SONY

MSX 2

MANUAL DE REFERENCIA
PARA PROGRAMACION

BASIC-MSX
Versión 2.0

HIT BIT

PROLOGO

En este manual se explica la forma de empleo de todas las órdenes y funciones del BASIC-MSX Versión 2.0.

Para aquellas personas que tengan alguna experiencia en BASIC, este Manual de Programación debe constituir una valiosa guía de consulta para programar en BASIC-MSX Versión 2.0. Las personas que se pongan por primera vez frente a un ordenador deben recurrir al manual "Introducción al BASIC-MSX Versión 2.0.".

El BASIC-MSX Versión 2.0 recibe en este manual la denominación BASIC-MSX2.

NOTA
Es posible que su ordenador MSX tenga varias teclas distintas a las mostradas en este manual.
Consulte la siguiente tabla:

En el teclado	En este manual
↵	RETURN
⇧	SHIFT
←	BS
⇪	CAP
SUP	DEL
EFE	HOME

INDICE

Capítulo 1 Particularidades del BASIC—MSX2

Capítulo 2 Ordenes, funciones y mensajes de error

Apéndice

CAPITULO 1

PARTICULARIDADES DEL BASIC–MSX2

CONSTANTES Y VARIABLES

CONSTANTES ... Corresponden siempre a un valor determinado.

Constantes manejadas por el BASIC–MSX2

- Constantes numéricas
  - Entero
    - Expresión decimal
    - Expresión hexadecimal
    - Expresión octal
    - Expresión binaria
  - Coma fija
  - Coma flotante
- Constantes alfanuméricas

Constantes numéricas

Entero	Gama: Desde –32768 hasta 32767 Expresión decimal: Tal como es (ej ... 123, –4567) Expresión hexadecimal: Añadir &H (ej ... 8HA0) Expresión octal: Añadir &0 o & (ej. &0146, &765) Expresión binaria: Añadir &B (ej ... &B11010110)
Coma flotante	6 (precisión simple) o 14 (precisión doble) dígitos significativos. Parte exponente: Desde –64 hasta +62 (la base del exponentes es 10) (Ej. 1.345E–10)

Precisión simple y precisión doble

coma fija o coma flotante } Precisión simple o precisión doble (la selección inicial es precisión doble).

Precisión simple (6 dígitos significativos)	Cuando se utiliza E como símbolo de potencia: Ejemplo: 12.23E3. Cuando se añade !al final: Ejemplo: 18.32!
Precisión doble (14 dígitos significativos)	Cuando se utiliza D como símbolo de potencia: Ejemplo: 23.456789D10 Cuando no se especifica ningún tipo en particular: Ejemplo: 3456789012.34 Cuando se añade #al final: Ejemplo: 3456789012.34 #

Constantes alfanuméricas
Cadenas de caracteres con hasta 255 caracteres encerrados entre comillas (" ").
La constante de cadena que no contiene ningún caracter o espacio recibe el nombre de "cadena nula".

"BASIC MSX2"
"AE830"
" "(cadena nula)

VARIABLES ... pueden tomar cualquier valor en el programa.

Nombre de las variables

- Solo son significativos los dos primeros caracteres (el primer caracter debe ser un caracter alfabético).
- No se pueden utilizar como nombre de variable las palabras reservadas del BASIC–MSX2 (nombres de órdenes, nombres de funciones, etc.) ni las cadenas de caracteres que contengan una palabra reservada.

Declaración del tipo de variable

Declaración mediante un caracter	Declaración mediante una instrucción DEF	Tipo declarado
Añadir % Ejemplo: A%	DEFINT Ejemplo: DEFINT A	Entero
Añadir ! Ejemplo: B!	DEFSNG Ejemplo: DEFSNG B	Precisión simple
Añadir # Ejemplo: C #	DEFDBL Ejemplo: DEFDBL C	Precisión doble
Añadir $ Ejemplo: D$	DEFSTR Ejemplo: DEFSTR D	Cadena

- Cuando se utilice un carácter de declaración de tipo diferente para una variable después de haberse ejecutado una sentencia de declaración (DEFINT ... etc.), tendrá prioridad el caracter de declaración.

Conversión de variables

Ejemplo	Descripción
A%=100.5 PRINT A% 100	Cuando se asigna a una variable declarada como número entero una constante distinta a la declarada, la asignación se efectúa de acuerdo con el tipo de variable.
B=8!/7! PRINT B 1.1428571428571	El resultado aritmético se asigna con precisión doble.
B!=8/7 PRINT B! 1.14286	Como se ha puesto en una variable un caracter de declaración de precisión simple, se asigna un resultado aritmético de precisión simple.
PRINT 6.7 OR 4.3 6	La operación lógica se efectúa después de que los datos han sido convertidos en enteros
A!=1.2345678 B=A! PRINT B 1.23457	Cuando se asigna a una variable de precisión doble un dato que solo tiene dígitos de precisión simple, el número de dígitos sigue siendo el mismo.

EXPRESIONES Y SU UTILIZACION

CONSTITUCION DE LAS EXPRESIONES

Constantes, variables, funciones y símbolos aritméticos.

EVALUACION DE LAS EXPRESIONES ARITMETICAS

Operador aritmético	Significado	Ejemplo	Orden de prioridad
+	Adición (X + Y)	X+Y	6
–	Sustracción (X–Y)	X–Y	
*	Multiplicación (X x Y)	X * Y	3
/	División (X ÷ Y)	X/Y	
^	Potencia (X^2)	X^2	1
–	Cambio de signo (–X)	–X	2
\	División entera	6.7\2.3	4
MOD	Resto de división entera	X MOD 10	5

EXPRESIONES RELACIONALES

Se compara el valor de dos datos y se da un resultado en forma de verdadero (–1) o falso.

Operador relacional	Significado	Ejemplo
=	Igual	X=Y, X$=Y$
<	Menor	X<Y, X$<Y$
>	Mayor	X>Y, X$>Y$
<>, ><	No igual	X<>Y, X$><Y$
<=, =<	Menor o igual	X<=Y, X$=<Y$
>=, =>	Mayor o igual	X>=Y, X$=>Y$

EXPRESIONES LOGICAS

Las expresiones lógicas realizan operaciones lógicas entre constantes, variables y funciones numéricas.
Operación lógica... Convierte el dato en un entero considerado como un número binario de 16 bits realizando la operación bit a bit.

Operación lógica	Resultado operación lógica para cada bit
NOT (negación)	X | NOT X 1 | 0 0 | 1
AND (producto lógico)	X | Y | X AND Y 1 | 1 | 1 1 | 0 | 0 0 | 1 | 0 0 | 0 | 0
OR (suma lógica)	X | Y | X OR Y 1 | 1 | 1 1 | 0 | 1 0 | 1 | 1 0 | 0 | 0
XOR (OR exclusiva)	X | Y | X XOR Y 1 | 1 | 0 1 | 0 | 1 0 | 1 | 1 0 | 0 | 0
EQV (negación OR exclusiva)	X | Y | X EQV Y 1 | 1 | 1 1 | 0 | 0 0 | 1 | 0 0 | 0 | 1
IMP (implicación)	X | Y | X IMP Y 1 | 1 | 1 1 | 0 | 0 0 | 1 | 1 0 | 0 | 1

EXPRESIONES ALFANUMERICAS

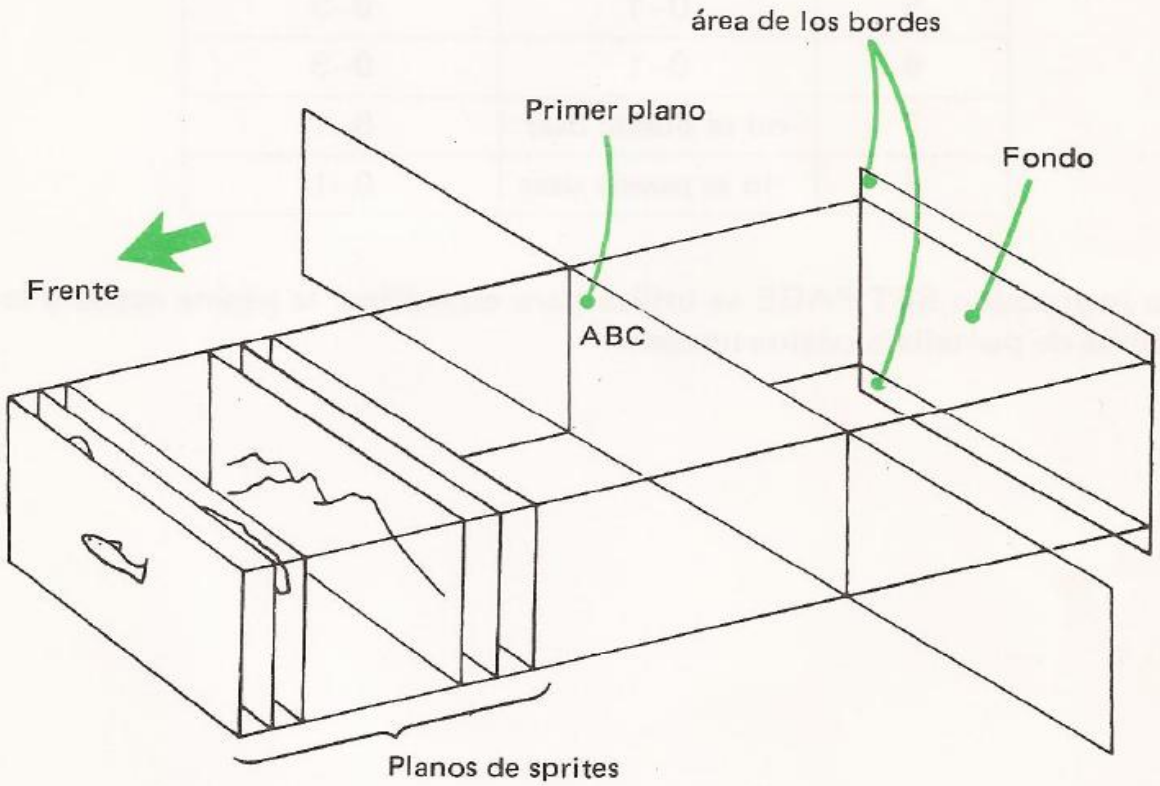
Para operaciones entre constantes, variables y funciones alfanuméricas solo es efectiva la adición (concatenación).

Ejemplo

```
PRINT "ABC"+"XYZ"
ABCXYZ
```

CONFIGURACION DE LA PANTALLA

La configuración de la pantalla de un ordenador MSX se muestra en la siguiente figura. Los caracteres y las figuras están en el primer plano. Detrás está el plano de fondo; se puede cambiar el color del fondo, pero no se pueden presentar caracteres o figuras en él. En el borde superior y en el borde inferior de la pantalla hay dos zonas límite en las que, como en el caso del fondo, no es posible representación alguna aunque sí se puede cambiar el color. Por delante del primer plano hay 32 planos de esprites que sirven para presentar y animar configuraciones de sprites.



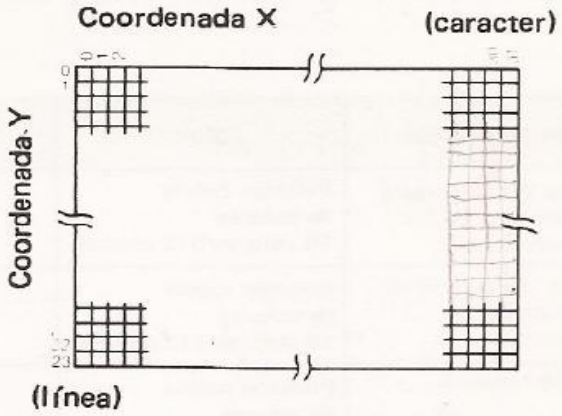
El MSX Versión 2.0 puede tener 64 o 128K bytes de memoria RAM de vídeo. La RAM de vídeo memorizará a la vez varias pantallas de datos con vistas a su presentación en el primer plano. Esta presentación se puede utilizar para dividir la RAM de vídeo en múltiples páginas y escribir datos en una página mientras se está presentando en pantalla otra distinta. La página donde se están escribiendo datos con órdenes BASIC recibe el nombre de página activa, y la página presentada en pantalla recibe el nombre de página de pantalla o página-imagen.
El número de páginas depende del modo de la pantalla (modo SCREEN) y del tamaño de la RAM de vídeo, como se muestra en la tabla siguiente.

Modo SCREEN	Número de páginas	
	64K	128K
5	0–1	0–3
6	0–1	0–3
7	no se puede usar	0–1
8	no se puede usar	0–1

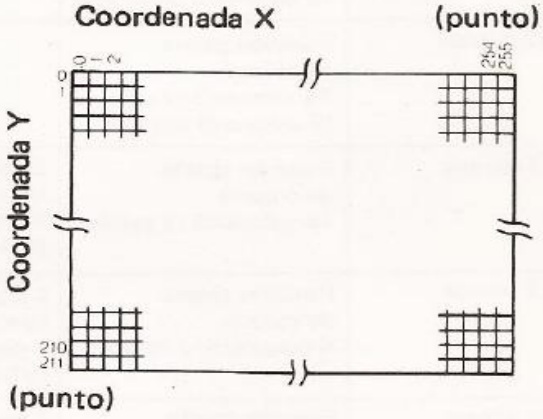
La instrucción SET PAGE se utiliza para especificar la página activa y la página de pantalla o página-imagen.

Coordenadas de la pantalla

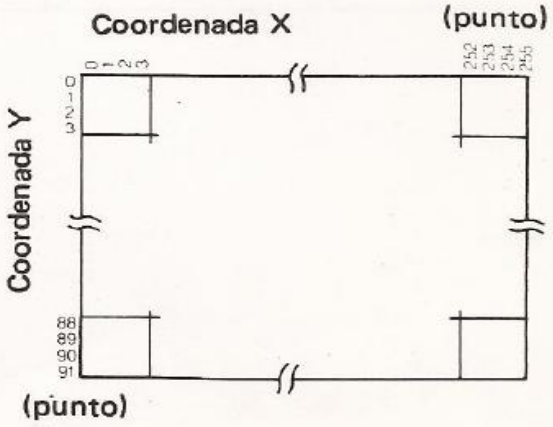
Modo Texto (screen 1)



Modo Gráfico (screen 5)



Modo Gráfico (screen 3, multicolor)



Las coordenadas (X,Y) cambiarán en función del modo de pantalla utilizado. Para más detalles, ver página 12.

TIPOS DE PANTALLAS (SCREEN)

El BASIC–MSX2 tiene 9 tipos de pantalla distintos (SCREEN) que proporcionan las diferentes resoluciones de la pantalla y funciones de presentación visual.

SCREEN	Modo	Imagen de pantalla	Color	Páginas	Sprites
0	Texto	Máximo: 80 caracteres horizontales y 24 líneas verticales.	Función paleta de colores 16 colores/512 colores	—	No utilizado
1		Máximo: 32 caracteres horizontales y 24 líneas verticales.	Función paleta de colores 16 colores/512 colores	—	Utilizado
2	Gráfico VRAM: 64 o 128K	256 x 192 puntos	Función paleta de colores 16 colores/512 colores (2 colores/8 puntos)	—	Utilizado
3		64 x 48 puntos (multicolor)	Función paleta de colores 16 colores/512 colores	—	Utilizado
4		256 x 192 puntos	Función paleta de colores 16 colores/512 colores (2 colores/8 puntos)	—	Utilizado (sprite multicolor)
5		256 x 212 puntos	Función paleta de colores 16 colores/512 colores	2 páginas VRAM 64K) 4 páginas (VRAM 128K	Utilizado (sprite multicolor)
6		512 x 212 puntos	Función paleta de colores 4 colores/512 colores	2 páginas (VRAM 64 K) 4 páginas (VRAM 128 K)	Utilizado (sprite multicolor)
7	Gráfico Solo VRAM de 128K	512 x 212 puntos	Función paleta de colores 16 colores/512 colores	2 páginas	Utilizado (sprite multicolor)
8		256 x 212 puntos	256 colores	2 páginas	Utilizado (sprite multicolor)

FUNCION PALETA DE COLORES

La función paleta de colores sirve para representar colores en los modos SCREEN 0 a 7. La función paleta de colores ofrece la posibilidad de especificar la intensidad del rojo, azul y verde correspondiente a cada color numerado de la paleta de colores con objeto de permitir componer cualquier color. Como hay ocho niveles de intensidad del rojo, verde y azul, se pueden componer un total de 512 colores. En los modos SCREEN 0 a 5 y 7, cabe la posibilidad de utilizar los 16 colores, del 0 al 15, pero en el modo SCREEN 6 solo se pueden utilizar los cuatro colores numerados del 0 al 3.

La selección inicial de la paleta de colores del BASIC—MSX2 es la siguiente:

Paleta N.º	Color	Intensidad rojo	Intensidad azul	Intensidad verde
0	Transparente	0	0	0
1	Negro	0	0	0
2	Verde	1	1	6
3	Verde claro	3	3	7
4	Azul oscuro	1	7	1
5	Azul claro	2	7	3
6	Rojo oscuro	5	1	1
7	Azul cielo	2	7	6
8	Rojo	7	1	1
9	Rojo claro	7	3	3
10	Amarillo oscuro	6	1	6
11	Amarillo claro	6	4	6
12	Verde oscuro	1	1	4
13	Magenta	6	5	2
14	Gris	5	5	5
15	Blanco	7	7	7

DISPOSITIVOS DE ARCHIVO

En el acceso a ficheros mediante órdenes BASIC, el elemento a utilizar se especifica mediante el nombre del dispositivo. Se pueden utilizar los siguientes dispositivos:

CAS: . . . . . Grabador/Reproductor de cassettes.
CRT: . . . . . Pantalla modo texto
GRP: . . . . . Pantalla modo gráfico
LPT: . . . . . Impresora
MEM: . . . . . RAM—DISK
A: —H: . . . . . Nombre de la unidad de discos

RAM—DISK

Para guardar ficheros de programas y de datos se utilizan discos flexibles (floppy Disk) y cintas de cassette. Además de ello, el BASIC—MSX2 tiene una función que ofrece la posibilidad de utilizar una parte de la memoria interna como dispositivo de memoria a efectos de almacenamiento de ficheros, de la misma forma que un disco flexible o una cinta cassette. Este tipo de dispositivo de memoria recibe el nombre de RAM—DISK.

La capacidad de la RAM—DISK se puede seleccionar dentro de la gama de 1024 a 32.768 bytes. Con esta capacidad se pueden almacenar 112 ficheros. De todas formas, hay que tener cuidado al trabajar con RAM—DISK, pues todos sus ficheros se borran cuando se desconecta el ordenador o se pulsa el botón **RESET**.

FUNCION DE LAS TECLAS Y SU UTILIZACION

El editor de pantalla completa sirve para efectuar la modificación y adición de caracteres mediante el desplazamiento del cursor a una posición arbitraria de la pantalla. Con esta función podrá realizar con suma facilidad la edición o la corrección de los programas.
El editor de pantalla se puede utilizar con todos los caracteres presentados en la pantalla durante el modo directo, en el que la línea donde está situado el cursor se introduce en la memoria del ordenador al pulsar la tecla [RETURN]. Cuando el principio de una línea es un número, la línea resulta almacenada en memoria como línea de programa, y cuando no es un número, la línea es ejecutada directamente.

TECLAS DE EDICION

Tecla	Función
Teclas de desplazamiento del cursor	Mueven el cursor un espacio, hacia arriba, hacia abajo, a la izquierda o a la derecha.
[HOME]	Mueve el cursor a la esquina superior izquierda de la pantalla.
[SHIFT] + [HOME]	Borra todos los caracteres de la pantalla y mueve el cursor a la esquina superior izquierda de la pantalla.
[INS]	Proporciona la posibilidad de entrada en el modo inserción (el cursor disminuye su tamaño). Todos los caracteres introducidos por el teclado tras la pulsación de esa tecla quedan insertados en la posición del cursor. Para volver al modo normal, pulsar otra vez la tecla [INS] y la tecla [RETURN] o una tecla de desplazamiento del cursor.
[DEL]	Borra el caracter que coincide con la posición del cursor. Los caracteres siguientes se desplazan un espacio a la izquierda.
[BS]	Borra el caracter situado delante del cursor. Los caracteres siguientes se desplazan un espacio a la izquierda.
[TAB]	Adelanta el cursor a la siguiente posición TAB, y si hay caracteres en medio, deja un espacio.

LA TECLA CTRL

Además de las teclas de edición, el BASIC–MSX2 ofrece un conjunto de funciones especiales que entran en escena simplemente pulsando la tecla CTRL simultáneamente con otra tecla.

Tecla pulsada	Función
CTRL + B	Desplaza el cursor al principio de una palabra (grupo de caracteres separados por un espacio). Cuando el cursor está ya al principio de una palabra, pasa al principio de la palabra inmediatamente anterior.
CTRL + C	Activa el estado de espera de entrada o la generación automática de número de línea por la orden AUTO para volver al estado de espera de orden.
CTRL + E	Borra todo lo escrito desde la posición del cursor hasta la última línea.
CTRL + F	Desplaza el cursor al principio de la palabra siguiente.
CTRL + G	Genera un pitido (bip).
CTRL + H	Igual que la tecla BS
CTRL + I	Igual que la tecla TAB
CTRL + J	Desplaza el cursor una línea hacia abajo.
CTRL + K	Igual que HOME.
CTRL + L	Igual que SHIFT + HOME.
CTRL + M	Igual que RETURN
CTRL + N	Desplaza el cursor a la posición siguiente al último caracter de la línea.
CTRL + R	Igual que INS
CTRL + U	Borra todos los caracteres de una línea.
CTRL + X	Igual que SELECT. No definida en el BASIC–MSX2.
CTRL + \	Igual que ◁.
CTRL + [	Igual que ESC. No definida en el BASIC–MSX2.
CTRL +]	Igual que ▷.
CTRL + ^	Igual que ▽.
CTRL + _	Igual que △.

TECLAS DE FUNCION

Las teclas **F1** a **F5** reciben el nombre de teclas de función. Para cada tecla de función se puede definir una cadena de caracteres de manera que la pulsación de la tecla de función surte los mismos efectos que la introducción de la cadena de caracteres definida para esa tecla. En el BASIC–MSX2 están definidas las siguientes cadenas de caracteres cuando se inicializa el BASIC–MSX2:

F1	color ␣
F2	auto ␣
F3	goto ␣
F4	list ␣
F5	run RETURN
SHIFT + F1	color 15, 4, 4 RETURN
SHIFT + F2	cload"
SHIFT + F3	cont RETURN
SHIFT + F4	list. RETURN
SHIFT + F5	SHIFT + HOME run RETURN

El contenido de una tecla de función se puede redefinir libremente mediante la instrucción KEY.

OTRAS TECLAS DE CONTROL

Tecla STOP
Detiene temporalmente la ejecución de un programa BASIC. Pulsar esta tecla otra vez para continuar la ejecución del programa.

Pulsación simultánea de CTRL + STOP
Interrumpe la ejecución de un programa BASIC. En el estado de generación automática de número de línea, consecuencia de una instrucción AUTO, produce el paso al estado de espera de ordenes.

Tecla SELECT **y tecla** ESC
Estas teclas no están definicas en el BASIC–MSX2 (no realizan ninguna función específica).

DISK–BASIC

Cuando hay una unidad de discos flexibles (floppy Disk) conectada a un ordenador que admite el BASIC–MSX2, se puede utilizar el DISK–BASIC MSX2.

El DISK–BASIC tiene todas las funciones del BASIC–MSX2 más las órdenes complementarias para trabajar con un disco como dispositivo de almacenamiento externo. Este manual describe tanto el BASIC del ordenador como el DISK–BASIC; las diferencias entre el BASIC y el DISK–BASIC están indicadas en cada entrada.

SIMBOLOS

En este manual se utilizan dos símbolos, 🖫 y 🖫, con el significado siguiente:

🖫 Indica los puntos donde se explican métodos de notación o funciones del DISK–BASIC.

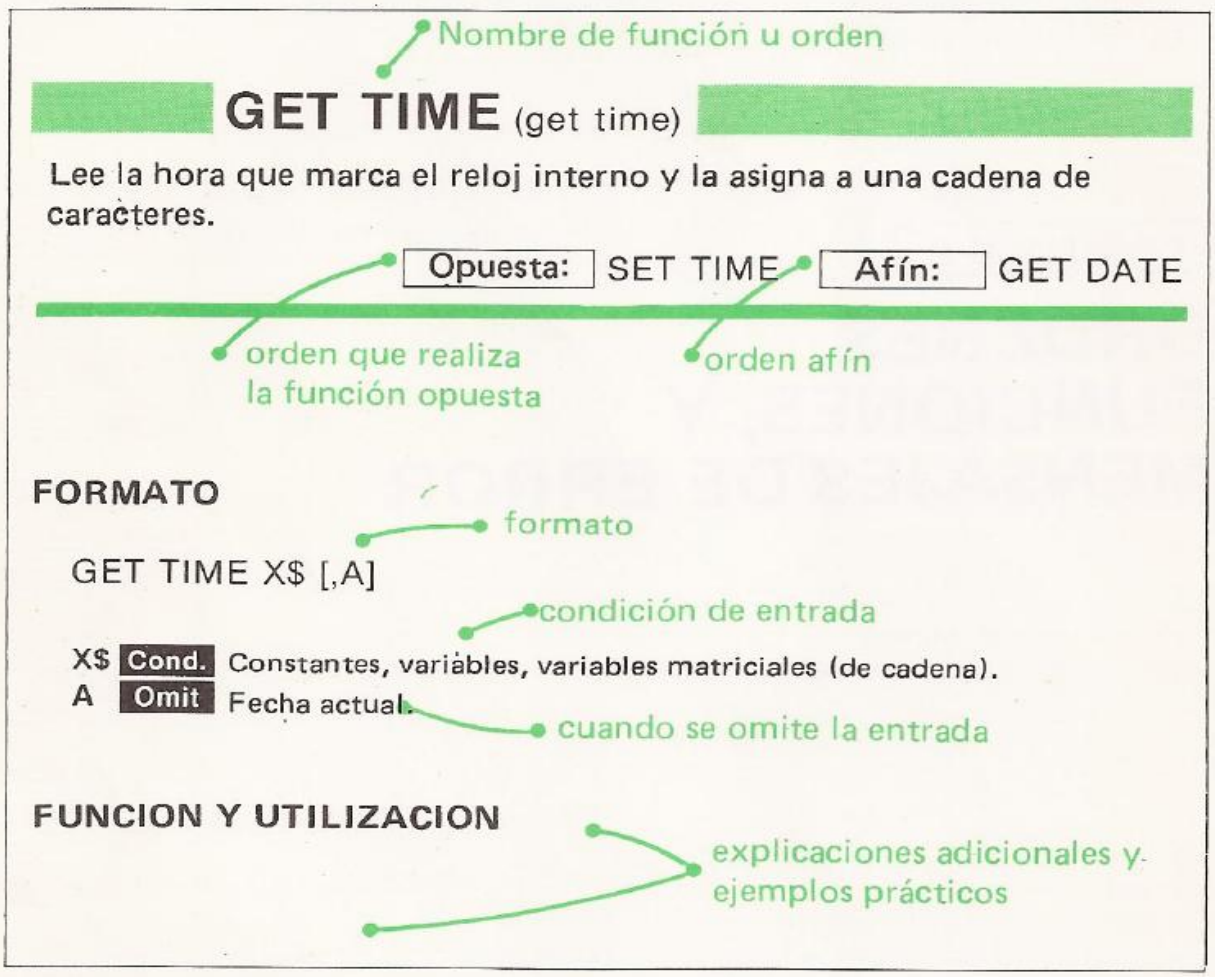
🖫 Indica los puntos donde se explican métodos de notación o funciones de la RAM–DISK.

CAPITULO 2

ORDENES, FUNCIONES, Y MENSAJES DE ERROR

ORDENES Y FUNCIONES

En este capítulo se describen todas las funciones y órdenes del BASIC–MSX2, ordenadas alfabéticamente. A continuación se muestra el formato de presentación.



delante de un nombre de función está escrita la palabra Función

Función **ABS** (absolute)

Ofrece el valor absoluto de un dato numérico.

Omisión de los conceptos de entrada

Los conceptos de entrada encerrados entre corchetes ( ) en la sección FORMATO, podrán omitirse.

EJEMPLO

SCREEN (modo), (tamaño del sprite), (sonido de las teclas), (velocidad de transmisión), (tipo de impresora), (modo de entrelazado)

Para especificar únicamente el modo y el tamaño del sprite:

```
SCREEN 2,3
```

los demás conceptos, incluyendo las comas, se pueden omitir

Para especificar únicamente la impresora:

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

las comas no se pueden omitir

Repetición del concepto de entrada

Los puntos suspensivos (.....) indican la repetición del mismo concepto.

EJEMPLO

DATA constante (,constante) .....

Después de DATA se puede poner el número de constantes que se quiera, dentro de los límites de una línea.

Función **ABS** (absolute)

Ofrece el valor absoluto de un dato numérico.

FORMATO

ABS(X)

X **Cond.** Constantes, variables, variables de matriz ó sus expresiones (numéricas)

Valor obtenido: Tipo numérico

FUNCION Y UTILIZACION

Da X cuando $X \geqslant 0$ y $-X$ cuando $X < 0$.

Ejemplo de ejecución

```
PRINT ABS(2)
 2
PRINT ABS (3-10)
 7
```

Función ASC (ascii)

Ofrece el código ASCII correspondiente al primer caracter del dato de cadena.

Opuesta: CHR$

FORMATO

ASC(X$)

X$ **Cond.** Constantes variables, variables de matriz, sus expresiones (de cadena)

Valor obtenido: Expresiones decimales, de tipo entero.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT ASC("d")
 100
```
Código de caracter correspondiente a "d"

```
PRINT ASC("data")
 100
```
Código de caracter correspondiente a "d"

Función **ATN** (arc tangent)

Ofrece el valor del arcotangente correspondiente a un dato numérico.

Opuesta: TAN Afín: : SIN, COS

FORMATO

ATN(X)

X **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas).

Valor obtenido: Tipo numérico

FUNCION Y UTILIZACION

La función ATN da un valor numérico de coma flotante que corresponde a un ángulo cuya tangente es X. Su unidad es el radian.

- Para obtener el resultado en grados, multiplicarlo por $180/\pi$.

Ejemplo de ejecución

```
PRINT ATN(1)
 .7853981633974 5          resultado en radianes.

PRINT ATN(1)*180/3.14159
 45.00003809905            resultado en grados.
```

AUTO (auto)

Produce la generación automática de números de línea a partir de un número de línea especificado y con un incremento especificado.

FORMATO

AUTO (número de línea inicial) (, incremento)

Número de línea inicial	Cond.	Constantes de tipo entero, 0 ≤número ≤65529
	Omit	0. Pero se omite el incremento, es 10.
Incremento	Cond.	Constantes de tipo entero, 1 ≤incremento ≤65529
	Omit	10.

FUNCION Y UTILIZACION

Sirve para eliminar la necesidad de teclear los números de línea a la hora de introducir un programa por el teclado.

- Cuando un número de línea generado automáticamente tiene ya asociada una instrucción de programa, sale un asterisco " * " a la derecha del número de línea. Para modificar esta instrucción de programa, mover el cursor al asterisco y luego introducir una nueva instrucción tras borrar el asterisco pulsando la tecla [DEL] o espacio. Cuando no es preciso modificarla, pulsar [RETURN].
- Para poner fin a la generación automática de números de línea, pulsar [STOP] manteniendo pulsada al mismo tiempo [CTRL], o pulsar [C] manteniendo pulsada al mismo tiempo [CTRL].

Ejemplo de ejecución

```
AUTO 100,50
100 PRINT "12345"
150*
```

indica que existe ya el número de línea 150

BASE (base)

Sirve para leer o escribir una dirección base de la tabla del VDP.

FORMATO

BASE (N)

N **Cond.** Valores a leer:
Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤ valores < 45.
Valores a asignar:
Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤ valores < 20.

FUNCION Y UTILIZACION

Para asignar valores se pueden utilizar los modos SCREEN 0 a 3.
N se calcula de la forma siguiente:

Modo SCREEN * 5 ≤ número de tabla

Por ejemplo, el valor N de la dirección de la tabla de atributos de los sprites en SCREEN 2 (tabla No. 3) se calcula así:

N = 2 * 5 + 3 = 13

Tabla No.	Tabla
0	Tabla de nombres de configuraciones
1	Tabla de colores
2	Tabla de generadores de configuraciones
3	Tabla de atributos de los sprites
4	Tabla de configuraciones de los sprites

Utilizar las fórmulas siguientes para determinar la dirección o el modo SCREEN cuando está incluido el valor de N:

Modo SCREEN INT (N/5)
No. de tabla (N MOD 5)

Por ejemplo, cuando N = 44,
44/5 = 8 (resto = 4)
Por lo tanto, el modo SCREEN cuando N = 44 es el 8 y el No. de tabla es el 4.

BEEP (beep)

Genera un pitido.

Afín: SOUND, PLAY

FORMATO

BEEP

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
10 FOR I=0 TO 9
20 BEEP
30 NEXT I
```

Este programa genera un pitido que se repite 10 veces consecutivamente.

