements dans un fichier relatif est tel qu'il dépassera la capacité de la disquette.
60	<b>WRITE FILE OPEN</b> fichier déjà ouvert en écriture	Un fichier en écriture n'a pas été fermé et on essaie de le rouvrir en lecture.
61	<b>FILE NOT OPEN</b> fichier non ouvert	On cherche à accéder à un fichier qui n'a pas été ouvert dans le DOS. Quelquefois, il n'y a pas de message et la demande est ignorée.

---



---

---

62	<b>FILE NOT FOUND</b> fichier non trouvé	Le fichier demandé n'existe pas sur la disquette indiquée.
63	<b>FILE EXISTS</b> fichier existant	Il existe déjà sur la disquette un fichier du nom de celui qu'on demande de créer.
64	<b>FILE TYPE MISMATCH</b> désaccord sur le type du fichier	Le type de fichier indiqué n'est pas celui qui est porté dans le répertoire pour le fichier demandé.
65	<b>NO BLOCK</b> bloc non disponible	Ce message se présente en liaison avec la commande B-A. Le bloc qu'on désire allouer l'a déjà été. Les paramètres indiquent la piste et le secteur du prochain bloc disponible. Si ces paramètres sont nuls, c'est que tous les blocs supérieurs ont déjà été alloués.
66	<b>ILLEGAL TRACK OR SECTOR</b> piste ou secteur illégal	Le DOS essaye de lire une piste ou un secteur inexistants. Ceci peut indiquer un problème de lecture sur le pointeur du prochain bloc.
67	<b>ILLEGAL SYSTEM TRACK OU SECTOR</b> piste ou secteur système illégal	Tentative d'accès à une piste ou un secteur réservé à l'usage du DOS.
70	<b>NO CHANNEL</b> plus de canal disponible	Le canal demandé n'est pas disponible ou tous les canaux sont déjà utilisés. A un instant donné, dans le DOS, on peut ouvrir au plus 5 fichiers séquentiels ou 6 canaux d'accès direct.
71	<b>DIRECTORY ERROR</b> erreur sur la BAM	La BAM n'est pas en accord avec le compte interne. Il y a un problème dans

---

- 
- 
- |    |  |   |
|----|--|---|
| 72 | <b>DISK FULL</b><br>disquette pleine                                   | <p>l'allocation de la BAM ou elle a été altérée dans la mémoire du DOS. Pour corriger ce problème, réinitialisez la disquette pour remettre la BAM en mémoire. Attention, cela peut détruire des fichiers actifs. NOTE : BAM = Block Availability Map (carte des blocs disponibles).</p>  |
| 73 | <b>DOS MISMATCH</b><br>73,CBM DOS<br>V2.6 1541)<br>désaccord de<br>DOS | <p>Tous les blocs de la disquette sont utilisés ou bien la répertoire est plein. Ce message est envoyé quand il ne reste plus que 2 blocs sur la 1541 pour permettre de fermer le fichier.</p> <p>Les DOS 1 et 2 sont compatibles en lecture mais pas en écriture. Les disquettes peuvent être interchangeables pour la lecture avec un des DOS, mais une disquette formatée avec une des versions, ne peut être utilisée en écriture avec une des autres versions. Cette erreur est générée si vous essayez d'écrire sur une disquette qui a été formatée avec un format non compatible (un utilitaire est disponible pour les conversions entre format). Ce message peut aussi apparaître après la mise sous tension.</p> |
| 74 | <b>DRIVE NOT READY</b><br>lecteur non<br>prêt                          | <p>Tentative d'accès à un lecteur alors qu'aucune disquette n'a été insérée. ou bien porte non fermée.</p>  |
- 
-

---

---

## SECTION 5

### TEDMON

#### INTRODUCTION

TEDMON est un programme en langage machine intégré, qui vous permet d'écrire aisément des programmes en langage machine. TEDMON comprend un moniteur de langage machine, un mini-assembleur et un désassembleur.

Les programmes en langage machine écrits avec TEDMON peuvent être exécutés seuls ou bien comme des sous-programmes très rapides pour vos programmes BASIC, puisque TEDMON a la possibilité de coexister pacifiquement avec le BASIC.

#### LES COMMANDES TEDMON

---

<b>A ASSEMBLE</b>	Assemble les lignes en code 6502.
<b>C COMPARE</b>	Compare deux parties de la mémoire et indique les différences.
<b>D DISASSEMBLE</b>	Désassemble les lignes en code 6502.
<b>F FILL</b>	Remplit la mémoire avec l'octet spécifié.
<b>G GO</b>	Démarré l'exécution à l'adresse spécifiée.
<b>H HUNT</b>	Cherche dans la mémoire la présence de certains octets.
<b>L LOAD</b>	Charge un fichier à partir du lecteur/enregistreur de cassettes ou d'une disquette.
<b>M MEMORY</b>	Affiche les valeurs hexadécimales des adresse mémoire.
<b>R REGISTERS</b>	Affiche les registres du 6502.
<b>S SAVE</b>	Sauvegarde sur le lecteur/enregistreur de cassettes ou disquette.
<b>T TRANSFER</b>	Transfère les codes d'une partie de la mémoire dans une autre.
<b>V VERIFY</b>	Compare la mémoire avec le lecteur/enregistreur de cassettes ou la disquette.
<b>X EXIT</b>	Sort de TEDMON.

---

---

- 
- 
- . (point) Assemble les lignes en code 6502.
  - > (supérieur à) Modifie la mémoire.
  - ; (point-virgule) Modifie l'affichage du registre 6502.

---

L'adresse \$7F8 contrôle si TEDMON pointe vers la ROM ou la RAM au-dessus de \$8000. Si l'adresse est à zéro, TEDMON affiche BASIC et le KERNAL, quand il doit désassembler ou lister la mémoire au-dessus de \$8000. Si l'adresse est à \$80, TEDMON affiche la RAM à la place du BASIC et du KERNAL. Ceci est souvent utile pour développer des programmes en langage machine. Notez que l'adresse \$7F8 n'affecte pas la commande GO. Cette commande démarre l'exécution dans la mémoire en cours (ROM ou RAM) sans regarder la valeur de l'adresse \$7F8.

## UTILISATION DE TEDMON

Vous entrez dans TEDMON en tapant:

**MONITOR**

TEDMON répond en affichant les registres du 6502 et en faisant clignoter le curseur. Le curseur est l'indication que TEDMON attend vos commandes.

## DESCRIPTION DES COMMANDES

### COMMANDE A

But : entrer une ligne en code assembleur.

Syntaxe: **A**<adresse><code mnémonique><opérande>

<adresse>, un nombre hexadécimal indique l'adresse mémoire où placer le code hexadécimal.

<code mnémonique>, un mnémonique standard du langage assembleur pour MOS technology, par exemple: LDA, STX, ROR, etc...

<opérande>, l'opérande, quand il est nécessaire, peut être un des modes d'adressage. (En page-zéro, vous devez mettre un nombre hexadécimal à 2 chiffres dont la valeur est inférieure à \$100. Pour les adresses en page différente de zéro, vous devez mettre un nombre hexadécimal à 4 chiffres.)

La touche  est utilisée pour indiquer la fin de la ligne assembleur. S'il y a une erreur sur la ligne, un point d'interrogation est affiché pour indiquer l'erreur et le curseur se met sur la ligne suivante. L'éditeur d'écran peut être utilisé pour corriger l'erreur.

---

---

Après le bon assemblage d'une ligne, l'assembleur affiche l'information contenant la prochaine adresse mémoire autorisée pour une instruction, ainsi vous n'avez pas besoin de répéter A et le numéro de ligne quand vous tapez un programme en langage machine sur le Plus/4.

Exemple :

**.A 1200 LDX#\$00**  
**.A 1202**

NOTE: Un point (.) est équivalent à la commande ASSEMBLE.

Exemple :

**.2000 LDA#\$23**

---

## COMMANDE C

But : compare deux zones mémoire

Syntaxe : **C**<adresse1><adresse2><adresse3>

<adresse1>, est un nombre hexadécimal indiquant l'adresse de début de la zone mémoire à comparer.

<adresse2>, est un nombre hexadécimal indiquant l'adresse de fin de la zone mémoire à comparer.

<adresse3>, est un nombre hexadécimal indiquant l'adresse de début de l'autre zone mémoire à comparer.

