

# **SANYO**

# **PHC-25**

## **Programmation**

## **BASIC**

Version du 2025-06-05



Dédicace à tous ceux qui préserve le patrimoine informatique.

Un grand merci à Olipix qui a lancé les GAME JAM.



# Table des matières

Table des matières.....	5
Introduction .....	9
ABS.....	12
ASC .....	13
CHR\$.....	14
CLEAR .....	15
CLOAD.....	16
CLOAD?.....	17
CLS.....	18
COLOR .....	19
CONSOLE.....	23
CONT.....	24
COS.....	25
CSAVE .....	26
CSRLIN .....	27
CTOFF.....	28
CTON.....	29
DATA .....	30
DEFFN .....	31
DIM.....	33
ELSE .....	34
END .....	35
EXEC.....	36
EXP .....	37

FRE.....	38
FOR .....	39
GOSUB .....	40
GOTO.....	41
IF .....	42
INKEY\$.....	43
INP .....	44
INPUT .....	45
INPUT# .....	46
INT.....	47
KEY .....	48
LCOPY .....	49
LEFT\$.....	50
LEN.....	51
LET .....	52
LINE.....	53
LIST.....	55
LLIST.....	56
LOCATE.....	57
LOG .....	58
LPOS .....	59
LPRINT.....	60
MID\$.....	61
NEW .....	62
NEXT.....	63
ON GOSUB .....	64

ON GOTO .....	65
OUT .....	66
PAINT .....	67
PEEK .....	68
PLAY .....	69
POINT .....	71
POKE .....	72
POS .....	73
PRESET .....	74
PRINT.....	75
PRINT#.....	76
PSET .....	77
READ .....	78
REM .....	79
RESTORE.....	80
RETURN .....	81
RIGHT\$ .....	82
RND .....	83
RUN .....	84
SCREEN.....	85
SCREEN 1 .....	86
SCREEN 2 .....	87
SCREEN 3 .....	88
SCREEN 4 .....	89
SCRIN .....	90
SGN .....	91

SIN .....	92
SLOAD .....	93
SOUND .....	94
SPC .....	95
SQR .....	96
SSAVE .....	97
STICK .....	98
STOP .....	99
STRIG .....	100
STR\$ .....	101
TAB .....	102
TAN .....	103
TIME .....	104
THEN .....	105
USR .....	106
VAL .....	107

# Introduction

Reprise des documentations sur le BASIC du SANYO PCH-25.

La documentation étant peu loquace, le but est d'avoir un document le plus juste et le plus complet possible.

Les tests de syntaxes et autres ont été effectuées pour la plupart sur émulateur.

D'autres ont pu être fait sur des machines réelles.

Le second but de ce document c'est la **GAME JAM 2025** sur **SANYO PHC-25**.

De façon à fournir une bonne base de documentation pour pouvoir créer un logiciel.

Il est toujours possible de participer aux GAME JAM après coup.

C'est d'ailleurs très utile pour compléter les logithèques de ces machines obscures.

La programmation du **Z80** n'est pas abordée ici.

Toutefois, cette documentation essaye d'expliquer le How To pour un programme binaire au travers les instructions pour le langage machine.

La documentation étant quasi nulle sur le sujet.

Pour les **INP** et **OUT** du **SANYO PHC-25** nous aimerais bien avoir plus d'indication.

## **Les variables**

Dans la documentation, elles sont nommées constantes.

Considérant le peu de place en mémoire leur étant réservée, ce n'est pas trop étonnant qu'elles s'appellent constante.

C'est un peu alambiqué, on ne parlera que de variables.

## **Chaîne de caractères**

Elle est constituée par un ou plusieurs caractères enfermés entre guillemets. La forme générale étant "caractère ..."

Exemple :

"PHC 25"

NB : Les guillemets ("") ne sont pas compris dans la chaîne de caractères.

Utilisée pour mémoriser des chaînes de caractères jusqu'à 255 caractères. Les chaînes de caractères variables utilisent le signe \$ après le nom variable.

Exemple : P\$, DX\$, XI\$, CD\$

Une restriction du BASIC fait qu'il n'est possible d'avoir que 2 caractères pour le nom de la chaîne de caractère.

Exemple :

AB\$  
ABC\$

Sont une même variable.

## **Valeurs**

Il y a 3 notations possible, voir ci-après.

Il est possible de stocker des nombres de  $10^{-39}$  à  $10^{+38}$ .

Il n'y a pas de possibilité de notation binaire.

## **Notation point (.) fixe**

On obtient un nombre négatif ou positif qui inclut le point décimal avec neuf chiffres significatifs.

Exemples :

33.169  
-45.765

## Notation décimal flottant

On obtient un nombre positif ou négatif exprimé en notation exponentielle. L'exposant va de -38 à +38.

La mantisse peut avoir jusqu'à 9 chiffres significatifs.

Exemple :

123456789 E-38

## Notation hexadécimale

La constante hexadécimale est exprimée par une combinaison de caractères de 0 à F précédés par &H.

Exemple :

&HC000

## Tableau

Le groupage de données peut être fait sous le terme TABLEAU. La dimension du tableau est définie par la valeur entre parenthèses.

Exemple :

A(3)  
K\$(4)

A contient 3 éléments [0,2].

K contient 4 éléments de type chaîne de caractère.

Il est possible d'avoir des tableaux à n dimensions.

Exemple :

X(3,3,3)

# **ABS**

## ***Objet***

Renvoie la valeur absolue de l'argument fourni.

## ***Syntaxe***

ABS(x)  
x : Nombre.

## ***Exemple***

```
10 PRINT ABS(9*(-2))
```

## ***Mots clés associés***

# **ASC**

## ***Objet***

Renvoie le code ASCII du premier caractère d'une chaîne de caractères.

## ***Syntaxe***

ASC(a\$)

a\$ : Chaîne de caractère.

## ***Exemple***

```
10 A$="TEST"  
20 PRINT ASC(A$)
```

Affiche 84 car la valeur ASCII du T est 84.

## ***Mots clés associés***

CHR\$

# **CHR\$**

## ***Objet***

Génère un caractère de code ASCII égal à la valeur fournie en argument.  
Cette fonction est utilisée pour envoyer un caractère spécial.

## **Syntaxe**

CHR\$(x)

x : nombre entier de 20 à 127.

## ***Exemple***

```
PRINT CHR$(42)
PRINT CHR$(20)+CHR$(49)
```

## **Mots clés associés**

ASC, LEFT\$, MID\$, RIGHT\$, STR\$, VAL

# **CLEAR**

## ***Objet***

Remettre à zéro toutes les variables numériques et à valeur nulle toutes les valeurs alphanumériques.

## **Syntaxe**

CLEAR I [J]

I : Taille de la mémoire pour y stocker des données. Obligatoire.

J : Optionnel. Limite supérieure de la mémoire pour le programme BASIC.

## ***Exemple***

```
CLEAR 400, &hE050
```

400 caractères réservé.

Limite de mémoire supérieur &hE050, adresse au-delà de laquelle on met un programme en langage machine.

## **Mots clés associés**

FRE

# **CLOAD**

## ***Objet***

Charger en mémoire un programme ou un fichier existant sur une cassette.

CLOAD efface le programme précédent avant de charger celui qui est sur la cassette.

## ***Syntaxe***

CLOAD"nom"

nom : Le nom est une chaîne dont les six premiers caractères sont significatifs et doivent être identiques au nom sous lequel le programme a été sauvegardé par CSAVE.

Le guillemet de fin n'est pas obligatoire.

CLOAD seul chargera le premier programme de la cassette.

## ***Exemple***

**CLOAD**

**CLOAD"othelo"**

## ***Mots clés associés***

CLOAD?

# **CLOAD?**

## ***Objet***

Une comparaison est faite entre le programme en mémoire et celui qui est sur la cassette.

## **Syntaxe**

CLOAD?"nom"

Nom : nom du programme à tester. Seul les 6 premier caractères compte.