Función **BIN$** (binary dollar)

Ofrece la expresión binaria de un dato numérico en forma de dato alfanumérico.

Afín: HEX$, OCT$

FORMATO

BIN$(X)

X — **Cond.** Constantes, variables, variables de matriz, sus expresiones (número); $-32768 \leqslant X < 65536$. En caso de un número negativo, tiene el mismo valor que si se sumara su valor a 65536.

Valor obtenido: Tipo alfanumérico.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT BIN$(100)
1100100

PRINT BIN$(-32768)
1000000000000000
```

BLOAD (binary load)

Carga un programa en lenguaje máquina, o lo carga y ejecuta.
💾 Carga un fichero en la RAM de vídeo.

Opuesta: BSAVE

FORMATO

BLOAD "(nombre del dispositivo) (nombre del fichero) (extensión)
(R)
(S) (decalaje)

Nombre del dispositivo	Cond.	CAS:
		💾 A:, B:, C:, D:, E:, F:, G:, H:
💾	Omit	Unidad de discos actual.
Nombre del fichero	Cond.	Una cadena de 6 caracteres o menos.
		💾 Una cadena de 8 caracteres o menos.
	Omit	Carga el primer fichero encontrado en la cinta. (No se puede omitir con DISK—BASIC).
💾 **Extensión**	Cond.	Una cadena de 3 caracteres o menos.
	Omit	Cadena nula.
Opción R	Omit	Unicamente carga (no ejecute el programa).
💾 **Opción S**	Omit	Carga el programa en la memoria principal.
Decalaje	Cond.	Un entero.
	Omit	0

FUNCION Y UTILIZACION

Carga un programa que había sido almacenado mediante la orden BSAVE en la zona de memoria comprendida entre la dirección inicial y la dirección final especificadas en la orden BSAVE. Si se ha especificado un decalaje, sumará dicho valor a la dirección inicial y a la dirección final.
Si se ha especificado la opción R, el programa será ejecutado inmediatamente después de ser cargado. La dirección de ejecución será la dirección especificada en la orden BSAVE.
Si se ha especificado la opción S, cargará el contenido de la RAM de vídeo almacenada mediante la orden BSAVE con la opción S. En el caso de SCREEN 5 a 8, cargará el contenido de la página activa.

Ejemplo de ejecución

```
  BLOAD "CAS:PROG4"
Ok
BLOAD "A:PROG.BIN",R
Ok
10 SCREEN 2
20 BLOAD"a:box",S
30 GOTO 30
```

La línea 20 carga el contenido de la RAM de vídeo que había sido guardado en el ejemplo incluido en la sección BSAVE Ėn consecuencia, aparece en pantalla un rectángulo.

BSAVE (binary save)

Almacena, en números binarios, el contenido de la memoria principal especificada.

💾 Almacena, en números binarios, el contenido especificado de la RAM de vídeo.

Opuesta: BLOAD

FORMATO

BSAVE "[nombre del dispositivo] [nombre del fichero [.Extensión]]", dirección inicial, dirección final [,dirección inicio ejecución] [,S]

Nombre del dispositivo
- Cond. CAS:
- 💾 A:, B:, C:, D:, E:, F:, G:, H:
- 💾 Omit Unidad de disco actual.

Nombre del fichero
- Cond. Una cadena de 6 caracteres o menos.
- 💾 Una cadena de 8 caracteres o menos.
- Omit Una cadena nula. (No se puede omitir en DISK—BASIC).

💾 Extensión
- Cond. Una cadena de 3 caracteres o menos.
- Omit Una cadena nula.

Dirección inicial, dirección final
- Cond. Constantes de tipo entero; $-32768 \leqslant$ dirección $\leqslant 65535$.

Dirección de inicio de la ejecución
- Cond. Constantes de tipo entero; $-32768 \leqslant$ dirección $\leqslant 65535$.
- Omit La misma que la dirección inicial.

💾 Opción S
- Omit Almacena el programa que hay en la memoria RAM de vídeo.

FUNCION Y UTILIZACION

Almacena, en binario, el contenido de la memoria principal comprendida entre la dirección inicial y la dirección final. Sirve para almacenar programas en lenguaje máquina.

Si se especifica la dirección de inicio de la ejecución, cuando se cargue el programa mediante una instrucción BLOAD con opción R, la ejecución comenzará en la dirección especificada. Si se omite, la dirección de inicio de la ejecución es la dirección inicial.

💾 Cuando se especifica la opción S, almacena, en forma de fichero, el contenido de la RAM de vídeo especificada.

En el caso de SCREEN 5 a 8, almacenará el contenido de la página activa.

Direcciones de la VRAM utilizadas para visualizar una página

SCREEN	Dirección
2	0 ~ &H3C20
3	0 ~ &H2040
4	0 ~ &H3C20
5	0 ~ &H76A0
6	0 ~ &H76A0
7	0 ~ &HFAA0
8	0 ~ &HFAA0

Ejemplo de ejecución

```
 BSAVE "CAS:PROG4",&HE000,&HE800,&HE100
BSAVE "B:PROG.BIN",&HE000,&HE800

10 SCREEN 2
 20 COLOR=(15,2,6,7)
 30 LINE(30,30)-(130,100),,BF
 40 BSAVE "A:BOX",0,&H3FFF,S
```

Este programa, tras dibujar en la pantalla un recuadro rectangular, línea 30, almacena todo el contenido de la RAM de vídeo (desde la dirección 0 a la dirección &H3FFF) con el nombre de fichero "BOX", línea 40 (orden BSAVE con la opción S).

CALL (call)

Ejecuta una orden extendida.

FORMATO

CALL orden extendida [(argumento, argumento...)]

Argumento **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo entero). Constantes, variables, variables de matriz, sus expresiones (de cadena).

FUNCION Y UTILIZACION

Cuando hay órdenes extendidas, se pueden ejecutar mediante una instrucción CALL.
En lugar de escribir CALL, se puede escribir el símbolo de subrayado (—).

Ordenes extendidas de DISK—BASIC y RAM—DISK.
La instrucción CALL sirve para extender el BASIC cuando se trabaja con un disco flexible o con RAM—DISK. Con la instrucción CALL se pueden utilizar las siguientes órdenes extendidas: FORMAT, SYSTEM, MEMINI, MFILES, MKILL, MNAME. (Ver "Ordenes complementarias para uso con disco flexible", página 242, y "Ordenes complementarias para uso con RAM—DISK", página 246).

Función CDBL (convert to double precisión)

Convierte datos numéricos en datos de precisión doble.

Afín: CINT, CSNG, FIX

FORMATO

CDBL(X)

X **Cond.** Constantes, variables, variables matriciales, sus expresiones (de tipo numérico).

Valor obtenido: Tipo numérico de doble precisión.

FUNCION Y UTILIZACION

La función CDBL trata internamente el dato numérico dado como dato de precisión doble.

Función CHR$ (character dollar)

Ofrece el caracter correspondiente a un código de caracter especificado.

Opuesta: ASC, Afín: STRING$

FORMATO

CHR$(X)

X **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $0 \leqslant X < 256$.

Valor obtenido: Tipo alfanumérico.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT CHR$(100)
d
PRINT CHR$(1);CHR$(67)
♥
```

Ver la tabla de códigos de los caracteres (página 258).

CINT (convert to integer)

Convierte un dato numérico en entero.

Afín: CDBL, CSNG, FIX

FORMATO

CINT(X)

X **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $-32768 \leqslant X < 32768$.

Valor obtenido: Tipo entero.

FUNCION Y UTILIZACION

Cuando el dato numérico X es un valor entero, lo mantiene como está, pero cuando es un valor del tipo coma flotante, lo convierte en un valor entero omitiendo los valores que siguen a la coma. Se diferencia de la función INT, en que ésta ofrece el número entero de X, mientras que CINT convierte el número X en un entero con un procesamiento distinto.

Ejemplo de ejecución

```
PRINT CINT(9/2)
 4
PRINT CINT(12*200*55)
Overflow
```

CIRCLE (circle)

Dibuja un círculo, un óvalo, un arco o una forma en abanico, en el modo gráfico.

FORMATO

CIRCLE [STEP] (coordenada del centro), radio, [color], [ángulo inicial], [ángulo final], [relación de ejes] .

Coordenadas del centro X,Y.

Cond. Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $-32768 \leqslant X < 32768$; $32768 \leqslant Y < 32768$.

Radio

Cond. Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $-32768 \leqslant$ radio < 32768.

Color

Cond. SCREEN 2 a 7:
Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); o $\leqslant$ color < 16.
SCREEN 8:
Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); o $\leqslant$ color < 256.

Omit Color de escritura actual.

Angulo inicial

Cond. Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $2\pi \leqslant$ ángulo $\leqslant 2\pi$ (en radianes).

Omit 0

Angulo final

Cond. Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $-2\pi \leqslant$ ángulo $\leqslant 2\pi$ (en radianes).

Omit 2π

Relación de ejes

Cond. Constantes, variables, variables de matriz, sus expresiones (numérico positivo).

Omit 1

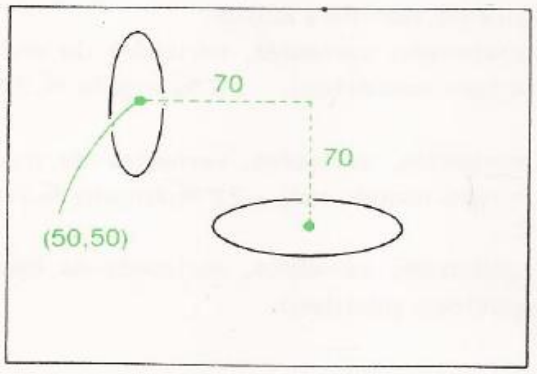
FUNCION Y UTILIZACION

Dibuja un círculo con el radio especificado que tiene como centro las coordenadas especificadas.
Cuando se especifica un ángulo inicial y un ángulo final, sólo dibuja un arco circular.
Se puede dibujar una forma en abanico especificando un ángulo inicial y un ángulo final negativos (−). Se puede dibujar una elipse con una determinada relación, especificando para el radio vertical un múltiplo del radio horizontal.

Cuando se especifica STEP, pasa X,Y a un sistema de coordenadas nuevo que toma como origen el último punto especificado en la instrucción gráfica inmediatamente anterior.

Ejemplo de ejecución

```
10 CLS
20 SCREEN 2
30 CIRCLE (50,50),30,,,,4
40 CIRCLE STEP(70,70),30,,,,,.25
50 GOTO 50
```



CLEAR (clear)

Inicializa todas las variables y selecciona el tamaño del área de caracteres y la dirección de memoria más alta utilizada en BASIC. Además, cierra todos los ficheros abiertos, si hay alguno.

Afines: OPEN, CLOSE

FORMATO

CLEAR [tamaño del área de caracteres] [, dirección más alta]

Tamaño del área de caracteres	**Cond.**	Constantes, variables, variables matriciales, sus expresiones (de tipo numérico).
	Omit	Valor seleccionado actual (el valor inicial es 200). De todas formas, no se puede omitir independientemente el tamaño del área de caracteres.
Dirección más alta	**Cond.**	Constantes, variables, variables matriciales, sus expresiones (de tipo numérico).
	Omit	Valor actual.

FUNCION Y FORMA DE EMPLEO

Ejemplo de ejecución

```
CLEAR 400,55296
```

Esta instrucción inicializa todas las variables. Además, pone el tamaño del área de caracteres en 400 bytes y la dirección más alta del área del programa BASIC en 55296.

CLOAD (cassette load)

Carga un programa BASIC desde una cinta de cassette.

Opuesta: CSAVE, Afines: CLOAD?, LOAD

FORMATO

CLOAD ["nombre de fichero"]

Nombre de fichero

Cond. Cadena de 6 o menos caracteres. Si se especifican 7 o más caracteres, ignora todos los caracteres a partir del sexto.

Omit Carga el primer programa que encuentra.

FUNCION Y UTLIZACION

Ejemplo de ejecución

```
CLOAD "PROG1"
```

Carga de cinta cassette a memoria el programa almacenado con el nombre de fichero PROG1.

- Si durante la carga ocurre un error, rebobine la cinta y vuelva a cargar.

CLOAD? (cassette load verify)

Compara un programa almacenado en cinta cassette con uno almacenado en memoria.

FORMATO

CLOAD? ("nombre de fichero")

Nombre de fichero	**Cond.**	Cadena de 6 caracteres o menos. Si se especifican 7 o más caracteres, ignora todos los caracteres a partir del sexto.
	Omit	Compara el primer programa encontrado con el residente en la memoria.

FUNCION Y UTILIZACION

Se trata de una orden que comprueba si un programa está bien almacenado o no. Cuando se ejecuta, compara el programa guardado en memoria con el programa almacenado en cinta de cassette bajo un nombre de fichero especificado.

- Tras hacer la comparación, si el resultado es positivo indica la correspondencia de los programas presentando en pantalla "OK". Cuando no hay correspondencia, presenta en pantalla el mensaje "Device I/O error" (error dispositivo de entrada/salida).
- Si se omite el nombre de fichero o se introduce CLOAD?"_", compara el primer programa que encuentra en la cinta con el programa de la memoria.

Ejemplo de ejecución

```
CLOAD? "PROG1"
```

CLOSE (close)

Cierra un fichero abierto anteriormente por una instrucción OPEN.

Opuesta: OPEN, Afin: CLEAR

FORMATO

CLOSE [#] [número de fichero] [, número de fichero] ...

Número de fichero

Cond. Constantes, variables, variables de matriz, sus expresiones (de tipo entero); 1 ≤ número de fichero ≤ número especificado por instrucción MAXFILES

Omit Cierra todos los ficheros.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
10 MAXFILES=3
20 SCREEN 2
30 OPEN "GRP:" FOR OUTPUT AS #1 ——— abre el fichero 1
40 OPEN "GRP:" FOR OUTPUT AS #2 ——— abre el fichero 2
50 OPEN "GRP:" FOR OUTPUT AS #3 ——— abre el fichero 3
60 PRINT #1,"ABC"
70 PRINT #2,"DEF"
80 PRINT #3,"GHI"
100 CLOSE ——————————————— cierra todos los ficheros
```

CLS (clear screen)

Borra la pantalla.

FORMATO

CLS

- En el modo gráfico, el color del fondo cambia ejecutando CLS después de la instrucción COLOR.

COLOR (color) (1)

Especifica los colores del primer plano, del fondo y de los bordes.

FORMATO

COLOR (color primer plano) , (color fondo) , (color bordes)

Color del primer plano, color del fondo, color de los bordes

Cond. SCREEN 0 a 7
Constantes, variables, variables matriciales, sus expresiones (de tipo numérico); 0 ≤color <16.
SCREEN 8
Constantes, variables, variables matriciales, sus expresiones (de tipo numérico); 0 ≤color <256.

Omit Color actual.

FUNCION Y UTILIZACION

Para calcular los códigos de color correspondientes a SCREEN 8, utilizar la fórmula siguiente:

Código de color = 32 x (intensidad verde) + 4 x (intensidad rojo) + (intesidad azul).
Intensidad verde: un entero del 0 al 7.
Intensidad rojo: un entero del 0 al 7.
Intensidad azul: un entero del 0 al 3.

Ejemplo de ejecución

```
COLOR 6
```
Sólo cambia el color del primer plano (el color de los caracteres en el modo texto y el color de los gráficos en el modo gráfico).

```
COLOR ,2
```
Sólo cambia el color del fondo

```
COLOR ,,11
```
Sólo cambia el color de los bordes

```
COLOR 15,4,4
```
Colores de inicialización.

- Consultar la tabla de los códigos de color en la página 13.
- Consultar la configuración de la pantalla en la página 9.
- En el modo gráfico no se puede cambiar el color del fondo especificándolo únicamente con una instrucción COLOR. Hay que ejecutar después la instrucción CLS.

COLOR (color) (2)

Asigna colores a los números de la paleta de colores.

FORMATO

COLOR = (No. de paleta, intensidad de rojo, intesidad de verde, intensidad de azul).

No. de paleta **Cond.** SCREEN 0 a 7
Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); 0 ≤ número < 16.

Intensidad de rojo, azul y verde
Cond. Constantes de tipo entero; 0 ≤ intensidad ≤ 7.

FUNCION Y UTILIZACION

Se pueden presentar en pantalla 512 colores combinando diferentes intensidades de rojo, azul y verde.
Los colores de los números de la paleta para los que no se ejecuta esta instrucción siguen teniendo los valores iniciales.

Ejemplo de ejecución

```
100 SCREEN 5
110 FOR I=7 TO 0 STEP -1
120   CIRCLE (120,100),I*10+5,I
130   PAINT (120,100),I,I
140 NEXT I
150 J=(J+1) MOD 8
160 FOR I=0 TO 7
170   COLOR=(I,0,J,0)
180   J=(J+1) MOD 8
190 NEXT I
200 GOTO 150
```

COLOR (color) (3)

Devuelve la paleta de colores a sus valores iniciales por omisión.

FORMATO

COLOR [= NEW]

NEW **Omit** La misma acción

No. paleta	Color	Intensidad rojo	Intensidad azul	Intensidad verde
0	Transparente	0	0	0
1	Negro	0	0	0
2	Verde	1	1	6
3	Verde claro	3	3	7
4	Azul oscuro	1	7	1
5	Azul claro	2	7	3
6	Rojo oscuro	5	1	1
7	Azul cielo	2	7	6
8	Rojo	7	1	1
9	Rojo claro	7	3	3
10	Amarillo oscuro	6	1	6
11	Amarillo claro	6	4	6
12	Verde oscuro	1	1	4
13	Magenta	6	5	2
14	Gris	5	5	5
15	Blanco	7	7	7

COLOR = RESTORE (color = restore)

Asigna el contenido de la tabla de colores de la RAM de vídeo al registro de la paleta de colores del VDP.

FORMATO

COLOR = RESTORE

FUNCION Y UTILIZACION

La ejecución de la instrucción BSAVE con la opción S almacena el contenido de la RAM de vídeo en el disco junto con los datos de la paleta de colores. Estos datos se cargan en la RAM de vídeo mediante la instrucción BLOAD con la opción S. Pero los datos de la paleta de color solo están especificados en la tabla de datos y el color presentado actualmente no cambia con respecto al color presentado antes de la carga. La ejecución de COLOR = RESTORE asigna los datos de la tabla a la paleta de color y se visualiza en pantalla el mismo color presentado cuando se ejecutó el comando BSAVE,S.

Ejemplo de ejecución

```
10 SCREEN 2
20 BLOAD "BOX",S
30 COLOR=RESTORE
40 GOTO 40
```

La línea 20 carga el contenido de la RAM de vídeo almacenado en el ejemplo dado en la sección BSAVE, y la línea 30 cambia el color haciéndolo coincidir con el color presente cuando se ejecutó la orden BSAVE.

COLOR SPRITE (color sprite)

Asigna el color del número de paleta especificado a un plano de sprites.

FORMATO

COLOR SPRITE (No. plano de sprites) = No. paleta

No. del plano de sprites	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); 0 ≤ No. plano < 32.
No. de paleta	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); 0 ≤ No. paleta < 16.

FUNCION Y UTILIZACION

Válida solo para SCREEN 4 a SCREEN 8.

COLOR SPRITE$ (color sprite dollar)

Especifica los colores correspondientes a las líneas horizontales de un sprite.

FORMATO

COLOR SPRITE$ (No. plano sprite) = "expresión de caracteres"

No. del plano del sprite **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); 0 ≤ no. plano < 32.

Expresión de caracteres **Cond.** 16 caracteres o menos.

FUNCION Y UTILIZACION

Válido sólo para SCREEN 4 a SCREEN 8

Cada expresión de caracteres corresponde a una línea de exploración, y cada bit de un caracter tiene el siguiente significado:

b_7	Si es 1, desplaza el sprite 32 puntos a la izquierda.
b_6	Si es 1, ignora la posición de prioridad del sprite y presenta el color suma (función OR) en la posición donde se han solapado varios sprites.
b_5	Si es 1, ignora el choque (solape) de sprites.
b_4	No utilizado.
b_3—b_0	Código de la paleta de colores.

Cuando la expresión de caracteres es menor de 15 caracteres, las partes no especificadas retendrán sus valores previos.

Ejemplo de ejecución

```
100 SCREEN 5,1
110 FOR I=0 TO 7
120   A$=A$+CHR$(2^I-1)
130   B$=B$+CHR$(I*2+1)
140 NEXT I
150 SPRITE$(0)=A$
160 COLOR SPRITE$(2)=B$
170 FOR I=0 TO 255
180   PUT SPRITE 2,(I,100),,0
190 NEXT I
200 GOTO 170
```

CONT (continue)

Reanuda la ejecución de un programa. Opuesta: STOP

FORMATO

CONT

FUNCION Y UTILIZACION

Reanuda un programa interrumpido anteriormente por una orden CTRL + STOP o por una instrucción STOP en un programa. Tras la ejecución de la instrucción CONT, la ejecución del programa vuelve a empezar a partir de la instrucción siguiente a la instrucción interrumpida. De todas formas, si durante la ejecución de una instrucción INPUT ha habido una interrupción, la ejecución comienza por el principio de la instrucción en cuestión.

COPY (copy) (1)

Copia un fichero del disco en el mismo disco o en otro.

FORMATO

COPY "[nombre de la unidad 1] nombre del fichero [.extensión]" [TO "[nombre de la unidad 2] nombre del fichero [.extensión]"]

Nombre de la unidad 1,2	Cond.	A:, B:, C:, D:, E:, F:, G:, H:
	Omit	Unidad de disco actual.
Nombre del fichero	Cond.	Una cadena de hasta 8 caracteres.
Extensión	Cond.	Una cadena de hasta 3 caracteres.
	Omit	Una cadena nula.

FUNCION Y UTILIZACION

Copia el contenido del disco de la unidad 1 en el disco de la unidad 2.

Copia de un fichero

Para hacer la copia en el mismo disco:

```
COPY "ABC.BAS" TO "XYZ.BAS"
```

Omitiendo los nombres de la unidad 1 y 2, se puede hacer una copia en el mismo disco bajo un nombre de fichero distinto. En este ejemplo, se ha grabado en el mismo disco el fichero de nombre "XYZ.BAS", que tiene el mismo contenido que el fichero de nombre "ABC.BAS".

Si se introduce únicamente:

```
COPY "ABC.BAS"
```

y se omite todo lo que vendría a continuación de TO, con la intención de copiar el fichero en el mismo disco y bajo el mismo nombre, saldrá un mensaje de error.

Para hacer la copia en otro disco:

Cuando solo se utiliza una unidad de discos

Cabe la posibilidad de copiar el contenido de un disco en otro disco aún en el caso de tener solo una unidad de discos. Al ejecutar

```
COPY "A:ABC.BAS" TO "B:"
```

Saldrá en pantalla el siguiente mensaje:

Insert diskette for drive B:
and strike a key when ready ⟵ (a)

Cuando sale este mensaje, sacar el disco de la unidad de discos e insertar un nuevo disco (que esté formateado) y pulsar una tecla.

Insert diskette for drive A:
and strike a key when ready ⟵ (b)

Sacar el disco nuevo, insertar el disco anterior y pulsar una tecla. Cuando salga otra vez el mensaje (a), insertar el disco nuevo, y cuando salga el mensaje (b), insertar otra vez el disco anterior. Este proceso se repite hasta que aparece "OK" en la pantalla, indicando que el proceso ha finalizado. El fichero "ABC.BAS" ha quedado copiado en el disco nuevo con el mismo nombre de fichero.

El número de veces que hay que cambiar los discos depende de la longitud del fichero a copiar.

Repitiendo el proceso anterior pero utilizando la orden siguiente, se copia el fichero "ABC.BAS" en el nuevo disco pero con el nombre de fichero "XYZ.BAS".

```
COPY "A:ABC.BAS" TO "B:XYZ.BAS"
```

Cuando se utilizan dos o más unidades de discos

Se puede especificar el disco fuente y el disco destino utilizando los nombres de unidad 1 y 2 de la forma siguiente:

```
COPY "A:ABC.BAS" TO "B:XYZ.BAS"
```

Esta orden copia el fichero "ABC.BAS" del disco de la unidad A: en el disco de la unidad B:, bajo el nombre "XYZ.BAS".

```
COPY "A:ABC.BAS" TO "B:"
```

Esta orden copia el fichero "ABC.BAS" del disco de la unidad A: en el disco de la unidad B: utilizando el mismo nombre ("ABC.BAS").

Si se omite el nombre de unidad 2, de la forma siguiente:

```
COPY "A:ABC.BAS" TO "XYZ.BAS"
```

el fichero quedará copiado en el disco de la unidad A.

```
COPY "A:ABC.BAS"
```

Esta orden, omitiendo todo lo que vendría a continuación de TO, copia el fichero "ABC.BAS" en el disco activo (A) y con el mismo nombre. En consecuencia, el disco activo en el ejemplo anterior no puede ser el disco de la unidad A:; si lo fuera se produciría un error.

Copia de más de un fichero a la vez

Se puede utilizar el signo de interrogación (?) para representar una letra cualquiera del nombre de fichero o de la extensión, y el asterisco (*) se puede utilizar para representar bien un nombre de fichero entero o bien una extensión entera. Utilizando el signo de interrogación o el asterisco, o una combinación de los dos, se pueden copiar varios ficheros con una sola orden COPY.

Cuando solo se dispone de una unidad de discos

```
COPY "A:*.BAS" TO "B:"
```

Esta orden copia todos los ficheros de un disco que tengan la extensión "BAS" en un disco diferente. Al hacerlo es necesario seguir intercambiando el primer disco y el disco nuevo de acuerdo con los mensajes que aparezcan en pantalla, exactamente igual que al copiar un fichero.

```
COPY "A:ABC.*" TO "B:"
```

Esta orden copia todos los ficheros de un disco que tengan el nombre "ABC" en otro disco distinto.

```
COPY "A:PROG??.BAS" TO "B:"
```

La ejecución de esta orden copia en el disco destino todos los ficheros del disco fuente que respondan al nombre PROG seguido de dos letras y con una extensión "BAS" (por ejemplo, PROG01.BAS, PROG02.BAS, PROGAB.BAS, etc.).

```
COPY "A:*.*" TO "B:"
```

Esta es la orden que sirve para copiar todos los ficheros de un disco en otro disco.

Si solo se trabaja con una unidad de discos, es preciso intercambiar el disco fuente y el disco destino de acuerdo con los mensajes de pantalla hasta que salga el OK.

Cuando se dispone de dos o más unidades de discos

La orden COPY se utiliza en este caso de la misma forma que en el caso de trabajar con una sola unidad de discos. Se especifica como nombre de unidad 1 el nombre de la unidad donde está el disco a copiar, y como nombre de unidad 2 el nombre de la unidad que aloja al disco destino.

```
COPY "A:*.*" TO "B:"
```

Esta orden copia todos los ficheros del disco de la unidad A en el disco de la unidad B.

COPY (copy) (2)

Transfiere datos de imagenes entre la RAM de vídeo, variables de matriz y ficheros de disco.

FORMATO 1

COPY (coordenada de comienzo) –[STEP] (coordenada final) [, página fuente] TO (coordenada de comienzo de la transferencia), [página destino], [operación lógica]

FORMATO 2

- COPY (coordenada de comienzo) –[STEP] (coordenada final) [, página fuente] TO "[nombre de unidad] nombre de fichero
  variable de matriz [.extensión]"

FORMATO 3

- COPY "[nombre de la unidad] nombre del fichero [.extensión]"
  variable de matriz
  [, dirección] TO (coordenada de inicio de la transferencia), [página de destino] , [operación lógica]

FORMATO 4

- COPY "[nombre de la unidad] nombre del fichero [.extensión]"
  TO variable de matriz

FORMATO 5

- COPY variable de matriz TO "[nombre de la unidad] nombre del fichero [.extensión]"

Coordenada de comienzo, coordenada final

Cond. Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $-32768 \leqslant$ coordenada < 32768.

Página fuente, página destino

Cond. SCREEN 5,6 (64K-bytes)
Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $0 \leqslant$ página < 2.
SCREEN 5,6 (128K-bytes)
Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $0 \leqslant$ página < 4.
SCREEN 7,8 (128K-bytes)
Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $0 \leqslant$ página < 2.

Omit Página activa.

Operación lógica

Cond. PSET, PRESET, XOR, OR, AND, TPSET, TPRESET, TXOR, TOR, TAND *

Omit PSET.

Variable de matriz

Cond. Numérica

Dirección

Cond. Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $0 \leqslant$ dirección < 4.

Omit 0

Coordenada de comienzo de la transferencia

Cond. Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $-32768 \leqslant$ coordenada < 32768.

Nombre de la unidad

Cond. A:, B:, C:, D:, E:, F:, G:, H:

Omit Unidad de discos actual.

FUNCION Y UTILIZACION

Válida únicamente para SCREEN 5 a SCREEN 8.
Transfiere los datos desde la página fuente a la página destino.

La dirección es la siguiente:

0	de arriba-izquierda a abajo-derecha
1	de arriba-derecha a abajo-izquierda
2	de abajo-izquierda a arriba-derecha
3	de abajo-derecha a arriba-izquierda

Las variables de matriz deben ser suficientemente grandes para contener todos los datos de la imágen a copiar. Determinar el tamaño con la fórmula siguiente:

INT ((((ABS (coordenada final X − coordenada inicial X) + 1) * (ABS (coordenada final Y − coordenada inicial Y) + 1) * tamaño del pixel + 7) /8) + 4)/tamaño del dato) + 1

- El tamaño del pixel (el número de bits de la RAM de vídeo correspondientes a un punto de la pantalla) varía en función del modo SCREEN.

Modo SCREEN	Tamaño del pixel
5	4
6	2
7	4
8	8

- El tamaño del dato está determinado por el tipo de variable de matriz.

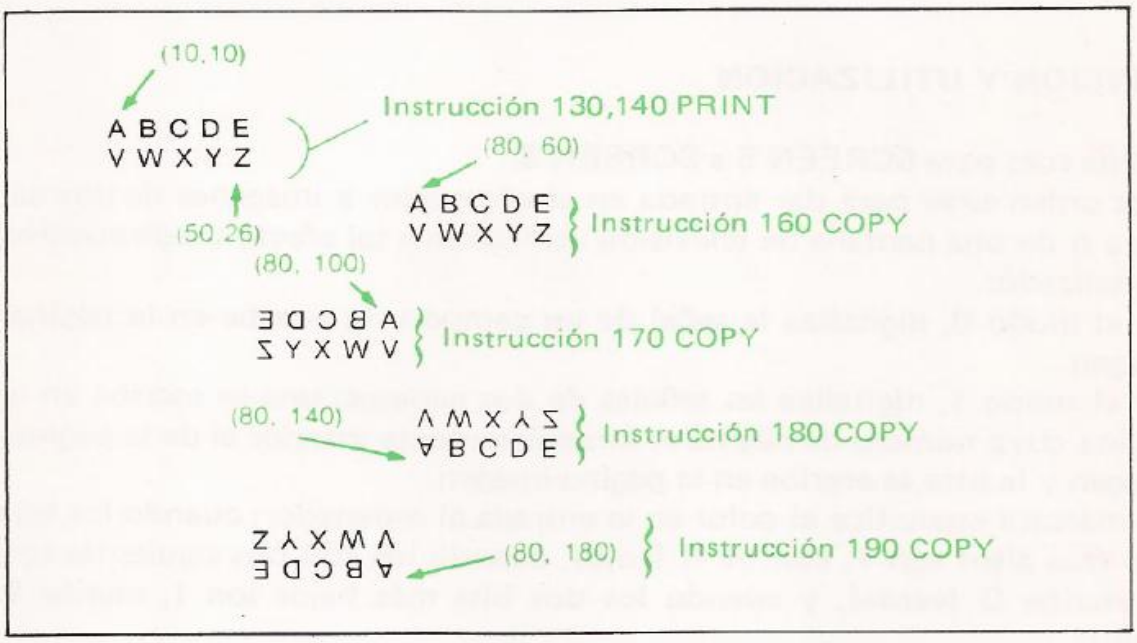
	Tamaño del dato
Entero	2
Precisión simple	4
Precisión doble	8

El tamaño normal del dato es 8.

Ejemplo de ejecución

```
100 SCREEN 5
110 DIM A(89)
120 OPEN "GRP:" FOR OUTPUT AS #1
130 PRESET(10,10):PRINT#1,"ABCDE"
140 PRESET(10,18):PRINT#1,"FGHIJ"
150 COPY (10,10)-(50,26),0 TO A
160 COPY A,0 TO (80,60),0
170 COPY A,1 TO (80,100),0
180 COPY A,2 TO (80,140),0
190 COPY A,3 TO (80,180),0
200 GOTO 200
```

Resultado del programa:



El valor 89 de la línea 110 DIM A(89) se ha calculado a partir de la siguiente fórmula:

$$\text{INT}(((((50-10)+1)*((26-10)+1)*4+7)/8)+4)/4)+1=89$$

COPY SCREEN (copy screen)

Digitaliza una señal de vídeo externa y la escribe en el VDP.

FORMATO

COPY SCREEN [modo] , [máscara]

Modo **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); 0 ≤ modo < 2.
Omit 0

Máscara **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); 0 ≤ máscara < 256.
Omit 255

FUNCION Y UTILIZACION

Válida solo para SCREEN 5 a SCREEN 8.
Esta orden sirve para dar entrada en el ordenador a imágenes de una cámara o de una pantalla de televisión utilizando a tal efecto un dispositivo digitalizador.
En el modo 0, digitaliza la señal de un campo y la escribe en la página-imagen.
En el modo 1, digitaliza las señales de dos campos; una se escribe en la página cuyo número de página es inmediatamente inferior al de la página-imagen y la otra se escribe en la página-imagen.
La máscara especifica el color en la entrada al ordenador; cuando los tres bits más altos son 1, escribe R (rojo), cuando los tres bits siguientes son 1, escribe G (verde), y cuando los dos bits más bajos son 1, escribe B (azul).
Con una máscara &B11111111, escribe R, G y B.

Ejemplo de ejecución

```
100 '--
110 VDP(7)=&HFF
120 SET VIDEO ,,1
130 COPY SCREEN
140 SET VIDEO ,,0
150 RETURN
160 '-- digitize
170 VDP(7)=&HFF
180 SET VIDEO ,,1
190 COPY SCREEN
200 RETURN
```

Función COS (cosine)

Da el valor del coseno de un dato numérico.

Afín: ATN, TAN, SIN

FORMATO

COS(X)

X **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico, en radianes).
Valor obtenido: Constantes de coma flotante; $-1 \leqslant$ valor $\leqslant 1$.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT COS(3.14/3)
 .500459689008l4
PRINT COS(60*3.14/180)
 .500459689008l4
```

Para poner X en grados, utilizar la fórmula COS (X * 3.14/180).