Si les deux zones mémoire sont équivalentes, TEDMON imprime un RETURN indiquant que la seconde zone mémoire est identique à la première. Les adresses des octets qui sont différents dans les deux zones, sont affichées à l'écran.

---

## COMMANDE D

But : désassemble le code machine en mnémoniques et opérandes du langage assembleur.

Syntaxe : **D**[<adresse>][<adresse2>]

<adresse>, un nombre hexadécimal donnant l'adresse de début de désassemblage.

<adresse2>, un nombre hexadécimal optionnel donne l'adresse de fin de désassemblage.

---

---

---

---

La présentation du désassemblage est légèrement différente de celle de l'assemblage. Cette différence est que le premier caractère du désassemblage est un point (.) plutôt qu'un A (pour la lisibilité) et la valeur hexadécimale du code est aussi affichée.

Une liste désassemblée peut être modifiée en utilisant l'éditeur d'écran. Modifiez le code mnémorique et l'opérande sur l'écran et appuyez ensuite sur RETURN. Cela valide la ligne et appelle l'assembleur pour les modifications.

Un désassemblage peut être paginé. Tapez un D pour passer à la page suivante de votre désassemblage à l'écran.

Exemple :

```
D 3000 3004
. 3000 A9 00      LDA#$00
. 3002 FF         ???
. 3003 D0 2B      BNE $3030
```

---

## COMMANDE F

But : remplir une série d'adresses avec un octet spécifié.

Syntaxe : **F**<adresse1><adresse2><octet>

<adresse1>, la première adresse à remplir avec l'<octet>.

<adresse2>, la dernière adresse à remplir avec l'<octet>.

<octet>, un nombre hexadécimal à 1 ou 2 chiffres.

Cette commande est utile pour initialiser des zones de données ou toute autre zone RAM.

Exemple : **F 0400 0518 EA**

Remplit les adresses mémoire entre \$0400 et \$0518 avec \$EA (l'instruction NOP).

---

## COMMANDE G

But : démarre l'exécution d'un programme à l'adresse spécifiée.

Syntaxe : **G** [<adresse>]

<adresse>, est un argument optionnel spécifiant la nouvelle valeur du compteur programme et de l'adresse où démarrera l'exécution. Si cet argument n'est pas indiqué, l'exécution commence au PC en cours (le PC sera vu avec la commande R).

---

---

La commande G remet à jour tous les registres (affichables par la commande R) et commence l'exécution à l'adresse de départ spécifiée. Vous devez faire attention à l'utilisation de cette commande. Pour revenir à TEDMON, après l'exécution du programme, utilisez l'instruction BRK.

Exemple : **G 140C**

Exécute le programme en commençant par l'adresse \$140C.

---

## COMMANDE H (HUNT)

But : Cherche dans des limites de la mémoire la présence d'un ensemble d'octets.

Syntaxe : **H**<adresse1><adresse2><données>

<adresse1> adresse de début de la procédure de recherche.

<adresse2> adresse de fin de la procédure de recherche.

<données> donne le but de la recherche des éléments qui peuvent être sous forme hexadécimale ou sous forme chaîne de caractères en ASCII. Une recherche sur l'ASCII est définie en faisant précéder les caractères par une apostrophe (par exemple 'CHAINE). les données peuvent être un argument unique à un seul ou à plusieurs éléments. Si vous recherchez une suite de nombres hexadécimaux, ils doivent être séparées par des blancs.

Exemple :

**H C000 FFFF 'READ**

(cherche la chaîne de caractères en ASCII entre C000 et FFFF)

**H A000 A101 A9 FF 4C**

(cherche la suite de nombres hexadécimaux \$A9, \$FF et \$4C de A000 à A101.)

---

## COMMANDE L (LOAD)

But : charger un fichier du lecteur/enregistreur de cassettes ou d'une disquette.

Syntaxe : **L**<"nom de fichier">,<unité>

<"nom de fichier"> est n'importe quel nom de fichier autorisé par le Plus/4, entre parenthèses.

---

---

<unité> est le nombre hexadécimal indiquant le périphérique devant être utilisé.

1 pour le lecteur/enregistreur de cassettes.

8 pour la disquette (ou 9 etc...)

La commande LOAD charge un fichier en mémoire. L'adresse de début est contenue dans le premier des deux octets du fichier (un fichier programme). En d'autres mots, la commande LOAD charge toujours le programme en mémoire tel qu'il a été sauvegardé. C'est très important pour travailler en langage machine, surtout que les programmes sont complètement relogeables.

Exemple :

L"ECRAN",1 (charge à partir du lecteur/enregistreur de cassettes)

L"AUTO",8 (charge à partir de la disquette)

---

COMMANDE : M (MEMORY DISPLAY, affichage de la mémoire)

But : affiche la mémoire avec les valeurs hexadécimales et les valeurs ASCII dans les limites spécifiées.

Syntaxe : M[<adresse1>][<adresse2>]

<adresse1> adresse de début de la liste mémoire. En option. Si on ne l'indique pas, une page est affichée, le premier octet est la dernière adresse utilisée.

<adresse2> adresse de fin de la liste mémoire. En option. Si on ne l'indique pas, une page est affichée. Le premier octet est celui de l'<adresse1>.

La mémoire est affichée dans le format suivant:

>A048 41 E7 00 AA AA 00 98 56 45:A1.\*..VE

Le contenu de la mémoire peut être modifié en utilisant l'éditeur d'écran. Amenez le curseur sur l'élément à modifier et tapez la modification désirée et appuyez sur RETURN. Si vous avez indiqué une mauvaise adresse RAM ou une adresse ROM, un signe d'erreur (?) est affiché.

La liste des valeurs ASCII est affichée en vidéo inverse (pour différencier l'ASCII de l'hexadécimal) sur la droite de l'écran. Si un caractère n'est pas affichable, il est représenté par un point (.)

---

Comme pour la commande D, vous pouvez passer à la page suivante en tapant M et RETURN.

Exemple :

```
M 1C00
>1C00 41 00 AA AA 00 98 56 45 :A.**..VE
>1C08 41 00 AA AA 00 98 56 45 :A.**..VE
>1C10 41 00 AA AA 00 98 56 45 :A.**..VE
>1C18 41 00 AA AA 00 98 56 45 :A.**..VE
>1C20 41 00 AA AA 00 98 56 45 :A.**..VE
>1C28 41 00 AA AA 00 98 56 45 :A.**..VE
>1C30 41 00 AA AA 00 98 56 45 :A.**..VE
>1C38 41 00 AA AA 00 98 56 45 :A.**..VE
>1C40 41 00 AA AA 00 98 56 45 :A.**..VE
>1C48 41 00 AA AA 00 98 56 45 :A.**..VE
>1C50 41 00 AA AA 00 98 56 45 :A.**..VE
>1C58 41 00 AA AA 00 98 56 45 :A.**..VE
```

---

### COMMANDE > (supérieur à)

But : peut être utilisé pour donner la valeur de 1 à 8 adresses mémoire en même temps.

Syntaxe : > *adresse donnée octet 1 <donnée octet 2...8>*

*adresse* : première adresse mémoire.

*donnée octet 1* : donnée à mettre à l'adresse indiquée.

*<donnée octet 2 à donnée octet 8>* : données à mettre aux adresses immédiatement supérieures à la première. En option.

Exemple :

>2000 08

(place à l'adresse 2000 la valeur \$08)

>3000 23 45 65

(place à l'adresse 3000 la valeur \$23, en 3001 \$45 et 3002 \$65)

---

### COMMANDE R (affichage des registres)

But : Montre les registres importants du 6502. Le registre d'état du programme, le compteur programme, l'accumulateur, les registres d'index X et Y et le pointeur de la pile sont affichés.

---

---

Syntaxe : **R**

Exemple : R  
          PC SR AC XR YR SP  
          ; 1002 01 02 03 04 F6

NOTE : le point-virgule (;) peut être utilisé pour modifier l'affichage des registres de la même manière que > peut être utilisé pour modifier les registres mémoires.

---

### COMMANDE **S** (SAVE)

But : Sauvegarde des contenus mémoire sur lecteur/enregistreur de cassettes ou disquette.