## ***Exemple***

`CLOAD?"prog"`

## **Mots clés associés**

# **CLS**

## ***Objet***

Tous les caractères affichés sur l'écran sont effacés.

Tous les caractères affichés sur l'écran définie par CONSOLE sont effacés.

## **Syntaxe**

CLS

## ***Exemple***

```
10 CLS
```

Efface l'écran.

```
10 CONSOLE 3,5  
20 CLS
```

Seules les lignes 3 à 8 seront effacées.

Les lignes sont numérotées de 0 à 15.

## ***Mots clés associés***

CONSOLE, SCREEN

# COLOR

## **Objet**

Désignation de la couleur des caractères, ou graphiques affichés sur l'écran.

Même si on désigne les mêmes chiffres, la couleur affichée est différente selon le mode écran choisi (1,2,3,4).

## **Syntaxe**

COLOR a,b,c

a : défini selon les modes la couleur du texte,

b : défini selon les modes 2, 3 et 4, la couleur de fond,

c : défini la palette de couleurs utilisées dans les modes écran 1,2,3 et 4.

## **Mode 1**

Dans ce mode, la couleur de l'écran est désignée par les valeurs a et c sans tenir compte de b.

La couleur de l'écran entier est changée en exécutant l'instruction COLOR et ensuite CLS.

Il n'y a pas de graphique possible en mode 1. Ce mode est textuel, de type console.

Les paramètres a et c peuvent être utilisés seul ou ensemble.

a	c	Caractère	Fond
1	1	Vert clair	Vert
2	1	Vert	Vert clair
1	2	Blanc	Orange
2	2	Orange	Blanc

Exemples :

**COLOR , ,1**

Passe la console sur la palette 1, Vert et Vert clair. Le changement de palette affecte tout l'écran.

**COLOR 1**

Passe le texte en mode normal.

**COLOR 2**

Passe le texte en mode inverse vidéo.

```
COLOR 1,1,1  
COLOR 1,2,1
```

Ces 2 instruction on le même résultat, le paramètre b est ignoré.

## Les étrangetés du SCREEN 1

Il est possible de faire COLOR de 1 à 4.

Ce qui permet d'avoir les 4 couleurs en même temps sur l'écran.

## Mode 2

La valeur de a définie la couleur des caractères. Il y a 4 couleurs disponibles.

La valeur de b définie la couleur de fond de l'écran. Il y a 9 couleurs disponibles.

La valeur de c définit la palette de couleur utilisée. Il y a 2 palettes disponibles.

Préalable :

```
SCREEN 2,1,1
```

Ci-dessus, le mode 2 est appliqué à l'écran 1 et est affiché.

NB : Le PHC-25 dispose de la possibilité d'avoir 2 écrans au détriment de l'espace mémoire.

Le choix de la palette se fait via la valeur de c.

Palette no 1 :

a	b	c	Caractères	Fond	Graphiques
0	0	1	Vert	Noir	Noir
1	1	1	Vert	Vert	Vert
2	2	1	Vert (inversé)	Jaune	Jaune
3	3	1	Orange	Bleu	Bleu
4	4	1	Orange inversé)	Rouge	Rouge
5	5	1	Orange inversé)	Blanc	Blanc
6	6	1	Orange inversé)	Bleu clair	Bleu clair
7	7	1	Orange inversé)	Violet	Violet
8	8	1	Orange inversé)	Orange	Orange

Exemple :

```
COLOR , ,1
```

Passe sur la palette 1, ceci affecte tout l'écran.

Palette no 2 :

a	b	c	Caractères	Fond	Graphiques
0	0	2	Orange	Noir	Noir
1	1	2	Orange	Blanc	Blanc
2	2	2	Orange (inversé)	Bleu clair	Bleu clair
3	3	2	Vert	Violet	Violet
4	4	2	Vert (inversé)	Orange	Orange
5	5	2	Vert (inversé)	Vert	Vert
6	6	2	Vert (inversé)	Jaune	Jaune
7	7	2	Vert (inversé)	Bleu	Bleu
8	8	2	Vert (inversé)	Rouge	Rouge

Exemple :

```
COLOR , ,2
```

Passe sur la palette 2, ceci affecte tout l'écran.

Exemples :

```
1000 SCREEN 2,1,1
1010 COLOR , ,1
1020 CLS
1030 PSET (120,120),3
1040 LINE(12,12)-(48,48),2,BF
1050 LOCATE 5,12:PRINT"Hello World!"
9000 GOSUB 9980
9910 END
9980 K$="" : K$=INKEY$ : IF K$="" THEN 9980
9990 RETURN
```

## Mode 3

La valeur de a définit la couleur des caractères. Il y a 4 couleurs disponibles.

La valeur de b définit la couleur de fond de l'écran. Il y a 5 couleurs disponibles.

La valeur de c définit la palette de couleur utilisée. Il y a 2 palettes disponibles.

NDR : La documentation originale est insuffisante.

## Mode 4

La valeur de a définit la couleur des caractères. Il y a 2 couleurs disponibles.

La valeur de b définit la couleur de fond de l'écran. Il y a 2 couleurs disponibles.

La valeur de c définit la palette de couleur utilisée. Il y a 2 palettes disponibles.

a	b	c	Caractères	Fond	Graphiques
1	0	1	Vert	Noir	Vert
0	1	1	Noir	Vert	Noir
1	0	2	Blanc	Noir	Blanc
0	1	2	Noir	Blanc	Noir

Exemple :

```
COLOR , ,1
```

Passe sur la palette 1, ceci affecte tout l'écran.

## Addendum

Les valeurs a, b, c, peuvent varier de 0 à 255 avec des effets de bord selon le mode choisi.

## Mots clés associés

SCREEN, CLS,

# **CONSOLE**

## ***Objet***

Désigne les limites de déplacement vers le bas et le haut de l'écran.  
Seules les lignes désignées vont scroller ou être effacées.

## **Syntaxe**

CONSOLE a,b

a : ligne de départ

b : nombre de ligne (>0) – [1,254]

## ***Exemple***

```
10 CLS
20 PRINT "TITRE"
30 CONSOLE 1,15
40 CLS
```

Le titre n'est pas effacé par le CLS de la ligne 40.

## **Mots clés associés**

CLS, LOCATE

# **CONT**

## ***Objet***

CONT veut dire continuer.

Utilisé pour reprendre l'exécution d'un programme après une suspension par la commande CTRL/C ou après l'exécution d'une instruction STOP.

L'exécution reprend à l'endroit où elle a été interrompue.

CONT n'est plus utilisable si une modification quelconque a été faite dans le programme, ni après une instruction END.

L'écran affiche « Can't continue ».

Il ne semble pas possible d'utiliser l'instruction CONT dans un programme BASIC.

## ***Syntaxe***

CONT

## ***Exemple***

CONT

## ***Mots clés associés***

END, STOP

# **COS**

## ***Objet***

Le cosinus de X exprimé en radians.  
Le résultat est donné en simple précision.

## **Syntaxe**

COS(x)  
X : angle en radians.

## ***Exemple***

```
PRINT COS(0.4)
```

Affichera 0.921060995.

## **Mots clés associés**

SIN, TAN

# **CSAVE**

## ***Objet***

Sauvegarde sur cassette d'un programme en mémoire.

Remarques : Chaque programme sauvegardé est identifié par un nom.

Le BASIC copie le programme existant en mémoire sur la cassette lorsque la commande CSAVE est exécutée. Elle se sert des six premiers caractères pour le « nom ».

Avant l'utilisation d'un CSAVE, il est nécessaire de vérifier que le magnétophone est branché correctement et que les touches RECORD et PLAY sont bien enfoncées.

## ***Syntaxe***

## ***Exemple***

```
CSAVE MonProgramme
```

La référence sur la cassette du programme sera « MonPro ». Les 6 premier caractères.

## ***Mots clés associés***

CLOAD

# **CSRLIN**

## ***Objet***

Obtient la ligne de la position du curseur.

## **Syntaxe**

CSRLIN

## ***Exemple***

```
10 LOCATE 12,7  
20 PRINT CSRLIN
```

Le curseur est en position colonne 12, ligne 7, le programme affichera 7.