CSAVE (cassette save)

Almacena en una cinta de cassette un programa BASIC.

Opuesta: CLOAD Afín: SAVE

FORMATO

CSAVE "nombre de fichero" [, velocidad de transmisión]

Nombre del fichero — **Cond.** Una cadena de 6 o menos caracteres. Si se especifican 7 o más caracteres, ignora los caracteres siguientes al sexto.

Velocidad de transmisión — **Cond.** Constantes, variables, variables matriciales, sus expresiones (de tipo numérico); 1 ≤ velocidad en baudios < 3.

Omit 1

FUNCION Y UTILIZACION

Respecto a la velocidad de transmisión (baudios), cabe decir que cuando se especifica 1 la velocidad es 1200 baudios y cuando se especifica 2 es 2400.

Ejemplo de ejecución

```
CSAVE "PROG1"
```

Almacena en cinta de cassette un programa BASIC en memoria, bajo el nombre de "PROG1".

Función CSNG (convert to single precision)

Convierte datos numéricos en datos de precisión simple

Afín: CINT, CDBL, FIX

FORMATO

CSNG(X)

X **Cond.** Constantes, variables, variables matriciales, sus expresiones (de tipo numérico).

Valor obtenido: Dato numérico de simple precisión

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
10 PRINT SQR(3)
20 PRINT CSNG(SQR(3))
RUN
 1.7320508075688
 1.73205
```

Función **CSRLIN** (cursor line)

Da la coordenada Y de la posición del cursor.

FORMATO

CSRLIN

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
10 CLS
20 INPUT A$
30 PRINT A$;
40 CL=CSRLIN
50 LOCATE 0,CL+3:PRINT "END"
```

El dato (caracteres) que presenta en pantalla la línea 30 ocupa una sola línea o varias líneas, en función de su longitud. Pero en todo caso, la línea 40 introduce la coordenada Y (posición vertical) del cursor tras la presentación visual en la variable CL, y por lo tanto la palabra "FIN" se visualiza en una posición vertical (coordenada Y) de un valor superior en 3 unidades al valor de CL. La palabra "FIN" se visualiza por lo tanto 3 líneas más abajo, sea cual sea la longitud del dato A$.

Función **CVI** (convert to integer)
CVS (convert to single precision)
CVD (convert to double precision)

Convierten una cadena de caracteres en un dato numérico.

FORMATO

CVI (X$)

X$ **Cond.** Dato de caracteres de 2 bytes.
Valor obtenido: Tipo entero.

CVS (X$)

X$ **Cond.** Dato de caracteres de 4 bytes.
Valor obtenido: Precisión simple.

CVD (X$)

X$ **Cond.** Dato de caracteres de 8 bytes.
Valor obtenido: Precisión doble.

FUNCION Y UTILIZACION

En los ficheros de acceso aleatorio de disco, los datos numéricos se convierten en cadenas de caracteres y se almacenan así. Esta conversión está a cargo de las funciones MKI$, MKS$ y MKD$, y para hacer lo contrario están las funciones CVI, CVS y CVD: convierten las cadenas de caracteres leídas en ficheros de acceso aleatorio en los datos numéricos originales; CVI convierte una cadena de 2 caracteres (2 bytes) en un dato numérico entero; CVS convierte una cadena de 4 caracteres (4 bytes) en un dato numérico de coma flotante de simple precisión; y CVD convierte una cadena de 8 caracteres (8 bytes) en un dato numérico de coma flotante y doble precisión.

DATA (data)

Ofrece los datos que serán leídos por una instrucción READ.

Afín: READ, RESTORE

FORMATO

DATA constante [, constante] ...

Constante **Cond.** De tipo numérico o alfanumérico.

FUNCION Y UTILIZACION

- Cuando hay varios grupos de datos en una instrucción DATA, se separan mediante comas.
- Si los datos de una instrucción data se corresponden secuencialmente con las variables de una instrucción READ, la instrucción DATA se puede poner en cualquier sitio en relación con la instrucción READ correspondiente.
  Cuando un dato de tipo alfanumérico incluye una coma (,) o un signo de dos puntos (:), o cuando hay un espacio delante o detrás, se pone entre dobles comillas (").

Ejemplo de ejecución

```
10 CLS
20 SCREEN 2
30 READ A,B,C,D
40 LINE (A,B)-(C,D)
50 DATA 0,0,255,191
60 GOTO 60
```

DEF FN (define function)

Define una función de usuario.

FORMATO

DEF FN nombre de función [(parámetro [, parámetro] ...)] = expresión

Nombre de función	**Cond.**	Variables numéricas, variables de cadena (en función de la expresión).
Parámetro	**Cond.**	Hasta 9 variables.
Expresión	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (numéricas o alfanuméricas).

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
10 DEF FNA(X,Y)=(X*2+Y*3)/(X-Y)
20 B=FNA(4,2)
30 PRINT B
RUN
 7
```

La línea 10 define la función FNA (X,Y) según la expresión del segundo término. En la línea 20 se dan los valores 4 y 2 para los parámetros X e Y, y luego se hace una llamada a la función definida. El resultado (7) se asigna a la variable B.

DEFINT (define integer)
DEFSNG (define single precision)
DEFDBL (define double precision)
DEFSTR (define string)

Definir la relación entre el primer caracter del nombre de variable y el tipo de variable.

FORMATO

DEFINT	caracter [-caracter] [,caracter [-caracter]] ...
DEFSNG	caracter [-caracter] [,caracter [-caracter]] ...
DEFDBL	caracter [-caracter] [,caracter [-caracter]] ...
DEFSTR	caracter [-caracter] [,caracter [-caracter]] ...

Caracter **Cond.** Un caracter alfabético.

FUNCION Y UTILIZACION

```
DEFINT A-C
```

El resultado de esta instrucción es que todas las variables que comiencen por los caracteres A, B y C son consideradas de tipo entero.

Prioridad de los caracteres de declaración de tipo (%, !, #, $)
Tras la declaración de DEFINT A, A queda convertida en una variable de precisión doble al declararla después como A #.

Ejemplo de ejecución

```
10 DEFINT A-C ———— las variables A a la C son de tipo entero
20 A=1.23456789#  }
30 ABC=1.23456789# } A consecuencia de la línea 20, las variables A, ABC quedan convertidas en enteros
40 B#=1.23456789# —— De doble precisión poniendo #
50 C!=1.23456789# —— De precisión simple poniendo !
60 PRINT A;ABC;B#;C!
RUN
 1   1   1.23456789   1.23457
```

DEFUSR (define user)

Especifica la dirección inicial de una subrutina en lenguaje máquina que se ejecutará después con una función USR.

FORMATO

DEFUSR [X] = dirección inicial

X **Cond.** Constantes enteras; 0 ≤ X ≤ 9.
Omit 0
Dirección inicial **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); 0 ≤ dirección < 65536.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
DEFUSR1=&HE000
```

El resultado de esta instrucción es la definición como USR1 de la subrutina en lenguaje máquina que empieza en la dirección &HE000.

- La dirección inicial se puede redefinir en un programa todas las veces que sea necesario sin cambiar el valor del número (X).

DELETE (delete)

Borra una línea específica de un programa.

FORMATO

DELETE [número de línea] [-número de línea]

Número de línea **Cond.** Constantes enteras; 0 ≤número ≤65529.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
DELETE 40 ———— Borra la línea 40

DELETE 20-40 ———— Borra las líneas 20 a 40

DELETE -50 ———— Borra todas las líneas hasta la 50

DELETE ———— Borra una línea presentada en último lugar por una
            instrucción LIST o una línea interrumpida a conse-
            cuencia de un error.
```

- Cuando solo hay que borrar una línea, basta con escribir el número de línea y pulsar RETURN.

DIM (dimension)

Define la dimensión, el tamaño, el tipo de dato y el nombre de una variable de matriz.

Opuesta: ERASE

FORMATO

DIM nombre de variable (valor máximo de un subíndice [,valor máximo de un subíndice] ...) [,nombre de variable ( ), ...]

Variable Cond. Variable numérica o de cadena.

Valor máximo de un subíndice

Cond. Constantes, variables, variables de matriz, sus expresiones (enteros mayores que 0).

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
DIM A(15)
```

Reserva en memoria una zona de 16 variables de matriz numéricas desde A(0) a A(15). El valor inicial de las variables es 0.

```
DIM B$(2,3)
```

Reserva en memoria una zona de 12 variables de cadena, mostrada en la tabla siguiente. El valor inicial de las variables es una cadena nula.

B$(0,0)	B$(1,0)	B$(2,0)
B$(0,1)	B$(1,1)	B$(2,1)
B$(0,2)	B$(1,2)	B$(2,2)
B$(0,3)	B$(1,3)	B$(2,3)

Definición de múltiples variables de matriz mediante una instrucción DIM

```
DIM A(2),B$(4,2),C(3,3)
```

Las variables están separadas por comas.

Variables de matriz multi-dimensionales

Las variables de matriz multi-dimensionales se generan especificando 2 o más subíndices. (Se pueden especificar 255 dimensiones como máximo).

```
DIM X(3,4,5)
```
3 dimensiones

Omisión de la instrucción DIM

Cuando se utiliza una variable de matriz sin declararla previamente con una instrucción DIM, se considera que el valor máximo del subíndice es 10.

DRAW (draw)

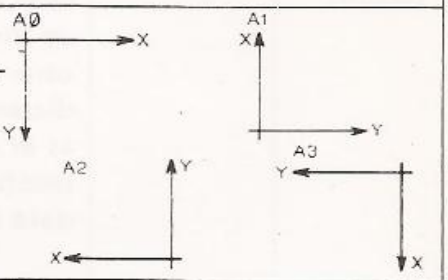
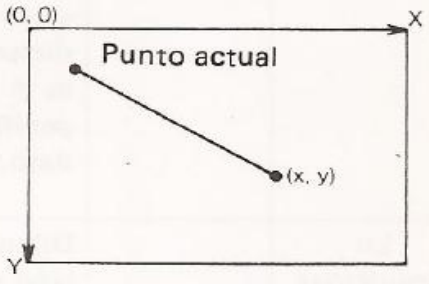
Dibuja gráficos en la pantalla en modo gráfico, siguiendo las especificaciones de los submandatos gráficos.

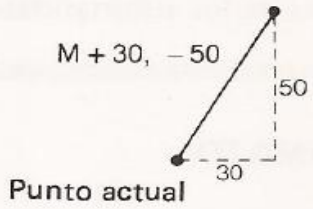
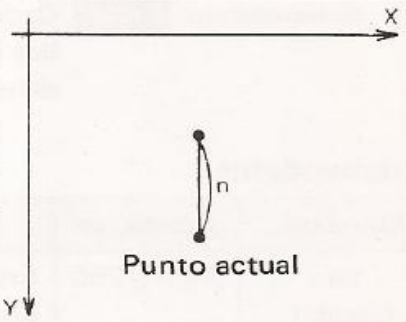
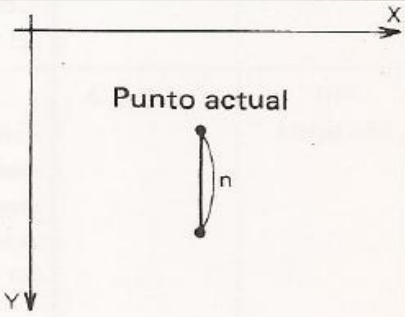
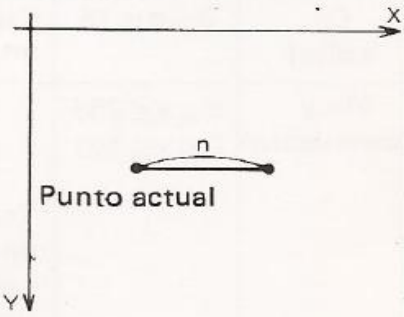
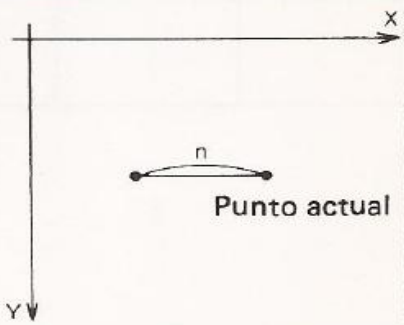
FORMATO

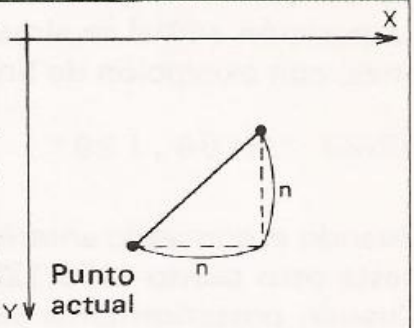
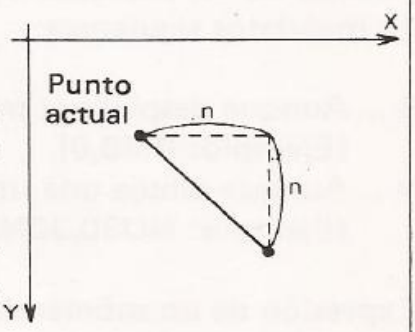
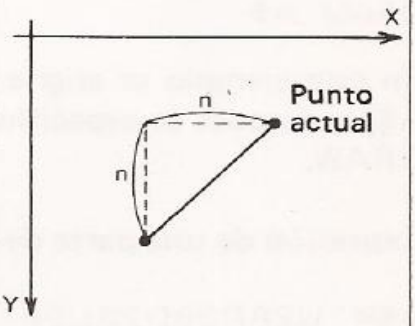
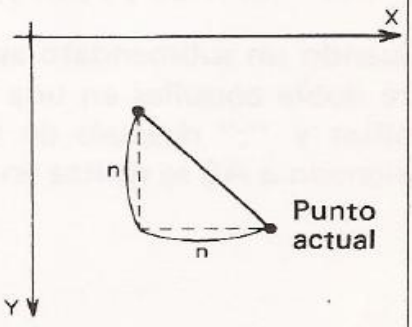
DRAW submandatos

Submandato **Cond.** Cadena de caracteres (constantes) encerradas entre comillas (" ") o variables de cadena que tienen asignada una cadena de caracteres. Caracteres en mayúsculas o minúsculas.

Submandatos

Mandato	Condición	Significado
Sn (escala)	$0 \leqq n \leqq 255$	Especifica el número de puntos correspondiente a una unidad cuando se dibuja una línea. Con n = 1, 1/4 de punto. El valor inicial es S4.
An (ángulo)	$0 \leqq n \leqq 3$	Gira el sistema de coordenadas un paso de 90° a partir de un eje de coordenadas estandar (0°). El valor inicial es A0. 
Cn (color)	$0 \leqq n \leqq 15$	Especifica un color para una línea dibujada mediante un código de color. El valor inicial es C15.
Mx,y (movimiento)	$0 \leqq x \leqq 255$ $0 \leqq y \leqq 191$	Traza una línea desde un punto actual hasta una posición absoluta (x,y). 

Mandato	Condición	Significado
M ± x, ± y (movimiento)	0 ≦ x ≦ 255 0 ≦ y ≦ 191	Desplaza horizontalmente ± x y verticalmente ± y desde un punto actual. La unidad de x,y es el número de puntos especificado por el submandato S. 
Un (arriba)		Dibuja una línea en dirección negativa del eje y, desde un punto actual hasta otro punto situado a una distancia n. La unidad de n es el número de puntos especificado por el submandato S (1 por omisión). 
Dn (abajo)		Dibuja una línea en dirección positiva del eje y, desde un punto actual hasta otro punto situado a una distancia n. La unidad de n es el número de puntos especificado por el submandato S (1 por omisión). 
Rn (derecha)		Dibuja una línea en dirección positiva del eje x, desde un punto actual hasta otro punto situado a una distancia n. La unidad de n es el número de puntos especificado por el submandato S (1 por omisión). 
Ln (izquierda)		Dibuja una línea en dirección negativa del eje x, desde un punto actual hasta otro punto situado a una distancia n. La unidad de n es el número de puntos especificado por el submandato S (1 por omisión). 

Mandato	Condición	Significado
En		Dibuja una línea en la dirección positiva del eje X y en la dirección negativa del eje Y, desde el punto actual hasta otro punto situado a una distancia n. La unidad de n es el número de puntos especificado por el submandato S (1 por omisión). 
Fn		Dibuja una línea en la dirección positiva del eje X y en la dirección positiva del eje Y, desde el punto actual hasta otro punto situado a una distancia n. La unidad de n es el número de puntos especificado por el submandato S (1 por omisión). 
Gn		Dibuja una línea en la dirección negativa del eje X y en la dirección positiva del eje Y, desde el punto actual hasta otro punto situado a una distancia n. La unidad de n es el número de puntos especificado por el submandato S (1 por omisión). 
Hn		Dibuja una línea en la dirección negativa del eje X y en la dirección negativa del eje Y, desde el punto actual hasta otro punto situado a una distancia n. La unidad de n es el número de puntos especificado por el submandato S (1 por omisión). 

FUNCION Y UTILIZACION

La posición actual se almacena siempre con un mandato para dibujar una línea, con excepción de Sn, An y Cn. Por ejemplo,

```
DRAW "M100,120"
```

Cuando el comando anterior dibuja una línea desde un punto determinado hasta otro punto (100,120), este último punto pasa a ser el punto actual. Cuando posteriormente se ejecute una orden de dibujo de una línea, dibujará una línea desde este último punto a un punto especificado.
Delante de un mandato de dibujo de una línea se pueden colocar uno de los mandatos siguientes:

B ... Aunque desplaza el punto actual, no dibuja línea alguna.
(Ejemplo: BM0,0)
N ... Aunque dibuja una línea, no desplaza el punto actual.
(Ejemplo: NU30,30NR30,30)

Expresión de un submandato con una variable

```
A$="BM100,150U50E50F50D50L100"
DRAW A$
```

En este ejemplo se asigna primero un submandato a la variable de cadena A$, y después se especifica que A$ es un submandato de una instrucción DRAW.

Expresión de una parte de un submandato con una variable (X variable;).

```
A$="U20R20D20L20"
DRAW "BM50,50XA$;"
DRAW "BM150,100XA$;"
```

Cuando un submandato asignado a una variable de cadena está puesto entre doble comillas en una instrucción DRAW, añadir "X" antes de las comillas y ";" después de tal variable. En este ejemplo, un submandato asignado a A$ se utiliza en dos instrucciones DRAW.

Expresión de n en un submandato con una variable (= variable;)

La n que expresa la distancia, el ángulo y el código de color con cada suborden puede ser una constante o una variable en una instrucción DRAW. Cuando está expresada mediante una variable, añadir "=" delante y ";" detrás de dicha variable.

```
X=40
DRAW "U=X;"
```

es lo mismo que

```
DRAW "U40"
```

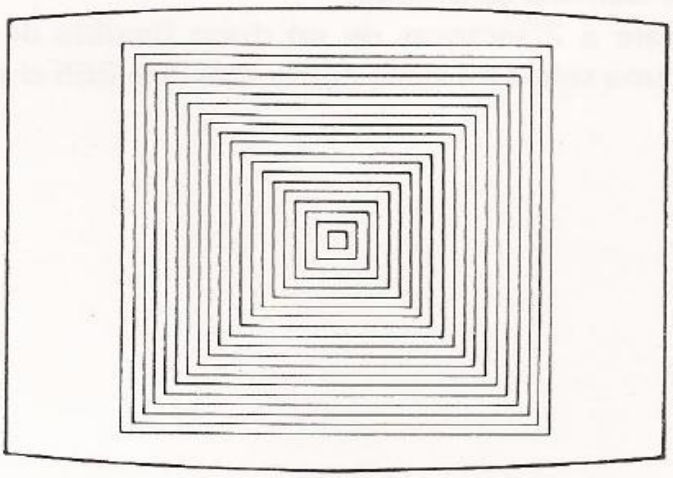
Ejemplo de ejecución

```
10 SCREEN 2
20 DRAW "BM125,100"
30 FOR I=4 TO 240 STEP 12
40 DRAW "S=I;BURD2L2U2RBD"
50 NEXT I
60 GOTO 60
```

Cuando se utilice una sentencia DRAW, habrá que usar el modo gráfico.

Movimiento hasta (125,100) sin dibujar nada

Dibuja continuamente cuadrados de diferentes tamaños



Función DSKF (disk free)

Da el espacio libre que queda en el disco en unidades de clusters.

FORMATO

DSKF (número de unidad)

Número de unidad **Cond.** Constantes, variables, variables matriciales, sus expresiones (de tipo numérico); 0 ≤número <9.

Valor obtenido: Tipo entero.

FUNCION Y UTILIZACION

El número de unidad 1 corresponde a la unidad A, el número 2 a la unidad B, el 3 a la C y así sucesivamente.
El número de unidad 0 corresponde a la unidad en activo.
Da el espacio libre que queda en el disco alojado en la unidad de discos especificada por el número de unidad.

- 1 cluster equivale a 2 sectores de un disco flexible de 3,5 pulgadas. Una unidad de una sola cara tiene como máximo 355 clusters.

END (end)

Da por terminada la ejecución de un programa y establece el estado de espera de mandatos.
Cuando hay un fichero abierto, lo cierra.

FORMATO

END

FUNCION Y UTILIZACION

La instrucción END se pone en la última línea de un programa cuando a continuación hay una subrutina, con el objeto de evitar que la subrutina sea ejecutada otra vez tras la finalización del programa principal. Se puede poner en un programa las veces que se quiera, por ejemplo, cuando el resultado de la ejecución de un programa tiene efectos en alguna bifurcación. Se puede poner al final de cada bifurcación.

- Para ejecutar otra vez el programa se utiliza una instrucción RUN o GOTO. No se puede continuar con una instrucción CONT.

```
        .
        .
100 GOSUB 1000
        .
        .
190
200 END
1000 ' SUBROUTINE
        .
        .
1100 RETURN
```

En este programa, si no estuviera la instrucción END de la línea 200, se ejecutaría la subrutina de la línea 1000 sin una instrucción GOSUB, tras regresar desde la subrutina y ejecutar la línea 190, se produciría un error.

Función EOF (end of file)

Cuando ha sido leído el último dato de un fichero da un −1; en los demás casos da un 0.

FORMATO

EOF (número de fichero)

Número de fichero **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo entero); 1 ⩽número de fichero ⩽número especificado por MAXFILE

Valor obtenido: Tipo entero (−1 o 0).

FUNCION Y UTILIZACION

```
IF EOF(1) THEN CLOSE #1
```

Cuando se lean los últimos datos del fichero, mientras se están leyendo datos del fichero cuyo número de fichero sea 1, la instrucción anterior cerrará el fichero.

ERASE (erase)

Borra variables de matriz. Opuesta: DIM

FORMATO

ERASE nombre de variable de matriz [,nombre de variable de matriz]

FUNCION Y UTILIZACION

```
10 DIM A(100),B$(4,3)
        .
        .
100 ERASE A,B$
```

En este ejemplo, la línea 100 borra las variables de matriz A y B$ declaradas en la línea 10. De esta forma se puede utilizar esa zona de memoria para otros fines. Además, mediante una instrucción DIM se puede volver a definir una variable de matriz con el mismo nombre.

Función ERL (error line)

Ofrece el número de línea donde se ha producido un error.

Afín: ERR

FORMATO

ERL

Valor obtenido: Tipo numérico.

FUNCION Y UTILIZACION

Cuando no hay ningún error da cero, y cuando hay un error que es consecuencia de un mandato directo, da un número comprendido entre 1 y 65535.
Se utiliza en combinación con una instrucción ON ERROR o con una instrucción ERROR.

Función ERR (error)

Ofrece el número de error de un error detectado.

Afín: ERL, ERROR

FORMATO

ERR

Valor obtenido: Tipo entero.

FUNCION Y UTILIZACION

Se puede usar para el proceso de errores en un programa en combinación con una instrucción ERROR o con la función ERL.

- Si no se ha producido ningún error, se obtendrá 0.

Ejemplo de ejecución

```
PRINT 10/0
Division by zero
Ok
PRINT ERR
 11
```

ERROR (error)

Simula el error de un número de error especificado o define un número de error.

Afín: ERR

FORMATO

ERROR número de error

Número de error **Cond.** Constantes, variables, variables matriciales, sus expresiones (numéricas); 0 ≤número <256.

FUNCION Y UTILIZACION

```
ERROR 1
```
Genera un error "NEXT without FOR"
(Detiene la ejecución del programa)

Definición por el usuario del número de error

```
IF A<0 THEN ERROR 250
```

Cuando se asigna a una variable un número negativo, la instrucción anterior da lugar al error 250. (Como en el BASIC–MSX2 están definidos los números de error del 0 al 71, hay que utilizar números superiores al 71).

Ejemplo de ejecución

Cuando se introduce un número negativo en el programa siguiente, se visualiza en pantalla un mensaje de error que indica que es necesario un número positivo, y la ejecución del programa continúa.

```
10 ON ERROR GOTO 90
20 FOR I=1 TO 10
30 INPUT A
40 IF A<0 THEN ERROR 250
50 SUM=SUM+A
60 NEXT I
70 PRINT SUM
80 END
90 IF ERR=250 THEN PRINT "Introduzca un número
positivo"      :RESUME 30
100 PRINT "Error!"
```

Función **EXP** (exponential)

Ofrece e^X que es la función exponencial natural de X.

- e (2.7182818284588) es la base de los logaritmos naturales.

FORMATO

EXP(X)

X **Cond.** Constantes, variables, variables de matriz, sus expresiones, de tipo numérico, de valor inferior a 145,06286085862.

Valor obtenido: Tipo numérico de coma flotante.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT EXP(100)
 2.688117141808 7E+43
```

FIELD (field)

Especifica el formato del registro de entrada/salida de un fichero de acceso aleatorio.

FORMATO

FIELD [#] número de fichero, longitud de caracteres AS variable de cadena [,longitud de caracteres AS variable de cadena] ...

Número de fichero	Cond.	Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); 1 ⩽ número de fichero ⩽ número de fichero especificado por MASFILES
Longitud de caracteres	Cond.	Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); 0 ⩽ longitud de caracteres ⩽ longitud del registro. El total de longitudes de caracteres de una instrucción FIELD no debe superar la longitud del registro.
Variable de cadena	Cond.	Una variable de cadena.

FUNCION Y UTILIZACION

El formato del registro se define mediante una instrucción FIELD antes de realizar la entrada/salida de datos de un fichero de acceso aleatorio mediante las instrucciones GET o PUT.

El número de fichero especifica el fichero ya abierto por una instrucción OPEN. Tras la ejecución de la instrucción FIELD, cuando se ejecuta la instrucción de asignación que utiliza la variable de cadena especificada por la instrucción FIELD, esta variable de cadena pasa a la zona de variables de cadena de la memoria y la especificación de la instrucción FIELD ya no es válida.

```
FIELD #1,20 AS A$,30 AS B$,40 AS C$
```

Esta instrucción FIELD asigna las variables de cadena A$, B$ y C$ al registro de entrada/salida que utiliza el fichero ≠ 1. La longitud de esas variables es 20, 30 y 40 bytes repectivamente. Si el total de las longitudes (en este caso 90 bytes) superará la longitud de registro especificada en la instrucción OPEN, daría un error.
La longitud máxima para un registro es de 256 bytes, así la suma total de las longitudes de las variables especificadas en una instrucción FIELD no debe ser superior a 256 bytes. La longitud de una variable no debe ser superior a 255 bytes.

FILES (ficheros)

Presenta en pantalla los nombres de los ficheros que hay en el disco especificado.

FORMATO

FILES [" [nombre de la unidad] [nombre del fichero [.extensión]] "]

Nombre de la unidad	**Cond.**	A:,B:,C:,D:,E:,F:,G:,H:
	Omit	Unidad de disco en activo
Nombre del fichero	**Cond.**	Una cadena de 8 o menos caracteres
Extensión	**Cond.**	Una cadena de 3 o menos caracteres

FUNCION Y UTILIZACION

El signo de interrogación (?) puede sustituir a uno o más caracteres del nombre de fichero o de la extensión. Un asterisco (*) puede sustituir a todo el nombre de fichero o a toda la extensión.

Ejemplo de ejecución

```
FILES "A:TEST.DAT"
```

Esta instrucción FILES pide al ordenador que busque el nombre de fichero TEST.DAT en el disco alojado en la unidad A y que saque en pantalla TEST.DAT si lo encuentra. Si no lo encuentra, saldrá en pantalla el mensaje "File not found" (fichero no encontrado).

Cuando se especifica

```
FILES "A:TEST??.DAT"
```

saldrán en pantalla los nombres de todos los ficheros cuyos nombres tengan las letras TEST como los cuatro primeros caracteres, seguidas por otros dos caracteres cualquiera, y cuya extensión sea DAT.

Cuando se especifica

```
FILES "A:TEST.*"
```

Saldrán en pantalla todos los ficheros con el nombre TEST independientemente de su extensión.