Syntaxe : S<"nom du fichier">,<unité>,<adresse1>,<adresse2>  
<"nom du fichier">, tout nom de fichier autorisé par le Plus/4. Pour sauvegarder les données, le nom de fichier doit être entre guillemets. L'apostrophe ne peut être utilisée.

<unité>, deux périphériques sont possibles, lecteur/enregistreur de cassettes et

disquettes. Pour sauvegarder sur lecteur/enregistreur de cassettes, utilisez l'unité n° 1. Le numéro d'unité du Plus/4 pour le lecteur de disquettes est habituellement le numéro 8. Cependant, ceci peut être changé (par exemple, lorsque plus d'un lecteur de disquettes est utilisé). Voyez votre manuel du lecteur de disquette du Plus/4.

<adresse1>, adresse de début de mémoire à sauvegarder.

<adresse2>, adresse de fin de mémoire à sauvegarder + 1. Toutes les données jusqu'à cette adresse sont sauvegardés, non compris l'octet de donnée de cette adresse.

Le fichier créé par cette commande est un fichier programme. Les deux premiers octets contiennent l'adresse de début de programme (<adresse1>). le fichier doit être rappelé par la commande L.

Exemple :

**S"jeux",08,0400,0BFF**

Sauvegarde la mémoire de l'adresse \$0400 à \$0BFF sur disquette.

---

### COMMANDE **T** (transfert)

**But : Transfert des parties de mémoire d'une zone mémoire sur une autre.**

---

---

**Syntaxe : T** <adresse1><adresse2><adresse3>

<adresse1>, adresse de début des données à transférer.

<adresse2>, adresse de fin des données à transférer.

<adresse3>, nouvelle adresse de début (où les données sont transférées).

Les données peuvent être transférées vers le haut ou vers le bas de la mémoire. Des parties de mémoire additionnelle de toutes longueurs peuvent être transférées vers le haut ou vers le bas de n'importe quel nombre d'octets.

Exemple :

**T 1400 1600 1401**

Déplace les données se trouvant entre \$1400 et 1600 inclus, d'un octet vers le haut de la mémoire.

---

### COMMAND V (vérification)

But : vérifie le fichier sur lecteur/enregistreur de cassettes ou disquette avec le contenu de la mémoire.

**Syntaxe : V** <"nom de fichier">,<unité>

<nom de fichier>, tout nom de fichier autorisé par le Plus/4.

<unité>, est un nombre hexadécimal indiquant le périphérique où le fichier est sauvegardé; lecteur/enregistreur de cassettes n°1 ou 01, disquette n°8 ou 08,09, etc...

La commande V compare le fichier avec le contenu de la mémoire. Le Plus/4 affiche: VERIFYING. Si une erreur est trouvée, le mot ERROR est ajouté. Si le fichier est vérifié avec succès, le curseur clignotant réapparaît.

Exemple : **V"travail",08**

---

### COMMANDE X (eXit, sortie)

But : retour au BASIC.

**Syntaxe : X**

Quand vous utilisez la commande X, le pointeur de la pile machine est mis à la valeur en cours du pointeur de la pile (voir la commande R). Si le pointeur a changé de valeur, utilisez la commande BASIC CLR pour remettre à zéro les pointeurs en revenant au BASIC.

---

---

## SECTION 6

### CODES D'AFFICHAGE ECRAN

Le tableau suivant donne la liste de tous les caractères résidant dans la police de caractères Commodore. Celui-ci montre quel nombre doit être mis (POKEd) dans la mémoire écran (adresse 3072 à 4095) pour obtenir le caractère désiré. (Souvenez-vous d'indiquer la couleur en mémoire de 2048 à 3071) Le tableau vous montre aussi quel caractère correspond au nombre en mémoire écran (PEEKed).

Deux polices de caractères sont autorisées, mais une à la fois seulement. Cela signifie que vous ne pouvez pas avoir les caractères d'une police à l'écran en même temps que des caractères de l'autre police. Les polices sont interverties en appuyant simultanément sur les touches  et .

En BASIC, PRINT CHR\$(142) vous mettra en mode majuscule/graphique et PRINT CHR\$(14) en mode minuscule/majuscule.

Tous nombres du tableau peuvent aussi être affichés en vidéo inversé. Le code caractère vidéo inverse est obtenu en ajoutant 128 à la valeur du tableau.

POLICE 1 POLICE 2 POKE

Les codes de 128 à 255 sont les images en vidéo inversé des codes 0 à 127.

SET 1	SET 2	POKE	SET 1	SET 2	POKE	SET 1	SET 2	POKE
"		0	T	t	20	(		40
A	a	1	U	u	21	)		41
B	b	2	V	v	22	*		42
C	c	3	W	w	23	+		43
D	d	4	X	x	24	,		44
E	e	5	Y	y	25	-		45
F	f	6	Z	z	26			46
G	g	7			27	/		47
H	h	8	£		28	0		48
I	i	9	J		29	1		49
J	j	10	↑		30	2		50
K	k	11	←		31	3		51
L	l	12	<b>SPACE</b>		32	4		52
M	m	13	!		33	5		53
N	n	14	"		34	6		54
O	o	15	#		35	7		55
P	p	16	\$		36	8		56
Q	q	17	%		37	9		57
R	r	18	&		38			58
S	s	19			39	:		59

SET 1	SET 2	POKE	SET 1	SET 2	POKE	SET 1	SET 2	POKE
<		60		T	84			108
=		61		U	85			109
		62		V	86			110
?		63		W	87			111
		64		X	88			112
	A	65		Y	89			113
	B	66		Z	90			114
	C	67			91			115
	D	68			92			116
	E	69			93			117
	F	70			94			118
	G	71			95			119
	H	72	<b>SPACE</b>		96			120
	I	73			97			121
	J	74			98			122
	K	75			99			123
	L	76			100			124
	M	77			101			125
	N	78			102			126
	O	79			103			127
	P	80			104			
	Q	81			105			
	R	82			106			
	S	83			107			

Les codes 128 à 255 sont les images à vidéo inverse des codes 0 à 127.

## SECTION 7

### CODES ASCII ET CHR\$

Cette section vous montre les caractères qui apparaîtront à l'écran si vous imprimez CHR\$(X), pour toutes les valeurs de X. Ce tableau vous montre aussi les valeurs obtenues en tapant PRINT ASC("X"), où X est un caractère que vous pouvez taper. Il est utile pour évaluer le caractère reçu d'une instruction GET, pour convertir des minuscules en majuscules et vice versa et pour imprimer des caractères de commande (comme la réduction de la taille de l'écran) qui ne peuvent pas être mis entre guillemets.

#### CODES 192-223 IDENTIQUES A 96-127

PRINTS	CHRS	PRINTS	CHRS	PRINTS	CHRS	PRINTS	CHRS
	0	↓	17	"	34	3	51
	1	RVS ON	18	#	35	4	52
	2	CLEAR HOME	19	\$	36	5	53
	3	INST DEL	20	%	37	6	54
	4		21	&	38	7	55
WHT	5		22	'	39	8	56
	6		23	(	40	9	57
	7		24	)	41		58
END OPERANT SHIFT C	8		25	*	42	:	59
END OPERANT SHIFT C	9		26	+	43	<	60
	10	ESCAPE	27		44	=	61
	11	RED	28	-	45	>	62
	12	→	29		46	?	63
RETURN	13	GRN	30	/	47	@	64
PASSAGE EN MODE MINUSCULES	14	BLU	31	0	48	A	65
	15	SPACE	32	1	49	B	66
	16	!	33	2	50	C	67

PRINTS	CHRS	PRINTS	CHRS	PRINTS	CHRS	PRINTS	CHRS
D	68		97		126	LT GREEN	155
E	69		98		127	PUR	156
F	70		99		128		157
G	71		100	ORANGE	129	VEL	158
H	72		101	FLASH ON	130	CYN	159
I	73		102		131	SPACE	160
J	74		103	FLASH OFF	132		161
K	75		104		133		162
L	76		105		134		163
M	77		106		135		164
N	78		107		136		165
O	79		108		137		166
P	80		109		138		167
Q	81		110		139		168
R	82		111		140		169
S	83		112	SHIFT RETURN	141		170
T	84		113	PASSAGE EN MODE MAJUSCULES	142		171
U	85		114		143		172
V	86		115	BLK	144		173
W	87		116		145		174
X	88		117	RVS OFF	146		175
Y	89		118	CLEAR HOME	147		176
Z	90		119	INST DEL	148		177
I	91		120	BROWN	149		178
E	92		121	YEL/GREEN	150		179
I	93		122	PINK	151		180
↑	94		123	BL /GREEN	152		181
←	95		124	LT BLUE	153		182
	96		125	BLUE	154		183