## **Mots clés associés**

LPOS, POS

# **CTOFF**

## ***Objet***

Fermer l'interrupteur de commande à distance du magnétophone (REMrte Control) arrêtant ainsi le défilement de la cassette.

## ***Syntaxe***

CTOFF

## ***Exemple***

## ***Mots clés associés***

CTON,

# **CTON**

## ***Objet***

Mettre le magnétophone en marche dans le cas où la touche PLAY est enfoncée.  
La commande à distance du magnétophone est enclenchée par cette commande, démarrant ainsi la cassette.

Vérifier que la touche PLAY du magnétophone est bien enclenchée.

## ***Syntaxe***

CTON

## ***Exemple***

### ***Mots clés associés***

CTOFF,

# **DATA**

## ***Objet***

Utilisé pour enregistrer dans un programme des valeurs numériques et des caractères constants qui seront appelés par des instructions READ.

Les chaînes de caractères doivent être encadrées de guillemets

Les lignes DATA peuvent se situer n'importe où dans le programme.

## ***Syntaxe***

DATA ...

## ***Exemple***

```
10 FOR I=1 TO 3: READA$(I): NEXT I
20 FOR I=1 TO 3: READB(I): NEXT I
30 FOR I=1 TO 3: PRINT A$(I); "="; B(I); "F": NEXT I
100 DATA "PAIN", "VIN", "JAMBON", 2, 6.50, 10
```

## ***Mots clés associés***

READ, RESTORE

# **DEFFN**

## ***Objet***

Définir une fonction écrite par l'utilisateur et lui donner un nom.

NDR : La documentation indique qu'il serait possible d'avoir N paramètre séparé par une virgule.  
Apparemment ce n'est pas fonctionnel, il faudrait approfondir le sujet.

Il est possible de se passer des espace séparateurs.

## ***Syntaxe***

DEF FN <nom>(x)=<fonction>

nom : le nom donné à la fonction.

x : le paramètre.

fonction : la formule de la fonction.

## ***Exemple***

```
1000 REM
1010 DEF FN SEC(X)=1/COS(X) : REM secante
1020 DEF FN CSC(X)=1/SIN(X) : REM cosecante
1030 DEF FN COT(X)=1/TAN(X) : REM cotangente
1040 DEF FN SINH(X)=(EXP(X)-EXP(-X))/2: REM sinus hyperbolique
1050 DEF FN COSH(X)=(EXP(X)~EXP(-X))/2: REM cosinus hyperbolique
1060 DEF FN TANH(X)=EXP(-X)/EXP(X)+EXP(-X))*2+1: REM tangente hyperbolique
1070 DEF FN SECH(X)=2/(EXP(X)+EXP(-X)): REM secante hyperbolique
1080 DEF FN CSCH(X)=2/(EXP(X)-EXP(-X)): REM cosecante hyperbolique
1090 DEF FN COTH(X)=EXP(-X)/(EXP(X)-EXP(-X)*2+1: REM cotangente hyperbolique
1100 DEF FN ARCSINH(X)=LOG(X+SQR(X*X+1)): REM inverse sinus hyperbolique
1110 DEF FN ARCCOSH(X)=LOG(X+SQR(X*X-1)): REM inverse cosinus hyperbolique
1120 DEF FN ARCTANH(X)=LOG((1+X)/(1-X))/2: REM inverse tangente hyperbolique
1130 DEF FN ARCSECH(X)=LOG((SQR(-X*X+1)+1/X)): REM inverse secante hyperbolique
1140 DEF FN ARCCSCH(X)=LOG((SGN(X)*SQR(X*X+1))/X: REM inverse cosecante hyperbolique
1150 DEF FN ARCCOTH(X)=LOG((X+1)/(X-1))/2: REM inverse cotangente hyperbolique
```

Défini des fonctions supplémentaires de trigonométrie.

Exemple :

```
10 DEFNF(X)=3*X+6  
20 PRINT FNF(5)
```

### ***Mots clés associés***

# **DIM**

## ***Objet***

Préciser les valeurs maximales des indices pour les tableaux et réserver la place utile en mémoire.  
En même temps on assure la place utile pour ranger la mémoire.

Lorsqu'un tableau de variables est utilisé sans instruction DIM, la valeur maximum est considérée comme étant 10 lignes en 3 dimensions.

Une erreur apparaît si l'on fait appel à une valeur plus grande.

La valeur minimale est zéro.

## **Syntaxe**

## ***Exemple***

```
10 DIM B$(20,10), C(30), E(10,3,5,7)
```

## **Mots clés associés**

## **ELSE**

Suivant les résultats d'un test, l'on prendra une décision ou l'on choisira une suite pour l'exécution du programme.

Si le résultat du test est vrai, c'est-à-dire différent de zéro, alors l'instruction THEN suivra jusqu'à ELSE ou la fin de la ligne et sera exécutée.

Si l'expression est égale à zéro (valeur fausse), seules les instructions suivant ELSE, si elles existent, seront exécutées.

### ***Syntaxe***

IF <expression> THEN <instruction> ELSE <instruction>

### ***Exemple***

```
5 A=-5: B=-10
10 IF A<>0 THEN B=10 ELSE B=0: GOTO 100
30 PRINT A,B
40 END
100 PRINT A,B
110 A=A+1
120 GOTO 10
```

### ***Mots clés associés***

IF, THEN

# **END**

## ***Objet***

Terminer l'exécution d'un programme.

L'instruction END peut être placée n'importe où dans le programme pour terminer l'exécution.

Le message BREAK n'est pas imprimé lorsque l'instruction END est exécutée.

La commande CONT ne peut être utilisée après.

## ***Syntaxe***

END

## ***Exemple***

```
9990 END
```

Le programme s'arrête lorsqu'il atteint la ligne 9990.

## ***Mots clés associés***

**BREAK, CONT,**

# **EXEC**

## ***Objet***

Le programme en basic ira à l'adresse indiquée par la variable X pour exécuter un programme écrit en langage machine (hexadécimal). Le programme aura été écrit grâce à l'instruction POKE et devra se finir automatiquement par la valeur (&HC9) (RET) qui permettra un retour au langage Basic.

## **Syntaxe**

EXEC &Hx

X est l'adresse hexadécimale du programme à exécuter.

## ***Exemple***

```
10 CLEAR 100, &HF000
20 LET J=00
30 INPUT "valeur Hex"; A$
40 N=VAL("&H"+A$)
50 IF A$="ZZ" OR A$="zz" THEN 90
60 POKE &HF000+(J), N
70 J=J+1
80 GOTO 30
90 END
100 CLS
110 EXEC &HF000
120 PRINT "SORTIE"
130 END
```

Valeur à passer : 3E, 41, 32, 8F, 60 C9

Le programme met le caractère A à la position 16,5 de l'écran (LOCATE).

Son adresse mémoire étant &h608F.

Voir chapitre sur l'écran du SANYO PHC-25 pour plus d'information.

## **Mots clés associés**

USR

# **EXP**

## ***Objet***

Renvoie la valeur de e à la puissance x.

## ***Syntaxe***

EXP(x)

x : Puissance pour e.

## ***Exemple***

```
10 PRINT EXP(6)
```

Retourne : 403.428794.

## ***Mots clés associés***

# FRE

## ***Objet***

Retourne la valeur de la mémoire disponible.

## ***Syntaxe***

FRE(x)

x :

FRE(x\$)

x\$ :

## ***Exemple***

```
10 PRINT FRE(x)
```

Nombre d'octets restant dans la mémoire pour le BASIC.

```
10 PRINT FRE(X$)
```

Nombre d'octets restant dans la mémoire pour les chaînes de caractères.

## ***Mots clés associés***

[CLEAR](#)

# **FOR**

## ***Objet***

FOR NEXT répète et exécute une chaîne de commandes le nombre de fois désigné. La première expression X, est la valeur initiale, Y est une valeur limite à ne pas dépasser. Les instructions entre FOR et NEXT sont exécutées autant de fois que nécessaire avec chaque fois incrémentation du compteur variable de la quantité indiquée après l'instruction STEP pour que le compteur dépasse la limite Y. Dans un tel cas, l'exécution passe à l'instruction suivante NEXT.