```
FILES "A:"
```

presentará en pantalla los nombres de todos los ficheros del disco de la unidad A.

Cuando se omite el nombre de la unidad, como en

```
FILES "TEST.*"
```

queda especificada, por omisión, la unidad de discos en activo.

Si se ejecuta solo la orden

```
FILES
```

saldrán en pantalla todos los nombres de fichero del disco de la unidad en activo.

Función **FIX** (redondear)

Ofrece el entero de un dato numérico. Afín: CINT, CDBL, CSNG

FORMATO

FIX(X)

X **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico).
Valor obtenido: Tipo numérico.

FUNCION Y UTILIZACION

Ofrece el valor de un dato numérico X después de eliminar las cifras situadas a la derecha del punto decimal.

Ejemplo de ejecución

```
PRINT FIX(3);FIX(-3);FIX(3.58);FIX(-3.58)
 3 -3  3 -3
```

FOR—NEXT (for—next)

Repite la ejecución de la parte del programa comprendida entre una instrucción FOR y su correspondiente instrucción NEXT

FORMATO

```
FOR variable = valor inicial TO valor final [STEP incremento]
  :
  :
NEXT [variable]
```

Variable — **Cond.** Variable de tipo numérico. Las variables de la instrucción FOR deben coincidir con las de la instrucción NEXT.

Valor inicial, valor final

Cond. Constantes, variables, variables de matriz, sus expresiones (de tipo numérico).

Incremento — **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico).

Omit 1

FUNCION Y UTILIZACION

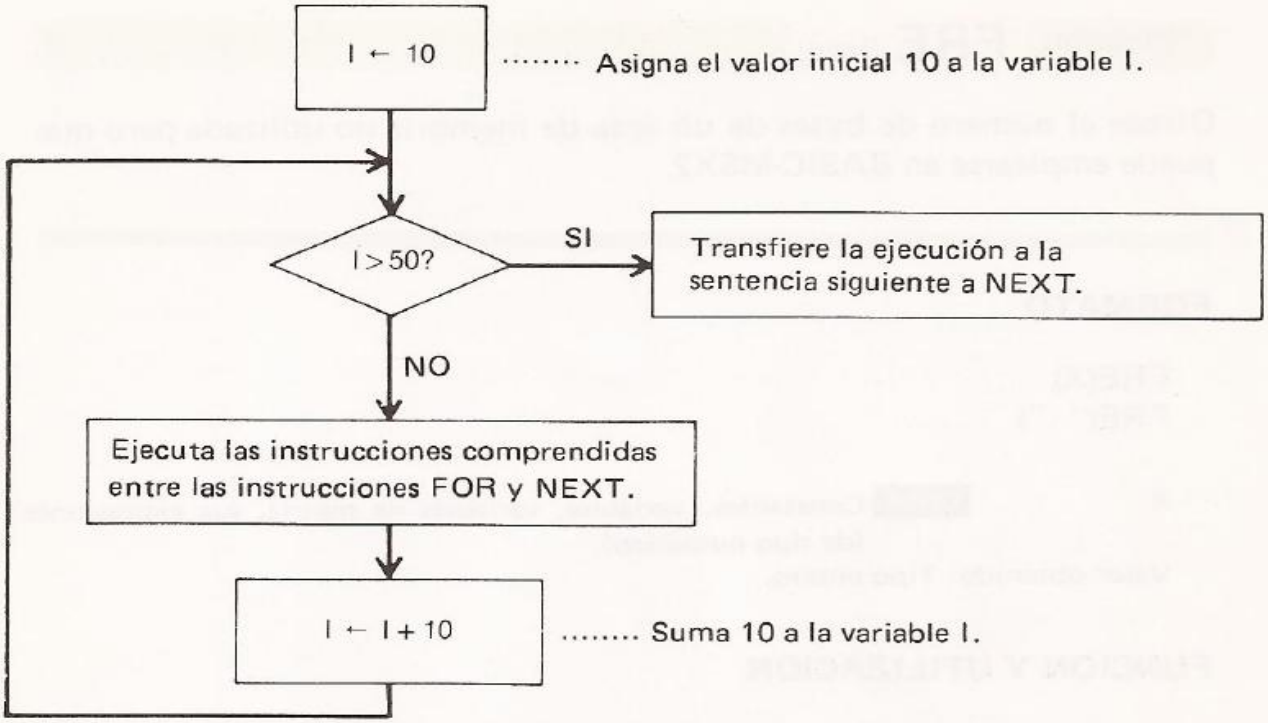
La parte del programa comprendida entre una instrucción FOR y la instrucción NEXT correspondiente, se ejecuta repetidamente mientras el valor de la variable especificada en la instrucción FOR va siendo incrementado desde un determinado valor inicial, hasta que alcanza el valor final determinado. El valor de la variable es incrementado una cantidad especificada cada vez que se llega al final de la ejecución de esa parte del programa.

- Aunque se puede omitir la variable de la instrucción NEXT, la correspondencia entre las instrucciones FOR y NEXT se puede ver más fácilmente si el listado del programa incluye tal variable, es decir, si se ha escrito.

Ejemplo de ejecución

```
10 FOR I=10 TO 50 STEP 10
20 PRINT "I=";I
30 NEXT I
```

La ejecución de este programa sigue la secuencia siguiente:



Bucles múltiples

Cabe la posibilidad de poner un bucle FOR—NEXT en el interior de otro bucle FOR—NEXT. En este caso, el bucle interior debe estar alojado totalmente en el bucle exterior. En cada bucle se utiliza una variable distinta.

```
10 FOR I=1 TO
20 FOR J=1 TO
30 PRINT "*";
40 NEXT J
50 PRINT
60 NEXT I
RUN
*
**
***
****
*****
```

Bucle interior

Bucle exterior

Cabe la posibilidad también de poner fin a varias instrucciones FOR con una instrucción NEXT. En tal caso, no se puede omitir en la instrucción NEXT el nombre de variable. Las variables están ordenadas secuencialmente (las del bucle interior en primer lugar) y separadas por comas.

```
FOR I=0 TO 10
FOR J=0 TO 5
        .
        .
NEXT J,I
```

Función FRE (free)

Ofrece el número de bytes de un área de memoria no utilizada pero que puede emplearse en BASIC-MSX2.

FORMATO

```
FRE(X)
FRE(" ")
```

X **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico).

Valor obtenido: Tipo entero.

FUNCION Y UTILIZACION

```
PRINT FRE(0)
```
Presenta en pantalla el número de bytes del área de memoria no utilizada.

```
PRINT FRE("")
```
Presenta en pantalla el número de bytes del área de cadenas no utilizada.

GET (get)

Lee un registro en el fichero de acceso aleatorio y asigna valores a cada variable siguiendo el formato definido por la instrucción FIELD.

Opuesta: PUT

FORMATO

GET [#] número de fichero [,número de registro]

Número de fichero **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); 1 ≤ número de fichero ≤ número especificado por MAXFILES

Número de registro **Cond.** Constantes (de tipo entero); 1 ≤ número de registro ≤ 65535.

Omit 1 + el número de registro utilizado en la instrucción GET o PUT ejecutada en último lugar.

FUNCION Y UTILIZACION

La instrucción

```
GET #1,3
```

lee el tercer registro del fichero de datos y asigna valores a cada variable especificada por la instrucción FIELD.

Ejemplo de ejecución

```
10 OPEN "A:TEST.DAT" AS #1
20 FIELD #1,2 AS A$,15 AS B$,10 AS C$
30 FOR I=1 TO LOF(1)/255
40 GET #1,I
50 PRINT CVI(A$);" ";B$;" ";C$
60 NEXT I
70 CLOSE #1
80 END
```

GET DATE (get date)

Lee la fecha en el reloj interno y la asigna a una variable de cadena.

Opuesta: SET DATE Afín: GET TIME

FORMATO

GET DATE T$ [,A]

T$ **Cond.** Constantes de cadena, variables, variables de matriz, sus expresiones.

A **Omit** Fecha actual.

FUNCION Y UTILIZACION

El formato de la fecha es DD/MM/AA, donde DD es el día, MM es el número del mes y AA es el año.
Cuando se especifica A se puede programar la alarma.
Para más información sobre el empleo de la función de alarma, ver el manual del ordenador.

Ejemplo de ejecución

```
10 GET DATE T$
20 PRINT T$
```

GET TIME (get time)

Lee la hora en el reloj interno y la asigna a una variable de cadena.

Opuesta: SET TIME Afín: GET DATE

FORMATO

GET TIME T$ [,A]

T$ **Cond.** Constantes de cadena, variables, variables de matriz, sus expresiones.

A **Omit** Hora actual.

FUNCION Y UTILIZACION

El formato de la hora es HH:MM:SS, donde HH es la hora, MM son los minutos y SS es el número de segundos.
Cuando se especifica A se puede programar la alarma.
Para más información sobre el empleo de la función de alarma, ver el manual del ordenador.

Ejemplo de ejecución

```
10 GET TIME X$
20 PRINT X$
```

GOSUB—RETURN (go to subroutine—return)

Transfiere el control del programa a una subrutina especificada. La instrucción RETURN señala el fin de la subrutina y devuelve el control de ejecución al número de línea especificado a continuación de RETURN o, en su defecto, al número de línea siguiente a la instrucción GOSUB.

Afín: GOTO

FORMATO

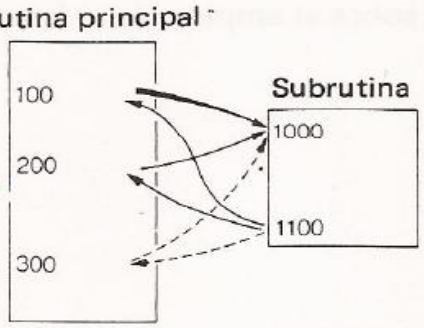
```
GOSUB número de línea
  :
RETURN [número de línea]
```

Número de línea **Cond.** Enteros, 0 ⩽ número ⩽ 65529.

Omit En ausencia de número de línea en la instrucción RETURN, se da por entendido que es el número de línea siguiente a la instrucción GOSUB.

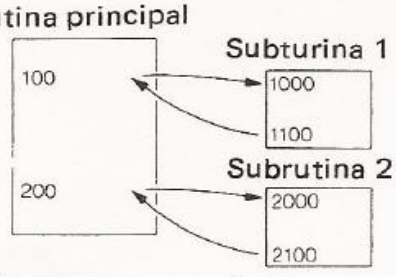
FUNCION Y UTILIZACION

```
        .
        .
100  GOSUB 1000
        .
        .
200  GOSUB 1000
        .
        .
300  GOSUB 1000
        .
        .
1000 ' SUBROUTINE
        .
        .
1100 RETURN
```



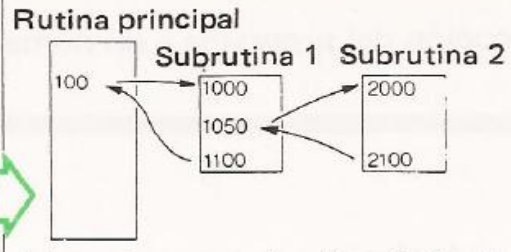
Llamadas a una subrutina mediante varias instrucciones GOSUB.

```
        .
        .
100  GOSUB 1000
        .
        .
200  GOSUB 2000
        .
        .
1000 ' SUBROUTINE 1
        .
        .
1100 RETURN
        .
        .
2000 ' SUBROUTINE 2
        .
        .
2100 RETURN
```



Programa con varias subrutinas.

```
        .
        .
100 GOSUB 1000
        .
        .
1000 ' SUBROUTINE 1
        .
        .
1050 GOSUB 2000
        .
        .
1100 RETURN
        .
        .
2000 ' SUBROUTINE 2
        .
        .
2100 RETURN
```



Llamada a una subrutina desde otra subrutina (multiplexado de instrucciones GOSUB).

La posibilidad de disposición de instrucciones GOSUB multiplexadas depende de la capacidad de memoria disponible.

GOTO (go to)

Transfiere la ejecución del programa a un número de línea especificado.

FORMATO

GOTO número de línea

Número de línea **Cond.** Enteros, 0 ⩽número ⩽65529.

FUNCION Y UTILIZACION

La ejecución del programa se transfiere a la línea especificada por la instrucción GOTO.

- Al trabajar en el modo de mandatos directos, la ejecución empieza en la línea especificada.

Función HEX$ (hexadecimal dollar)

Ofrece la expresión haxadecimal de un dato numérico en forma de datos alfanuméricos.

Afín: BIN$, OCT$

FORMATO

HEX$(X)

X **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $-32768 \leqslant X < 65535$. En el caso de números negativos, su valor es el que resulta de sumarles 65536.

Valor obtenido: Tipo alfanumérico.

FUNCION Y UTILIZACION

```
PRINT HEX$(100)
64

PRINT HEX$(-32768)
8000

PRINT HEX$(255)
FF
```

IF—THEN—ELSE (if—then—else)

Bifurca la ejecución del programa de acuerdo con el valor de una expresión.

FORMATO

IF expresión THEN instrucción [ELSE instrucción]

Expresión	**Cond.**	Una expresión de relación cuyo resultado es una expresión numérica, una expresión lógica o una expresión aritmética.
ELSE sentencia	**Omit**	A la sentencia siguiente a THEN si el valor de la expresión es cierto (se cumple) y a la línea siguiente si es falso (si no se cumple).

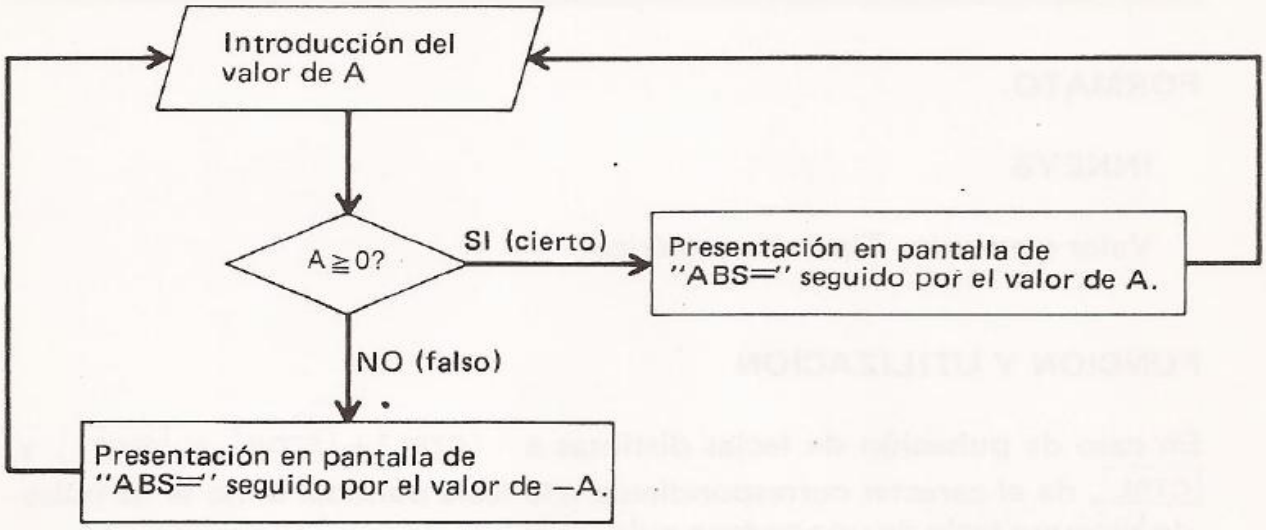
FUNCION Y UTILIZACION

Si el valor de una expresión es cierto (si se cumple esa expresión), se ejecuta la sentencia siguiente a THEN, y si es falso (si no se cumple esa expresión), se ejecuta la sentencia siguiente a ELSE; a continuación se ejecuta la línea siguiente.

- En caso de omisión de la instrucción ELSE, si el valor de la expresión es cierto se ejecuta la instrucción siguiente a THEN, y si el valor de la expresión es falso, se ignora la instrucción siguiente a THEN y la ejecución se transfiere a la línea siguiente.
- En el formato IF—THEN GOTO, cabe la posibilidad de omitir THEN o GOTO.
  IF A = 0 THEN 30
  IF A = 0 GOTO 30 } — igual significado
  Tras THEN va una instrucción o número de línea.
  Tras GOTO va un número de línea.
- Cuando GOTO va detrás de ELSE, se puede omitir.
- Cuando después de THEN o ELSE van varias instrucciones, son ejecutadas secuencialmente comenzando por la izquierda; estas instrucciones están separadas por el signo de dos puntos (:).

Ejemplo de ejecución

```
10 INPUT A
20 IF A>=0 THEN PRINT "ABS=";A ELSE PRINT
"ABS=";-A
30 GOTO 10
```



Sentencias IF—THEN multiplexadas

Tras THEN o ELSE puede ir otra sentencia IF—THEN. Se pueden poner múltiples instrucciones IF—THEN dentro de los límites de una línea.

Función INKEY$ (inkey dollar)

Ofrece el caracter correspondiente a la tecla pulsada y una cadena nula en caso de no haber pulsado ninguna tecla.

Afín: INPUT

FORMATO

INKEY$

Valor obtenido: Tipo alfanumérico.

FUNCION Y UTILIZACION

En caso de pulsación de teclas distintas a CTRL + STOP, y SHIFT, y CTRL, da el caracter correspondiente a la tecla pulsada. Si no se ha pulsado ninguna tecla da una cadena nula.

Ejemplo de ejecución

```
10 CLS
20 PRINT   "Pulse una tecla".
30 K$=INKEY$
40 IF K$="" THEN GOTO 30
50 PRINT K$;
60 GOTO 30
```

Se repite hasta que haya sido pulsada una tecla

Cuando se pulsa una tecla, la línea 30 asigna el caracter correspondiente a la variable K$ y la línea 50 lo presenta en pantalla.

Función INP (input)

Leè datos del port de entrada/salida especificado. Afín: OUT

FORMATO

INP (número de port)

Número de port **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); 0 ≤número <256.

Valor obtenido: Tipo numérico.

FUNCION Y UTILIZACION

Introduce y presenta datos procedentes del port de E/S especificado.
Para más información sobre la asignación de ports de entrada/salida consulte la página 264.

INPUT (input)

Introduce el valor de una variable desde el teclado.

Afín: INPUT$, INPUT #, LINE INPUT

FORMATO

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

Variable	**Cond.**	Numérica, de cadena, sus variables de matriz.
Mensaje	**Cond.**	Sentencia de comentario para la introducción de datos (cadena).
	Omit	Presenta visualmente solo "?" en ausencia del mensaje de petición de entrada.

FUNCION Y UTILIZACION

Introduce datos desde el teclado y los asigna a una variable. Los espacios en blanco introducidos antes de los datos son ignorados.

- La instrucción INPUT correspondiente a una variable de tipo numérico ignora también los espacios dispuestos en el centro de los datos.
- Cuando se introduce una coma, la instrucción INPUT la considera un signo de separación de datos; considera que los conceptos anteriores a la coma son un conjunto de datos asignados a una variable, y no asigna la coma.
- Cuando se escribe un mensaje de entrada, lo presenta visualmente cuando hace la petición de entrada de datos. Si se omite el mensaje de petición de entrada, presenta únicamente el signo "?".
- El número de variables debe coincidir con los datos.

Ejemplo de ejecución

```
10 INPUT A
RUN
?
```

Cuando se omite el mensaje de petición de entrada

```
10 INPUT "A=";A
RUN
A=?
```

Cuando se incluye un mensaje de petición de entrada

```
10 INPUT "A AND B ";A,B
RUN
A AND B ? 7
??
```

Como el número de datos introducidos es inferior al número de variables, pide los datos que faltan con ??

```
10 INPUT "A AND B ";A,B
RUN
A AND B ? 1,2,3,4
?Extra ignored
```

Presenta ese mensaje (Extra ignored) cuando el número de datos introducidos es mayor que el número de variables.

Función INPUT$ (input dollar)

1. Introduce un número especificado de caracteres desde el teclado.
2. Introduce un número especificado de caracteres desde un fichero.

Afín: INPUT, INPUT#, LINE INPUT

FORMATO

1. INPUT$(N)
2. INPUT$(N, [#] número de fichero)

N — **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $1 \leqslant N < 256$.

Número de fichero — **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo entero); 1 ⩽ número de fichero ⩽ número especificado por la instrucción MASFILES

Valor obtenido: Tipo alfanumérico.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
10 X$=INPUT$(5)
20 PRINT X$
```

La ejecución de la línea 10 acarrea el estado de espera de entrada de datos por teclado, y tras la introducción de 5 caracteres, los asigna a la variable X$. Durante la introducción de caracteres por el teclado, no los presenta en pantalla.

```
10 OPEN "CAS:TEST" FOR INPUT AS #1
20 X$=INPUT$(50,#1)
30 CLOSE
```

Este programa introduce 50 caracteres desde un fichero almacenado en una cinta cassette y los asigna a una variable de cadena X$; luego cierra el fichero.

Límites de "N"

En el estado inicial, si N sobrepasa los límites 1 a 200, se produce un error. Cuando se dispone que el área de caracteres sea de tamaño superior a 255, mediante una instrucción CLEAR, se puede seleccionar un valor comprendido entre 1 y 256.

INPUT # (input number)

Lee datos de un fichero abierto por una instrucción OPEN y los asigna a una variable.

Afín: INPUT, INPUT$, LINE INPUT #

FORMATO

INPUT # número de fichero, variable [,variable] ...

Número de fichero — **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo entero); 1 ⩽número de fichero ⩽número especificado por la instrucción MAXFILES=.

Variable — **Cond.** Numérica o de cadena, o sus variables de matriz.

FUNCION Y UTILIZACION

Lee datos de un fichero. Si es numérico, ignora los códigos de espacio, de retorno y de salto de línea que preceden al dato.
Si el dato es una cadena, considera como un solo dato los caracteres comprendidos entre el primer caracter y el caracter anterior a un código de espacio, coma, retorno y salto de línea. Si los caracteres están encerrados entre dobles comillas (" "), solo lee y considera como datos estos caracteres.

Ejemplo de ejecución

```
10 OPEN "CAS:DATA" FOR INPUT AS #1
20 IF EOF(1) THEN GOTO 50
30 INPUT #1,A$:PRINT A$
40 GOTO 20
50 CLOSE #1
```

Abre un fichero para su lectura

Lee datos, asigna a la variable AS y los presenta en pantalla

Función INSTR (in string)

Localiza una cadena de caracteres especificada entre otras cadenas y da su posición.

FORMATO

INSTR ([N,] X$, Y$)

N	Cond.	Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); o $\leqslant N < 256$.
	Omit	1
X$, Y$	Cond.	Constantes, variables, variables de matriz, sus expresiones (de tipo de cadena).

Valor obtenido: Tipo entero.

FUNCION Y UTILIZACION

Ofrece en forma de datos numéricos el número de caracter de la cadena X$ (contando a partir de la izquierda) que coincide con el de la cadena Y$. Cuando se especifica N, empieza a contar desde el caracter enésimo de X$.

Ejemplo de ejecución

```
PRINT INSTR(3,("BASIC MSX2","SIC")
 3
```

- Cuando el valor de N es superior a la longitud de X$, o X$ es una cadena nula, o no encuentra Y$, ofrece el valor 0.

Función INT (integer)

Ofrece el valor entero máximo menor que el dato numérico dado.

Afín: FIX, CINT

FORMATO

INT(X)

X **Cond.** Constantes, variables, variables matriciales, sus expresiones (de tipo numérico).

Valor obtenido: Tipo numérico.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT INT(3);INT(-3);INT(3.58);INT(-3.58)
 3 -3  3 -4
```

INTERVAL ON (interval on)
INTERVAL OFF (interval off)
INTERVAL STOP (interval stop)

Valida, invalida o retiene una interrupción mediante el temporizador interno.

FORMATO

INTERVAL ON — Interrupción habilitada
INTERVAL OFF — Interrupción inhabilitada
INTERVAL STOP — Interrupción retenida

FUNCION Y UTILIZACION

Se trata de una orden que habilita (INTERVAL ON), inhabilita (INTERVAL OFF) o retiene (INTERVAL STOP) una interrupción tras la declaración de una interrupción con el temporizador interno, utilizando ON INTERVAL GOSUB.

KEY (key)

Asigna una cadena de caracteres a una tecla de función.

Afín: KEY LIST, KEY ON, KEY OFF

FORMATO

KEY número de tecla de función, cadena de caracteres

Número de tecla de función	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (de tipo numérico).
Cadena de caracteres	**Cond.**	Cadena de 15 caracteres o menos.

FUNCION Y UTILIZACION

Cuando está definida una cadena de caracteres para una tecla de función, con solo pulsar esa tecla queda introducida la cadena de caracteres.

- Los números 1 al 5 corresponden a las teclas F1—F5, mientras que los números 6 al 10 corresponden a la pulsación de cada una de las teclas de función manteniendo pulsada al mismo tiempo la tecla SHIFT
- Las definiciones de las teclas de función quedan borradas e inicializadas al pulsar el botón RESET o al desconectar la alimentación.
- Mediante la función CHR$ se puede definir un código distinto al correspondiente a un caracter (por ejemplo, el código return).

Ejemplo de ejecución

```
KEY 1,"JAPAN"                    ——— Define "JAPON" para F1
KEY 2,"CLS"+CHR$(13)             ——— Define CLS + RETURN para F2
```

KEY LIST (key list)

Visualiza el contenido de las teclas de función.

Afín: KEY, KEY ON, KEY OFF

FORMATO

KEY LIST

FUNCION Y UTILIZACION

La ejecución de esta orden produce la presentación en pantalla de la cadena de caracteres definida para cada una de las teclas de función.

Ejemplo de ejecución

```
KEY LIST
color ---------- F1
auto ----------- F2
goto ----------- F3
list ----------- F4
run ------------ F5
color 15,4,4 --- SHIFT + F1
cload" --------- SHIFT + F2
cont ----------- SHIFT + F3
list. ---------- SHIFT + F4
 run ----------- SHIFT + F5
```

Este es el listado del estado inicial. En esta lista podemos ver que la tecla de función 6 tiene definida la cadena "color 15,4,4". (La tecla de función 6 corresponde a la pulsación simultánea de la tecla F1 y la tecla SHIFT).

KEY ON, KEY OFF (key on, key off)

Visualiza o borra en/de la pantalla la cadena asignada a las teclas de función.

Related: KEY, KEY LIST

FORMATO

KEY ON o KEY OFF

FUNCION Y UTILIZACION

Inicialmente, se visualizan en la última línea de la pantalla las cadenas de caracteres definidas para cada una de las teclas de función. Para borrarlo, ejecutar KEY OFF.

- Se pueden visualizar caracteres en esa última línea de la pantalla mediante una instrucción PRINT, tras ejecutar KEY OFF para borrar esa línea.
- Ejecutar KEY ON para visualizar de nuevo el contenido de las teclas de función.

KEY (n) ON (key (n) on)
KEY (n) OFF (key (n) off)
KEY (n) STOP (key (n) stop)

Habilita, inhabilita o retiene una interrupción de tecla de función.

FORMATO Y FUNCION

KEY (número de tecla de función) ON — Habilita interrupción.
KEY (número de tecla de función) OFF — Inhabilita interrupción.
KEY (número de tecla de función) STOP — Retiene interrupción.

Número de tecla de función **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); 1 ≤número <11.

FUNCION Y UTILIZACION

Especifica una tecla de función, mediante su número de tecla de función, a efectos de interrupción.

```
KEY(1) ON ——— Habilita una interrupción de la tecla F1
KEY(2) OFF ——— Inhabilita una interrupción de la tecla F2
KEY(3) STOP ——— Retiene una interrupción de la tecla F3
```

KILL (kill)

Borra un fichero del disco.

FORMATO

KILL " [nombre de unidad] nombre de fichero [.extensión]"

Nombre de unidad	**Cond.**	A:, B:, C:, D:, E:, F:, G:, H:
	Omit	Unidad de disco actual.
Nombre de fichero	**Cond.**	Una cadena de 8 caracteres o menos.
Extensión	**Cond.**	Una cadena de 3 caracteres o menos.

FUNCION Y UTILIZACION

Borra un fichero del disco especificado.
El signo de interrogación (?) puede sustituir a uno o más caracteres del nombre del fichero o de la extensión. El asterisco (*) puede sustituir a todo el nombre del fichero o de la extensión.

Ejemplo de ejecución

```
KILL "A:TEST.DAT"
```

Esta instrucción KILL buscará en el disco de la unidad A el fichero de nombre TEST.DAT y, si lo encuentra, lo borrará.

```
KILL "A:TEST??.DAT"
```

Esta instrucción borrará todos los ficheros cuyo nombre empieza por las letras TEST seguidas por dos caracteres cualesquiera (incluyendo el carácter nulo), y cuya extensión es DAT.

```
KILL "A:TEST.*"
```

Borra del disco todos los ficheros con el nombre de fichero TEST, sea cual sea su extensión.

```
KILL "A:*.*"
```

Borra todos los ficheros del disco de la unidad A.

En todos los casos, si se omite la unidad de discos, la instrucción actua sobre la unidad actualmente en activo.

Función LEFT$ (left dollar)

Ofrece una cadena de caracteres compuesta por un número especificado de caracteres extraidos de otra cadena, de izquierda a derecha.

Afín: RIGHT$, MID$

FORMATO

LEFT$ (X$, N)

X$	Cond.	Constantes, variables, variables de matriz, sus expresiones (de tipo cadena).
N	Cond.	Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $0 \leqslant N < 256$.

Valor obtenido: Tipo alfanumérico.

FUNCION Y UTILIZACION

```
PRINT LEFT$("MSX-BASIC",3)
MSX
OK

PRINT LEFT$("MSX-BASIC",3.8)
MSX
OK

PRINT LEFT$("MSX-BASIC",0)
OK
```

Si N no es un entero, ignora los números que siguen a la coma decimal.

Si N es 0, da una cadena nula.

Función LEN (length)

Ofrece, en forma de datos numéricos, el número de caracteres (longitud) de una cadena de caracteres.

FORMATO

LEN(X$)

X$ **Cond.** Constantes, variables, variables de matriz, sus expresiones (del tipo cadena).

Valor Obtenido: Tipo entero.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT LEN ("BASIC")
5
PRINT LEN ("BASIC MSX2")
10
```

Cuando una cadena de caracteres incluye un espacio, el espacio se cuenta como un caracter.

- Cuando una cadena de caracteres incluye la función CHR$, esta se cuenta como un caracter.

LET (let)

Asigna datos a una variable.

FORMATO

[LET] variable = X

Variable **Cond.** Variables y variables de matriz de tipo numérico y de tipo de cadena.

X **Cond.** Variables, variables de matriz, sus expresiones, de tipo numérico y tipo cadena.

FUNCION Y UTILIZACION

Asigna el valor del término de la derecha al término de la izquierda.

- Las constantes de cadena van encerradas entre dobles comillas (" ").
- Se puede omitir LET
- Cuando se asigne un determinado tipo de dato numérico a una variable numérica de otro tipo distinto, convierte el dato numérico del primer tipo en el dato numérico correspondiente al tipo de la variable.

Ejemplo de ejecución

```
LET N=N+1
A%=45.6:PRINT A%
 45
A$=3+4
Type mismatch
```

Aumenta el valor de N una unidad

Como se asigna un dato numérico a una variable de cadena surge un error y se visualiza en pantalla el mensaje de error: "Type mismatch".

LINE (line)

En el modo gráfico, dibuja una línea recta o un rectángulo.

FORMATO

LINE [[STEP] (coordenadas del punto inicial)]–[STEP] (coordenada del punto final), [color] {[,B] / [,BF]} [, operación lógica]

Coordenadas del punto inicial (X,Y)

Cond. Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $-32768 \leqslant$ coordenadas < 32767

Omit La última posición especificada en la última instrucción gráfica.

Coordenadas del punto final (X,Y)

Cond. Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); $-32768 \leqslant$ coordenadas < 32767

Color

Cond. SCREEN 2 a 7:
Constantes, variables, variables matriciales, sus expresiones (de tipo numérico); $0 \leqslant$ color < 16.
SCREEN 8:
Constantes, variables, variables matriciales, sus expresiones (de tipo numérico); $0 \leqslant$ color < 256.

Omit Color actual del primer plano.

B, BF

Omit Dibuja una línea recta.

Operación lógica

Cond. SCREEN 5 a 8:
PSET, PRESET, XOR, OR, AND.

FUNCION Y UTILIZACION

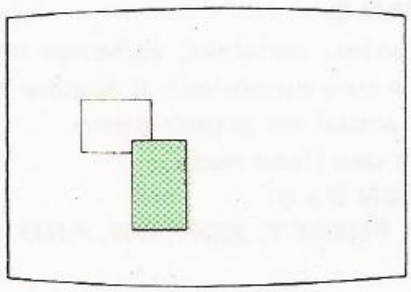
Dibuja una línea recta entre las coordenadas del punto inicial y las coordenadas del punto final (cuando se omiten B, BF).

- Cuando se especifica "B", dibuja un rectángulo cuya diagonal es una línea recta entre los dos puntos especificados.
- Cuando se especifica "BF", dibuja un rectángulo cuya diagonal es una línea recta entre los dos puntos especificados, y colorea el rectángulo.

Cuando se especifica STEP, pasa X,Y a un nuevo sistema de coordenadas cuyo origen está en el último punto especificado en la instrucción gráfica inmediatamente anterior.

Ejemplo de ejecución

```
10 CLS
20 SCREEN 5
30 LINE (60,60)-(100,100),1,B,AND
40 LINE STEP(-10,-10)-(120,160),8,BF
50 GOTO 50
```



LINE INPUT (line input)

Da entrada desde el teclado a una cadena de hasta 254 caracteres, en forma de variable de cadena.

Afín: INPUT, LINE INPUT #

FORMATO

LINE INPUT ["sentencia de comentario para la introducción de datos";] variable

"Sentencia de comentario"

Cond. Constantes de cadena

Omit Presenta en pantalla solo el signo de interrogación "?", sin sentencia de petición de entrada.

Variable Cond. Variables, variables de matriz, (de cadena).

FUNCION Y UTILIZACION

Considera los códigos de return como separación del dato, y asigna una cadena de caracteres introducida por el teclado a una variable. Cuando en una cadena de caracteres se incluye una coma, la asigna como un caracter más de la cadena.

Ejemplo de ejecución .

```
10 CLS
20 LINE INPUT "NOMBRE, TELEFONO?"; N$
30 PRINT N$
RUN
NOMBRE, TELEFONO? JAVIER, 27-27-07
JAVIER, 27-27-07
```

LINE INPUT # (line input number)

Lee de un fichero una cadena de hasta 254 caracteres y la asigna a una variable de cadena.

