systèmes divers

ROUTINE TESTCHR POUR HP 71B

Olivier Arbey

Voici une routine pour HP 71B qui permet de déterminer le code du prochain caractère spécial que l'on va créer avec CHARSET. Rappelons que ce code (ASCII) est un nombre à partir de 128 qui représente le numéro d'ordre de création du caractère.

Trois mois plus tard, vous n'êtes plus censé vous souvenir du nombre exact de caractères que vous avez redéfini. TESTCHR est là pour vous aider : un petit coup de RUN affiche le code que prendra le prochain caractère redéfini.

Cette routine est présentée sous la forme d'un sous-programme, et la variable locale C est utilisable à partir de n'importe quel fichier programme.

L'astuce préconisée consiste à utiliser la longueur de la chaîne CHARSET$ car celle-ci contient la représentation colonne par colonne de tous les caractères spéciaux (ligne 50). Comme un caractère est codé sur six colonnes, un rapide calcul... donne la ligne 60. La variable C est transférée par référence, c'est-à-dire qu'elle est utilisée par le programme appelant, mais peut être modifiée par le sous-programme, ce qu'il ne se prive pas de faire en ligne 60 !

```
10 CALL TESTCHR(C)
20 DISP "N0 CHR$ suivant
   :";C
30 END
40 SUB TESTCHR(C)
50 C=LEN(CHARSET$)
60 C=C/6+128
70 END SUB

END SUB          99BYTES
```

DEPLACER UN POINT A TRAVERS L'ECRAN

Pierre Barnouin

Vous avez disposé un superbe labyrinthe sur l'écran de votre X07, et il ne manque plus que le point dont vous commanderez la progression à l'aide des quatre curseurs ; la position initiale est (X, Y) et s'étend jusqu'au bord droit de l'écran (X = 119), mais doit respecter tous les obstacles.

Les trois lignes ci-dessous rendent caducs tous les échafaudages complexes d'INKEY$, PSET ou PRESET conventionnels auxquels vous pourriez vous référer. Le BEEP Z,3 facilite votre progression en « couinant » pour signaler les obstacles qui entravent son déplacement.

Pensez à programmer les valeurs initiales de X et Y, ainsi qu'à encadrer votre labyrinthe afin d'empêcher le point de quitter l'écran.

Programme pour X07

```
100 PSET(X,Y) : Z=STICK(0) : U=X-(Z=3)+(Z=7)
    : V=V+(Z=1)-(Z=5)
110 IFPOINT(U,V)THENBEEPZ,3ELSEPRESET(X,Y)
    : X=U : Y=U
120 IFX<119THEN100
```

INVERSION VIDEO

Marc Dutendas

Ce programme pour Dragon 32 permet de réaliser l'inversion vidéo, texte ou graphique. Il sert simplement à charger le langage machine et à contrôler qu'aucune erreur ne se trouve dans les DATA. Toutes les lignes REM (') peuvent être omises ; par conséquent, une fois le langage machine chargé, on peut effacer le programme.

Quelques explications. La dernière valeur dans chaque ligne de DATA est la somme des autres valeurs.

$31-$32 (ligne 210) est le numéro de la ligne de DATA en cours de lecture.
$BA (ligne 510) est le début de la page graphique.
$B7 (ligne 560) est la fin de la page graphique.
Les routines se trouvent aux adresses 32700 pour le texte et 32715 pour le graphique.
Pour les appeler, il suffit de faire EXEC 32700 ou EXEC 32715.

Par ailleurs, voici une petite astuce qui vous permettra peut-être de trouver des choses intéressantes. Entrez :

10 POKE 65480, 1 : GOTO 10

Puis faites RUN.

A partir de ce moment-là vous visualisez les 512 premiers octets. Pour revenir en mode normal, faites BREAK.

Programme pour Dragon 32

```
10 'Inversion video
20 CLEAR200,32700
30 FOR T=32700 TO 32725 STEP5
40 C=0
50 FOR T1=0 TO 4
60 READ A$
70 A=VAL("&H"+A$)
80 POKE T+T1,A
90 C=C+A
100 NEXT T1
110 READ B
120 IF C<>B THEN 210
130 NEXT T
140 PRINT"FIN"
150 END
200 'Ligne de data avec erreur
210 X=PEEK(&H31)*256+PEEK(&H32)
220 PRINT"ERREUR LIGNE";X
230 END
250 DATA 8E,04,00,A6,84,444
260 DATA 88,40,A7,80,8C,635
270 DATA 06,00,26,F5,39,346
280 DATA 9E,BA,EC,84,43,779
290 DATA 53,ED,81,9C,B7,788
300 DATA 26,F6,39,39,39,455
400 'Programme en assembleur
410 'Inversion texte
420 'LDX #$0400
430 'LDA ,X
440 'EORA #$40
450 'STA ,X+
460 'CMPX #$0600
470 'BNE F5
480 'RTS
500 'Inversion graphique
510 'LDX #BA
520 'LDD ,X
530 'COMA
540 'COMB
550 'STD ,X++
560 'CMPX #B7
570 'BNE F6
580 'RTS
```