## ***Syntaxe***

FOR variable=X TO Y (STEP Z)

## ***Exemple***

### ***Mots clés associés***

NEXT, STEP, TO

# **GOSUB**

## ***Objet***

Aller à un sous-programme et revenir.

## **Syntaxe**

GOSUB adresse

adresse : numéro de ligne du sous-programme.

## ***Exemple***

```
1010 PRINT"BCD"
1020 GOSUB 1050
1030 PRINT"DEF"
1040 END
1050 PRINT"123"
1060 RETURN
```

## **Mots clés associés**

**GOTO, ON, RETURN,**

# **GOTO**

## ***Objet***

Saut inconditionnel à une ligne qui a été désignée.  
Une erreur est signalée si la ligne désignée n'existe pas.

## ***Syntaxe***

GOTO adresse

adresse : ligne cible du saut inconditionnel.

## ***Exemple***

### ***Mots clés associés***

GOSUB, ON, RETURN

# **IF**

## ***Objet***

Suivant les résultats d'un test, l'on prendra une décision ou l'on choisira une suite pour l'exécution du programme.

Si le résultat du test est vrai, c'est-à-dire différent de zéro, alors l'instruction THEN suivra jusqu'à ELSE ou la fin de la ligne et sera exécutée.

Si l'expression est égale à zéro (valeur fausse), seules les instructions suivant ELSE, si elles existent, seront exécutées.

## ***Syntaxe***

IF <expression> THEN <instruction> ELSE <instruction>

## ***Exemple***

```
5 A=-5: B=-10
10 IF A<>0 THEN B=10 ELSE B=0: GOTO 100
30 PRINT A,B
40 END
100 PRINT A,B
110 A=A+1
120 GOTO 10
```

## ***Mots clés associés***

THEN, ELSE

# **INKEY\$**

## ***Objet***

Obtient le caractère de la touche lorsque l'on appuie sur le clavier.  
Permet de tester quelle touche a été appuyée.

## ***Syntaxe***

INKEY\$

## ***Exemple***

```
100 IF INKEY$="" THEN GOTO 100
110 IF INKEY$<>"A" THEN GOTO 100
```

## ***Mots clés associés***

STICK,STRIG

# **INP**

## ***Objet***

Lecture d'un octet sur le port I (adresse de &H00 à &HFF).

## ***Syntaxe***

INP(x)

x : adresse du port de 0 à 255.

## ***Exemple***

```
PRINT INP(&H8F)
```

Renvoie la valeur du port &H8F.

## ***Mots clés associés***

**OUT**

# **INPUT**

***Objet***

**Syntaxe**

***Exemple***

**Mots clés associés**

# **INPUT#**

## ***Objet***

Lecture de données dans un fichier séquentiel et affectation des valeurs aux variables de la liste.  
Le numéro du fichier est le numéro associé au fichier au moment de son ouverture.

La liste contient le nom des variables où l'on doit affecter les données existantes sur le fichier.  
Il faut que les types concordent et il n'y a pas de point d'interrogation sur l'écran comme dans le "INPUT".

Lorsque l'on entre des données par l'intermédiaire du magnétophone, il est nécessaire  
d'enregistrer ces données par une instruction PRINT£

## ***Syntaxe***

INPUT#-n,val

n :

0 : clavier

1 : magnétophone

Val : élément à lire, la donnée enregistrée doit correspondre au type.

## ***Exemple***

## ***Mots clés associés***

# **INT**

## ***Objet***

Retourne le plus grand entier inférieur ou égal à X.

## ***Syntaxe***

INT(x)

x : nombre dont on veut l'entier.

## ***Exemple***

```
PRINT INT(99.99)
```

Renvoie 99.

```
PRINT INT(-32,12)
```

Renvoie -33, c'est bien la valeur inférieur du nombre passé en argument.

## ***Mots clés associés***

# **KEY**

## ***Objet***

Les touches F1, F2, F3 et F4 sont reprogrammables en mode normal et en mode SHIFT. Ce qui donne 8 touches au total.

Une touche peut contenir 8 caractères exécutables au maximum.

## ***Syntaxe***

KEYn,char

## ***Exemple***

```
KEY1 , "CONSOLE"
```

La touche F1 écrira « CONSOLE ».

```
KEY7 , "CTON"+CHR$(13)
```

La touche F3 (avec SHIFT) mettra le moteur du magnétophone en marche.

## ***Mots clés associés***

# **LCOPY**

## ***Objet***

Envoi de l'information contenue sur l'écran vers l'imprimante. Aussi appelée "HARD COPY". Ne marche que sur imprimante graphique.

## **Syntaxe**

LCOPY

## ***Exemple***

## **Mots clés associés**

LPRINT

# **LEFT\$**

## ***Objet***

Sélectionne le nombre spécifié de caractère à gauche de la chaîne fourni en argument.

## ***Syntaxe***

LEFT\$(A\$,x)

A\$: Chaîne de caractère sur laquelle on veut faire l'extraction.

x : nombre de caractère à extraire.

Si x vaut 0, renvoie une chaîne vide.

La valeur de x ne doit pas dépasser 255.

Si la chaîne est plus courte que le x spécifié, LEFT\$ renvoie toute la chaîne.

## ***Exemple***

## ***Mots clés associés***

# **LEN**

## ***Objet***

Renvoie la longueur de la chaîne fournie en argument.

## ***Syntaxe***

LEN(A\$)

A\$ : Chaîne dont on veut connaître la longueur

## ***Exemple***

```
10 D$ = "LES QUATRE SAISONS"  
20 PRINT LEN(D$)
```

## ***Mots clés associés***

# **LET**

## ***Objet***

La valeur de "expression" est substituée et devient une variable.

En fait, le signe égal (=) suffit pour constituer l'affectation de la variable.

L'instruction LET a le même but.

## **Syntaxe**

LET NomVariable = Expression

## ***Exemple***

```
10 LET A=B+1
```

## **Mots clés associés**

# **LINE**

## ***Objet***

Relier les points par une ligne ou les insérer dans un rectangle.

Deux points désignés par des références données sont reliés par une ligne qui a une couleur désignée.

Lorsque "B" est désigné, un rectangle est dessiné utilisant les deux points comme angles opposés.  
De plus, si F est désigné, le rectangle est coloré par la couleur spécifiée ou celle de la dernière instruction COLOR.

Les définitions en ligne et en colonne sont assujetties à la définition du SCREEN (1,2, 3 ,4).

## **Syntaxe**

LINE(X1,Y1)-(X2,Y2),(couleur),(BF)

X1 : coordonnées sur l'axe horizontale

Y1 : coordonnées sur l'axe verticale

X2 : coordonnées sur l'axe horizontale

Y2 : coordonnées sur l'axe verticale

Couleur :

B : Bordure, il faut utiliser la lettre « B »

F : Fill (remplissage), il faut utiliser la lettre « F »

## ***Exemple***

```
LINE (5,5)-(20,20) , 3
```

Les points de coordonnées 5,5 et 20,20 sont reliés par une ligne de la couleur 3

```
LINE (5,5)-(20,20) , 3,B
```

Les points 5,5 et 20 ,20 appartiennent à un rectangle. Ils sont les deux points opposés.

```
Exemple : LINE(10,8)-(190,130) ,2,BF
```

Un rectangle colorié de couleur 2 passe par les points de coordonnées 10-8 et 190-130

## ***Mots clés associés***

COLOR, SCREEN

# **LIST**

***Objet***

**Syntaxe**

***Exemple***

**Mots clés associés**

## **LLIST**

***Objet***

**Syntaxe**

***Exemple***

**Mots clés associés**

# **LOCATE**

## ***Objet***

Le curseur est déplacé vers une position sur l'écran qui est exprimée par une position horizontale et verticale.

Le coin gauche en haut de l'écran est 0,0.