Afín: INPUT, LINE INPUT #

FORMATO

LINE INPUT # número de fichero, variable

Número de fichero	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (de tipo entero); 1 ⩽ número de fichero ⩽ número especificado en la instrucción MAXFILES
Variables	**Cond.**	Variables y variables de matriz alfanuméricas.

FUNCION Y UTILIZACION

Lee datos de cadena de un fichero, pero no considera los códigos de la coma, el espacio y el cambio de línea como signos de separación de los datos, lo cual la diferencia de la instrucción INPUT #: asigna las cadenas de caracteres que incluyen tales elementos a una variable, en forma de datos alfanuméricos. Solo considera signo de separación de los datos, al código de return.

Ejemplo de ejecución

```
10 OPEN "CAS:DATA" FOR INPUT AS #1
20 IF EOF(1) THEN GOTO 60
30 LINE INPUT #1,A$
40 PRINT A$
50 GOTO 20
60 CLOSE #1:END
```

Cuando se haya preparado un archivo mediante el procedimiento siguiente con un nombre de fichero denominado DATA,

```
PRINT #1,"ABC";",";"DEF"
PRINT #1,"GHI JKL";
PRINT #1,"MNO"
PRINT #1,"PQR"
```

Cuando el programa anterior lea estos datos y los visualice en la pantalla, observará que estos fueron leídos como datos de 3 caracteres, de la forma siguiente:

```
ABC,DEF
 GHI JKLMNO
 PQR
```

LIST (list out)

Visualiza por pantalla el listado del programa almacenado en memoria.

Afín: LLIST

FORMATO

LIST [número de línea inicial] [—] [número de línea final]

Número de línea inicial Cond. Constantes enteras, 0 ≤número <65529.
Omit Número de línea más bajo.

Número de línea final Cond. Constantes enteras, 0 ≤número <65529.
Omit Número de línea más alto.

FUNCION Y UTILIZACION

Pulsar STOP para detener temporalmente la visualización del listado en la pantalla, y volver a pulsar STOP para reanudarla. Para suspenderla, pulsar CTRL + STOP .

Ejemplo de ejecución

```
LIST ———————— Visualiza todas las líneas

LIST 40 ———————— Visualiza la línea 40

LIST 20-40 ———— Visualiza las líneas 20 a 40

LIST -50 ———————— Visualiza todas las líneas desde la primera hasta la 50

LIST 30- ———————— Visualiza todas las líneas desde la 30 hasta la última

LIST. ———————————— Visualiza la última línea presentada por una
                    instrucción LIST o una línea cuya ejecución
                    haya estado interrumpida por un error
```

LLIST (line printer list out)

Imprime con una impresora el listado del programa almacenado en memoria.

Afín: LIST

FORMATO

LLIST [número de línea inicial] [—] [número de línea final]

Número de línea inicial Cond. Constantes enteras, 0 ≤número ≤65529.
Omit Número de línea más abajo.

Número de línea final Cond. Constantes enteras, 0 ≤número ≤65529.
Omit Número de línea más alto.

FUNCION Y UTILIZACION

Las especificaciones son idénticas a las de la instrucción LIST. La instrucción LLIST no presenta el listado del programa en pantalla.

- Si se ejecuta una instrucción LLIST cuando la impresora no está conectada o cuando, estándolo, no funciona, el ordenador se para y no acepta entradas por teclado. En tal caso pulse al mismo tiempo las teclas CTRL y STOP para volver al estado de entrada de comandos.

LOAD (load)

Carga en memoria un programa BASIC procedente del dispositivo especificado.

Opuesta: SAVE Afín: CLOAD, MERGE

FORMATO

LOAD " [nombre de dispositivo] [nombre de fichero [.extensión]]" [,R]

Nombre de dispositivo	Cond.	CAS: A:, B:, C:, D:, E:, F:, G:, H: MEM:
	Omit	CAS: Unidad de disco actual.
Nombre de fichero	Cond.	Una cadena de 6 o menos caracteres. Una cadena de 8 o menos caracteres.
	Omit	Carga el primer fichero que encuentra (no se puede omitir el nombre de fichero cuando se trabaja con un disco flexible o con la RAM-DISK)
Extensión	Cond.	Una cadena de 3 o menos caracteres.
	Omit	Cadena nula.
Opción R	Omit	Solo carga.

FUNCION Y UTILIZACION

Cuando se especifica CAS: en el nombre del dispositivo, carga los programas almacenados en la cinta cassette en formato ASCII mediante la instrucción SAVE"CAS: nombre de fichero".

- Cuando se especifica la opción R, ejecuta el programa tras su carga.

Ejemplo de ejecución

```
LOAD "CAS:PROG2"
LOAD "A:PROG.ASC"
```

Función LOC (location)

Ofrece la situación actual en el fichero.

FORMATO

LOC (número de fichero)

Número de fichero	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); 1 ⩽número de fichero ⩽número especificado en la instrucción MAXFILES

Valor obtenido: Tipo entero.

FUNCION Y UTILIZACION

Cuando se aplica esta función a un fichero secuencial, ofrece el número de los registros leídos o escritos. La longitud de un registro es 256 bytes.

Cuando se aplica esta función a un fichero de acceso aleatorio, ofrece el número del último registro leído o escrito.

LOCATE (locate)

Desplaza el cursor a la posición especificada, en el modo texto.

FORMATO

LOCATE [coordenada X], [coordenada Y], [interruptor del cursor]

Coordenada X, coordenada Y

Cond. Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); 0 ⩽coordenada <256.

Omit 0

Interruptor del cursor

Cond. 0 o 1

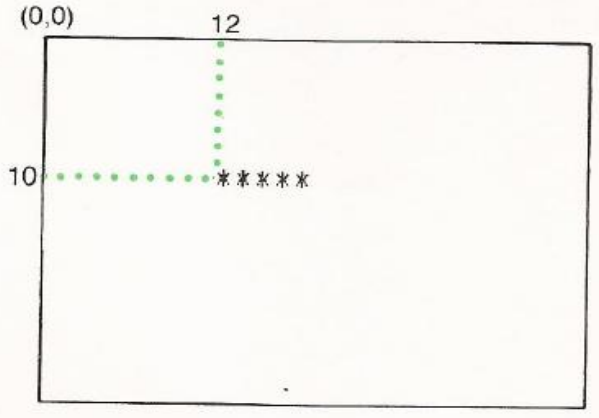
Omit 1

FUNCION Y UTILIZACION

0 ... Cursor no presente
1 ... Cursor presente

Ejemplo de ejecución

```
10 CLS
20 LOCATE 12,10
30 PRINT "*****"
```



Función LOF (length of file)

Ofrece el tamaño del fichero especificado.

FORMATO

LOF (número de fichero)

Número de fichero Cond. Constantes, variables, variables matriciales, sus expresiones (de tipo numérico); 1 ≤número de fichero ≤número especificado en la instrucción MAXFILES

Valor obtenido: Entero.

FUNCION Y UTILIZACION

Ofrece en bytes, el tamaño del fichero especificado.

Función LOG (natural logarithm)

Ofrece el valor de un logaritmo natural (Log.)

FORMATO

LOG (X)

X **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico, mayores que 0).
Valor obtenido: Tipo numérico.

FUNCION Y UTILIZACION

La función LOG da el valor de un logaritmo natural; su base es :
e (2,7182818284588).

- Mediante LOG (b)/LOG(a) se puede obtener el valor del logaritmo de b en base a ($Log_a b$). Si $b > 0$ y $a \neq 1$.

Ejemplo de ejecución

```
PRINT LOG(10)
 2.302585092994
```

Función LPOS (line printer position)

Ofrece la posición del cabezal de la impresora, procedente del buffer de la impresora.

FORMATO

LPOS(X)

X **Cond.** Un número cualquiera (argumento ficticio).
Valor obtenido: Tipo entero.

FUNCION Y UTILIZACION

Ofrece la posición en el buffer (memoria imtermedia de la impresora) del caracter que esté escribiendo actualmente (Principio = 0).

LPRINT (line printer)

Escribe por impresora el valor de una expresión.

Afín: PRINT, LLIST

FORMATO

LPRINT [expresión] [separador] [expresión] [separador] [expresión]

Expresión **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas y de cadena).
Omit Avance de línea.
Separador **Cond.** "," o ";"

FUNCION Y UTILIZACION

La instrucción LPRINT envía datos a la impresora mientras que la instrucción PRINT los envía a la pantalla. Para más detalles, ver PRINT.

LPRINT USING (line print using)

Escribe datos por impresora con un formato especificado.

Afín: LPRINT, PRINT USING

FORMATO

LPRINT USING símbolo del formato; expresión [separador] [expresión] [separador] ...

Expresión **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico y de cadena).

FUNCION Y UTILIZACION

La instrucción LPRINT USING envía datos a una impresora con un formato especificado mientras que la instrucción PRINT USING envía datos a la pantalla también con un formato especificado. Para más detalles, por ejemplo los símbolos de los formatos, ver PRINT USING.

LSET (left set)

Escribe los datos en un registro de un fichero de acceso aleatorio con justificación a la izquierda.

Afín: RSET

FORMATO

LSET variable de cadena = expresión de cadena

Variable de cadena	**Cond.**	Variable de cadena
Expresión de cadena	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (de cadena).

FUNCION Y UTILIZACION

Dispone los datos de cada variable en el registro especificado por la instrucción FIELD como preparación para la escritura de los datos en un fichero de acceso aleatorio con una instrucción PUT. La instrucción LSET alinea los datos desde el lado izquierdo del registro; si el dato es más corto que la variable especificada por la instrucción FIELD, los espacios vacíos se llenan con espacios en blanco, y si el dato es más largo que la longitud especificada por la instrucción FIELD, se ignora la porción derecha sobrante de la cadena.

- En el proceso de organización del dato numérico, se convierte en primer lugar el dato numérico en dato de cadena mediante las funciones MKI$, MKS$ y MKD$.

```
LSET A$=X$_________ coloca el dato de cadena X$ de la
                    variable A$ en el registro
LSET B$=MKS$(N)____ convierte el dato numérico N en dato de cadena
                    y lo coloca en la variable B$ del registro
```

MAXFILES (maxfiles)

Declara el número de ficheros que pueden abrirse simultáneamente en un programa.

Afín: OPEN

FORMATO

MAXFILES = expresión

Expresión **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo numérico); 0 ≤ expresión < 16.

FUNCION Y UTILIZACION

Declara el número de ficheros que pueden abrirse simultáneamente en un programa; abiertos simultáneamente de archivos significa que se abre un fichero y después otro sin haber cerrado el primero.

Ejemplo de ejecución

```
10 MAXFILES=3
20 OPEN "GRP:" FOR OUTPUT AS #1
30 OPEN "CRT:" FOR OUTPUT AS #2
40 OPEN "LPT:" FOR OUTPUT AS #3
          .
          .
          .

1000 CLOSE
```

Dado que la línea 10 establece que 3 es el máximo número de ficheros que pueden abrirse simultáneamente, en la línea 20 y siguientes se pueden abrir 3 ficheros.

En ausencia de una instrucción MAXFILES que especifique el número de ficheros que pueden abrirse simultáneamente, solo puede estar abierto un fichero cada vez.

- La declaración de un número innecesariamente alto de ficheros en la instrucción MAXFILES reduce el área de memoria a disposición del usuario.

MERGE (merge)

Carga un programa almacenado en formato ASCII y lo fusiona con el programa almacenado en la memoria del ordenador.

Afín: LOAD

FORMATO

MERGE " [nombre de dispositivo] [nombre de fichero [.extensión]]"

Nombre de dispositivo
- **Cond.** CAS:
  - A:, B:, C:, D:, E:, F:, G:, H:
  - MEM:
- **Omit** CAS:
  - Unidad de discos actual.

Nombre de fichero
- **Cond.** Una cadena de 6 o menos caracteres.
  - Una cadena de 8 caracteres o menos.
- **Omit** Funde el primer fichero que encuentra. (No se puede omitir el nombre del fichero cuando se opera con disco flexible o con RAM-DISK.

Extensión
- **Cond.** Una cadena de 3 caracteres o menos.
- **Omit** Una cadena nula.

FUNCION Y UTILIZACION

Carga un programa almacenado por una instrucción SAVE en el formato ASCII. El programa que hay en la memoria sigue ahí y la instrucción combina con él el programa cargado.

- Si el programa cargado por la instrucción MERGE tiene números de línea que coinciden con otros del programa almacenado en memoria, prevalecen los números de línea cargados por la instrucción MERGE.

Ejemplo de ejecución

```
MERGE "CAS:PROG3"
MERGE "A:PROG.ASC"
MERGE"MEM:PROG. ASC"
```

Función **MID$** (middle dollar)

Localiza y ofrece una parte de los datos de una cadena de caracteres.

Afín: LEFT$, RIGHT$

FORMATO

MID$ (X$, M [,N])

X$	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (de cadena).
M	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (numéricas); 1 ≤ M < 256.
N	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (numéricas); 1 ≤ N < 256.
	Omit	Ofrece todos los caracteres que siguen al caracter M.ésimo.

Valor obtenido: Cadena de caracteres.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT MID$("JAPANUKFRANCE",6,2)
UK
PRINT MID$("JAPANUKFRANCE",6,2.6)
UK
PRINT MID$("JAPANUK",6,4)
UK
PRINT MID$("JAPANUK",12,5)

PRINT MID$("JAPANUK",6,0)
```

Si N no es un valor entero, ignora las cifras que siguen a la coma decimal.

Si tras el caracter M.ésimo no hay un número N de caracteres, da todos los caracteres que hay a continuación del emésimo

Cuando el valor de M es mayor que la longitud de X$ o cuando N es 0, da una cadena nula.

MID$ = Y$ (middle dollar)

Sustituye una parte de una cadena de caracteres por otra cadena.

Afín: MID$

FORMATO

MID$(X$,M[,N]) = Y$

X$, Y$	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (de cadena).
M	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (numéricas); 1 ⩽M <256.
N	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (numéricas); 1 ⩽N <256.
	Omit	Sustituye los caracteres M.ésimo y siguientes de X$ por Y$.

FUNCION Y UTILIZACION

Sustituye los caracteres M.ésimo y siguientes de la cadena de caracteres X$, de izquierda a derecha, por los primeros N caracteres de Y$. De todas formas, esta instrucción no cambia la longitud de X$.

Ejemplo de ejecución

```
10 X$="ABCDEFG"
20 Y$="QRSTUVWXYZ"
30 MID$(X$,4,2)=Y$
40 PRINT X$
RUN
ABCQRFG
```

Función **MKI$** (make integer dollar)
MKS$ (make single precision dollar)
MKD$ (make double precision dollar)

Convierten un dato numérico en una cadena de caracteres en función de su formato interno.

FORMATO

MKI$ (X)

X **Cond.** Constantes, variables, variables de matriz, sus expresiones (de tipo entero).

Valor obtenido: Cadena de 2 caracteres (2 bytes)

MKS$ (Y)

Y **Cond.** Constantes, variables, variables de matriz, sus expresiones (de precisión simple).

Valor obtenido: Cadena de 4 caracteres (4 bytes)

MKD$ (Z)

Z **Cond.** Constantes, variables, variables de matriz, sus expresiones (de precisión doble).

Valor obtenido: Cadena de 8 caracteres (8 bytes)

FUNCION Y UTILIZACION

Como en un fichero de acceso aleatorio solo se pueden escribir datos de cadena, hay que convertir primero los datos numéricos en datos de cadena haciendo uso de las funciones MKI$, MKS$ y MKD$. Estas conversiones se hacen cuando se asignan los datos a los lugares especificados del registro mediante las instrucciones LSET/RSET.

La cantidad de espacio del registro que ocupa un dato numérico depende de su tipo. Los datos de tipo entero ocupan 2 bytes, los datos de coma flotante de precisión simple ocupan 4 bytes, y los datos de coma flotante de precisión doble ocupan 8 bytes.

Cuando los datos numéricos son de tipo entero, como en

```
LSET A$=MKI$(X)
```

(justificación a la izquierda)

se convierte el dato numérico entero en una cadena de caracteres (dos caracteres = dos bytes) mediante una función MKI$, y se asigna al registro. En consecuencia, es preciso haber dejado reservado, mediante una instrucción FIELD, un espacio de 2 bytes o más para alojar a la cadena A$ resultado de la conversión.

De una forma similar, la función MKS$ convierte los datos de precisión simple en cadenas de 4 caracteres (4 bytes), y la función MKD$ convierte los datos de precisión doble en cadenas de 8 caracteres (8 bytes).

A la hora de leer los datos del fichero, las cadenas de caracteres que antes eran datos numéricos son convertidas de nuevo en cadenas de caracteres mediante las funciones CVI, CVS y CVD.

MOTOR (motor)

Pone en marcha y para el motor del grabador/reproductor de cassettes.

FORMATO

MOTOR [{ ON / OFF }]

FUNCION Y UTILIZACION

Conecte el terminal TAPE (cinta) del ordenador al terminal de control remoto de un grabador/reproductor de cassette y ponga el aparato en el modo reproducción o grabación. MOTOR ON pone la cinta en funcionamiento y MOTOR OFF la para.
Cuando solo se ejecuta MOTOR, si estaba en ON lo pone OFF y si estaba en OFF lo pone ON.

NAME (name)

Cambia el nombre de un fichero del disco.

FORMATO

NAME "[nombre de unidad] nombre anterior de fichero. [.extensión anterior]" AS "nuevo nombre de fichero [.nuevo nombre de la extensión]"

Nombre de la unidad
Cond. A:, B:, C:, D:, E:, F:, G:, H:
Omit Unidad de discos actual.

Nombre anterior de fichero, nombre nuevo de fichero
Cond. Una cadena de 8 caracteres o menos.

Nombre anterior de la extensión, nombre nuevo de la extensión
Cond. Una cadena de 3 caracteres o menos.
Omit Una cadena nula.

FUNCION Y UTILIZACION

Sirve para cambiar el nombre del fichero, pero no su contenido.

Ejemplo de ejecución

```
NAME "A:OLD.BAS" AS "NEW.BAS"
```

Esta orden cambia el nombre del fichero OLD.BAS del disco de la unidad A por NEW.BAS.

- El nombre anterior de fichero [.nombre anterior de la extensión] deben existir en ese disco. El nuevo nombre de fichero [.nuevo nombre de la extensión] no deben coincidir con el nombre de un fichero existente en ese disco.

NEW (new)

Borra un programa BASIC de la memoria y anula todas las variables.

FORMATO

NEW

FUNCION Y UTILIZACION

NEW se ejecuta antes de dar entrada en memoria a un nuevo programa, con objeto de borrar todos los programas existentes y pasar al estado de entrada de comandos.

- Cuando hay en memoria un programa en lenguaje máquina, este se conserva aunque se ejecute NEW.

Función OCT$ (octonary dollar)

Ofrece una expresión octal de un dato numérico en forma de dato alfanumérico.

Afín: BIN$, HEX$

FORMATO

OCT$(X)

X — **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas); $-32768 \leqslant X < 65536$. Si se trata de un número negativo, su valor es el resultado de la suma de 65536.

Valor obtenido: Una cadena de caracteres.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT OCT$(100)
144

PRINT OCT$(65536-32768)
100000
```

ON ERROR GOTO (on error go to)

Transfiere la ejecución del programa al número de línea especificado cuando ocurre un error.

FORMATO

ON ERROR GOTO número de línea

Número de línea **Cond.** Constantes enteras, 0 ⩽número ⩽65529.

FUNCION Y UTILIZACION

Sirve para prevenir la interrupción de la ejecución del programa causada por un error durante la ejecución. Cuando surge un error después de la declaración de ON ERROR GOTO, la ejecución se transferirá al número de línea especificado. (Cuando el error sea resultado de una orden directa, el control de la ejecución del programa también se transferirá al número de línea especificado).

Ejemplo de ejecución

```
10 ON ERROR GOTO 100
20 INPUT A
30 B=SQR(A)
40 PRINT "SQR(A)=";B
50 END
100 IF ERR=5 AND ERL=30 THEN PRINT
"Input a positive number."
110 RESUME 20
```

Instrucción END que distingue una rutina principal de la subrutina de procesamiento de errores.

Rutina de proceso de errores.

Invalidación de una instrucción ON ERROR GOTO

Ejecutar ON ERROR GOTO 0.

ON—GOSUB (on—go to subroutine)

Bifurca la ejecución de un programa a las subrutinas que comienzan por los números de línea especificados, dependiendo del valor de la expresión.

FORMATO

ON expresión GOSUB número de línea [,número de línea] ...

Expresión **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ⩽expresión <256.

Número de línea **Cond.** Constantes enteras, 0 ⩽número ⩽65529.

FUNCION Y UTILIZACION

```
100 ON X GOSUB 500,600,700
```

En este programa, si el valor de X es 1, la ejecución salta a la subrutina que empieza en la línea 500, si el valor de X es 2, la ejecución salta a la subrutina que empieza en la línea 600, y si es 3 salta a la subrutina que comienza en la línea 700.
La instrucción RETURN realiza el retorno del control de ejecución al programa principal.

Valor de la expresión y resultado de la ejecución
Cuando el valor de la expresión no es un entero ... ignora las cifras que siguen al punto decimal.
Cuando el valor de la expresión es 0 o mayor que el número de línea especificado por GOSUB ... transfiere la ejecución a la instrucción siguiente a la ON—GOSUB
Cuando el valor de la expresión es negativo o mayor que 255 ... se produce un error.

ON—GOTO (on—go to)

Bifurca la ejecución del programa a números de línea que dependen del valor de una expresión.

FORMATO

ON expresión GOTO número de línea [,número de línea] ...

Expresión	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (numéricas).
Número de línea	**Cond.**	Constantes enteras, 0 ⩽número ⩽65529.

FUNCION Y UTILIZACION

```
100 ON X GOTO 120,130,180
```

En este programa, si el valor de X es 1, la ejecución salta a la línea 120, si es 2 salta a la línea 130, y si es 3 a la línea 180.

Valor de la expresión y resultado de la ejecución
Cuando el valor de la expresión no es un entero ... ignora las cifras que siguen al punto decimal.
Cuando el valor de la expresión es 0 o mayor que los número de línea especificados por GOTO ... transfiere la ejecución a la instrucción siguiente a ON—GOTO.
Cuando el valor de la expresión es negativo o mayor que 255 ... se produce un error.

ON INTERVAL GOSUB

(on interval go to subroutine)

Declara la subrutina a la que salta la ejecución del programa en caso de interrupción debida al temporizador interno.

FORMATO

ON INTERVAL = tiempo de intervalo GOSUB número de línea

Tiempo de intervalo	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (numéricas); $-32768 \leqslant$ tiempo $\leqslant 65529$, distinto a 0.
Número de línea	**Cond.**	Constantes enteras, $0 \leqslant$ número $\leqslant 65529$.

FUNCION Y UTILIZACION

Se trata de una instrucción que dá el número de línea al que debe saltar la ejecución del programa en caso de una interrupción, causada por el temporizador interno, con el intervalo de tiempo especificado. Cada intervalo de la interrupción es aproximadamente: intervalo de tiempo x 1/50 segundo; en otras palabras, cuando se ha especificado un intervalo de 50, se produce una interrupción aproximadamente cada segundo.

Ejemplo de ejecución

```
10 ON INTERVAL=50 GOSUB 100
20 INTERVAL ON
30 SCREEN 2,1
40 SPRITE$(1)=CHR$(&H18)+CHR$(&H3C)+CHR$
(&H66)+CHR$(&HDB)+CHR$(&HE7)+CHR$(&H7E)+
CHR$(&H24)+CHR$(&H42)
50 GOTO 50
100 X=INT (RND(1)*256):Y=INT (RND(1)*192)
110 C=INT (RND(1)*14)+2
120 PUT SPRITE 1,(X,Y),C,1
130 RETURN 50
```

En este programa hay una interrupción cada segundo, consecuencia de las líneas 10 y 20, y cada vez que hay una interrupción, la ejecución salta a la subrutina que empieza en la línea 100. Después de que esta subrutina presente en pantalla la configuración de sprites en forma de OVNI, la línea 130 (RETURN 50) devuelve la ejecución a la línea 50.

- Cuando el intervalo de tiempo especificado es un número negativo, el valor seleccionado es el resultado de sumar a ese número 65536.

ON KEY GOSUB (on key go to subroutine)

Declara la subrutina a la que se bifurcará el programa cuando se provoque una interrupción mediante una tecla de función.

FORMATO

ON KEY GOSUB número de línea [,número de línea] ...

Número de línea **Cond.** Constantes enteras, 0 ⩽número ⩽65529.

FUNCION Y UTILIZACION

Se trata de una instrucción que declara el número de línea de comienzo de una subrutina a la que salta el programa en caso de provocar una interrupción mediante una tecla de función. Tras GOSUB se pueden especificar hasta 10 números de línea, separados por comas, que corresponderían secuencialmente a las teclas de función [F1], [F2], ... etc.

Ejemplo de ejecución

```
10 ON KEY GOSUB 1000,,2000
20 KEY(1) ON:KEY(3) ON
```

Cuando se pulsa [F1] la ejecución del programa salta a la subrutina que comienza en la línea 1000, y cuando se pulsa [F3] salta a la subrutina que comienza en la línea 2000 (líneas 10 y 20).
La vuelta de la subrutina al programa principal se realiza mediante la instrucción RETURN.

ON SPRITE GOSUB

(on sprite go to subroutine)

Define la subrutina a la que se bifurcará el programa cuando se produzca una interrupción por choque de dos sprites.

FORMATO

ON SPRITE GOSUB número de línea

Número de línea **Cond.** Constantes enteras, 0 ⩽número ⩽65529.

FUNCION Y UTILIZACION

Esta instrucción define el número de línea de comienzo de una subrutina a la que salta el programa en caso de una interrupción debida al choque de dos sprites.

Ejemplo de ejecución

```
10 ON SPRITE GOSUB 1000
20 SPRITE ON
```

Cuando se produce el choque de dos sprites, las líneas anteriores transfieren la ejecución a una subrutina que comienza en la línea 1000. El retorno de la subrutina se especifica mediante una instrucción RETURN.

ON STOP GOSUB (on stop go to subroutine)

Define la subrutina a la que se bifurcará el programa cuando se produce una interrupción provocada por la pulsación de CTRL + STOP

FORMATO

ON STOP GOSUB número de línea

Número de línea **Cond.** Constantes enteras, 0 ⩽número ⩽65529.

FUNCION Y UTILIZACION

Esta instrucción define el número de línea de comienzo de una subrutina a la que salta el programa en caso de una interrupción producida por la pulsación de las teclas CTRL + STOP .

Ejemplo de ejecución

```
10 ON STOP GOSUB 1000
20 STOP ON
```

Las dos líneas anteriores transfieren la ejecución del programa a la subrutina que comienza en la línea 1000, en caso de pulsación simultánea de las teclas CTRL y STOP . El retorno de la subrutina se realiza mediante una instrucción RETURN.

Precauciones

Al ejecutar una subrutina es necesario terminar de alguna forma el programa. La única forma de poner fin al programa siguiente es pulsando la tecla RESET .

```
10 ON STOP GOSUB 100
20 STOP ON
30 PRINT "MAIN ROUTINE"
40 GOTO 40
100 PRINT "CTRL+STOP EXECUTED"
110 RETURN 30
```

ON STRIG GOSUB

(on stick trigger go to subroutine)

Define la subrutina a la que salta el programa en caso de una interrupción causada por la pulsación de la barra espaciadora, del botón de disparo del mando de juegos o del pulsador correspondiente de otros periféricos (ratón, bola gráfica, etc.)

FORMATO

ON STRIG GOSUB número de línea [,número de línea] ...

Número de línea **Cond.** Constantes enteras, 0 ⩽número ⩽65529.

FUNCION Y UTILIZACION

Esta instrucción define el número de línea de comienzo de una subrutina a la que se bifurca el programa en caso de una interrupción debida a la pulsación de la barra espaciadora o del pulsador correspondiente de un periférico conectado en los ports del Joystick. Pueden especificarse hasta cinco números de línea separados por coma para determinar el elemento causante de la interrupción.

ON STRIG GOSUB No. línea ❶, , No. línea ❷, , No. línea ❸, No. línea ❹, No. línea ❺

No. línea 1 ❶ ... Bifurcación ante la pulsación de la barra espaciadora.
No. línea 2 ❷ ... Periférico A, pulsador 1. (conector Joystick A)
No. línea 3 ❸ ... Periférico B, pulsador 1. (conector Joystick B)
No. línea 4 ❹ ... Periférico A, pulsador 2. (conector Joystick A)
No. línea 5 ❺ ... Periférico B, pulsador 2. (conector Joystick B)

Ejemplo de ejecución

```
10 ON STRIG GOSUB 1000,2000,3000
20 STRIG(0) ON:STRIG(1) ON:STRIG(2) ON
```

Al pulsar la barra espaciadora, transfiere la ejecución del programa a la subrutina que empieza en la línea 1000, al apretar el pulsador 1 del periférico A, transfiere la ejecución a la subrutina que empieza en el número 2000; al apretar el pulsador 1 del periférico B, transfiere la ejecución a la subrutina que empieza en la línea 3000.
El retorno de la subrutina se realiza mediante una instrucción RETURN.

OPEN (open)

Abre un fichero. Opuesta: CLOSE Afín: MAX FILES

FORMATO

OPEN "[nombre de dispositivo] [nombre de fichero [.extensión]"] [FOR modo] AS [#] número de fichero [LEN = longitud del registro]

Nombre del dispositivo
- Cond. CAS:, CRT:, GRP:, LPT:,
  - 💾 A:, B:, C:, D:, E:, F:, G:, H:
  - MEM:
- Omit CAS:
  - 💾 Unidad de disco en servicio.

Nombre del fichero
- Cond. Una cadena de 6 o menos caracteres.
  - MEM💾 Una cadena de 8 o menos caracteres.
- Omit Una cadena nula (no se puede omitir el nombre del fichero cuando se opera con un disco flexible o con la RAM DISK).

MEM💾 **Extensión**
- Cond. Una cadena de 3 caracteres o menos.
- Omit Una cadena nula.

Modo
- Cond. OUTPUT, INPUT, MEM💾 APPEND.

Número de fichero
- Cond. Constantes, variables, variables de matriz y sus expresiones (numéricas); 1 ⩽número ⩽número especificado en la instrucción MAXFILES

💾 **Longitud de registro**
- Cond. Constantes, variables, variables de matriz y sus expresiones (enteras); 1 ⩽longitud ⩽ 256.
- Omit 256.

FUNCION Y UTILIZACION

La instrucción OPEN abre el fichero con el número de fichero especificado a efectos de realizar una operación de entrada/salida de datos en el dispositivo especificado. Entre los dispositivos que se pueden especificar, CRT:, GRP: y LPT: se utilizan únicamente para escribir, por lo tanto, con estos dispositivos hay que especificar el modo OUTPUT (salida). En el caso del dispositivo CAS:, es posible la escritura y la lectura, por lo tanto se puede especificar el modo OUTPUT o el modo INPUT. El modo APPEND se puede especificar también en el caso de unidades de discos (como A: o B:) y la RAM DISK.

- El número de fichero puede ser cualquier número siempre que sea menor al número especificado en la instrucción MAXFILES.

Omisión de FOR modo

La omisión de FOR modo, especifica que se abre un fichero de acceso aleatorio. Los ficheros de acceso aleatorio solo pueden ser en disco.

Especificación de la longitud del registro

En la instrucción OPEN se puede especificar la longitud máxima del registro en los ficheros de acceso aleatorio. La especificación por omisión es 256.

Ejemplo de ejecución

```
10 SCREEN 2
20 OPEN "GRP:" FOR OUTPUT AS #1
30 PSET (120,90)
40 PRINT #1,"ABC"
50 GOTO 50
```

Este programa visualiza caracteres en la pantalla en modo gráfico (SCREEN 2).

OUT (out)

Envía un dato de 1 byte al port de entrada/salida especificado.

Afín: INP

FORMATO

OUT número de port, expresión

Número de port, expresión	**Cond.**	Constantes, variables, variables de matriz y sus expresiones (numéricas); 0 ≤ número < 256.

FUNCION Y UTILIZACION

Esta instrucción envía datos directamente a una port de E/S.
Para información relativa a las asignaciones de los port E/S, ver página 264.

Función PAD (pad)

Ofrece el estado del dispositivo señalizador (touch pad, lapiz óptico, ratón y bola gráfica).

FORMATO

PAD(N)

Cond. Constantes, variables, variables de matriz y sus expresiones (numéricas); $0 \leqslant N < 20$.

Valor obtenido: Tipo numérico.