*a*  
*u*  
*a*  
*f*  
*e*  
*r*  
*r*  
*i*

PRINTS	CHRS	PRINTS	CHRS	PRINTS	CHRS	PRINTS	CHRS
	ô <sup>^</sup> 184		186		188		190
	u 185		187		189		191

**CODES**            **192-223**  
**CODES**            **224-254**  
**CODE**              **255**

**IDENTIQUES À**  
**IDENTIQUES À**  
**IDENTIQUES À**

**96-127**  
**160-190**  
**126**



---

---

## CARTE MEMOIRE COULEUR

	COLONNE				
	0	10	20	30	39
2048					2087
2088					
2128					
2168					
2208					
2248					
2288					
2328					
2368					
2408					
2448					10
2488					
2528					
2568					
2608					
2648					
2688					
2728					
2768					
2808					
2848					20
2888					
2928					
2968					
3008					24

3047

Les valeurs réelles permettant de changer la couleur d'un caractère, sont les suivantes :

---

1 NOIR	9 ORANGE
2 BLANC	10 BRUN
3 ROUGE	11 JAUNE-VERT
4 TURQUOISE	12 ROSE
5 POURPRE	13 BLEU-VERT
6 VERT	14 BLEU CLAIR
7 BLEU	15 BLEU FONCE
8 JAUNE	16 VERT CLAIR

---

La brillance d'une couleur est choisie en multipliant la valeur de brillance (0-7) par 16 et vous ajoutez le numéro de la couleur. Pour faire clignoter le caractère, augmentez la valeur de la couleur de 128.

# SECTION 9

## CARTE DES REGISTRES MEMOIRE DU PLUS/4

REG	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
0	TIMER # 1 RELOAD VALUE, BITS 0-7 (LOW)								
1	TIMER # 1 RELOAD VALUE, BITS 8-15 (HIGH)								
2	TIMER # 2 RELOAD VALUE, BITS 0-7 (LOW)								
3	TIMER # 2 RELOAD VALUE, BITS 8-15 (HIGH)								
4	TIMER # 3 RELOAD VALUE, BITS 0-7 (LOW)								
5	TIMER # 3 RELOAD VALUE, BITS 8-15 (HIGH)								
6	TEST	ECM	BMM	BLANK	# ROWS	Y2	Y1	Y0	
7	RVS OFF	PAL	FREEZE	MCM	# COLS	X2	X1	X0	
8	KEYBOARD LATCH								
9	IRQ	I-T3	NC	I-T2	I-T1	I-LP	I-RAS	NC	
10	NC	EI-T3	NC	EI-T2	EI-T1	EI-LP	EI-RAS	RC8	
11	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	
12	NC	NC	NC	NC	NC	NC	C9	CUR8	
13	CUR7	CUR6	CUR5	CUR4	CUR3	CUR2	CUR1	CUR0	
14	SND1-7	SND1-6	SND1-5	SND1-4	SND1-3	SND1-2	SND1-1	SND1-0	
15	SND2-7	SND2-6	SND2-5	SND2-4	SND2-3	SND2-2	SND2-1	SND2-0	
16	NC	NC	NC	NC	NC	NC	SND2-9	SND2-8	
17	SND-REL	NOISE	V2-SEL	V1-SEL	VOL3	VOL2	VOL1	VOLO	
18	NC	NC	BMB2	BMB1	BMB0	R BANK	S1-9	S1-8	
19	CB5	CB4	CB3	CB2	CB1	CB0	SCLOCK	STATUS	
20	VM4	VM3	VM2	VM1	VM0	NC	NC	NC	
21	BKGD0	NC	LUM2	LUM1	LUM0	COLOR3	COLOR2	COLOR1	COLOR0
22	BKGD1	NC	LUM2	LUM1	LUM0	COLOR3	COLOR2	COLOR1	COLOR0
23	BKGD2	NC	LUM2	LUM1	LUM0	COLOR3	COLOR2	COLOR1	COLOR0
24	BKGD3	NC	LUM2	LUM1	LUM0	COLOR3	COLOR2	COLOR1	COLOR0
25	BKGD4	NC	LUM2	LUM1	LUM0	COLOR3	COLOR2	COLOR1	COLOR0
26	BKGD4	NC	NC	NC	NC	NC	BRE9	BRE8	
27	BRE7	BRE6	BRE5	BRE4	BRE3	BRE2	BRE1	BRE0	
28	NC	NC	NC	NC	NC	NC	NC	VL8	
29	VL7	VL6	VL5	VL4	VL3	VL2	VL1	VL0	
30	H8	H7	H6	H5	H4	H3	H2	H1	
31	NC	BL3	BL2	BL1	BL0	VSUB2	VSUB1	VSUB0	
62	ROM SELECT								
63	RAM SELECT								

ADDRESS	CONTENTS	NOTES
	<----->	
\$FFFE-FFFF	<IRQ VECTOR	> *
\$FFFC	<RES VECTOR	> *
\$FFFA	<NMI VECTOR (NOT USED)	> *
	<	> *
\$FFB1-FFFF	<KERNAL JUMP TABLE	> *
\$FF40-FF80	<	>
\$FF00-FF3F	<TED CHIP	> *
	<	> *
\$FE00-FE7F	<DMA DISK SYSTEM	> *
\$FDE0-FDEF	<	> *
\$FDD0-FDDF	<CARTRIDGE BANK PORT	> *
\$FD10-FD1F	<6529 PARALLEL PORT	> *
\$FD00-FD0F	<ACIA	> *
	<	>
\$FCD0-FCFF	<	>
	<	>
\$D800-FCFF	<	> *
	<	> *
\$D000-D7FF	<CHARACTER ROM	> *
	<	> *
\$C000-D7FF	<MORE BASIC	> *
	<	>
\$8000-BFFF	<BASIC	>
	<	> *
\$4000-FFFF	<RAM, ALSO START OF BASIC TEXT	> *
	<AREA WHEN HIRES GRAPHICS ARE USED	>
\$2000-3FFF	<BIT MAP SCREEN DATA	>
	<	>
\$1C00-1FFF	<HIRES SCREEN VIDEO MATRIX	>
	<	>
\$1800-1BFF	<HIRES SCREEN ATTRIBUTE BYTES	>
	<	>
\$1000-	<BASIC TEXT AREA (BIT MAP OFF)	>
	<	>
\$0C00-0FFF	<TEXT VIDEO MATRIX (SCREEN MEMORY)	>
	<	>
\$0800-0BFF	<TEXT ATTRIBUTE BYTES (COLOR MEMORY)	>
	<	>
\$0000-07FF	<SYSTEM STORAGE	>

\*NOTE In the 64K RAM system, RAM goes from \$0000-\$FCFF, and from \$FF40-\$FFFF

# SECTION 10

## DERIVATION DE FONCTIONS MATHÉMATIQUES

Les fonctions qui ne sont pas intrinsèques au BASIC 3.5 peuvent être calculées comme suit:

FONCTION	EQUIVALENT BASIC
SECANTE COSANTE COTANGENTE	$SEC(X) = 1/COS(X)$ $CSC(X) = 1/SIN(X)$ $COT(X) = 1/TAN(X)$
SINUS INVERSE COSINUS INVERSE COTANGENTE INVERSE SECANTE INVERSE COSECANTE INVERSE	$ARSIN(X) = ATN(X/SQR(1-X^2))$ $ARCOS(X) = -ATN(X/SQR(1-X^2))+\pi/2$ $ARCOT(X) = ATN(X)+\pi/2$ $ARSEC(X) = ATN(X/SQR(X^2-1))$ $ARCSC(X) = ATN(X/SQR(X^2-1))$ $+ (SGN(X)-1)*\pi/2$
SINUS HYPERBOLIQUE COSINUS HYPERBOLIQUE TANGENTE HYPERBOLIQUE	$SINH(X) = (EXP(X)-EXP(-X))/2$ $COSH(X) = (EXP(X)+EXP(-X))/2$ $TANH(X) = EXP(-X)/(EXP(X)+EXP$ $(-X))^2+1$
COTANGENTE HYPERBOLIQUE	$COTH(X) = EXP(-X)/(EXP(X)-EXP$ $(-X))^2+1$
SECANTE HYPERBOLIQUE COSECANTE HYPERBOLIQUE	$SECH(X) = 2/(EXP(X)+EXP(-X))$ $CSCH(X) = 2/(EXP(X)-EXP(-X))$
SINUS HYPERBOLIQUE INVERSE COSINUS HYPERBOLIQUE INVERSE TANGENTE HYPERBOLIQUE INVERSE COTANGENTE HYPERBOLIQUE INVERSE SECANTE HYPERBOLIQUE INVERSE COSECANTE HYPERBOLIQUE INVERSE	$ARSIHN(X) = LOG(X+SQR(X^2+1))$ $ARCOSH(X) = LOG(X+SQR(X^2-1))$ $ARTANH(X) = LOG((1+X)/(1-X))/2$ $ARCOTH(X) = LOG((X+1)/(X-1))/1$ $ARSECH(X) = LOG((SQR(1-X^2)+1)/X)$ $ARCSCH(X) = LOG((SQR(1+X^2)+1)/X)$ $*SGN(X)$

---

## SECTION 11

### TABLEAU DES NOTES MUSICALES

NOTE	NOTE DU REGISTRE SON	FREQUENCE (Hz)
A	7	110
B	118	123.5
C	169	130.8
D	262	146.8
E	345	164.7
F	383	174.5
G	453	195.9
A	516	220.2
B	571	246.9
C	596	261.4
D	643	293.6
E	685	330
F	704	349.6
G	739	392.5
A	770	440.4
B	798	494.9
C	810	522.7
D	834	588.7
E	854	658
F	864	699
G	881	782.2
A	897	880.7
B	911	989.9
C	917	1045
D	929	1177
E	939	1316
F	944	1398
G	953	1575

Le tableau ci-dessus contient les valeurs du registre son de quatre octaves. Les valeurs du registre son sont utilisées comme le second paramètre de la commande SOUND. Pour utiliser la première note

---

---

du tableau (A=LA, registre son valeur 7) mettez le 7 comme second paramètre de la commande SOUND - SOUND 1,7,30.

Utilisez la formule suivante pour trouver la valeur du registre son pour les fréquences qui ne sont pas dans le tableau:

$$\text{VALEUR REGISTRE SON} = 1024 - (111860.781 / \text{FREQUENCE})$$

Le tableau des valeurs du registre SOUND et la formule ci-dessus sont toutes les deux valables pour les télévisions NTSC. C'est le standard pour la télévision utilisée aux Etats-Unis et au Canada. Si vous êtes dans un pays à standard PAL, vous devez utiliser la formule suivante pour calculer les nouvelles valeurs du registre son pour le tableau entier :

$$\text{VALEUR REGISTRE SON} = 1024 - (111840.45 / \text{FREQUENCE})$$

---

---

## SECTION 12

### PROGRAMMES A ESSAYER

```
5 GRAPHIC 3, 1: GRAPHIC 0, 1
10 INPUT "DOIS-JE FAIRE DU NETTOYAGE?"; A$
20 INPUT "DOIS-JE FAIRE UNE ROTATION?"; B$
30 INPUT "DOIS-JE CHANGER LA VITESSE?"; C$
40 INPUT "DOIS-JE CHOISIR L'ORIGINAL?"; D$
50 IF A$ = "O" THEN DIM A (3,200)
60 DEF FNA (X) = INT(RND(1) * X)
70 IF D$ = "O" THEN X1 = FNA(80)+80: X2=FNA(80)-80:
    Y1=FNA(100)+100
75 IF D$ = "O" THEN Y2 = FNA(100)+ 100
80 IF D <> "O" THEN X1=80: X2=80: Y1=100: Y2=100
90 GRAPHIC 3: FOR L=1 TO 3: COLOR L, FNA(15)+2, FNA(8):
    NEXT
100 IF C1< 1 THEN COLOR FNA(3)+1, FNA(15)+2, FNA(8):
    C1=FNA(40)+20
110 IF C2<=0 THEN 140: ELSE XA=FNA(11)-5: XB=FNA(11)-5:
    YA=FNA(13)-6
115 YB=FNA(13)-6
120 IF C$= "O" THEN C2 = FNA(10)+5
130 IF B$ = "O" THEN XB = - XA: YB= - YA
140 IF C3<1 THEN C=FNA(3)+1: C3=FNA(10)
145 IF A$= "O" THEN DRAW 0, A(0,P), A(1,P) TO A(2,P), A(3,0)
150 X1= X1+ XA: X2= X2+ XB: Y1= Y1+ YA: Y2= Y2+ YB
160 IF X1<0 OR X1>159 THEN XA= -XA: X1= X1+XA
170 IF X2<0 OR X2>159 THEN XB= -XB: X2= X2+XB
180 IF Y1<0 OR Y1>199 THEN YA= -YA: Y1= Y1+YA
190 IF Y2<0 OR Y2>199 THEN YB= -YB: Y2= Y2+YB
200 DRAW C, X1, Y1 TO X2, Y2
210 IF $= "O" THEN A(0,P)= X1: A(1,P)=Y1: A(2,P)=X2:
    A(3,P)=Y2: P= P+1
215 IF A$= "O" THEN IFP>200THENP=0
220 C1= C1-1: C2= C2-1: C3= C3-1: GOTO 100
```

---

---

## EFFETS SONORES

### LE HURLEMENT DU LOUP

```
10 VOL7
20 FORL=400TO800STEP20
30 SOUND1,L,3:NEXT
40 FORL=300TO600STEP40
50 SOUND1,L,3:NEXT
60 FORL=600TO300STEP-40
70 SOUND1,L,3:NEXT
```

---

---

---

---

## LE MANIAQUE DE L'ORDINATEUR

```
10 VOL7
20 FORL = 1TO100
30 SOUND1,INT(RND(0)*500)+400,4
40 NEXT
```

---

## TELEPHONE

```
10 VOL7
20 FORL = 1TO5
30 FORM = 1TO60
40 SOUND1,466,1
50 SOUND1,1020,1
60 NEXT
70 FORZ = 1TO2000:NEXT
80 NEXT
```

---

## SIGNAL OCCUPE

```
10 VOL7
20 FORL = 1TO15
40 SOUND1,466,20
50 SOUND1,1020,15
80 NEXT
```

---

## BULLES

```
10 VOL7
20 GRAPHIC1,1
30 FORM = 1TO50
40 GOSUB80
50 SOUND1,900 - R*20, (YR/2) + 50
60 CIRCLE1,X,Y,R,YR
70 NEXT:GRAPHIC0:END
80 X=INT(RND(0)*280)+20
90 Y=INT(RND(0)*160)+20
100 R=INT(RND(0)*40)+5
110 YR=R/1.3
120 RETURN
```

---

## FAISCEAU ZAP

```
10 VOL7
20 FORM = 1TO 20
30 FORL = 900TO850STEP - 10
40 SOUND1,L,1
```

---

---

---

```
50 NEXT
60 FORL=850TO900STEP10
70 SOUND1,L,1
80 NEXT
100 NEXT
```

---

### PORTEES DE MUSIQUE

```
10 VOL7
15 X1=0:Y1=0
20 GRAPHIC1,1
30 GETA$:IFAS$="" THENGRAPHIC0:END
40 GOSUB80
45 FORL=1TODSTEP2
50 SOUND1,X*3,5
55 SOUND2,Y*3,5
60 DRAW1,X,Y
65 X=X+2*DX:Y=Y+2*DY
70 NEXT:GOTO30
80 X=X1:X1=INT(RND(0)*280)+20
90 Y=Y1:Y1=INT(RND(0)*160)+20
100 A=X1-X:B=Y1-Y:D=SQR(A*A+B*B)
110 DX=A/(D+1):DY=B/(D+1)
120 RETURN
```

---

## SECTION 13

### INTERFACE RS-232

#### INTRODUCTION

Votre Commodore Plus/4 a une interface RS-232 résidant en mémoire pour la connection de tous modems, imprimantes ou autres périphériques RS-232. Pour connecter un périphériques à votre Plus/4, vous avez besoin d'un câble et d'un programme complémentaire. Le modem Commodore se connecte directement.