La position horizontale va de 0 à 31 pour les modes 1,2,4 d'écran.

Toutefois, lorsque l'écran 3 est utilisé, la position horizontale s'arrête à 15.

La position verticale est de 0 à 15 pour tous les modes d'écran.

## ***Syntaxe***

LOCATE X,Y

X : Colonne.

Y : Ligne.

## ***Exemple***

```
10 FOR I=1 TO 100
20 LOCATE 10,10
30 PRINT I
40 NEXT
```

## ***Mots clés associés***

# **LOG**

## ***Objet***

Renvoie le logarithme népérien de l'argument. Cet argument doit être positif.

## ***Syntaxe***

LOG(x)

## ***Exemple***

```
PRINT LOG(5.4)
```

## ***Mots clés associés***

# **LPOS**

## ***Objet***

Renvoie la position de tête du pointeur de ligne dans le tampon de sortie de l'imprimante.

## ***Syntaxe***

LPOS(X)

## ***Exemple***

## ***Mots clés associés***

# **LPRINT**

## ***Objet***

Écriture de données sur l'imprimante.

Cette instruction est identique à PRINT, la sortie se fait sur l'imprimante connectée au PHC 25.

## **Syntaxe**

LPRINT val

Val est soit du texte soit une valeur numérique.

## ***Exemple***

```
LPRINT"Bonjour . "
```

## **Mots clés associés**

# **MID\$**

## ***Objet***

Renvoie une sous chaîne de la longueur spécifiée depuis un rang spécifié.

## ***Syntaxe***

MID\$(A\$,R,L)

R : Rang de début d'extraction.

L : Nombre de caractère à extraire.

R et L doivent être compris entre 0 et 255.

## ***Exemple***

```
10 A$= "BONJOUR"
20 PRINT MID$(A$,2,4)
```

## ***Mots clés associés***

LEFT\$, RIGHT\$

# **NEW**

***Objet***

**Syntaxe**

***Exemple***

**Mots clés associés**

# **NEXT**

***Objet***

***Syntaxe***

***Exemple***

***Mots clés associés***

FOR, STEP

# **ON GOSUB**

## ***Objet***

Branchement multiple suivant la valeur de l'expression. C'est cette valeur qui détermine sur quelle ligne le programme sera branché.

Si la valeur est I, le programme sera branché sur I.

Si l'expression n'est pas intégrale (entière) les décimales seront arrondies.

Si la valeur de l'expression est zéro ou négative ou si elle est plus grande que les numéros de ligne, le programme passera à la ligne suivante.

## ***Syntaxe***

ON Valeur GOSUB L1,L2,Ln...

Valeur est la valeur sur laquelle doit s'effectué le test.

En fonction du résultat, branchement sur la ligne n.

## ***Exemple***

```
ON K GOSUB 100,200,300,400
```

Si K<=0, ligne suivante

Si K>=5, ligne suivante

Si K=1, ligne 100

Si K=2, ligne 200

Si K=3, ligne 300

Si K=5, ligne 400

Après le RETURN, ligne suivante.

## ***Mots clés associés***

GOSUB, GOTO, RETURN

# **ON GOTO**

## ***Objet***

Branchement multiple suivant la valeur de l'expression. C'est cette valeur qui détermine sur quelle ligne le programme sera branché.

Si la valeur est I, le programme sera branché sur I.

Si l'expression n'est pas intégrale (entière) les décimales seront arrondies.

Si la valeur de l'expression est zéro ou négative ou si elle est plus grande que les numéros de ligne, le programme passera à la ligne suivante.

## **Syntaxe**

ON Valeur GOTO L1,L2,Ln...

Valeur est la valeur sur laquelle doit s'effectué le test.

En fonction du résultat, branchement sur la ligne n.

## ***Exemple***

ON K GOTO 100,200,300,400

Si K<=0, ligne suivante

Si K>=5, ligne suivante

Si K=1, ligne 100

Si K=2, ligne 200

Si K=3, ligne 300

Si K=5, ligne 400

## **Mots clés associés**

GOSUB, GOTO, RETURN

# **OUT**

## ***Objet***

Cette instruction permet d'envoyer sur le port I l'octet J.

## **Syntaxe**

OUT I,J

I : No de port,

J : Octet.

## ***Exemple***

```
OUT &h20,&h64
```

La valeur &h64 sera envoyée sur le port no &h20.

## **Mots clés associés**

INP

# **PAINT**

## ***Objet***

Les coordonnées x et y définissent 1 point se situant dans une zone de l'écran. Cette zone est peinte avec la couleur C sauf le périmètre de couleur P.

## **Syntaxe**

PAINT (X,Y),c,p

X : coordonnée horizontale.

Y : coordonnée verticale.

c : couleur selon le SCREEN.

p : couleur selon le SCREEN.

## ***Exemple***

```
10 LINE(0,0)-(255,0),2,BF  
20 LINE(255,0)-(255,191),2,BF  
30 LINE (255,191)-(5,186),2,BF  
40 LINE(5,186)-(0,0,),2,BF  
50 PAINT (80,80),4,2
```

NDR : exemple à revoir car nécessite SCREEN.

## **Mots clés associés**

LINE, PRESET, PSET, SCREEN

# **PEEK**

## ***Objet***

Renvoie la valeur de l'octet de l'adresse spécifiée. Les valeurs retournées vont de &H00 à &HFF.

## ***Syntaxe***

PEEK(x)

X : adresse mémoire dont on veut lire l'octet.

## ***Exemple***

```
A=PEEK(&H2AOO)
```

## ***Mots clés associés***

POKE

# PLAY

## **Objet**

La musique est produite par le générateur de son (vendu en option).

## **Syntaxe**

PLAY"ox [sx] [mx] [vx] [lx] [tx] [rx] [x+] [x-] n"

Les fonctions entre crochet sont facultatives

Avec n: (cdefgab), notation Anglo-saxonne de la musique.

c	d	e	f	g	a	b
do	ré	mi	fa	sol	la	si

Dans l'instruction PLAY, pour la chaîne de caractères, les caractères suivants sont utilisés pour donner des significations spéciales :

ox	Désigne l'octave (l'octave plus haute est o4) initial 4
sx	Désigne la forme de l'enveloppe (se référer à l'annexe D.)
mx	Désigne la période de l'enveloppe (1<=x<=65535)
vx	Désigne le volume (0<=x<=15) initial 8
lx	Désigne la durée du son (0<=x<=64)
tx	Désigne la vitesse (tempo) du son initial 170
rx	Désigne la durée de la période de silence (aucun son) (1<=x<=64)
x+	Le son est remonté d'un demi-ton
x-	Le son est descendu d'un demi-ton

Tous les caractères de PLAY doivent être en minuscule.

## **Exemple**

PLAY"o4 18 v2 s5 a"

octave 4 ; longueur 8 ; volume 2 ; enveloppe 5 voix ; note LA

```
10 FOR I=0 TO 10
20 PLAY"o6132c"
30 NEXT I

10 FOR I=0 TO 10
20 PLAY"o41cdfgab05c"
30 NEXT I

10 PLAY
"o7fardl34fafdbcfa05farecferafbcl56gbfddeaedaeaddcffdfbfebacdfeadaeo4129ggabgag
afdfdfaffaddeeaddbccffcfdo7ffacca"
```

## Mots clés associés

SOUND

# **POINT**

## ***Objet***

Renvoie le numéro de la couleur sélectionnée par les coordonnées (x,y).

## ***Syntaxe***

POINT(x,y)

## ***Exemple***

```
10 PSET(7,30),3  
20 PRINT POINT(7,30)
```

## ***Mots clés associés***

PRESET, PSET

# **POKE**

## ***Objet***

Insert un octet à une adresse désignée de la mémoire.

## **Syntaxe**

POKE Expression 1, Expression 2

Expression 1 : Adresse mémoire

Expression 2 : Octet à placer (&H00 à &HFF)

## ***Exemple***

```
10 POKE &H6010,&H81  
20 POKE 4053,24
```

```
POKE &h6080,42
```

Affiche le caractère « \* » première colonne 5<sup>ème</sup> ligne.