FUNCION Y UTILIZACION

	Valor de N	Significado del valor obtenido
Touch pad	0 o 4	0: sin contacto – 1: contacto
	1 o 5	Coordenada X de la posición de contacto
	2 o 6	Coordenada Y de la posición de contacto
	3 o 7	0: interruptor no pulsado – 1: interruptor pulsado
Lapiz óptico	8	0: sin contacto – 1: contacto
	9	Coordenada X de la posición de contacto
	10	Coordenada Y de la posición de contacto
	11	0: Interrup. del lápiz luminoso no pulsado – 1: Interrup. del lápiz luminoso pulsado
Ratón o bola gráfica	12 o 16	– 1: petición de entrada
	13 o 17	Coordenada X del ratón o bola gráfica
	14 o 18	Coordenada Y del ratón o bola gráfica
	15 o 19	0

Cuando N es un valor comprendido entre 0 y 3 ó entre 12 y 15, ofrece el estado del dispositivo señalizador conectado en el conector del controlador A. Cuando N es un valor comprendido entre 4 y 7 ó entre 16 y 19, ofrece el estado del dispositivo señalizador conectado en el conector del controlador B.

Para leer el estado del ratón o de la bola gráfica, utilizar en primer lugar PAD(12) o PAD(16) y leer las coordenadas. Para introducir el estado del botón se utiliza la función STRIG. La entrada sólo es posible cuando se utiliza en primer lugar PAD(12) o PAD(16). Si el intervalo de tiempo es demasiado largo, el contador introducirá un valor de coordenadas arbitrario.

Ejemplo de ejecución

```
100 SCREEN 5
110 FOR I=0 TO 7
120  A$=A$+CHR$(2^I-1)
130 NEXT
140 SPRITE$(0)=A$
150 D=PAD(12)
160 X=X+PAD(13):Y=Y+PAD(14)
170 PUT SPRITE 2,(X,Y),8,0
180 GOTO 150
```

PAINT (paint)

Colorea el área rodeada por un contorno.

FORMATO

PAINT [STEP] (coordenada X, coordenada Y), [color de pintura], [color del contorno]

Coordenada X — **Cond.** Constantes, variables, variables de matriz y sus expresiones (numéricas); $0 \leqslant X < 513$.

Coordenada Y — **Cond.** Constantes, variables, variables de matriz y sus expresiones (numéricas); $0 \leqslant Y < 213$.

Color de pintura, color del contorno

Cond. SCREEN 2 a 7:
Constantes, variables, variables de matriz y sus expresiones (numéricas); $0 \leqslant$ color < 16.
SCREEN 8:
Constantes, variables, variables de matriz y sus expresiones (numéricas); $0 \leqslant$ color < 256.

Omit Color del primer plano actual.

FUNCION Y UTILIZACION

Colorea el área delimitada por el contorno especificado, incluyendo la posición de las coordenadas X,Y; el color es el especificado en el término color de la pintura.

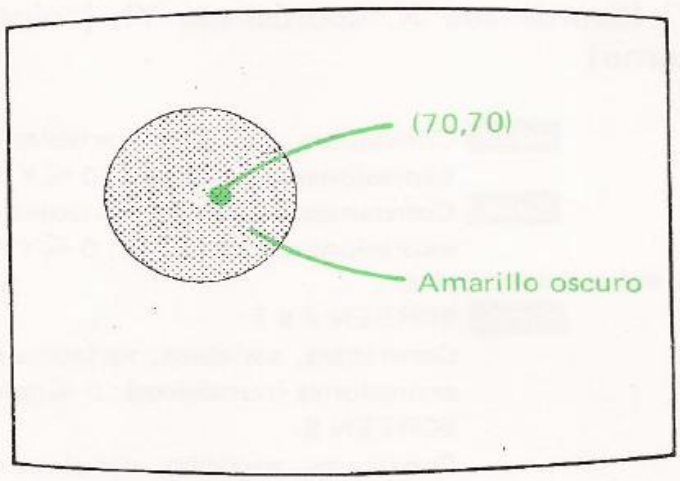
- Si la línea del contorno no es cerrada, colorea toda la pantalla.
- En los modos SCREEN 2 y SCREEN 4 colorea toda la pantalla si no coinciden los colores de pintura y del contorno.

Cuando se especifica STEP, pasa X,Y a un nuevo sistema de coordenadas que tiene su origen en el último punto especificado en la instrucción gráfica inmediatamente anterior.

Ejemplo de ejecución

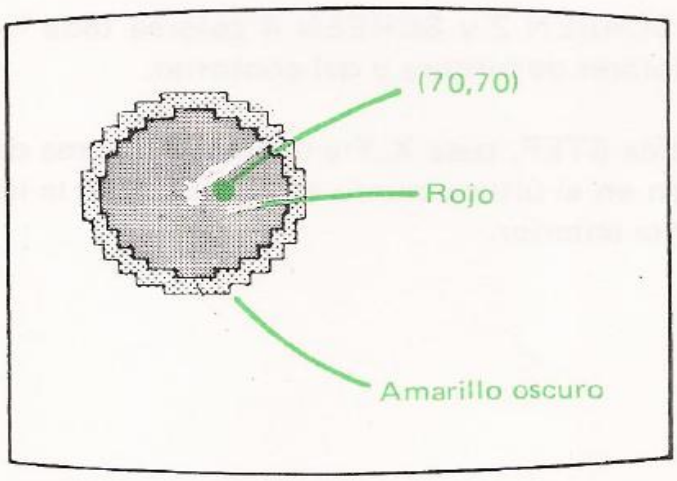
```
10 CLS
20 SCREEN 2
30 CIRCLE (70,70),40,10
40 PAINT (70,70),10,10
50 GOTO 50
```

En los modos SCREEN 2 y SCREEN 4 hay que especificar el mismo color en el término "color de pintura" y en el término "color del contorno".




```
10 CLS
20 SCREEN 3
30 CIRCLE (70,70),40,10
40 PAINT (70,70),8,10
50 GOTO 50
```

En los modos SCREEN 3, 5, 6 y 7 se pueden especificar colores diferentes para el área a colorear y la línea de contorno.



Función PDL (paddle)

Ofrece el valor procedente de un "paddle".

FORMATO

PDL(N)

Cond. Constantes, variables, variables de matriz y sus expresiones (enteros); $0 \leqslant X < 13$.

Valor obtenido: Tipo numérico, $0 \leqslant$ valor $\leqslant 255$.

FUNCION Y UTILIZACION

Ofrece el valor procedente de un "paddle", en forma de dato numérico. Cuando N es un número impar, proporciona los datos de un "paddle" conectado al controlador A, y cuando N es un número par proporciona los datos de un "paddle" conectado al controlador B.

Función PEEK (peek)

Ofrece el contenido de una dirección de memoria especificada.

Opuesta: POKE

FORMATO

PEEK (dirección)

Dirección **Cond.** Constantes, variables, variables de matriz y sus expresiones (numéricas); $-32768 \leqslant$ dirección < 65536. En el caso de números negativos, su valor es el resultado de sumarlos a 65536.

Valor obtenido: Tipo numérico en notación decimal.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
M=PEEK(50000)
```

Asigna el contenido de la dirección de memoria 5000 a la variable M.

PLAY (play) (1)

Genera un sonido de acuerdo con los submandatos especificados.

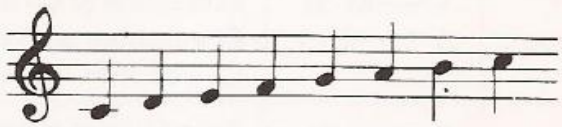
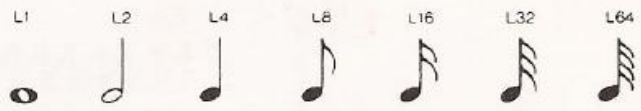
Afín: SOUND, PLAY, BEEP

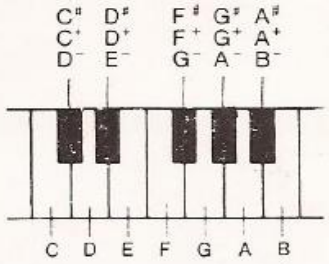
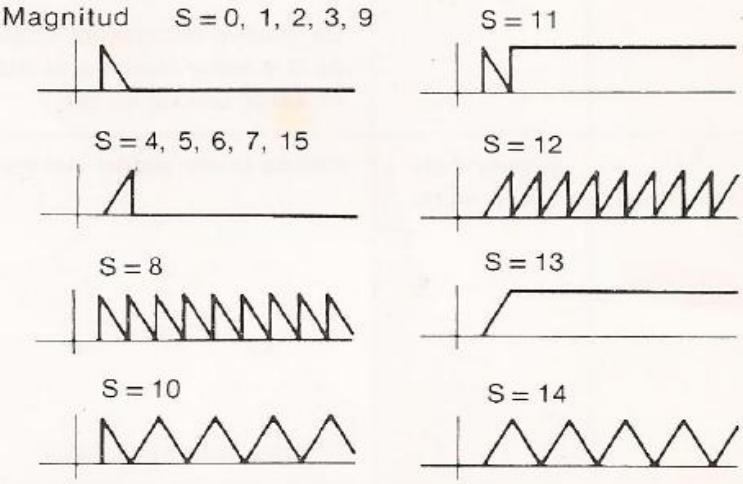
FORMATO

Play submandato [,submandato] [,submandato]

Submandato **Cond.** Constantes y variables de cadena.

Submandatos

Submandato	Condición	Significado
Tn (Tiempo)	Enteros de 32 ≤n ≤255	Especifica la velocidad de ejecución de la música. El valor de n indica la cuenta de un cuarto de nota durante un minuto. La selección inicial es T120.
On (octava)	Enteros de 1 ≤n ≤8	Especifica una de las 8 octavas. Cuando se especifica 04, ejecuta la música dentro de la escala mostrada a continuación.  La octava desciende cuando disminuye el valor de n y sube cuando el valor de n aumenta. El valor inicial es 04.
Ln (duración)	Enteros de 1 ≤n ≤64	Indica la duración del sonido.  El valor inicial es L4.
Nn (Nota)	Enteros de 0 ≤n ≤96	Especifica una nota musical. N36 es 04C  NO es un silencio La escala cromática aumenta al aumentar n en una unidad.

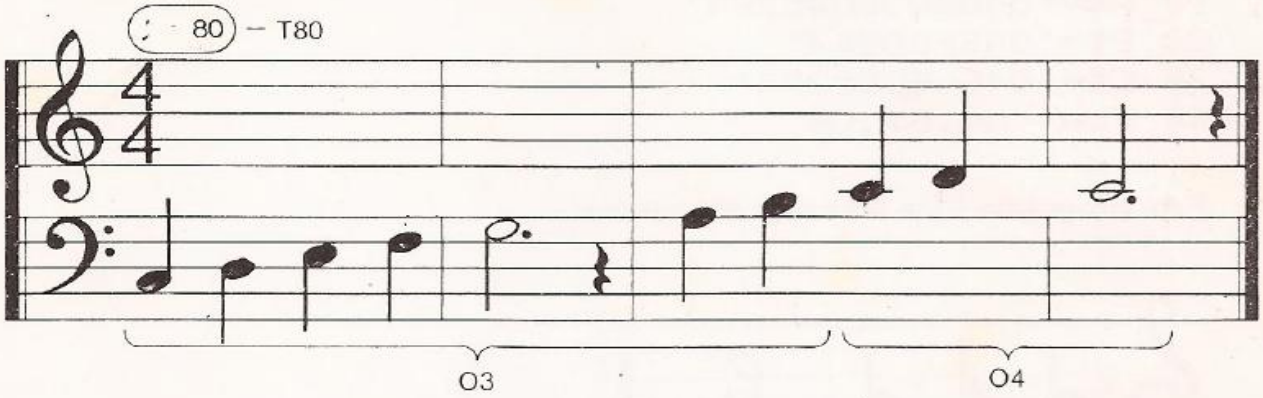
Mandato	Condición	Significado
A—G An—Gn	Enteros de 1 ≤n ≤64	Especifica la nota musical de una octava especificada.  Para el semitono se utilizan # (0 +) y −. La longitud del sonido se puede especificar con n. (C4 es igual que L4C). La longitud por omisión es la especificada por Ln.
Rn (silencio)	Enteros de 1 ≤n ≤64	Especifica un silencio. 
		Especifica el puntillo musical. La duración de la nota se amplia en 1/2 de la nota ó silencio que lo lleva delante. C4. = ♩. R4. = 𝄽.
Vn (volumen)	Enteros de 0 ≤n ≤15	Especifica el volumen. El volumen aumenta al aumentar n. La posición inicial es V8.
Sn (forma)	Enteros de 0 ≤n ≤15	Especifica el modelo de variación del volumen entre los siguientes:  La selección inicial es S1. La generación de muchos sonidos diferentes es el resultado de la combinación del submandato S con el submandato M.

Mandato	Condición	Significado
Mn (modulación)	. Enteros de 1 ⩽n ⩽65535	Determina el ciclo del modelo especificado por el submandato S. El ciclo se alarga al aumentar el valor de n. La selección inicial es M255.

FUNCION Y UTILIZACION

```
PLAY "T80O3L4CDEFG2.RAB04CDC2."
```

La instrucción anterior produce un sonido compuesto por las siguientes notas.



Expresión de un submandato con una variable

```
M$="T80O3L4CDEFG2.RAB04CDC2."
PLAY M$
```

Estas instrucciones asignan un submandato a la variable de cadena M$ que después se especifica en la sentencia PLAY como submandato.

Expresión de una parte de un submandato con una variable (X variable;)

```
10 M$="CD[illegible]G2.R"
20 PLAY "O4L4XM$;GAGAG2.R"
30 PLAY "XM$;AB05CDC2."
```

Cuando en una instrucción PLAY se utiliza un submandato asignado a una variable de cadena encerrada entre comillas (" "), añadir antes X y después ; . En el ejemplo anterior, el submandato asignado a M$ se utiliza en dos instrucciones PLAY.

Expresión de un submandato con una variable (= variable;)
El valor de n especificado en un submandato puede ser una constante o una variable, incluidas como tal en una instrucción PLAY. Cuando se expresan como una variable, añadir "=" antes y ";" después.

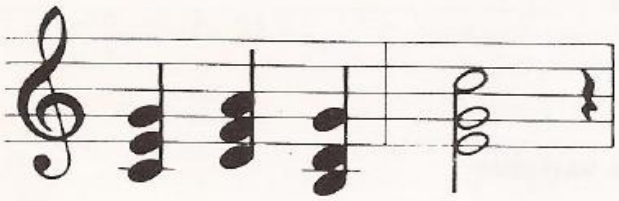
```
10 FOR I=1 TO 8
20 PLAY "O=I;CEG"
30 NEXT I
```

Este programa ejecuta una octava musical, desde PLAY "01CEG" hasta PLAY "08CEG".

Generación de acordes
Se pueden ejecutar simultáneamente hasta 3 voces, por ejemplo, PLAY A$, B$, C$.

```
10 A$="O4CDO3BO4E2R4"
20 B$="O4EFDG2R4"
30 C$="O4GAGO5C2R4"
40 PLAY A$,B$,C$
```

Este programa toca las notas siguientes:



Función PLAY (play) (2)

Comprueba si se está ejecutando música o no.

Afín: PLAY

FORMATO

PLAY (N)

N **Cond.** Constantes, variables, variables de matriz, sus expresiones (enteros); $0 \leqslant N < 4$.

Valor obtenido: Tipo numérico.

FUNCION Y UTILIZACION

Con una instrucción PLAY se pueden ejecutar simultáneamente tres sonidos diferentes.
En el caso concreto de PLAY A$, B$, C$;
el sonido del submandato A$ sale por el canal 1, el sonido del submandato B$ sale por el canal 2, y el sonido del submandato C$ sale por el canal 3.
La función PLAY comprueba si hay datos en el buffer (memoria intermedia) de los datos musicales del canal 1 cuando N = 1, del canal 2 cuando N = 2, y del canal 3 cuando N = 3. Cuando hay datos en la memoria intermedia da un −1 y cuando no hay datos un 0. Cuando N = 0, da la suma lógica (OR) de los estados de las memorias intermedias (0 o 1); en otras palabras, si una de ellas está en −1, da un −1.

Función **POINT** (point)

Ofrece el código de color de un punto de una posición especificada.

FORMATO

POINT (X, Y)

X, Y **Cond.** Constantes, variables, variables de matriz y sus expresiones (numéricas); $-32768 \leqslant$ coordenada < 32768.

Valor obtenido: Tipo numérico (−1 cuando la posición especificada esté fuera de la pantalla.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
10 SCREEN 3
20 FOR I=1 TO 250
30 X=INT(RND(1)*255)
40 Y=INT(RND(1)*191)
50 PSET (X,Y),1
60 NEXT I
70 FOR Y=0 TO 191 STEP 4
80 FOR X=0 TO 255 STEP 4
90 C=POINT(X,Y)
100 IF C=4 THEN PSET (X,Y),15
110 NEXT X,Y
120 GOTO 120
```

La línea 90 asigna el código de color correspondiente a una posición (X,Y) a la variable C, y la línea 100 lo cambia a blanco si C es 4 (azul oscuro).

POKE (poke)

Escribe datos en la dirección de memoria especificada. Opuesta: PEEK

FORMATO

POKE dirección, expresión

Dirección **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas); $-32768 \leqslant$ dirección < 65536.
En caso de números negativos, su valor es el resultado de sumarlos a 65536.

Expresión **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant$ expresión < 256.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
POKE 50000,255
```
Escribe el dato 255 en la dirección de memoria 5000

```
POKE &HD000,&HA8
```
Escribe el dato A8H en la dirección de memoria D000H.

Función **POS** (position)

Ofrece la coordenada X de la posición del cursor.

FORMATO

POS (X)

Cond. Constante, variable, variable de matriz o sus expresiones (numéricas) (argumento ficticio).

Valor obtenido: Tipo entero.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
10 INPUT A$
20 PRINT A$;:X=POS(X)
30 IF X>=5 THEN CLS
40 PRINT:GOTO 10
```

La línea 20 X = POS (X), asigna el valor de la coordenada X de la posición del cursor, a la variable X. En consecuencia, la pantalla se borra al asignar una cadena con cinco o más caracteres para A$.

PRESET (point reset)

Marca o borra un punto en la pantalla, en el modo gráfico.

FORMATO

PRESET [STEP] (coordenada X, coordenada Y) [,color] [,operación lógica]

Coordenada X,Y	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (numéricas); $-32768 \leqslant$ coordenada < 32768.
Color	**Cond.**	SCREEN 2 a 7: Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant$ color < 16.
	Omit	Color actual del fondo.
Operación lógica	**Cond.**	SCREEN 5 a 8: PSET, PRESET, XOR, OR, AND.
	Omit	PSET.

FUNCION Y UTILIZACION

Cuando se ejecuta con omisión de la especificación del color, marca un punto con el mismo color del fondo. En consecuencia, si hay algo dibujado en la posición especificada, en un color distinto al color del fondo, da la impresión de que ha borrado un punto en esa posición.

- Cuando se especifica un color, la función es idéntica que cuando se especifica un color con PSET.
- Consulte el programa ejemplo de PSET.

Cuando se especifica STEP, pasa X,Y a un nuevo sistema de coordenadas que tiene su origen en el punto especificado en último lugar en la instrucción gráfica inmediatamente anterior.

PRINT (print)

Visualiza datos numéricos o alfanuméricos en la pantalla en modo texto.

Afín: LPRINT, PRINT USING

FORMATO

PRINT [expresión] [separador] [expresión] [separador] [expresión]...

Expresión Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas o de cadena).
Separador Cond. Coma (,) o punto y coma (;).

FUNCION Y UTILIZACION

Método de escritura de expresiones (datos)
Las constantes de tipo numérico y las variables numéricas y de cadena se escriben tal como son, y las constantes de cadena se escriben encerradas entre comillas (" ").

Función del separador
Cuando los datos están separados por una coma, la función tab de 14 dígitos inserta espacios entre los datos, y cuando están separados por un punto y coma, los datos se visualizarán uno al lado del otro.
Si al final de la instrucción no se escribe ningún separador, realiza un salto de línea tras la presentación de los datos. Si al final se ha puesto un separador, los datos de la siguiente instrucción PRINT continuan en la misma línea.

Signos y datos numéricos
Con respecto a los signos positivo (+) y negativo (−), omite el positivo (+) y presenta el negativo (−) donde esté. (Si se utiliza el separador punto y coma con datos numéricos, inserta dos espacios entre los datos para dejar sitio al signo.

Omisión del formato

Se obtiene el mismo resultado escribiendo el signo de interrogación (?) en vez de PRINT.

Ejemplo de ejecución

```
10 A$="ABC":B$="DEF"
20 PRINT A$;B$
30 PRINT A$,B$
40 PRINT
50 PRINT "MSX"
60 PRINT +50,-50
70 ?"PERSONAL COMPUTER"
RUN
ABCDEF                          Resultado de la línea 20
ABC             DEF             Resultado de la línea 30

                                Resultado de la línea 40
MSX                             Resultado de la línea 50
 50             -50             Resultado de la línea 60
PERSONAL COMPUTER               Resultado de la línea 70
```

Función PRINT USING (print using)

Visualiza datos a la pantalla usando un formato especificado.

Afín: PRINT, LPRINT USING

FORMATO

PRINT USING símbolo del formato; expresión [,expresión] ...

Expresión **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas y de cadena).

FUNCION Y UTILIZACION

Presenta en pantalla el valor de una expresión en un formato especificado por un símbolo de formato.

Símbolos de formato para datos alfanuméricos

Símbolo	Formato de la expresión y ejemplo de ejecución

"!"	Imprime el primer caracter
	`PRINT USING "!"; "BASIC", "MSX"`
	`BM`

"\ ␣ \"
↑
n espacios
	Imprime n + 2 caracteres. Cuando los datos sean menores a N + 2 caracteres, inserta espacios para los caracteres residuales.
	`PRINT USING "\ \";"ABCDEF","GHI","JKLMN"`
	`ABCDGHI JKLM`

"&"	Imprime todas las cadenas de caracteres.
	`10 A$= "Norte": B$ = "Sur"`
	`20 PRINT USING " Polo & "; A$, B$`
	`RUN`
	`Polo Norte Polo Sur`

Símbolos de formatos para datos de tipo numérico

<table>
<tr><th>Símbolo</th><th>Formato de la expresión y ejemplo de ejecución</th></tr>
<tr><td>"#"</td><td>
Escribir tantos dígitos como número de símbolos #
La coma decimal es ".".
<pre>PRINT USING "POINT:###.#";123.4
POINT:123.4</pre>
● Cuando el número de dígitos enteros es menor que el número de símbolos # especificado, presenta los datos con justificación a la derecha, y si es mayor, añade el símbolo %antes del dato.
<pre>10 PRINT USING "####";12
20 PRINT USING "####";12345
RUN
  12
%12345</pre>
● Cuando el número de dígitos siguientes a la coma decimal de un dato numérico sea inferior al número de símbolos # especificados, añade un cero, y cuando es mayor, redondea el número.
<pre>10 PRINT USING "##.##";25.3
20 PRINT USING "##.##";25.345
RUN
25.30
25.35</pre>
Ignora el signo "+" de los datos numéricos y cuenta el signo "–" como un dígito.
<pre>10 PRINT USING "###";+123
20 PRINT USING "###";-123
RUN
123
%-123</pre>
</td></tr>
<tr><td>"+"</td><td>
Antes o después del dato numérico, pone el signo "+" si se trata de un número positivo y el signo "–" si se trata de un número negativo.
<pre>10 PRINT USING "+####";123,-123
20 PRINT USING "####+";123,-123
RUN
 +123 -123
 123+ 123-</pre>
</td></tr>
<tr><td>"_"</td><td>
Pone el signo menos "–" a continuación de un dato numérico negativo.
<pre>PRINT USING "###-";123,-123
123 123-</pre>
</td></tr>
</table>

" ** "	Rellena con asteriscos los espacios anteriores al dato numérico. Cada asterisco del formato expresa un dígito.
	`10 PRINT USING "**########";123`
	`20 PRINT USING "**########";-234`
	`RUN`
	`******123`
	`*****-234`

"££"	Añade el signo "£" antes del dato numérico. Cada £ del formato expresa un dígito.
	`10 PRINT USING "££###"; 1234`
	`20 PRINT USING "+££###"; -1234`
	`RUN`
	`£ 1234`
	`- £ 1234`

" * * £"	Añade el signo "£" justo antes del dato numérico, y rellena los espacios anteriores con asteriscos "*".
	`PRINT USING " **£###.##"; 12.34`
	`*** £ 12.34`

" , "	Cuando se especifica la coma en algún sitio anterior a la coma decimal (el punto), inserta comas entre cada tres dígitos, por delante de la coma decimal (el punto).
	`PRINT USING "#,######.##";12345.67`
	`12,345.67`

"^^^^"	Presenta datos numéricos con el formato de coma flotante. "^^^^" corresponde a los dígitos de la parte exponente.
	`PRINT USING "##.##^^^^";234.56`
	`2.35E+02`

(1) El BASIC–MSX2 incorporado en los ordenadores comercializados en Francia y Alemania Federal utiliza el símbolo "$" en vez del "£".

PRINT # (print number)

Escribe datos en un fichero abierto por una instrucción OPEN.

Afín: PRINT # USING, PRINT

FORMATO

PRINT # número de fichero, [expresión] [separador] [expresión]

Número de fichero	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (de tipo entero); 1 ≤número de fichero ≤número especificado en instrucción MAXFILES
Expresión	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (numéricas y de cadena).
Separador	**Cond.**	Coma (,) o punto y coma (;).

FUNCION Y UTILIZACION

Introduce datos en un fichero abierto por una instrucción OPEN.

Ejemplo de ejecución

```
10 OPEN "CAS:DATA" FOR OUTPUT AS #1
20 FOR I=0 TO 4
30 READ A$
40 PRINT #1,A$;",";
50 NEXT I
60 CLOSE #1
70 DATA TOKYO,LONDON,PARIS,PEKING,NEW YORK
```

Abre un fichero para escritura

Escribe datos en el fichero

Este programa escribe secuencialmente en una cinta cassette los datos escritos en la línea 70, en un fichero de nombre "DATA".

PRINT # USING (print number using)

Escribe datos con un formato especificado en un fichero abierto por una instrucción OPEN.

Afín: PRINT #, PRINT USING

FORMATO

PRINT # número de fichero USING símbolo del formato; expresión [,expresión] ...

Número de fichero	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (enteras); 1 ⩽ número de fichero ⩽ número especificado por instrucción MAXFILES
Expresión	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (numéricas y de cadena).

FUNCION Y UTILIZACION

Se puede especificar este formato para enviar datos a un fichero. Para información relativa a los símbolos de formato, ver PRINT USING.

PSET (point set)

Marca un punto en la pantalla, en el modo de gráficos.

FORMATO

PSET [STEP] (coordenada X, coordenada Y), [color] , [operación lógica]

Coordenada X,Y	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (numéricas); $-32768 \leqslant$ coordenada < 32768.
Color	**Cond.**	SCREEN 2 a 7: Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant$ color < 16. SCREEN 8: Constantes, variables, variables matriciales, sus expresiones (numéricas); $0 \leqslant$ color < 256.
	Omit	Color actual del primer plano.
Operación lógica	**Cond.**	SCREEN 5 a 8: PSET, PRESET, XOR, OR, AND.
	Omit	PSET.

FUNCION Y UTILIZACION

Cuando se especifica STEP, las coordenadas X,Y pasan a un nuevo sistema de coordenadas que tiene su origen en el punto especificado en último lugar en la instrucción gráfica inmediatamente anterior.

Ejemplo de ejecución

```
10 SCREEN 2
20 FOR X=0 TO 255
30 PSET (X+1,100)
40 PRESET (X,100)
50 NEXT X
```

Dibuja un punto

Borra el punto dibujado por la línea 30

PUT (put)

Escribe el registro definido por la instrucción FIELD en un fichero de acceso aleatorio.

Opuesta: GET

FORMATO

PUT [#] número de fichero [,número de registro]

Número de fichero **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas); 1 ≤número de fichero ≤número especificado en la instrucción MAXFILES

Número de registro **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas); 1 ≤número <65536.

Omit El número de registro utilizado en la instrucción GET o PUT ejecutada en último lugar más uno.

FUNCIÓN Y UTILIZACION

```
PUT #1,3
```

La ejecución de esta instrucción graba en el fichero de acceso aleatorio, preparado por la instrucción LSET/RSET, el registro número 3 (tercer registro).

Ejemplo de ejecución

```
10 OPEN "A:TEST.DAT" AS #1
20 FIELD #1,2 AS CODE$,15 AS NAM$,10 AS  TEL$
30 READ A%,B$,C$
40 LSET CODE$=MKI$(A%)
50 LSET NAM$=B$
60 LSET TEL$=C$
70 PUT #1,1
80 CLOSE #1
90 DATA 100, JORGE, 211-71-71
100 END
```

PUT SPRITE (put sprite)

Visualiza un sprite en una posición arbitraria del plano de sprite especificado.

Afín: SPRITE ON, SPRITE OFF, SPRITE STOP, SPRITE$

FORMATO

PUT SPRITE número de plano de sprite, [[STEP] (coordenada X, coordenada Y)], [color] , [número de sprite]

Número de plano de sprite

Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤ número < 32.

Coordenada X

Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); −32 ≤ X < 256.

Coordenada Y

Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); −32 ≤ Y < 212.

Color

Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤ color < 16.

Omit Color actual del primer plano.

Número de sprite

Cond. Para 8x8 puntos ... Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤ número < 256.

Para 16 x 16 puntos ... Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤ número < 64.

Omit El mismo que el del plano de sprite.

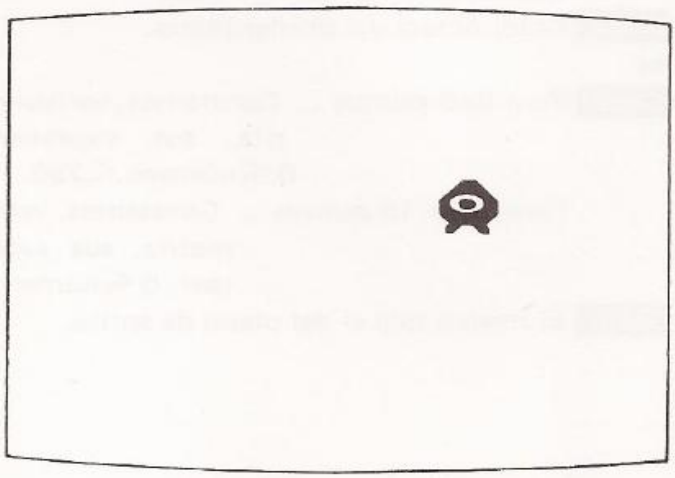
FUNCION Y UTILIZACION

Cuando se especifica STEP, las coordenadas X,Y pasan a un nuevo sistema de coordenadas que tiene su origen en el punto especificado en último lugar en la instrucción gráfica inmediatamente anterior.

Ejemplo de ejecución

```
10 SCREEN 2
20 SPRITE$(1)=CHR$(&H18)+CHR$(&H3C)+CHR$
(&H66)+CHR$(&HDB)+CHR$(&HE7)+CHR$(&H7E)+
CHR$(&H24)+CHR$(&H42)
30 X=0:Y=0:DX=1:DY=1
40 PUT SPRITE 0,(X,Y),,1
50 X=X+DX:Y=Y+DY
60 IF X>250 OR X<0 THEN DX=-DX
70 IF Y>190 OR Y<0 THEN DY=-DY
80 GOTO 40
```

La línea 20 define un sprite en forma de OVNI, asignado al sprite número 1. La instrucción 40, PUT SPRITE, presenta este sprite en pantalla; el número del plano de sprite es el 0. Como se ha omitido el color, es el mismo que el color de primer plano seleccionado anteriormente. El OVNI da la impresión de volar por la pantalla debido al cambio de los valores de las coordenadas X,Y, que especifican su posición.



READ (read)

Lee los datos especificados por una instrucción DATA.

Afín: DATA, RESTORE

FORMATO

READ variable [,variable] [,variable]

Variable **Cond.** Numérica o de cadena.

FUNCION Y UTILIZACION

Lee los datos secuencialmente, empezando por el primer dato de la instrucción DATA de número de línea más bajo del programa, y los asigna secuencialmente a variables de la instrucción READ.

- Cuando en una instrucción READ hay más de una variable numérica o de cadena, van separadas por comas.
- El tipo de variable debe coincidir con el dato correspondiente.

```
10 READ A,B,C,D$,E$
20 PRINT A,B,C,D$,E$
100 DATA 5,10,20,ABC,XYZ
```

- Cuando en un programa hay varias instrucciones READ, la segunda instrucción READ comienza a leer el dato siguiente al último dato leído por la anterior instrucción READ.
- La ejecución de una instrucción RESTORE hace que la lectura de la instrucción READ ejecutada a continuación vuelva a la instrucción DATA de número más pequeño después del número de línea especificado por la instrucción RESTORE.