Le RS-232 de votre Plus/4 est au format standard RS-232, mais le voltage est au niveau TTL (0-5 V) plutôt qu'au voltage standard RS-232 (-12 +12). Le câble entre le Plus/4 et le périphérique RS-232, doit comprendre la conversion au voltage nécessaire (la cartouche interface RS-232 Commodore le fait bien).

Le logiciel interface RS-232 est accessible du BASIC ou du KERNAL pour la programmation en langage machine. Cette section s'occupe de l'utilisation de la RS-232 à partir du BASIC. Pour plus d'informations et pour l'utilisation à partir du langage machine, consultez le "Guide de référence du programmeur du Plus/4". L'utilisation de la RS-232 à partir du BASIC nécessite les commandes BASIC suivantes: OPEN, CLOSE, CMD, INPUT#, GET#, PRINT# et la variable réservée ST. INPUT# et GET# reçoivent des données du buffer-récepteur, tandis que PRINT# et CMD placent des données dans l'émetteur.

#### OUVERTURE D'UN CANAL RS-232

Un seul canal RS-232 peut être ouvert au même moment; une seconde instruction OPEN entrainerait la mise à zéro du pointeur du buffer-récepteur, effaçant tous les caractères se trouvant dans le buffer.

4 caractères au plus peuvent être envoyés dans le nom de fichier. Les deux premiers sont les caractères des registres contrôle et commande; les deux autres sont réservés à des options futures. Le nombre de Bauds, leur parité et les autres options peuvent être sélectionnés comme suit:

#### Syntaxe :

**OPEN /f,2,0,<">registre contrôle registre commande<">**

#### Exemple :

**OPEN 2,2,0,CHR\$(5)+CHR\$(15)**

/f - numéro de fichier logique (1-255). Si /f > 127, alors un saut à la ligne est généré après le retour chariot.

*registre contrôle* - caractère d'un seul octet (voir Figure1) (utilisé pour spécifier le nombre de Bauds).

*registre commande* - caractère d'un seul octet (voir Figure2)

## OBTENIR DE DONNEES A PARTIR DU CANAL RS-232

Quand il reçoit des données, le buffer-récepteur du Plus/4 contient 127 caractères avant de déborder. Ceci est indiqué dans le mot d'état (ST en BASIC) du RS-232. Tous les caractères reçus quand le buffer est plein, sont perdus. Evidemment, il est intéressant d'essayer de garder le buffer aussi vide que possible.

La réception de données par la RS-232 à grande vitesse, nécessite l'utilisation du langage machine, car le BASIC est trop lent.

### REGISTRE DE CONTROLE

Le registre de contrôle est utilisé pour obtenir le mode désiré pour le 6551. La longueur du mot, le nombre de bits stop et le contrôleur d'horloge sont tous déterminés par le registre de contrôle qui est décrit dans la Figure 1.

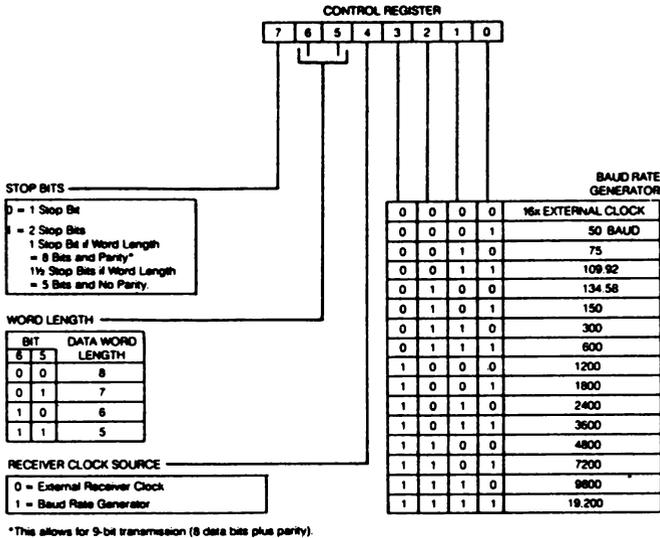


FIGURE 1. FORMAT DU REGISTRE DE CONTROLE

### Syntaxe :

Recommandée : **GET# /f,<chaîne de caractères>**

Non recommandée : **INPUT# /f,<liste de variables>**

## REGISTRE DE COMMANDE

Le registre de commande est utilisé pour contrôler les fonctions spécifiques d'émission/réception comme le montre la Figure 2.

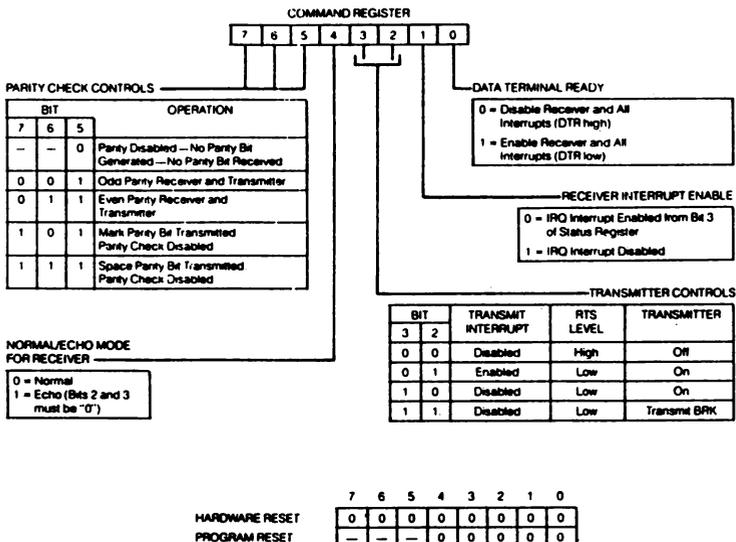


FIGURE 2. FORMAT DU REGISTRE DE COMMANDE

## NOTES

Si la longueur du mot est inférieure à 8 bits, tous les bits inutilisés sont mis à 0.

Si GET# ne trouve aucune donnée dans le buffer, le caractère vide "" est retourné.

Si INPUT# est utilisé, alors le système attend un caractère non nul suivi d'un retour chariot. Ainsi, si les lignes CTS et DSR disparaissent pendant INPUT#, le système sera dans une boucle infinie. C'est pourquoi, les routines INPUT# et CHRIN ne sont pas recommandées.

---

---

## ENVOYER DES DONNEES AU CANAL RS-232

### Syntaxe :

**CMD** /f - opère de la même manière qu'en BASIC.

**PRINT** # /f, <liste de variables>

## FERMETURE D'UN CANAL RS-232

La fermeture d'un fichier RS-232 abandonne toutes les données dans le buffer au moment de l'exécution, arrête toute émission/réception RS-232 et change les lignes RTS et Sout.

### Syntaxe :

**CLOSE** /f

---

**TABLEAU 1. LIGNES DES PORTS RS-232.**

---

PIN ID	DESCRIPTION	EIA	ABV	OUT
C	RECEIVED DATA	(BB)	Sin	IN
D	REQUEST TO SEND	(CA)	RTS	OUT
E	DATA TERMINAL READY	(CD)	DTR	OUT
F	RING INDICATOR	(CE)	RI	IN
H	RECEIVED LINE SIGNAL	(CF)	DCD	IN
J	UNASSIGNED	( )	XXX	IN
K	CLEAR TO SEND	(CB)	CTS	IN
L	DATA SET READY	(CC)	DSR	IN
B	RECEIVED DATA	(BB)	Sin	IN
M	TRANSMITTED DATA	(BA)	Sout	OUT
A	PROTECTIVE GROUND	(AA)	GND	
N	SIGNAL GROUND	(AB)	GND	

---

---

---

MODE

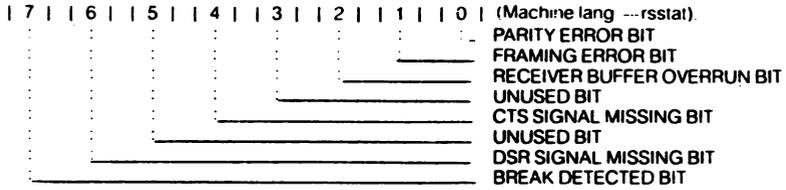


FIGURE A-3: REGISTRES D'ETAT RS-232

**NOTES**

Si le BIT=0, aucune erreur n'a été détectée.