## ***Mots clés associés***

EXEC, PEEK, USR

# **POS**

## ***Objet***

Renvoie la position du curseur sur la colonne.

## **Syntaxe**

POS(X)

## ***Exemple***

```
10 LOCATE 12,7  
20 PRINT POS(X)
```

## **Mots clés associés**

# **PRESET**

## ***Objet***

Effacer le point de coordonnées (X,Y) sur l'écran.

Forte dépendance vis-à-vis de l'instruction SCREEN.

## **Syntaxe**

PRESET (X,Y)

X : coordonnée horizontale.

Y : coordonnée vertical.

## ***Exemple***

```
5 SCREEN 4,1,1 TO 20
6 CLS
9 FOR X=1 TO 100
10 PSET (X,100),3
11 IF X> 20 THEN PRESET (X,100)
12 IF X> 50 THEN PSET(X,100),3
15 NEXT
20 GOTO 20
```

## **Mots clés associés**

LINE, PAINT, PSET, SCREEN

# **PRINT**

## ***Objet***

Écrire des données sur l'écran.

Si la liste d'expressions est absente, une ligne blanche s'en suivra.

La position de chaque élément de la ligne est déterminée par la ponctuation.

## ***Syntaxe***

PRINT élément[,|;[élément]]...

Les caractères doivent obligatoirement être encadrés entre guillemets.

Lorsque l'on écrit un nombre multiple d'expressions, chacune doit être séparée par une virgule (,) ou un point-virgule (;).

Une virgule entre deux expressions de la liste fait que l'expression suivant la virgule est imprimée au début de la zone suivante.

Si c'est un point-virgule, l'expression se fait immédiatement après la dernière expression écrite.

Un ou plusieurs espaces entre deux expressions ont le même effet que le point-virgule.

## ***Exemple***

```
10 A$="PROGRAMME"
20 B$="No . " :C=15
30 D$="exemple de programme"
40 PRINT D$
50 PRINT
60 PRINT A$;B$;C
70 PRINT "suivant"
```

## ***Mots clés associés***

LOCATE, SPC, TAB

# **PRINT#**

## ***Objet***

Écriture des données dans un appareil désigné.

La désignation de l'appareil se fait par le numéro suivant PRINT #

## **Syntaxe**

PRINT#-n

N=0 : écran

N=1 : magnétophone

N=3 : imprimante.

## ***Exemple***

### **Mots clés associés**

LPRINT, PRINT

# **PSET**

## ***Objet***

Un point ayant une couleur désignée est affiché à l'endroit spécifié par les coordonnées X,Y.  
La couleur c dépend du SCREEN 1,2,3,4 et de la couleur choisie (COLOR).

## **Syntaxe**

PSET(X, Y) ,c

X : coordonnée horizontale

Y : coordonnée verticale

C : couleur

## ***Exemple***

## **Mots clés associés**

COLOR, LINE, PRESET, PSET, SCREEN

# **READ**

## ***Objet***

Lecture des valeurs dans une instruction DATA et affectation aux variables citées.

Une instruction READ doit toujours être utilisée en relation avec une ou plusieurs instructions DATA.

L'instruction READ distribue les valeurs données dans l'instruction 'DATA aux variables mentionnées.

Ces variables sont numériques ou de type chaîne et doivent s'accorder, sinon il peut y avoir "erreur de syntaxe".

S'il reste des DATA inutilisés, le READ suivant les utilise.

S'il n'en reste pas, ils seront ignorés.

On peut relire les DATA grâce à l'instruction RESTORE.

## ***Syntaxe***

READ valeur[,valeur]

## ***Exemple***

## ***Mots clés associés***

DATA, RESTORE

# **REM**

## ***Objet***

Insérer des remarques et des notes dans le programme.

Bien que l'instruction REM ne soit pas exécutable, elle figure dans la liste avec le contenu du texte qui l'accompagne.

L'instruction REM ne peut pas être continuée en la séparant des autres instructions par un point-virgule (;).

## ***Syntaxe***

REM <texte>

## ***Exemple***

```
10 REM Menu
```

## ***Mots clés associés***

# **RESTORE**

## ***Objet***

Permettre de ·relire des données à partir d'une certaine ligne.

Après l'instruction RESTORE, le premier READ prend ses valeurs dans la première instruction DATA.

Lorsque le numéro de ligne est spécifié, la lecture se fera à partir de cette ligne.

## ***Syntaxe***

**RESTORE [n]**

N : numéro de ligne pour relire les données.

## ***Exemple***

**RESTORE**

Restaure la lecture au premier DATA du programme.

**RESTORE 100**

Restaure la lecture à la ligne 100

## ***Mots clés associés***

DATA, READ

# **RETURN**

*Objet*

*Syntaxe*

*Exemple*

**Mots clés associés**

GOSUB

# **RIGHT\$**

## ***Objet***

Sélectionne le nombre spécifié de caractère à droite de la chaîne fourni en argument.

## ***Syntaxe***

RIGHT\$(A\$,)

A\$: Chaîne de caractère sur laquelle on veut faire l'extraction.

x : nombre de caractère à extraire.

Si x vaut 0, renvoie une chaîne vide.

La valeur de x ne doit pas dépasser 255.

Si la chaîne est plus courte que le x spécifié, RIGHT\$ renvoie toute la chaîne.

## ***Exemple***

```
10 A$="ABCDEFG"  
20 PRINT RIGHT$(A$, 4)
```

## ***Mots clés associés***

# RND

## ***Objet***

Initialiser le générateur de nombres aléatoires.

Si l'expression facultative manque, l'exécution est suspendue.

Si l'on utilise RND sans initialiser le générateur à chaque exécution du programme, la même suite de nombres aléatoires sera produite.

## **Syntaxe**

RND(x)

x>0 commence une nouvelle séquence

x=0 donne le dernier nombre généré

x<0 génère un nouveau nombre aléatoire

## ***Exemple***

```
10 FOR I=1 TO 5
20 PRINT INT(RND(1.0)*100)
30 NEXT I

10 CLS:PRINT"RND TEST"
20 A=1:B=13
30 I=RND(-RND(1)*100):PRINT"SEED:";I: REM NEW SEED
40 FOR I=1 TO 10
50 N=A+INT((B-A+1)*RND(1)): REM BETWEEN [A,B]
60 PRINT "E";I;" :";A;" -"; B; " ="; N
70 NEXT I
80 END
```

## **Mots clés associés**

# **RUN**

## ***Objet***

Lancement de l'exécution du programme stocké en mémoire.  
La touche F1 contient par défaut l'instruction RUN.

## **Syntaxe**

RUN [n]

N : numéro de ligne, n'est pas obligatoire.

## ***Exemple***

RUN

## ***Mots clés associés***

END, STOP

# **SCREEN**

## ***Objet***

Le PHC 25 a deux pages d'écran qui sont définies à la mise sous tension.

- mode avec 1 page d'écran, la mémoire disponible sera de 14265 octets.
- mode avec 2 pages d'écran, la mémoire disponible sera de 8121 octets.

Ce qui correspond à la question posée à l'allumage « SCREEN ? ».

Il ne faut pas confondre le choix du nombre de pages utilisées et la façon d'utiliser ces pages.

## ***Syntaxe***

SCREEN a,b,c

La valeur a fixe le choix de la taille d'écran.

Il y a 4 possibilités d'utilisation d'une page suivant le tableau ci-dessous.

a	1	2	3	4
<b>Texte</b>	16L 32C	16L 32C	16L 16C	16L 32C
<b>Graphique</b>	(16x32)*	64x48	128x192	256x192

(\*) Les instructions graphiques fonctionnent.

La valeur b désigne le numéro de la page affectée par les instructions qui suivent.

La valeur c désigne le numéro de la page affichée à l'écran. (NDR : selon le choix fait au démarrage de la machine).

## ***Exemple***

Voir ci-après pour les 4 mode d'écran.