Ejemplo de ejecución

```
10 READ A,B,C
20 READ D$,E$
30 PRINT A;B;C;D$;E$
100 DATA 10,20,30,ABC,DEF
RUN
 10  20  30 ABCDEF
```

REM (remark)

Inserta un comentario en un programa.

FORMATO

REM comentario

FUNCION Y UTILIZACION

La instrucción REM sirve para insertar un comentario con vistas a facilitar la lectura del listado del programa.

Ejemplo de ejecución

```
10 REM MUSIC
20 PLAY "T60CEGEC1"
```

Aunque en el listado del programa aparece la instrucción REM, al ejecutar el programa se ignora esta línea.

```
10 'MUSIC
20 PLAY "T60CEGEC1
```

En vez de escribir REM, se puede poner apóstrofe (')

```
10 PRINT "MSX":REM Output
20 PRINT "PERSONAL COMPUTER" 'Output
RUN
MSX
PERSONAL COMPUTER
```

Aunque es obligatorio escribir los dos puntos (:) cuando se utiliza una sentencia REM junto con otra instrucción coma (,), éstos pueden omitirse, utilizando el apóstrofe (') en lugar de REM.

RENUM (renumber)

Renumera las líneas de un programa.

FORMATO

RENUM [nuevo número de línea inicial] , [antiguo número de línea inicial] , [incremento]

Nuevo número de línea inicial
Cond. Constantes enteras, 0 ⩽número ⩽65539.
Omit 10

Antiguo número de línea inicial
Cond. Constantes enteras, 0 ⩽número ⩽65529.
Omit Número de línea más bajo antes de la ejecución.

Incremento
Cond. Constantes enteras, 0 ⩽incremento ⩽ 65529.
Omit 10.

FUNCION Y UTILIZACION

Sirve para volver a numerar las líneas tras una corrección del programa.

- El número de línea al que se salta tras una instrucción GOTO o GOSUB se puede renumerar correctamente mediante la ejecución de una instrucción RENUM. De todas formas, si el número de línea especificado al que se salta con una instrucción GOTO etc no existe en el momento de la ejecución de la instrucción RENUM, no cambia ese número de línea y por lo tanto se produce un error.

Ejemplo de ejecución

```
RENUM
```
Renumera todas las líneas à partir de la 10 con un incremento de 10.

```
RENUM 100,,100
```
Renumera todas las líneas que comienzan a partir de 100 con incrementos de 100.

```
RENUM 100
```
Renumera todas las líneas con números que empiezan por el 100, con incrementos de 10.

```
RENUM 100,38,20
```
Renumera la línea 38 y siguientes con números de línea que comienzan por el 100, con incrementos de 20.

RESTORE (restore)

Especifica la instrucción DATA que será leída por una instrucción READ.

Afín: DATA, READ

FORMATO

RESTORE [número de línea]

Número de línea **Cond.** Constantes enteras, 0 ⩽ número ⩽ 65529.
Omit Instrucción DATA con el número de línea más bajo.

FUNCION Y UTILIZACION

La instrucción RESTORE sirve para leer varias veces los mismos datos.
La ejecución de la instrucción RESTORE hace que la siguiente instrucción READ empiece a leer datos a partir de la instrucción DATA con el número de línea más bajo siguiente al número de línea especificado en la instrucción RESTORE.

Ejemplo de ejecución

```
10 READ A,B,C
20 READ D,E,F
30 RESTORE 110
40 READ G,H,I
50 PRINT A;B;C;D;E;F;G;H;I
100 DATA 10,20,30
110 DATA 40,50,60
RUN
 10   20   30   40   50   60   40   50   60
```

RESUME (resume)

Devuelve la ejecución al programa principal tras la ejecución de una rutina de proceso de errores.

Afín: ON ERROR GOTO, RETURN

FORMATO

RESUME [{ 0 / número de línea / NEXT }]

Número de línea — **Cond.** Constantes enteras, 0 ⩽ número ⩽ 65529.
Omit Línea donde haya ocurrido el error.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
RESUME 0   or   RESUME      — Devuelve la ejecución a la sentencia donde se
                              haya producido un error.
RESUME 100  — Devuelve la ejecución a la línea 100.
```

(Ver el programa ejemplo de ON ERROR GOTO).

Función RIGHT$ (right dollar)

Ofrece un número arbitrario de caracteres, en forma de datos alfanuméricos, tomados a partir del extremo derecho de una cadena de caracteres.

Afín: LEFT$, MID$

FORMATO

RIGHT$ (X$, N)

X$	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (de cadena).
N	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant N < 256$.

Valor obtenido: Tipo cadena.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT RIGHT$ ("HIT-BIT", 3)
BIT
PRINT RIGHT$ ("HIT-BIT", 5.3)
BIT

PRINT RIGHT$ ("HIT-BIT", 0)

Ok
```

Cuando N no es un valor entero, omite las cifras siguientes a la coma decimal.

Cuando N es 0 da una cadena nula.

Función RND (random)

Ofrece un número aleatorio positivo menor que 1 (incluyendo el 0).

FORMATO

RND (X)

X **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas).

Valor obtenido: Tipo numérico.

FUNCION Y UTILIZACION

Cuando X es mayor que 0

Genera siempre números aleatorios en la misma secuencia.

```
 .5952194399462 3
 .1065862805015 8
 .7659765177282 3
 .5775639293595 8
 .7347475950302 3
 .1842681290975 8
 .3707537790522 3
 .9495415165155 8
 .6379955689942 3
 .4704111764135 8
```

Cuando X es negativo
Genera una serie que corresponde al valor de X y después genera números aleatorios con esta serie.

```
10 PRINT RND(-1)
20 FOR N=1 TO 10
30 PRINT RND(N)
40 NEXT N
RUN
 .4389820420821
 .9624816692
 .2106965585230 1
 .326517363050 4
 .4777512433658 1
 .3409147084636
 .1297118408166 1
 .0977770174288
 .3515786017554 1
 .8353896966666
 .6390264138622 1
```

Cuando X es 0
Se obtendrá el mismo valor que el generado anteriormente.

```
10 PRINT RND(1)
20 PRINT RND(0)
30 PRINT RND(-1)
40 PRINT RND(0)
RUN
 .5952194399462 3
 .5952194399462 3
 .4389820420821
 .4389820420821
```

RSET (right set)

Escribe datos en un registro de un fichero de acceso aleatorio alineándolos por la derecha.

Afín: LSET

FORMATO

RSET variable de cadena = expresión de cadena

Variable de cadena	**Cond.**	Variable de cadena.
Expresión de cadena	**Cond.**	Constantes, variables, variables de matriz, sus expresiones (de cadena).

FUNCION Y UTILIZACION

Pone los datos de cada variable en el registro especificado por la instrucción FIELD como preparación para la escritura de datos en un fichero de acceso aleatorio mediante una instrucción PUT. La instrucción RSET alinea los datos en el lado derecho del registro; si el dato es más corto que la longitud de la variable especificada, rellena con espacios en blanco los sitios vacios; si el dato es más largo que el espacio especificado, ignorará la parte derecha del dato de cadena.

- Al organizar los datos numéricos, pasa primero los datos a cadenas mediante las funciones MKI$, MKS$S y MKD$.

```
RSET A$=X$
```
Pone el dato de cadena X$ en la variable A$ del registro.

```
RSET B$=MKS$(N)
```
Pasa el dato numérico N a cadena asignándolo a la variable B$ del registro.

RUN (run)

1. Ejecuta un programa a partir de una línea especificada.
2. Carga un fichero de disco 💾 o de la RAM–Disk 🟫 y lo ejecuta.

FORMATO

1. RUN [número de línea]

Número de línea	**Cond.**	Constantes de tipo entero, 0 ⩽ número ⩽ 65529.
	Omit	Ejecuta el programa a partir de la primera línea.

🟫💾 2. RUN "[nombre de unidad] nombre de fichero [.extensión]", [,R]

Nombre de unidad	**Cond.**	A:, B:, C:, D:, E:, F:, G:, H:
	Omit	Unidad de disco actual.
Nombre de fichero	**Cond.**	Una cadena de 8 o menos caracteres.
Extensión	**Cond.**	Una cadena de 3 caracteres o menos.
	Omit	Una cadena nula.
Opción R	**Omit**	Cierra todos los ficheros de datos.

FUNCION Y UTILIZACION

FORMATO 1

La instrucción RUN pone todos los valores numéricos a 0, las variables alfanuméricas a cadena nula y las variables de matriz en una condición indefinida; luego ejecuta el programa.

Tras completar la ejecución del programa, entra en el estado de espera de mandatos.

Pulsar la tecla STOP para detener temporalmente la ejecución del programa. La ejecución se reanuda pulsando STOP otra vez.

Pulsar CTRL + STOP para interrumpir el programa. El programa se puede reanudar introduciendo la orden CONT.

FORMATO 2

La ejecución de la instrucción RUN pone a 0 todos los valores numéricos, a cadena nula las variables alfanuméricas, y en una condición indefinida las variables de matriz, y cierra todos los ficheros. Luego carga del disco especificado el programa especificado, y lo ejecuta.

SAVE (save)

Almacena un programa BASIC en el dispositivo especificado.

Opuesta: LOAD Afín: CSAVE

FORMATO

SAVE "[nombre de dispositivo] [nombre de fichero [.extensión]]" [,A]

Nombre de dispositivo **Cond.** CAS:, CRT:, GRP:, LPT:,
MEM:
A:, B:, C:, D:, E:, F:, G:, H:
Omit CAS:
Unidad de disco actual.

Nombre de fichero **Cond.** Una cadena de 6 caracteres o menos.
Una cadena de 8 caracteres o menos.
Omit Una cadena nula.
(No se puede omitir el nombre de fichero cuando se trabaja con discos flexibles o con RAM-Disk).

Extensión **Cond.** Una cadena de 3 o menos caracteres.
Omit Una cadena nula.

Opción A **Omit** Almacena el programa en formato intermedio.

FUNCION Y UTILIZACION

Cuando se especifica CAS: como nombre de dispositivo, almacena el programa BASIC de memoria en la cinta cassette en formato ASCII. Cuando se especifica una unidad de discos (por ejemplo A:, o B:), almacena el programa en el formato ASCII cuando se especifica la opción A, y en formato intermedio cuando se omite la opción A.

Nota
La instrucción CSAVE sirve para almacenar un programa en cinta cassette en formato intermedio.

Ejemplo de ejecución

```
SAVE "CAS:PROG2"
SAVE "PROG2",A
```

- El programa a fusionar con un programa en memoria mediante la instrucción MERGE, se debe almacenar en formato ASCII.
- Los programas se almacenan en la RAM—Disk en formato ASCII.

SCREEN (screen)

Establece el modo de visualización de la pantalla, el tamaño de los sprites, la activación o nó del chasquido de las teclas, la velocidad de transmisión del cassette (en baudios), el modo de entrelazado, y el tipo de impresora.

FORMATO

SCREEN [modo], [tamaño del sprite] , [interruptor de chasquido de las teclas] , [velocidad de transmisión en baudios], [tipo de impresora] , [modo de entrelazado]

Modo

Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant$ modo < 9.

Omit Modo actual.

Tamaño del sprite

Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant$ tamaño < 4.

Omit Tamaño actual.

Interruptor de chasquido de las teclas

Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant$ interruptor < 256.

Omit Estado actual.

Velocidad en baudios

Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant$ velocidad < 3.

Omit Velocidad actual.

Tipo de impresora

Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant$ tipo < 256.

Omit Tipo de impresora actual.

Modo de entrelazamiento

Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant$ modo < 4.

Omit Modo actual.

FUNCION Y UTILIZACION

Modos

Valor especificado	Modo
0 *	Modo texto; 40 caracteres x 24 líneas o 80 caracteres x 24 líneas.
1	Modo texto; 32 caracteres x 24 líneas.
2	Modo alta resolución; 256 x 192 puntos.
3	Modo multicolor; 64 x 48 puntos.
4	Modo alta resolución resaltada con función de sprites; 256 x 192 puntos.
5	Modo alta resolución, acceso punto a punto (bit a bit), 16 colores; 256 x 212 puntos.
6	Modo alta resolución, acceso punto a punto (bit a bit), 4 colores; 512 x 212 puntos.
7 **	Modo alta resolución, acceso punto a punto (bit a bit), 16 colores; 512 x 212 puntos.
8 **	Modo alta resolución, acceso punto a punto (bit a bit), 256 colores, 256 x 212 puntos.

* Si la anchura especificada en la instrucción WIDTH es 40 caracteres o menos, se selecciona el modo texto de 40 caracteres x 24 líneas, y si la anchura es 41 caracteres o más selecciona el modo texto de 80 caracteres x 24 líneas.

** No se puede operar en los modos 7 y 8 con ordenadores de 64K de VRAM.

Para la descripción de las funciones de los diferentes modos SCREEN, ver la página 12.

Tamaño del sprite

Valor especificado	Tamaño
0	8 x 8 puntos
1	8 x 8 puntos ampliado
2	16 x 16 puntos
3	16 x 16 puntos ampliado

Interruptor del chasquido de las teclas

Valor especificado	Clik
0	NO
Distinto de 0*	SI

* entre 1 y 255.

Velocidad de transmisión (baudios)

Valor especificado	Velocidad (baudios)*
1	1200 baudios
2	2400 baudios

*Velocidad en baudios del interface de cassette.

Tipo de impresora

Valor especificado	Impresora
0	Impresora MSX**
Distinto de 0*	Impresora no-MSX***

* Desde 1 a 255

** Impresora con caracteres gráficos compatible con ordenadores MSX.

*** Las impresoras no-MSX convierten los caracteres gráficos en espacios en blanco.

Modo de entrelazado

Valor especificado	Entrelazado
0	Normal
1	Entrelazado
2 *	Par/impar
3 *	Par impar, entrelazado.

*La página de visualización debe ser una página de número impar. Presenta alternativamente la página de visualización y la página de número inmediatamente inferior a la página de visualización.

Valores iniciales y especificaciones por omisión

En caso de omisión de una especificación, no cambia el modo actual, pero si se omiten todas las especificaciones, no se ejecutará la instrucción SCREEN.

Los valores iniciales (en la puesta en marcha del BASIC-MSX2) se pueden cambiar con una instrucción SET SCREEN. En cualquier caso, los valores iniciales seleccionados en fábrica son:

modo: modo texto de 40 caracteres x 24 líneas (WIDTH 37).
Tamaño del sprite: 8 x 8 puntos.
Interruptor de chasquido de las teclas: sonido.
Velocidad de transmisión en baudios: 1200 baudios.
Tipo de impresora: Impresora MSX.
Modo de entrelazamiento: Normal.

Ejemplo de ejecución

```
10 SCREEN 0,,1
```
Modo texto de 40 caracteres x 24 líneas, teclas con el sonido "click".

```
10 SCREEN ,,,2
```
Especificación de la velocidad de 2400 baudios.

```
10 SCREEN 2,3
```
Modo alta resolución; el sprite es de 16 x 16 puntos ampliados.

```
10 SCREEN 2
20 FOR I=0 TO 255
30 PSET (I,100)
40 NEXT I
50 GOTO 50
```

Al final de un programa la pantalla vuelve al modo texto (SCREEN 0 ó 1). Para seguir en el modo gráfico hay que añadir al final del programa una instrucción que mantenga el ordenador en este estado, como es el caso de la línea 50 del programa anterior. Esta línea mantiene el programa en ejecución; para poner término a la ejecución, pulsar CTRL + STOP .

SET ADJUST (set adjust)

Ajusta la posición de la imagen en la pantalla.

FORMATO

SET ADJUST (coordenada X, coordenada Y)

Coordenada X, coordenada Y

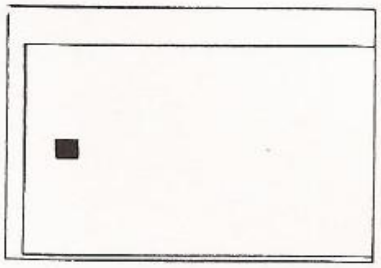
Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); $-8 \leqslant$ coordenada < 9.

FUNCION Y UTILIZACION

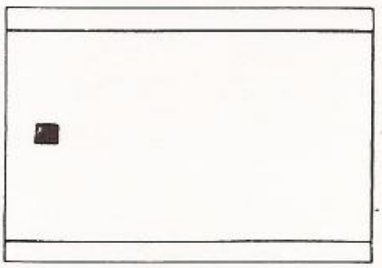
Los valores positivos producen el desplazamiento correspondiente de la coordenada X hacia la derecha y de la coordenada Y hacia abajo, cambiando la posición de la imagen en la pantalla. Los valores por omisión son 0 para las dos coordenadas.

Ejemplo de ejecución

```
SET ADJUST (8,8)
```



```
SET ADJUST (0,0)
```



SET BEEP (set beep)

Selecciona el sonido BIP.

FORMATO

SET BEEP [sonido] , [volumen]

Sonido	Cond.	Constantes, variables, variables de matriz, sus expresiones (numéricas); 1 ≤sonido <5.
Volumen	Cond.	Constantes, variables, variables de matriz, sus expresiones (numéricas); 1 ≤volumen <5.

FUNCION Y UTILIZACION

Hay cuatro sonidos diferentes disponibles.
Especificando el volumen 1 se selecciona el nivel de volumen más bajo, y especificando el volumen 4 se selecciona el nivel de volumen más alto.

SET DATE (set date)

Pone la fecha en el reloj interno.

Opuesta: GET DATE Afín: SET TIME, GET TIME

FORMATO

SET DATE T$ [,A]

T$ **Cond.** Constantes, variables, variables de matriz, sus expresiones (de cadena).

A **Omit.** Fecha actual.

FUNCION Y UTILIZACION

El formato es DD/MM/AA; DD es el número correspondiente al día, MM el número correspondiente al mes, y AA el número de dos dígitos correspondiente al año.

Cuando se especifica A se puede programar la alarma.

Para más instrucciones relativas a la forma de empleo de la función alarma, consultar el manual del ordenador.

Cuando se ejecuta SET TIME X$,A, se borrará la fecha seleccionada para la alarma. Para ajustar la fecha y la hora de disparo de la alarma, ejecutar en primer lugar SET TIME y después SET DATE.

Ejemplo de ejecución

```
10 T$="30/01/86"
20 SET DATE T$
30 PRINT T$
```

SET PAGE (set page)

Selecciona la página de escritura de datos y la página de presentación en pantalla cuando se opera con múltiples páginas de RAM de vídeo.

FORMATO

SET PAGE [página de visualización] [, página activa]

Página de visualización, página activa

Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤página <4.

FUNCION Y UTILIZACION

Válida únicamente para SCREEN 5 hasta SCREEN 8.
La página donde se escriben los datos recibe el nombre de **página activa** y la página que en un determinado momento está presentada en pantalla recibe el nombre de **página de visualización**.

SCREEN	Número de páginas	
	64 kB	128 kB
5	0—1	0—3
6	0—1	0—3
7	No utilizado	0—1
8	No utilizado	0—1

Ejemplo de ejecución

```
100 SCREEN 5
110 SET PAGE 0,1:CLS
120 LINE(70,50)-(170,150),,BF
130 SET PAGE 0,0
140 LINE(60,60)-(180,140),,BF
150 DP=0:AP=1
160 SET PAGE DP,AP:SWAP DP,AP
170 FOR I=0 TO 100:NEXT I
180 GOTO 160
```

SET PASSWORD (set password)

Selecciona la contraseña del sistema.

Afín: SET PROMPT, SET TITLE

FORMATO

SET PASSWORD "contraseña"

Contraseña **Cond.** Una cadena de hasta 255 caracteres.

FUNCION Y UTILIZACION

Si hay una contraseña seleccionada, el ordenador pedirá la introducción de esa contraseña al ser conectado. Si se introduce correctamente, el sistema funcionará con normalidad, pero si se introduce erróneamente, el sistema no pasará de ese punto y seguirá solicitando la introducción de la contraseña correcta. La selección de una contraseña impide que extraños pongan en marcha el sistema.

En caso de olvido de la contraseña, cabe la posibilidad de poner en marcha el sistema manteniendo pulsadas las teclas GRAPH y STOP mientras se pulsa el botón RESET.

Solo una de las tres órdenes SET PASSWORD, SET TITLE o SET PROMPT se puede ejecutar en un determinado momento. Si se ejecuta SET TITLE o SET PROMPT una vez que ha sido ejecutada SET PASSWORD, queda cancelada la función de la contraseña.

SET PROMPT (set prompt)

Especifica el mensaje (prompt) que se visualizará en pantalla cuando el BASIC entre en el estado de espera de comandos.

Afín: SET PASSWORD, SET TITLE

FORMATO

SET PROMPT "mensaje"

Indicación Cond. Una cadena de 6 caracteres o menos.

FUNCION Y UTILIZACION

Solo se puede ejecutar en un determinado momento una de las tres órdenes siguientes: SET PROMPT, SET PASSWORD o SET TITLE. Si tras la ejecución de SET PROMPT se ejecuta SET TITLE o SET PASSWORD, saldra de nuevo en pantalla el mensaje "OK".

SET SCREEN (set screen)

Registra los valores actuales seleccionados por las instrucciones SCREEN, COLOR, WIDTH y KEY ON/OFF como valores iniciales de puesta en marcha.

FORMATO

SET SCREEN

FUNCION Y UTILIZACION

Los valores que pueden seleccionarse para la inicialización son:

Screen (modo texto)
Anchura de la imagen (modo texto)
Color del primer plano
Color del fondo
Color del margen
Conmutador de las teclas de función
Sonido "click" de las teclas
Impresora
Velocidad de transmisión (en baudios)
Modo de visualización

SET TIME (set time)

Pone el reloj en hora.

Opuesta: GET TIME Afín: TIME, SET DATE, GET DATE

FORMATO

SET TIME X$ [,A]

X$ **Cond.** Constantes, variables, variables de matriz, sus expresiones (de cadena).
A **Omit** Hora actual.

FUNCION Y UTILIZACION

El formato es HH:MM:SS; HH es la hora, MM los minutos y SS los segundos.
Cuando se especifica A, se puede programar la alarma.
Para instrucciones relativas a la forma de empleo de la función alarma, consulte el manual del ordenador.
Cuando se ejecuta SET TIME X$ se borra la fecha de la alarma. Para poner la fecha y la hora de activación de la alarma, ejecutar en primer lugar SET TIME y luego SET DATE.

Ejemplo de ejecución

```
10 X$="11:00:00"
20 SET TIME X$
30 PRINT X$
```

SET TITLE (set title)

Selecciona el título a presentar en pantalla cuando se pone en marcha el ordenador.

Afín: SET PASSWORD, SET PROMPT

FORMATO

SET TITLE ["título"] , [color]

Título
- Cond. Una cadena de caracteres (6 o menos).
- Omit La cadena de caracteres actual.

Color
- Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant color < 5$.
- Omit El color actual.

FUNCION Y UTILIZACION

Selecciona un título, elegido por el usuario, que se visualizará en la pantalla cuando se ponga en marcha el ordenador.
Cuando se especifica una cadena de 6 caracteres, la imagen inicial de la pantalla continúa en pantalla hasta pulsar una tecla.
Sólo se puede ejecutar en un determinado momento una de las tres órdenes siguientes: SET TITLE, SET PROMPT o SET PASSWORD. Si se ejecuta SET PROMPT o SET PASSWORD una vez ejecutada una instrucción SET TITLE, el título seleccionado se borrará.

Ejemplo de ejecución

```
SET TITLE "SONY",1
```

SET VIDEO (set video)

Especifica los modos de trabajo relativos a superposición de imágenes de vídeo, mezcla de sonidos, etc.

FORMATO

SET VIDEO [modo] , [YM] , [CB] , [sinc] , [sonido] , [entrada vídeo], [control AV]

Modo	Cond.	Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤ modo < 4.
	Omit	0
YM	Cond.	Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤ YM < 2.
	Omit	0
CB	Cond.	Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤ CB < 2.
	Omit	0
Sinc	Cond.	Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤ sinc < 2.
	Omit	0
Sonido	Cond.	Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤ sonido < 4.
	Omit	0
Entrada vídeo	Cond.	Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤ entrada < 2.
	Omit	0
Control AV	Cond.	Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤ control < 2.
	Omit	1

FUNCION Y UTILIZACION

Modo

Modo	S1	S0	TP	Sinc	Función
0	0	0	0	Interno	Ordenador
1	0	1	1	Externo	Ordenador
2	0	1	0	Externo	Sobreimpresión
3	1	0	0	Externo	TV

(S1, S0 y TP son flags del registro VDP).

No se puede seleccionar el sinc externo en el modo 0.
En los modos 1–3 no se puede utilizar la salida compuesta del VDP, pero sí se puede utilizar la salida RGB analógica.

YM: 0: Brillo normal
1: Reducción del brillo de la TV a la mitad.
CB: 0: Bus de color del VDP en estado de salida.
1: Bus de color del VDP en estado de entrada.
Sinc: 0: Sinc interno.
1: Sinc externo.

Sonido

Sonido	Función
0	Señal de sonido externa no mezclada.
1	Señal de sonido externa del canal derecho mezclada.
2	Señal de sonido externa del canal izquierdo mezclada.
3	Señal de sonido externa de ambos canales mezclada.

Entrada de vídeo
0: Selecciona la entrada del multiconector de RGB.
1: Selecciona la entrada del conector de entrada de vídeo.

Control Audio/Vídeo Especifica la salida de señal de control del terminal de control AV del multiconector de RGB. Esta señal controla el circuito interno de la TV conectada al multiconector de RGB y sirve para superponer la imagen del ordenador sobre una imagen de vídeo.
0: Selecciona la señal TV como vídeo de fondo.
1: Selecciona una señal de vídeo externa, suministrada desde el ordenador, como vídeo de fondo.

Función SGN (sign)

Ofrece un 1 cuando el dato numérico es positivo, un 0 cuando es 0, y un −1 cuando es negativo.

FORMATO

SGN (X)

X **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas).

Valor obtenido: Tipo entero.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
10 INPUT A
20 IF SGN(A)=-1 THEN PRINT "Negativo"
30 GOTO 10
```

La línea 20 visualizará en pantalla "Negativo" únicamente cuando el valor asignado a A es negativo.

Función **SIN** (sine)

Ofrece el valor del seno de un dato numérico.

Afín: TAN, COS, ATN

FORMATO

SIN(X)

X **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas, en radianes).

Valor obtenido: Constantes en coma flotante comprendidas entre —1 y 1.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT SIN(3.14/3)
 .865759839492 39
PRINT SIN(60*3.14/180)
 .865759839492 39
```

● Para poner X en grados, utilizar la fórmula SEN ($X * \pi/180$).

SOUND (sound)

Genera efectos sonoros escribiendo datos directamente en el registro del PSG (Generador de Sonidos Programable).

Afín: BEEP, PLAY

FORMATO

SOUND número de registro, expresión

Número de registro — Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤ número < 14.

Expresión — Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas) comprendidas en la gama especificada para cada registro.

FUNCION Y UTILIZACION

Funciones de los registros del PSG

Registro No.	Función	Gama de datos
0	Frecuencia del canal A	0—255
1		0—15
2	Frecuencia del canal B	0—255
3		0—15
4	Frecuencia del canal C	0—255
5		0—15
6	Frecuencia de ruido	0—31
7	Selecciona un canal para generación de tonos y ruido.	0—63
8	Volumen canal A	0—15 Cuando se selecciona 16 hay variación del volumen.
9	Volumen canal B	
10	Volumen canal C	
11	Frecuencia del patrón de variación de volumen	0—255
12		0—255
13	Selección del patrón de variación de volumen	0—14

Determinación de la frecuencia del sonido

Las frecuencias generadas por los tres diferentes canales se determinan mediante seis registros, numerados del 0 al 5. Los datos escritos en los registros se pueden obtener con la siguiente fórmula:

$$\frac{1789772.5\ (\text{Hz})}{16 \times (\text{frecuencia de salida (Hz)})} = 256 \times (\text{datos de los registros 1, 3, 5}) + (\text{datos de los registros 0, 2, 4})$$

Por ejemplo, cuando se ha de generar un sonido de 300 Hz por el canal A, se cumple la siguiente expresión:

$$\frac{1789772,5}{16 \times 300} = 373 = 256 \times 1 + 117$$

Por lo tanto, hay que escribir un 117 en el registro 0 y un 1 en el registro 1.

Determinación de la frecuencia de ruído

En el registro 6, que determina la frecuencia de ruido, se pueden escribir datos comprendidos entre 0 y 31. El dato y la frecuencia están relacionados por la siguiente expresión:

$$\text{Valor del dato} = \frac{1789772,5\ (\text{Hz})}{16 \times \text{frecuencia de ruido (Hz)}}$$

Por ejemplo, cuando en el registro 6 se ha escrito el dato 15,

$$15 = \frac{1789772,5}{16 \times 7457}$$

Por lo tanto, la frecuencia de ruido es aproximadamente 7457 Hz.

Especificación del canal
El canal utilizado está determinado por el dato escrito en el registro 7.

Ruido			Sonido		
Canal C	B	A	C	B	A
32	16	8	4	2	1

Sumar los datos numéricos que corresponden al canal utilizado en base a la tabla anterior y restar el resultado de 63 para obtener el dato a escribir. Por ejemplo, cuando los canales A y B han de generar únicamente sonido y el canal C sonido y ruido, se cumple la siguiente expresión (24 es el dato a escribir).

$$63 - (32 + 4 + 2 + 1) = 24$$

Generación de sonido tras la determinación del volumen
Escribir los datos que determinan el volumen de los canales A, B y C en los registros 8, 9 y 10, respectivamente. Se pueden escribir datos comprendidos entre 0 y 15 (15 corresponde al volumen máximo).

Generación de efectos sonoros mediante patrones de variación de volumen
Con una instrucción SOUND se pueden poner en práctica funciones idénticas a las de los submandatos S y M de una instrucción PLAY. Los patrones de variación de volumen están determinados por los datos escritos en el registro 13, que es el mismo que corresponde a la especificación n del submandato S de una instrucción PLAY (Sn).
Para información relativa a los valores de n de los patrones correspondientes, ver la tabla de la página 162.

El ciclo de un patrón de variación de volumen está determinado por los datos escritos en los registros 11 y 12, para los cuales se cumple la siguiente expresión:

$$\frac{1789772{,}5\ (\text{Hz})}{256 \times \text{ciclo (Hz)}} = 256 \times (\text{dato registro 12}) + (\text{dato registro 11})$$

Por ejemplo, cuando el ciclo ha de ser 10 Hz, escribir 187 en el registro 11 y 2 en el registro 12, pues se cumple la siguiente expresión:

$$\frac{1789772{,}5}{256 \times 10} = 699 = 256 \times 2 + 187$$

Ejemplo de ejecución

```
10 SOUND 0,56
20 SOUND 1,1
30 SOUND 7,62
40 SOUND 8,8
```

Selecciona la frecuencia del canal A en 400 Hz

Selecciona un tono del canal A

Selecciona el volumen del canal A

La ejecución de este programa produce un sonido continuo de 400 Hz.
de 400 Hz.
Pulsar [CTRL] + [STOP] para detener el sonido.

Función SPACE$ (space dollar)

Ofrece un número arbitrario de espacios en forma de dato alfanumérico.

Afín: SPC

FORMATO

SPACE$ (N)

N **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant N < 256$.

Valor obtenido: Tipo cadena.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT SPACE$(5);"ABC"
     ABC
```

5 espacios

- Cuando N no es un valor entero, no se tienen en cuenta las cifras siguientes a la coma decimal.

Función SPC (space)

Ofrece un número arbitrario de espacios. Afín: SPACE$

FORMATO

SPC(N)

N **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant N < 256$.

Valor obtenido: Tipo cadena.

FUNCION Y UTILIZACION

La función SPC solo se puede utilizar en instrucciones PRINT y LPRINT.

Ejemplo de ejecución

```
PRINT "ABC";SPC(10);"DEF"
ABC          DEF
```

10 espacios

- Cuando N no es un valor entero, no se tienen en cuenta las cifras siguientes a la coma decimal.

SPRITE ON (sprite on)
SPRITE OFF (sprite off)
SPRITE STOP (sprite stop)

Valida, invalida o retiene una interrupción provocada por el choque de sprites.

Afín: PUT SPRITE

FORMATO

SPRITE ON — Valída interrupción
SPRITE OFF — Invalída interrupción
SPRITE STOP — Retiene interrupción

FUNCION Y UTILIZACION

Esta instrucción se utiliza para validar (SPRITE ON), invalidar (SPRITE OFF) o retener (SPRITE STOP) una interrupción tras la declaración de una interrupción provocada por el choque de sprites, (ON SPRITE GOSUB).

SPRITE$ (sprite dollar)

Define los datos de un sprite.

Afín: PUT SPRITE

FORMATO

SPRITE$ (número de sprite)

Número de sprite

Cond. Caso de 8 x 8 puntos ... Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤ número < 256.

Caso de 16 x 16 puntos ... Constantes, variables, variables matriz, sus expresiones (numéricas); 0 ≤ número < 64.

FUNCION Y UTILIZACION

Cuando una variable SPRITE$ define un modelo de sprite, este se mantiene con un número especificado. Para detalles relativos a la definición del sprite, ver PUT SPRITE.

Función **SQR** (square root)

Ofrece la raíz cuadrada de un dato numérico.

FORMATO

SQR (X)

X **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant X$.

Valor obtenido: Tipo numérico.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT SQR(100)
 10
```

Function STICK (stick)

Ofrece la dirección de las teclas del cursor y de los mandos para juegos. (Joysticks).