Le registre d'état RS-232 peut être lu en BASIC avec la variable ST.

Si ST est lu en BASIC ou en utilisant la routine KERNAL READST, le mot d'état RS-232 est mis à zéro à sa sortie. Si vous devez utiliser plusieurs fois le mot d'état, la variable ST doit être assignée à une autre variable, par exemple :

**SR=ST:REM DONNE LA VALEUR ST A SR**

Le mot d'état RS-232 est lu (et mis à zéro) seulement lorsque le canal RS-232 est la dernière opération d'Entrée/Sortie utilisée.

---

---

## EXEMPLE DE PROGRAMME BASIC

Ce programme ouvre le canal de télécommunication pour vous permettre de communiquer en utilisant un modem.

```
100 OPEN 5, 2, 2, CHR$(22)+CHR$(5): RESERVE UN BUFFER ET
    OUVRE UN CANAL
110 DIM F%(255), T%(255)
120 FOR J= 32 TO 64: T%(J)=J:NEXT
130 T%(13)=13: T%(20)=8: RV=18: CT=0
220 FOR J=65 TO 90: K=J+32: T%(J)=K: NEXT
230 FOR J=91 TO 95: T%(J)=J: NEXT
240 FOR J=193 TO 218: K=J-128: T%(J)=K: NEXT
250 T%(146)=16: T%(133)=16
255 T%(137)=3: T%(134)=17: T%(138)=19
260 FOR J=0 TO 255
270 K=T%(J)
280 IF K<>0 THEN F%(K)=J: F%(K+128)=J
290 NEXT
300 PRINT""CHR$(147)
310 GET#5, A$
320 IF A$="" THEN 360
330 PRINT""CHR$(157);CHR$(F%(ASC(A$)));
340 IF F%(ASC(A$))=34 THEN PRINT CHR$(27)"O";
350 GOTO 310
360 PRINT CHR$(RV)""CHR$(157);CHR$(146):: GET A$
370 IF A$<>"" THEN PRINT#5,CHR$(T%(ASC(A$)));
380 CT=CT+1
390 IF CT=8 THEN CT=0: RV=164-RV
410 GOTO 310
```

---

---

# OBSERVATIONS

---

---

# INDEX

## A

- Abréviations des instructions
  - BASIC 169-171
- Animation 87-89
- Assigner des données
  - instructions DATA...READ 74-75, 133-134, 151, 170
  - INPUT 73,141,170
  - GET 138, 169
  - LET 142, 170

## B

- BASIC
  - abréviations 169-171
  - commandes 69-82, 118-129
  - conversion au BASIC
  - Commodore 172, 173
  - fonctions 28-31, 158-163
  - instructions 69-82, 130-157
- Boîte de connexion 8,12-13,16-17
- Brillance 90

## C

- Cable RF 9, 11
- Calculs
  - addition 59, 63
  - décimales 59
  - division 59, 63
  - ordre d'exécution 62-63
  - exponentiation 65
  - fractions 59
  - opérateurs mathématiques 59, 167-168
  - multiplication 59, 63
  - parenthèses 62-63
  - instruction PRINT 61-62

- opérateurs relationnels 59, 167-168
- notation scientifique 60
- soustraction 59, 63
- Cartes de la mémoire 199-202
- Carte des zones écran 90
- Cartouches
  - chargement 35
  - installation 35
- Cassettes
  - Lecteur/enregistreur de cassettes 4, 19
  - LOAD-chargement 36
  - SAVE-sauvegarde 37, 127
  - logiciel 36-37
- Chargement
  - cartouches 35
  - cassettes 36
  - disquettes 38-39, 121
  - commande DLOAD 30-31, 38-39, 121, 169
- Clavier
  - touches couleur 27
  - touches curseur 25-26
  - touches graphiques 28-31
  - HELP 30-32
  - touches de fonction programmable 30-31
  - touches spéciales 24, 29
- Codes
  - ASCII 169, 196-198
  - CHR\$ 169, 196-198
- Commandes (voir BASIC, commandes)
  - BACKUP 118, 169
  - CLR 25-26
  - modes graphiques 28, 84-89
  - écran graphique 93, 169
  - mémoire 199-202
  - écran 193-195
  - COLLECT 119, 169
  - CONT 119, 169
  - COPY 119, 169
  - HEADER 39-40, 122, 170

---

---

KEY 31, 123 170  
LOAD 124-125, 170  
Connecter l'ordinateur 8-18  
Copies de disquettes 119  
Couleur  
  zones 90-91  
  fond 90-91  
  bords 90-91  
  changement 27, 100-101  
  instruction COLOR 90-91, 133, 169  
  remplir des zones 100-101  
  touches 27  
  brillance 90  
  carte de la mémoire 199-202  
  PAINT 100-101, 145  
  écran 193-195  
Curseur  
  contrôler le mouvement 25-26  
  touches curseur 25-26  
  dans les instructions PRINT 25, 49-51

## D

Définitions des touches de fonctions 30-31  
Définitions de fonctions dans des programmes 30-31  
DEL, effacer  
  commande 25, 120, 169  
  mise en forme 25  
  fichiers d'une disquette 25  
  touche 25  
  lettres d'un mot 25, 49-51  
Démarrage de l'ordinateur 8-18  
Dimension d'un tableau 134-135  
DIRECTORY 41, 120-121, 169  
Disquettes  
  instruction COPY 119, 169  
  commande DIRECTORY 41, 120-121, 169  
  lecteur de disquettes 4, 19, 38  
  messages d'erreur disquettes 39, 174-183

commande DLOAD 30-31, 38-39, 121, 169  
DS\$ 41  
  duplication 119  
  format 39-40  
  commande HEADER 39-40, 122, 170  
  liste d'un répertoire 41, 120-121  
  chargement 38-39, 121, 124-125  
  SAVE, sauvegarder 40-41, 127  
  table des matières 41  
DLOAD 30-31  
DO/LOOP/WHILE/UNTIL/EXIT 78, 135-136  
DRAW 94-95, 136, 169  
DS\$ 41  
DSAVE 30-31, 122, 169  
Dupliquer des disquettes 119

## E

Ecran  
  vider 25-26, 132, 169  
  vider en mode graphique 93, 169  
  codes d'affichage 26, 52, 193-195  
  commande LIST 30-31, 48-49, 170  
  carte de la mémoire 199-202  
  affichage programme 193-195  
  reprenre l'affichage 152  
  taille 57, 86  
  ralentissement affichage 26  
  fenêtres 57  
Effets sonores 108, 206-208  
Encyclopédie 115-214  
Erreurs  
  instructions de mise au point 174-183  
  erreurs disquettes 39, 174-183  
  explication des messages 39, 174-183