## ***Mots clés associés***

# **SCREEN 1**

## ***Objet***

Voir information avec SCREEN.

## ***Syntaxe***

SCREEN 1,b,c

## ***Exemple***

## ***Mots clés associés***

CLS

## **SCREEN 2**

### ***Objet***

Voir information avec SCREEN.

### ***Syntaxe***

SCREEN 2,b,c

### ***Exemple***

### ***Mots clés associés***

## **SCREEN 3**

### ***Objet***

Voir information avec SCREEN.

### ***Syntaxe***

SCREEN 3,b,c

### ***Exemple***

### ***Mots clés associés***

## **SCREEN 4**

### ***Objet***

Voir information avec SCREEN.

### ***Syntaxe***

SCREEN 4,b,c

### ***Exemple***

### ***Mots clés associés***

# **SCRIN**

## ***Objet***

Renvoie la valeur ASCII du caractère de coordonnées d'écran (c,l).

## ***Syntaxe***

SCRIN(c,l)

c : numéro de la colonne

L : numéro de la ligne

## ***Exemple***

```
5 CLS
10 PRINT"ABCDEF"
20 Z=SCRIN(5,0)
30 PRINT Z
```

La valeur renvoyée est bien 70 qui est le code ASCII de F.  
Les coordonnées texte commence en haut à gauche (0,0).

## ***Mots clés associés***

CHR\$,

# **SGN**

## ***Objet***

Connaître le signe d'un nombre.

## ***Syntaxe***

SGN(x)

x : Nombre dont on veut connaître le signe.

Renvoie 1 si  $x \geq 0$

Renvoie -1 si  $x < 0$

## ***Exemple***

```
PRINT SGN(912)
```

## ***Mots clés associés***

# **SIN**

## ***Objet***

Calcule le sinus de l'angle en radians. Le résultat est obtenu en précision simple.

## ***Syntaxe***

SIN(x)

x : angle dont on veut le sinus.

## ***Exemple***

```
PRINT SIN(0.32)
```

Résultat : .314566561

## ***Mots clés associés***

COS, DEF FN, TAN

# **SLOAD**

## ***Objet***

L'information de l'écran mémorisée par la cassette est enregistrée dans la mémoire de l'écran.  
Si la page de l'écran utilisée n'est pas celle qui a été mémorisée, la commande SLOAD est arrêtée et une ERREUR est affichée.

## ***Syntaxe***

SLOAD "nom"

Le nom est sensible à la casse.

8 caractères maximum.

## ***Exemple***

```
SLOAD "data"
```

## ***Mots clés associés***

SSAVE

# **SOUND**

## ***Objet***

Une sortie de son est produite directement en envoyant les commandes au générateur de son. La sortie du son désiré peut être faite en désignant l'information (DATA) qui est dans le registre et les registres de 0 à 13 listés ci-dessous.

## ***Syntaxe***

SOUND registre,data

Registre :

Data :

## ***Exemple***

### ***Mots clés associés***

PLAY

# **SPC**

## ***Objet***

Écrit le nombre spécifié d'espace.

## **Syntaxe**

SPC(x)  
x : nombre d'espace.

## ***Exemple***

```
10 PRINT"GAME"
20 PRINT SPC(3)
30 PRINT"OVER"
```

Place 3 espaces entre les 2 mots.

## **Mots clés associés**

PRINT, LPRINT

# **SQR**

## ***Objet***

Renvoie la racine carrée de x.

## ***Syntaxe***

SQR(x)  
x doit être zéro ou positif

## ***Exemple***

```
PRINT SQR ( 10)
```

Résultat : 3.16227766

## ***Mots clés associés***

DEF FN

# **SSAVE**

## ***Objet***

L'information affichée sur l'écran est sauvegardée sur la cassette du magnétophone.

SSAVE peut servir de commande ou d'instruction.

Avant l'exécution d'un SAVE, s'assurer que le magnétophone est correctement branché et que les touches RECORD et PLAY sont bien enfoncées.

## **Syntaxe**

SSAVE "fichier"

Fichier : nom du fichier qui contiendra les informations de l'écran. Le nom est sensible à la casse.

## ***Exemple***

```
SSAVE "data"
```

## **Mots clés associés**

# **STICK**

## ***Objet***

Permet de tester les manettes et le clavier (touches de direction)  
Nécessite d'avoir le synthétiseur pour les joysticks.

## **Syntaxe**

STICK(x)

x=0 : Touches du clavier

x=1 : Joystick 1

x=2 : Joystick 2.

## ***Exemple***

```
10 IF STICK(1)=1 then print "nord"
20 IF STICK(1)=2 then print "nord-est"
30 IF STICK(1)=3 then print "est"
40 IF STICK(1)=4 then print "sud-est"
50 IF STICK(1)=5 then print "sud"
60 IF STICK(1)=6 then print "sud-ouest"
70 IF STICK(1)=7 then print "ouest"
80 IF STICK(1)=8 then print "nord-ouest"
90 IF STICK(1)=0 then print "manette inactive"
```

## **Mots clés associés**

STRIG

# **STOP**

## ***Objet***

Arrête le programme en exécution.

Le STOP peut être mis à n'importe quel endroit du programme pour arrêter son exécution. Toutefois, contrairement à l'instruction END, lorsque STOP est exécuté, le message suivant apparaît :

BREAK in nnnnn

Nnnnn étant le numéro de ligne où se situe le STOP.

## ***Syntaxe***

STOP

## ***Exemple***

```
9990 STOP
```

## ***Mots clés associés***

END

# **STRIG**

## ***Objet***

Test du bouton de Joystick ou barre espace.

## ***Syntaxe***

STRIG(x)

x=0 : Barre espace.

x=1 : Bouton Joystick 1.

x=2 : Bouton Joystick 2.

Renvoie 1 si l'appuie est détecté, sinon 0.

## ***Exemple***

```
10 IF STRIG(0)=0 THEN GOTO 10
```

Attend l'appuie sur la barre d'espace.

## ***Mots clés associés***

STICK

# **STR\$**

## ***Objet***

Transforme un nombre en chaîne de caractères.

## ***Syntaxe***

STR(x)

## ***Exemple***

### ***Mots clés associés***

LEFT\$, RIGHT\$, MID\$, VAL

# **TAB**

## ***Objet***

Met des espaces jusqu'à la colonne spécifiée.

## ***Syntaxe***

TAB(x)

x : numéro de colonne.

## ***Exemple***

```
10 PRINT"XXX";TAB(10);"XXX"  
20 PRINT"PHC-25";TAB(10);"PROGRAMME"
```

Affichera la 2<sup>ème</sup> partie du PRINT à partir de la colonne 10.

## ***Mots clés associés***

SPC

# **TAN**

## ***Objet***

Renvoie en simple précision la tangente de l'angle, exprimée en radians

## ***Syntaxe***

**TAN(x)**

x : angle dont on veut la tangente.

## ***Exemple***

```
PRINT TAN(1.34)
```

Renvoie : 4.67344123

## ***Mots clés associés***

COS, SIN

# **TIME**

## ***Objet***

Compteur de temps qui ajoute 1 chaque 1/360 secondes.  
Le compteur à une précision faible.

## **Syntaxe**

TIME

## ***Exemple***

```
10 A=TIME  
20 B=TIME-A  
30 PRINT INT(B/360)  
40 GOTO 20
```

## ***Mots clés associés***

# **THEN**

## ***Objet***

Suivant les résultats d'un test, l'on prendra une décision ou l'on choisira une suite pour l'exécution du programme.

Si le résultat du test est vrai, c'est-à-dire différent de zéro, alors l'instruction THEN suivra jusqu'à ELSE ou la fin de la ligne et sera exécutée.

Si l'expression est égale à zéro (valeur fausse), seules les instructions suivant ELSE, si elles existent, seront exécutées.

## ***Syntaxe***

IF <expression> THEN <instruction> ELSE <instruction>

## ***Exemple***

```
5 A=-5: B=-10
10 IF A<>0 THEN B=10 ELSE B=0: GOTO 100
30 PRINT A,B
40 END
100 PRINT A,B
110 A=A+1
120 GOTO 10
```

## ***Mots clés associés***

ELSE, IF

# **USR**

## ***Objet***

Exécute un programme fait par l'utilisateur en langage machine (Z80).

## ***Syntaxe***

USR(x)

## ***Exemple***

### ***Mots clés associés***

EXEC, PEEK, POKE

# **VAL**

## ***Objet***

Renvoie la valeur de la chaîne de caractères fournie en argument.

Si le premier caractère de la chaîne n'est pas +, -, &, ou un chiffre, la valeur renvoyée est 0.

En notation hexadécimale [0-9],[A-F], les autres caractères sont ignorés.

## ***Syntaxe***

VAL(x\$)

x\$ : Chaîne à convertir

## ***Exemple***

```
10 FOR I=1 TO 3
20 READ A$
30 PRINT VAL("&H"+A$);
40 NEXT
50 DATA C3,40,FF
```

## ***Mots clés associés***

# L'écran du SANYO PCH-25

## ***Les 2 choix au boot***

Lors du démarrage de la machine, apparaît :

SCREEN ?

La réponse est soit 1 soit 2.

À ne pas confondre avec l'instruction SCREEN.

Réponses :

1 – Un seul écran de travail

2 – 2 écrans de travail

La bascule entre les 2 écrans se fera avec CTRL + Q.

## ***Les résolutions d'écran***

Le tableau ci-dessous donne les possibilités du PHC-25.

a	1	2	3	4
Texte	16L 32C	16L 32C	16L 16C	16L 32C
Graphique	32x16	48x64	128x192	256x192

Les coordonnées texte commencent en haut à gauche en 0,0.

Les coordonnées graphiques commencent en haut à gauche en 0,0.

La première position est l'axe des abscisses.

La deuxième position est l'axe des ordonnées.

Exemple :

```
10 LINE(0,0)-(255,0)
```

Trace une ligne en haut de l'écran.

Selon le SCREEN choisi, la dimension de la ligne changera.

Pour les instructions graphiques, il faut toujours se référer à la dimension maximum, soit 256x192 pixels.

## ***Caractères non affichables par CHR\$***

Les caractères spéciaux de &h10 à &h1C (voir table page 75 de la documentation) ne sont pas affichable par PRINT CHR\$(n).

Il faut donc passer par la solution du poke ci-dessus.

## **Le screen 1**

Le screen 1 est de 32 colonnes sur 16 lignes.

Il s'agit d'un mode texte uniquement. Son adresse mémoire va de &h6000 à &h61FF.

Soit 512 octets, ce qui correspond bien à 32x16 caractères.

La numérotation par du coin en haut à gauche de l'écran, de 0 à 31 et 0 à 15.

Table d'adressage simplifiée :

Ligne	Colonne 1	Colonne 32
0	&h6000	&h601F
1	&h6020	&h603F
2	&h6040	&h605F
3	&h6060	&h607F
4	&h6080	&h609F
5	&h60A0	&h60BF
6	&h60C0	&h60DF
7	&h60E0	&h60FF
8	&h6100	&h611F
9	&h6120	&h613F
10	&h6140	&h615F
11	&h6160	&h617F
12	&h6180	&h619F
13	&h61A0	&h61BF
14	&h61C0	&h61DF
15	&h61E0	&h61FF

Il est possible de simuler le LOCATE x,y directement sur l'écran en tapant directement l'adresse avec la valeur du caractère souhaité.

La valeur de l'octet mis à l'écran correspond au générateur de caractère (voir page 75 de la documentation) ou l'annexe.

Il est possible de tracer des lignes, des rectangles, des rectangles pleins, etc. avec les fonctions BASIC. Il faut cependant considérer que la résolution est de 32x16 pixel.

Il est donc possible de faire du ASCII art sur cette résolution.

Exemple :

```
1000 REM Poke direct sur le screen 1
1010 SCREEN 1,1,1: CLS
1020 A=0
1030 FOR I=&h10 to &h85
1040 POKE &h6000+A,I
1050 A=A+1
1060 NEXT I
1070 GOSUB 9980
1080 END
9980 K$="" :K$=INKEY$ :IF K$="" THEN 9980
9990 RETURN
```

Ce programme affiche les caractères de &h10 à &h85 sur un screen 1.

Ce principe permet d'aller assez vite pour afficher à l'écran car on écrit directement dans la mémoire vidéo du PHC-25.

Exemple :

Pour faire un LOCATE 10,10 (colonne 10, ligne 10).

L'adresse de la ligne 10 est &h6140.

La colonne 10 est un incrément de &hA.

La localisation est donc &h614A.

**POKE &h614A,&h2A**

Ou

**POKE &h6140+10,&h2A**

Affichera « \* » en colonne 10 ligne 10.

CONSOLE ne bloquera pas le POKE.

## **Le Screen 2**

### **Description**

Le Screen 2 est un mixe entre mode texte et mode graphique. La résolution texte est de 32 colonnes et 16 lignes. La résolution graphique est de 64x48 pixels. Il y a de nombreuse restriction sur le fonctionnement de ce mode.

Un graphique ne peut pas s'afficher correctement sur du texte.

Le fond pour la partie graphique doit toujours être BLACK (valeur 0).

Il faut donc utiliser CONSOLE pour définir les lignes sur lesquelles mettre du texte ou du graphique.

Il y à 2 palettes de couleurs disponibles (voir COLOR). Un changement de palette affecte l'écran entier.

Concernant le texte, il est possible de faire des PRINT CHR\$, sauf des caractères spéciaux de &h10 à &h1C.

La mémoire vidéo est la même qu'avec le SCREEN 1. Soit de &h600 à &h61FF.

Il est donc aussi possible de poker dedans une valeur de &h00 à &hFF.

### **Particularité des pixels en mode 2**

Si la couleur de fond n'est pas noire (valeur 0), alors cette partie d'écran est considérée comme du texte. La couleur de fond doit donc être obligatoirement noire pour y mettre du graphique.

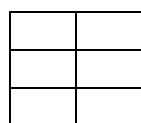
L'écran se subdivise en 32 colonnes par 16 lignes.

Il y a alors 512 cases disponibles qui ont la dimension d'un caractère.

Ces cases ne peuvent avoir que 2 couleurs dont obligatoirement le noir (le fond).

L'autre couleur étant choisie dans la palette (8 possibles, voir COLOR).

Les cases ont une dimension de 2x3.



Subdivisé à la résolution :

O	O	O	O	O	O	O	O
O	O	O	O	O	O	O	O
O	O	O	O	O	O	O	O
O	O	O	O	O	O	O	O
O	O	O	O	O	O	O	O
O	O	O	O	O	O	O	O
O	O	O	O	O	O	O	O
O	O	O	O	O	O	O	O

Chaque « O » est un pixel de la résolution de 256x192.

Si la figure ci-dessus représente le coin haut, gauche :

Elle couvre les points de (0,0) à (7,11), un PSET dans ces limites mets la même couleur sur tous les pixels déjà allumés.

Un PSET mis sur 0,0 allume tous les pixels de (0,0) à (3,3).

L'adresse de cette case mémoire vidéo est &h6000.

Un POKE sur &h6000 va donc allumer 0 à 6 des portions avec une des 9 couleur selon la palette (1 ou 2, voir COLOR).

Quand 2 lignes graphiques se croisent, à l'intersection, la case prend la couleur de la dernière ligne qui s'y affiche Pour les pixels déjà allumés (ils font 4x4 en screen 2).

Positionner un pixel (PSET) implique la mise à la même couleur des pixels déjà présent (0 étant non présent).

## POKE mémoire vidéo

La plage d'adresse va de &h6000 à &h61FF.

256 valeurs sont possibles (&h00 à &hFF).

Soit l'octet &x00000000. Dont nous trouvons les bits dans la matrice.

5	4
3	2
1	0

&x00000000, matrice éteinte.

&x00111111, matrice allumée, couleur 0 ?

## ***Le Screen 3***

## ***Le Screen 4***