---

---

## F

Fenêtres 57, 86

Fonctions

autres 163

chaînes de caractères

162-163, 171

CHR\$ 162, 196-198

ESCape 28-29, 57

EXP 158, 169

INSTR 158-159, 170

INT 66-67, 159, 170

LEFT\$ 162, 170

mathématique 59-67, 158-162,  
203

MID\$ 162-170

numériques 59-67, 158-162,  
203

FOR...TO...STEP 77-78, 136-137

Format disquettes 39-40

Format sortie

PRINT USING 145-150

PUDEF 150-151

zones d'impression 55-56  
ponctuation 55-56

## G

GET# 138-139, 169

GOSUB 80, 139, 169

GOTO 47, 76-77, 139, 169

Graphiques

BOX 97-98, 100-101, 130, 169

CIRCLE 98-100, 131-132, 169

vider 93, 169

COLOR 90-91, 169

DRAW 94-95, 136 169

commande GRAPHIC

92-92, 139-140, 157, 169

haute résolution 92-93

touches 28-29

modes 28, 84-89

modes multi-couleur 102-103

PAINT 100-101, 145

imprimer des symboles  
graphiques 84-86

mode majuscule graphique  
24,28

GSHAPE 153-154, 170

## H

Haute résolution graphique 92-93

## I

IF...THEN...ELSE 79, 140-141,  
170

Instructions

BOX 100-101, 130, 169

CHAR 95-95, 130-131, 169

CIRCLE 98-100, 131-132, 169

CLOSE 132, 169

CLR 25-26, 132, 169

CMD 132-133, 169

conditionnelles

IF/THEN 79, 140-141, 170

contrôle

GOTO 47, 76-77, 139

FOR/NEXT/STEP 77-78

DATA 74-74, 133-134

DEF FN 134, 169

DIM 134-135, 169

END 136

entrée/sortie

PRINT 71-73

INPUT 73, 141, 170

GETKEY 74, 138, 169

READ/DATA 74-75, 133-134,

151, 170

GET 138, 169

GETKEY 74, 138, 169

INPUT 73, 141, 170

LET 142, 170

LIST 30-31, 48-49, 124, 170

INPUT# 141, 170

- 
- 
- Insertion  
mise en forme 25  
touche 25  
mode 25
- Installation de l'ordinateur 8-18
- L**
- LOCATE  
instruction 142, 170  
boucles  
DO...LOOP...WHILE/UNTIL 78,  
135-136  
GOSUB 80,139  
GOTO 47, 76-77, 139  
FOR...TO...STEP 77-78,  
136-137  
IF...THEN...ELSE 79, 140-141,  
170  
ON...GOSUB/GOTO 143, 170
- Logiciel résident 34  
résident 34  
cartouches 35  
cassettes 36-37  
disquettes 38-41  
LOADING 35-36, 38-40  
sauvegarde de votre propre  
logiciel 37, 40-41
- M**
- Mannette de jeux 5, 12, 159, 170
- Mise au point  
CONT 119, 169  
DS\$ 41  
STOP 155, 171  
RESUME 152, 171  
TRAP 155, 171
- Mise en forme  
touche INSert 25  
mode INSert 25, 28-29  
touche DELeTe 25  
commande DELeTe 25, 120,  
169  
commande RENUMBER 70,  
126-127, 171
- Modes  
clignotant  
FLASH ON/OFF 27  
touches 27  
utilisation dans l'instruction  
PRINT 27  
direct 70  
guillemets  
accès 45,56  
touche 47  
utilisation avec PRINT 47-48,56  
graphique 84-89  
immédiat 62, 70  
insertion  
accès 25, 28-29  
touche 25  
multi-couleur 102-103  
vidéo-inversé  
accès 27, 45-46  
touches 27, 45  
utilisation avec les instructions  
PRINT 45-46  
majuscule/graphique 28,84-86  
accès 24, 26, 28  
imprimer des graphiques  
84-89  
touche SHIFT 24,28  
majuscule/minuscule 28,84-86  
accès 24, 26, 28  
imprimer des graphiques  
84-89  
touche SHIFT 24,28
- Modems 214
- Moniteur langage machine  
184-192
- Moniteur  
connection à l'ordinateur 13-16
- MONITOR 142-170
- Musique  
durée des notes 107
- 
-

---

---

instruction SOUND  
106,204-205  
voix 106-107  
volume 105,156,171

## **N**

NEW commande 51,126,170  
NEXT instruction 77-78,143,170  
Nombres  
aléatoires 65-67, 171  
calcul 61-67,72-73,203  
puissance 65  
ordre d'exécution 62-63,65-67  
fractions 59-60  
opérateur mathématique 59  
pi 59,163  
opérateur relationnel  
59,167-168  
notation scientifique 60  
signes (+ -) 59  
variables 64,170  
Notation scientifique 60

## **O**

ON...GOSUB 143,170  
ON...GOTO 143  
OPEN commande 144-145,170  
Opérateurs 59,167-168

## **P**

PAINT 100-101,145,170  
PEEK fonction 160,170  
Périphérique 19-21  
Pi 59,163  
PC curseur point  
modes graphiques 84-89  
instruction LOCATE 142,170  
positionnement 25-26,49-51  
POKE instruction 146,170  
Points-virgules  
dans les instructions PRINT

contre les virgules 55-56  
PRINT  
calcul 61-62,72-73,203  
message 71-73  
format de sortie 145-150  
mode direct 62-70  
mode programme 71-73  
zones 55-56  
ponctuations 55-56  
PRINT# 147,170  
PRINT USING 145-150,170  
Programme de conversion  
170-173  
Programmation  
Commande BASIC  
69-82,118-129  
fonctions BASIC  
30-31,158-163  
Instructions BASIC  
69-82,130-157  
Touches de fonctions 28-31,123  
moniteur de langage machine  
184-192  
Mode 70  
PUDEF 150-151,170

## **R**

Ralentir le défilement de l'écran  
26  
READ instruction 74-75, 151, 170  
Redéfinition des touches 30-31  
REM instruction 80-81, 151-152  
RENAME commande 126,171  
RENUMBERING lignes de  
programme 70, 126-127  
Renumérotation AUTOMatique  
118,169  
Reprendre l'affichage d'un  
programme 152,171  
Reset touche 10  
RESTORE 152,171  
RESUME 152,171  
RETURN 152, 171  
RIGHT\$ fonction 163, 171

---

---

RND fonction 66-67, 160-161,  
171  
RS-232  
interface 209-213  
port 12  
RUN commande 30-31, 70, 127,  
171

## S

SAVE commande 127-128, 171  
Sauvegarde de programmes  
cassettes 37  
erreurs disquettes 41, 171-183  
disquettes 40-41, 127-128  
DSAVE 30-31, 40, 127  
SAVE 127-128, 171  
SCALE 152-153, 171  
SCNCLR instruction 93, 153, 171  
SCRATCH commande 128, 171  
SOUND instruction 106, 153,  
171, 204-205  
SPC fonction 163, 171  
SSHAPE 153-154, 171  
STOP instruction 155, 171  
Stopper l'affichage du  
programme 24, 52  
Sous-programmes 80  
SYS instruction 34, 155, 171

## T

TAB fonction 88, 163, 171  
Tableaux 164  
Tableau de dépannage 17-18  
TEDMON 184-192  
Texte  
dans les graphiques 92-93  
dans les instructions PRINT  
71-73  
chaînes de caractères 70-73,  
162-163

TI\$ fonction 164, 171  
Touches  
Commodore 26  
Contrôle CTRL 26  
ESCAPE 28-29  
de fonction 30-31  
HELP 30-32, 123  
TRAP instruction 155, 171  
TROFF 156, 171  
TRON 155, 171  
TV

types d'antenne 13, 17  
sélection du canan 11, 15-17  
raccordement 13-18  
alimentation secteur 9,  
11, 13, 15-17

## V

Variables  
virgule flottante 64, 164  
entière 64, 164  
noms réservés 116-117,  
165-166  
texte chaîne de caractères 64,  
73, 164  
types 64, 164-166  
voir aussi Assignation des  
données  
VERIFY commande 70, 128-129,  
171, 192  
Vider 25-26  
Virgules  
dans les instructions PRINT  
55-56  
séparations des nombres 61  
contre points-virgules 55-56  
VOLUME 105, 156, 171  
Voix 106-107

## W

WAIT instruction 156, 171











































## **VOTRE GUIDE UTILISATEUR PLUS/4...**

**Le Plus/4 a un concept unique : quatre logiciels intégrés.**

**Il s'adapte avec efficacité à la maison, il est idéal pour effectuer les premiers pas en informatique.**

**Il permet également d'effectuer la gestion des petites entreprises.**

**Ses capacités : 64 Ko de mémoire centrale, graphique, son, possibilités de programmation en BASIC (évolué 3.5).**

**Ce manuel vous aidera à faire les premiers pas en informatique et vous donnera toutes les informations concernant votre machine. Les cours sont simples, progressifs et à la portée de tous, même pour les personnes n'ayant jamais abordé l'informatique.**

**Pour ceux déjà familiarisés avec ce domaine, Commodore a prévu un chapitre « encyclopédie » regroupant toutes les commandes utilisées pour le BASIC. Ce guide vous donnera aussi les informations concernant les capacités les plus « pointues » de votre Commodore Plus/4.**

**Vous pourrez également apprendre à utiliser les fonctions des logiciels intégrés (traitement de texte, fichier, tableur, graphique), en utilisant le « Guide des logiciels intégrés ».**

 **commodore**  
COMPUTER

**Commodore France SARL  
8, rue Copernic  
75116 Paris**



**This was brought to you  
from the archives of**

**<http://retro-commodore.eu>**