Afín: STRIG

FORMATO

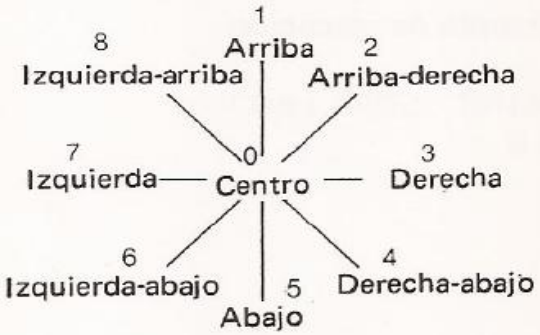
STICK (N)

N **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas).

Valor obtenido: Tipo entero.

FUNCION Y UTILIZACION

Cuando N = 0 da la dirección de las teclas del cursor, cuando N = 1 la del Joystick A, y cuando N = 2 la del Joystick B. La gama de valores correspondientes a la dirección va desde 0 a 8. Cuando no se pulsa ninguna tecla ni se activa ningún mando, da 0.



Ejemplo de ejecución

```
10 CLS
20 X=14
30 LOCATE X,10:PRINT " ";
40 D=STICK(0)
50 IF D=0 THEN LOCATE X,10:PRINT "*"
60 IF D=3 THEN X=X+1:IF X>28 THEN X=28
70 IF D=7 THEN X=X-1:IF X<0 THEN X=0
80 LOCATE X,10:PRINT "*";
90 GOTO 30
```

Este programa mueve un asterisco (*) a la izquierda y derecha de la pantalla mediante las teclas del cursor a derecha y a izquierda. El valor asignado a la variable D en la línea 40 depende de si la tecla está pulsada o no. Las líneas 50, 60 y 70 modifican la coordenada X, donde está ubicado el *, según un valor dado.

STOP (stop)

Interrumpe la ejecución del programa. Opuesta: CONT

FORMATO

STOP

FUNCION Y UTILIZACION

La ejecución de una instrucción STOP produce la interrupción de la ejecución del programa.

- Cuando se ejecuta una instrucción CON en modo directo, la ejecución se reanuda a partir de la instrucción siguiente a la interrumpida.

STOP ON (stop on)
STOP OFF (stop off)
STOP STOP (stop stop)

Valida, invalida o retiene una interrupción debida a la pulsación de las teclas [CTRL] + [STOP].

Afín: END, CONT

FORMATO

STOP ON	— Valída interrupción
STOP OFF	— Invalída interrupción
STOP STOP	— Retiene interrupción

FUNCION Y UTILIZACION

Ordenes que validan (STOP ON), invalidan (STOP OFF) o retienen (STOP STOP) una interrupción tras su declaración con la instrucción ON STOP GOSUB, (interrupción provocada al pulsar las teclas [CTRL] + [STOP]).

Función STRIG (stick trigger)

Ofrece un −1 cuando se pulsa la barra espaciadora o el botón de un dispositivo señalizador (Joystick, ratón, etc.), y un 0 en caso contrario.

Afín: STICK

FORMATO

STRIG (N)

N — Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant N < 5$.

Valor obtenido: Tipo entero.

FUNCION Y UTILIZACION

Ofrece el estado de la barra espaciadora cuando N = 0, el estado del botón del dispositivo señalizador A cuando N = 1, y el estado del botón del dispositivo señalizador B cuando N = 2. El valor resultante es 0 cuando no están pulsados y −1 cuando lo están.

Ejemplo de ejecución

```
10 CLS
20 COLOR ,C,C
30 IF STRIG(0)=0 THEN GOTO 20
40 C=C+1:IF C>15 THEN C=0
50 GOTO 20
```

Este programa cambia el color de la pantalla cada vez que se pulsa la barra de espacios.

Nota

Dispositivo señalizador A: Periférico conectado en el port del Joystick A.
Dispositivo señalizador B: Periférico conectado en el port del Joystick B.

STRIG ON (stick trigger on)
STRIG OFF (stick trigger off)
STRIG STOP (stick trigger stop)

Valida, invalida o retiene una interrupción provocada por la pulsación de la barra espaciadora o del botón de un dispositivo señalizador (Joystick, ratón, etc.).

Afín: STRIG

FORMATO

STRIG(N) ON — Valída interrupción
STRIG(N) OFF — Invalída interrupción
STRIG(N) STOP — Retiene interrupción

N **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant N < 5$.

FUNCION Y UTILIZACION

Especifica, según el valor de "N", la barra espaciadora o el botón del dispositivo señalizador A o B utilizados en una interrupción. El número de línea de la subrutina correspondiente, debe estar especificado por una instrucción ON STRIG GOSUB.

Valor de N	Especifica
0	Barra de espacio
1	Botón 1 del dispositivo A
2	Botón 2 del dispositivo A
3	Botón 1 del dispositivo B
4	Botón 2 del dispositivo B

```
STRIG(0) ON
```
Valída una interrupción de la barra espaciadora

```
STRIG(1) OFF
```
Valída una interrupción de botón 1 del dispositivo A

```
STRIG(2) STOP
```
Valída una interrupción de botón 1 del dispositivo B

Función STR$ (convert to string)

Convierte datos de tipo numérico en datos alfanuméricos.

Opuesta: VAL

FORMATO

STR$(X)

X **Cond.** Constantes, variables, variables de matriz, sus expresiones.
Valor obtenido: Tipo cadena.

FUNCION Y UTILIZACION

Cuando el dato numérico es negativo, el primer caracter de la cadena resultante es el—, y cuando es 0 o positivo, el primer caracter es un espacio en blanco.

Ejemplo de ejecución

```
10 X=100:Y=200
20 X$=STR$(X):Y$=STR$(Y)
30 PRINT X+Y
40 PRINT X$+Y$
RUN
 300
 100  200
 X$   Y$
```

Función STRING$ (string dollar)

Ofrece, en forma de datos alfanuméricos (de cadena), continuamente y el número especificado de veces, el caracter de un código de caracter dado o el de comienzo de una cadena.

Afín: CHR$

FORMATO

```
STRING$ (N, J)
STRING$ (N, X$)
```

N	Cond.	Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant N < 256$.
J	Cond.	Un código de caracter (ver la tabla de los códigos de caracteres en la página 258).
X$	Cond.	Constantes, variables, variables de matriz, sus expresiones (de cadena).

Valor obtenido: Tipo cadena.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT STRING$(10,70)
FFFFFFFFFF
PRINT STRING$(5,"ABC")
AAAAA
```

SWAP (swap)

Intercambia el valor de dos variables.

FORMATO

SWAP variable, variable

Variable **Cond.** Variables, variables de matriz numéricas o de cadena. Las dos variables deben ser del mismo tipo.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
10 A=3:B=5
20 SWAP A,B
30 PRINT "A=";A
40 PRINT "B=";B
RUN
A= 5
B= 3
```

Función TAB (tab)

Mueve el cursor hacia la derecha desde el principio de una línea el número de caracteres especificado.

FORMATO

TAB(N)

N **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas); $0 \leqslant N < 256$.

FUNCION Y UTILIZACION

La función TAB solo se puede utilizar en instrucciones PRINT o LPRINT. Cuando N es 0 corresponde al extremo izquierdo y cuando es un valor que coincide con el número de caracteres de una línea menos 1, corresponde al extremo derecho.

Ejemplo de ejecución

```
PRINT TAB(5);"AAA"
     AAA
```

5 espacios

Función TAN (tangent)

Ofrece el valor de la tangente de un dato numérico.

Opuesta: ANT Afín: COS, SIN

FORMATO

TAN (X)

X **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas, en radianes).

Valor obtenido: Constantes de coma flotante.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT TAN(3.14/3)
 1.72992922009
PRINT TAN(60*3.14/180)
 1.72992922009
```

● Para obtener X en grados, utilizar la fórmula TAN (X * π/180).

TIME (time)

Retiene el valor del temporizador incorporado. Afín: SET TIME

FORMATO

TIME
TIME = expression

Expresión **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ⩽expresión <65536.

FUNCION Y UTILIZACION

Esta variable, retiene el valor del temporizador incorporado, activado en el momento de inicializarse el BASIC, y suma una unidad cada 1/50 seg., en una gama que va desde 0 hasta 65536. Cuando llega a 65536, salta a 0 otra vez.
El valor de la variable se puede re-escribir mediante una instrucción LET. Cuando la CPU está en un estado de prohibición de interrupciones (por ejemplo, durante la entrada/salida de cinta cassette), el temporizador está parado. Cuando la alimentación está desconectada tampoco funciona el temporizador.

Ejemplo de ejecución

```
10 CLS:TIME=0
20 LOCATE 12,8:PRINT INT(TIME/50)
30 GOTO 20
```

Este programa presenta en pantalla continuamente el valor entero de TIME/50; el valor de TIME se divide por 50 tras hacer inicializar a cero la variable TIME. El número aumenta una unidad cada segundo, aproximadamente.

TROFF (trace off)

Desactiva TRON y detiene por lo tanto la presentación en pantalla de los números de línea ejecutados.

Opuesta: TRON

FORMATO

TROFF

FUNCION Y UTILIZACION

La ejecución de una instrucción TROFF en el modo directo o indirecto, mientras se ejecuta una instrucción TRON, anula la presentación de los números de línea en pantalla.

TRON (trace on)

Presenta en pantalla los números de línea ejecutados. Opuesta: TROFF

FORMATO

TRON

FUNCION Y UTILIZACION

La ejecución de la instrucción TRON, en el modo directo o indirecto, produce la presentación en pantalla en el modo texto y entre corchetes, del número de línea ejecutado a continuación.

- Cuando la pantalla está en modo gráfico, consecuencia de una instrucción SCREEN, no se visualizará el número de línea.

Ejemplo de ejecución

```
10 TRON
20 FOR I=0 TO 3
30 A=I+1:PRINT A
40 NEXT I
50 TROFF
RUN
[20][30] 1
[40][30] 2
[40][30] 3
[40][30] 4
[40][50]
```

Función **USR** (user)

Ofrece el resultado obtenido tras la ejecución de una subrutina en lenguaje máquina que comienza en una dirección definida por una instrucción DEFUSR.

Afín: DEFUSR

FORMATO

USR [X] (I)

X — **Cond.** Constantes de tipo entero; $0 \leqslant X < 9$.
Omit 0

I — **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas o de cadena).

Valor obtenido: Depende de la función de usuario definida.

FUNCION Y UTILIZACION

X es el número de la subrutina de usuario, concretamente el número especificado por DEFUSR. I es una constante o una variable que indica el valor a transferir desde el BASIC a la subrutina.

EJECUCION DE SUBRUTINAS EN LENGUAJE MAQUINA

Variable = USR N(I)

La subrutina en lenguaje máquina definida por el usuario se ejecuta mediante la instrucción anterior. Cuando esta subrutina ha sido ejecutada, el valor del resultado queda asignado a una variable, y se sigue ejecutando el programa BASIC.

Cuando la ejecución resulta transferida a una subrutina en lenguaje máquina, se da a esa subrutina el valor de "I" especificado como parámetro de una función USR.

X = USR 1 (I)

La instrucción anterior almacena el valor de la variable I en la siguiente posición de memoria, y, al mismo tiempo, introduce el dato que indica el tipo en el registro A, en función del tipo de I. La dirección inicial de la zona de memoria donde se almacena el valor de I es introducida en el registro HI.

Tipo de I	Datos introducidos al registro A*	Indicación de la dirección; registro HL	Dirección donde se almacena el valor de I
Entero	2	&HF7F6	&HF7F8—&HF7F9
Precisión simple	4		&HF7F6—&HF7F9
Precisión doble	8		&HF7F6—&HF7FD

*Introduce el mismo dato en la dirección de memoria &HF663.

Cuando I es una variable de cadena, el proceso es:

Datos introducidos al registro A	Datos introducidos en el registro DE	Descriptor de la cadena
3	Dirección inicial del descriptor de la cadena	1er byte: longitud de la cadena 2º y 3er bytes: Dirección inicial de la zona donde se almacena la cadena de caracteres.

Cuando la ejecución de la subrutina en lenguaje máquina ha llegado a su fín, el valor del resultado queda asignado a la variable X y los registros y la memoria quedarán de la siguiente forma:

Tipo del valor del resultado	Dirección de memoria &HF663	Registro DE	Registro HL	Dirección de almacenamiento del resultado
Entero	2		&HF7F6	&HF7F8—&HF7F9
Precisión simple	4		&HF7F6	&HF7F6—&HF7F9
Precisión doble	8		&HF7F6	&HF7F6—&HF7FD
Cadena	3	Dirección inicial del descriptor de la cadena		Dirección inicial del área de memoria indicada por el 2º y 3º byte del descriptor de la cadena.

Ejemplo de ejecución

```
DEFUSR0=&HE000
X=USR0(I)
```

Las instrucciones anteriores producen la ejecución de la subrutina que comienza en la dirección &HE000 y la asignación del resultado al BASIC.

Función **VAL** (value)

Ofrece un dato alfanumérico en forma de dato numérico. Afín: STR$

FORMATO

VAL (X$)

X$ **Cond.** Constantes, variables, variables de matriz, sus expresiones que expresan números (de cadena).

Valor obtenido: Tipo numérico.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
PRINT VAL("5")
 5
PRINT VAL("      5")
 5
```

Ignora el espacio anterior al dato de cadena

Función VARPTR (variable pointer) (1)

Ofrece la dirección inicial de memoria donde está almacenado el dato asignado a una variable.

FORMATO

VARPTR (variable)

Variable **Cond.** Constantes, variables, variables de matriz (numéricas y alfanuméricas).

Valor obtenido: Constantes de tipo entero; $-32768 \leqslant$ valor $\leqslant 32767$.

FUNCION Y UTILIZACION

Ofrece la dirección inicial (decimal) de la zona de memoria donde está almacenado un valor asignado a una variable dada. Si la dirección decimal es un número negativo, la dirección real es la que resulta de sumar a ese número 65536. La función VARPTR se utiliza, por ejemplo, en el caso de transferir una dirección de memoria con datos a una subrutina en lenguaje máquina.

Ejemplo de ejecución

```
10 A%=15
20 X=VARPTR(A%)
30 M$=HEX$(X):PRINT M$
RUN
802E
```

Este programa comprueba la dirección de memoria donde está almacenado el valor asignado a una variable (A%) y la presenta en pantalla tras su conversión al sistema hexadecimal.

- Antes de hacer uso de la función VARPTR, asignar un valor a la variable.

Función VARPTR (variable pointer) (2)

Ofrece la dirección inicial del bloque de control de ficheros asignado al fichero especificado.

FORMATO

VARPTR (# número de fichero)

Número de fichero **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas); 1 ⩽ número de fichero ⩽ número especificado en la instrucción MAXFILES

Valor obtenido: Tipo entero; −32768 ⩽ valor ⩽ −1.

FUNCION Y UTILIZACION

Ofrece la dirección inicial del bloque de control de ficheros asignado al fichero especificado. La dirección inicial de la zona de memoria (memoria intermedia de ficheros) especificada para entrada al fichero, está almacenada en el bloque de control de ficheros.

VDP (video display processor)

Lee o escribe el contenido de los registros del procesador de vídeo (VDP).

FORMATO

VDP (N)

N **Cond.** Constantes de tipo entero; $0 \leqslant N \leqslant 24$, $33 \leqslant N \leqslant 47$, $-9 \leqslant N \leqslant -1$.

FUNCION Y UTILIZACION

Sirve para leer o especificar el contenido de los registros del procesador de vídeo de un ordenador MSX.

N	Registro	Función
0—7	0—7	Sin especificar
8	Registro de estado 0	Sólo lectura
9—24	8—23	Sin especificar
33—47	32—46	Sin especificar
−1— −9	Registro de estado 1—9	Sólo lectura

PRECAUCION

Para realizar operaciones con una variable VDP y reescribir el valor de los registros del VDP, es necesario conocer adecuadamente el procesador de vídeo. Si se reescriben inadecuadamente los registros del VDP, no podrá conseguir la visualización normal de la pantalla.

Función VPEEK (video RAM peek)

Lee datos de la RAM vídeo. Afín: VPOKE

FORMATO

VPEEK (dirección)

Dirección **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤dirección <65536.

Valor obtenido: Tipo entero, 0 ≤valor ≤255.

FUNCION Y UTILIZACION

Ofrece el dato escrito en la dirección especificada de la RAM de vídeo. Para los modos SCREEN 5 a 8, suma la dirección inicial de la página activa a la dirección especificada para componer la dirección absoluta de la RAM de vídeo.

SCREEN	Página activa	Primera dirección
5, 6	0	0
	1	&H08000
	2	&H10000
	3	&H18000
7, 8	0	0
	1	&H10000

Ejemplo de ejecución

El siguiente programa presentará en pantalla el contenido de la dirección &H10200 (SCREEN 6).

```
10 SCREEN 6
20 SET PAGE ,2
30 A=VPEEK(&H200)
40 SCREEN 0
50 PRINT A
```

La línea 20 especifica la página 2 como página activa.
La línea 30 lee el contenido de la VRAM. La dirección absoluta de la VRAM es &H10200:

&H200 (Dirección VPEEK) + &H10000 (dirección inicial de la página activa).

VPOKE (video RAM poke)

Escribe un byte de datos en la RAM de vídeo. Afín: VPEEK

FORMATO

VPOKE dirección, expresión

Dirección Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤dirección ≤65536.

Expresión Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤expresión <256.

FUNCION Y UTILIZACION

Escribe el dato designado en la dirección especificada de la RAM de vídeo. Para los modos SCREEN 5 a 8, suma la dirección inicial de la página activa a la dirección especificada para componer la dirección absoluta de la RAM de vídeo.
Para la composición del mapa de direcciones de la RAM de vídeo, la dirección base de cada tabla se puede determinar con la función BASE. Por lo tanto, al hacer uso de la instrucción VPOKE, determine la dirección de la RAM vídeo con la función BASE.

WAIT (wait)

Espera hasta que la entrada del port E/S alcanza un valor determinado.

FORMATO

WAIT número de port, expresión 1 [,expresión 2]

Número de port, expresión 1, expresión 2

Cond. Constantes, variables, variables de matriz, sus expresiones (numéricas); 0 ≤número/expresión <256.

FUNCION Y UTILIZACION

Cuando se ejecute esta sentencia, los datos se introducirán a través del port de Entrada/Salida especificado, se realizará la operación XOR (suma lógica exclusiva) con el valor de la expresión 2, y se obtendrá el resultado final de la operación AND (producto lógico) del resultado anterior y el valor de la expresión 1. Si el valor del resultado final es 0, los datos procedentes del port de E/S se introducirán continuamente, y si es cualquier otro, se pasará a ejecutar la siguiente línea del programa. Cuando se omita la expresión 2, su valor se considerará cero.

WIDTH (width)

Especifica el número de caracteres por línea en el modo texto.

FORMATO

WIDTH número de caracteres

Número de caracteres Cond. SCREEN 0:
Constantes, variables, variables de matriz, sus expresiones; numéricas, 1 ≤ número < 81.
SCREEN 1:
Constantes, variables, variables de matriz, sus expresiones; numéricas, 1 ≤ número < 32.

FUNCION Y APLICACION

En el modo SCREEN 0, cuando el número de caracteres especificado es 40 o menos de 40, selecciona el modo texto de 40 caracteres x 24 líneas. Cuando es superior a 40, selecciona el modo texto de 80 caracteres x 24 líneas.

Ejemplo de ejecución

```
SCREEN 0
WIDTH 40
```

Estas sentencias, inicializan el modo texto SCREEN 0, con 40 caracteres por línea.

ORDENES ADICIONALES PARA USO DE DISKETTES

Los discos flexibles utilizados con el Disk BASIC, han de ser formateados antes de su utilización. En el MSX Disk-BASIC, el tipo de órdenes que sirven para realizar este tipo de operaciones, reciben el nombre de órdenes extendidas u órdenes adicionales. Se pueden ejecutar con la instrucción CALL.

FORMAT (format)

Formatea un disco.

FORMATO

CALL FORMAT
o
_FORMAT

FUNCION Y UTILIZACION

La operación de formateado escribe unos datos específicos en el disco en cumplimiento de un conjunto de reglas. El MSX Disk BASIC utiliza estos datos como "postes indicadores" en la búsqueda de los nombres de fichero y de los contenidos de los ficheros. Por lo tanto, es imprescindible formatear un disco virgen antes de utilizarlo por primera vez.

Cuando se ejecuta

CALL FORMAT

o _FORMAT

se visualiza en pantalla el mensaje

Drive name? (A,B) —¿Unidad de discos? (A,B)—

Especifique el nombre de la unidad (A o B) que contiene el disco a formatear introduciendo por el teclado una A o una B. En caso de disponer de una unidad de discos de una cara, se visualizará en pantalla el siguiente mensaje:

Strike a key when ready —Pulsar una tecla cuando esté listo—

Comprobar que el disco a formatear está en la unidad de discos especificada y luego pulsar cualquier tecla.
En caso de disponer de unidades de discos de dos caras, se visualizará en pantalla el siguiente mensaje:

1 — Single sided, 9 sectors	1 — una cara, 9 sectores
2 — Double sided, 9 sectores	2 — dos caras, 9 sectores

Para formatear un disco de una sola cara especificar un 1, y para formatear un disco de dos caras especificar 2. Una vez especificado el tipo de disco, se visualizará en pantalla el siguiente mensaje:

Strike a key when ready	Pulsar una tecla cuanto esté listo

Pulsar una tecla, exactamente igual que en el caso de disponer de una sola unidad de discos. El disco será formateado y luego saldrá el mensaje:

Format complete	Formateado completado
OK	

Y el Disk BASIC volverá al estado de espera de comandos.

- La operación de formateado se puede cancelar (con anterioridad al mensaje "Strike a key when ready") pulsando CTRL + STOP.

Formateado con una sola unidad de discos
Especificar unidad de discos A:. Si especifica B:, el ordenador dará por supuesto que la unidad de discos es la unidad B: y emitirá el siguiente mensaje:

Insert diskette for drive B:	Insertar disco en unidad B:
and strike a key when ready	y pulsar una tecla cuando esté listo

Aunque se haya indicado la unidad B:, se puede insertar un disco en la unidad que se dispone; quedará formateado según el proceso expuesto anteriormente.

SYSTEM

Devuelve el control al sistema operativo MSX.DOS.

FORMATO

CALL SYSTEM
o
_SYSTEM

FUNCION Y UTILIZACION

Devuelve el control al MSX-DOS después de cerrar todos los ficheros y borrar los programas y datos de la memoria.
Para pasar el control desde BASIC al sistema operativo MSX-DOS, es indispensable que anteriormente se haya pasado control desde MSX-DOS a BASIC y que la unidad de discos tenga insertado el diskette del sistema MSX-DOS. (MSXDOS.SYS y COMMAND.COM).
Para cargar el sistema operativo, introduzca el diskette MSX-DOS en la unidad de discos y pulse la tecla RESET del ordenador.

ORDENES COMPLEMENTARIAS PARA USO DE LA RAM–DISK

Para la inicialización y tratamiento de ficheros con la RAM DISK se necesitan órdenes BASIC adicionales. A tal efecto, se dispone de las siguientes órdenes.

MEMINI, MFILES, MKILL, MNAME

Para acceder a los ficheros de la RAM-Disk se utilizan las órdenes normales pero precedidas por MEM:. Las órdenes adicionales se ejecutan con la instrucción CALL.

MEMINI (memory disk initialize)

Asigna una zona de memoria a utilizar como Ram-Disk. Inicializa la Ram-Disk y borra todos los ficheros de la Ram-Disk.

FORMATO

CALL MEMINI [(tamaño)]
o
_MEMINI [(tamaño)]

Tamaño **Cond.** Constantes, variables, variables de matriz, sus expresiones (numéricas); 1023 ≤ tamaño < 32768.
Omit 32767.

FUNCION Y UTILIZACION

Se puede utilizar como RAM-Disk la parte de RAM comprendida entre las direcciones 0000H y 7FFFH.
Esta orden ha de ser ejecutada antes de utilizar la función RAM-Disk.
El tamaño especifica la cantidad de memoria de la RAM-Disk en bytes.
Si se ejecuta CALL MEMINI (0), quedará cancelada la función de la RAM-Disk y ésta dejará de ser utilizable.

Ejemplo de ejecución

Cuando se ejecuta

```
CALL MEMINI
```

or `_MEMINI`

Se visualizará el siguiente mensaje en la pantalla:

32000 bytes allocated

MFILES (memory disk files)

Visualiza en pantalla los nombres de todos los ficheros almacenados en la RAM-Disk.

FORMATO

CAL MFILES
o
_MFILES

MKILL (memory disk kill)

Borra un fichero especificado de la RAM-Disk.

FORMATO

CALL MKILL ("nombre de fichero [.extensión]")
o
_MKILL ("nombre de fichero [.extensión]")

Nombre de fichero	Cond.	Una cadena de 8 o menos caracteres.
Nombre del tipo	Cond.	Una cadena de 3 o menos caracteres.
	Omit	Una cadena nula.

FUNCION Y UTILIZACION

Ejemplo de ejecución

```
CALL MKILL ("TEST.BAS")

o

_MKILL ("TEST.BAS")
```

MNAME (memory disk name)

Cambia el nombre de un fichero de la RAM-Disk.

FORMATO

CALL MNAME ("nombre de fichero anterior [.extensión anterior]" AS "nuevo nombre de fichero [.nueva extensión]")
_MNAME ("nombre de fichero anterior [.extensión anterior]"AS "nuevo nombre de fichero [.nueva extensión]")

Nombre de fichero anterior, nuevo nombre de fichero

Cond. Una cadena de 8 o menos caracteres.

Extensión anterior, nueva extensión

Cond. Una cadena de 3 o menos caracteres.
Omit Una cadena nula.

Ejemplo de ejecución

```
CALL MNAME ("TEST.BAS" AS "SONY.BAS")
```

o

```
_MNAME ("TEST.BAS" AS "SONY.BAS")
```

MENSAJES DE ERROR

En caso de producirse un error, la ejecución del programa se detendrá. Se entrará en el estado de espera de comandos y se visualizará en pantalla un mensaje de error. El mensaje de error explica de una forma concisa la causa del error. A continuación se indican los mensajes de error y su significado (los números de los errores están encerrados entre paréntesis).

Bad FAT (60)

- El disco no está formateado.

Bad drive name (62)

- Se ha especificado como nombre de dispositivo una unidad de discos que no está en uso.

Bad file mode (61)

- Se ha utilizado una instrucción PUT, una instrucción GET o una función LOF con un fichero secuencial.
- Se ha ejecutado una instrucción LOAD con un fichero de acceso aleatorio.
- Se ha especificado un modo inadecuado en una instrucción OPEN.

Bad file name (56)

- El nombre del archivo es erróneo.
- Se ha especificado, mediante una sentencia OPEN, SAVE o LOAD, un nombre de dispositivo erróneo.

Bad file number (52)

- Se ha utilizado un número de archivo que supera la gama especificada en la instrucción MAXFILES.
- Se ha intentado ejecutar una instrucción PRINT # con un número de fichero no abierto.

Bad sector number (63)

- El número de registro especificado en una instrucción PUT o en una instrucción GET es 0 o mayor que 32767.

Can't CONTINUE (17)

- Se ha intentado reanudar un programa modificado tras una interrupción.
- No hay ningún programa en la memoria del ordenador.
- Se ha utilizado una instrucción CONT en un programa.

Devide I/O error (19)

- Carga imposibilitada por causa de la cinta o problemas del reproductor/grabador de cassettes.
- Nivel de volumen del cassette inadecuado.
- Orden interrumpida con anterioridad a la terminación de la carga.
- Error del dispositivo de entrada/salida.

Direct statement in file (57)

- Una instrucción de un programa ASCII en proceso de carga no tiene número de línea.
- Se ha intentado cargar un fichero que no está escrito en BASIC, por ejemplo un fichero de datos.

Disk full (66)

- El disco está lleno.

Disk I/O error (69)

- Se ha producido un "error fatal", en el proceso de entrada/salida de disco.

Disk offline (70)

- No hay disco en la unidad.

Disk write protected (68)

- Se ha intentado escribir en un disco protegido contra la escritura.

División by zero (11)

- Se ha intentado hacer una división por cero.
- Se ha intentado hacer una división por una variable indefinida.

Field overflow (50)

- El área especificada en la instrucción FIELD es superior a la longitud del registro.

File already exists (65)

- El nombre de fichero especificado como nombre de nuevo fichero en una instrucción NAME, ya existe en el disco.

File already open (54)

- Se ha intentado abrir un fichero que ya está abierto.
- Se ha intentado utilizar la orden KILL con un fichero abierto.

File not found (53)

- El fichero especificado en una instrucción LOAD, KILL o OPEN no existe en ese disco.

Fiel not OPEN (59)

- Se ha intentado ejecutar una instrucción PRINT # o INPUT # etc. con un fichero que no había sido abierto por la instrucción OPEN.

File still open (64)

- El fichero no ha sido cerrado.

Illegal direct (12)

- Se ha intentado ejecutar en el modo "mandato directo" una instrucción que solo puede ir dentro de un programa, por ejemplo una instrucción DEFFN.

Illegal function call (5)

- Se ha utilizado un valor erróneo en un comando.
- El valor de una función excede los límites de la gama permitida.

Input past end (55)

- Se ha intentado leer otra vez un fichero cuando ya han sido leídos todos los datos.
- No hay datos en el fichero.

Internal error (51)

- Hay un defecto en el intérprete BASIC.

Line buffer overflow (25)

- La memoria intermedia de entrada (buffer) está llena.

Missing operand (24)

- No hay ningún parámetro a continuación de un mandato.
- Los parámetros necesarios están incompletos.

NEXT without FOR (1)

- No hay ninguna instrucción FOR correspondiente a la instrucción NEXT ejecutada.
- La ejecución del programa se ha bifurcado al interior de un bucle FOR–NEXT a causa de una instrucción GOTO.

NO RESUME (21)

- Una rutina de procesamiento de errores no tiene ninguna instrucción RESUME. (Las rutinas de procesamiento de errores deben terminar con END, RESUME u ON ERROR GOTO O.)

Out of DATA (4)

- No hay instrucciones DATA o las que hay son insuficientes con respecto a la(s) instrucción(es) READ.

Out of memory (7)

- Programa demasiado largo.
- Demasiadas variables.
- Matriz demasiado larga.
- El programa contiene demasiadas instrucciones FOR—NEXT o GOSUB.

Out of string space (14)

- Area de caracteres rebasada.
- Area de caracteres especificada por una instrucción CLEAR demasiado pequeña.

Over flow (6)

- Un dato numérico o un resultado aritmético rebasa la gama permitida.
- Un parámetro de dirección rebasa los límites de la gama especificada.

Rename across disk (71)

- Se ha intentado la ejecución de un mandato NAME entre unidades de disco diferentes.

RAM disk full (66)

- Se ha utilizado toda la capacidad de la RAM-Disk. (En Disk-BASIC el mensaje emitido será "Disk full").

RAM offline (70)

- Se ha utilizado la RAM-Disk antes de ejecutar una instrucción CALL MEMINI. (En Disk-BASIC el mensaje emitido será "Disk offline").

RESUME without error (22)

- Hay una instrucción RESUME que no tiene su correspondiente instrucción ON ERROR.
- Transferencia a una rutina de procesamiento de errores por una instrucción GOTO.
- Como no hay ninguna instrucción END al final de una rutina principal, ejecuta continuamente una rutina de procesamiento de errores.

RETURN without GOSUB (3)

- Hay una instrucción RETURN que no tiene su correspondiente instrucción GOSUB.
- Transferencia a una subrutina por una instrucción GOTO.
- Como no hay ninguna instrucción END al final de una rutina principal, ejecuta continuamente una subrutina.

Redimensioned array (10)

- Se ha intentado definir matrices con el mismo nombre.
- Se han utilizado variables de matriz sin haber sido definidas mediante la instrucción DIM, y después se han definido.

Squential I/O only (58)

- Se ha utilizado una orden de fichero de acceso aleatorio para un fichero secuencial.

String formula too complex (16)

- Expresión de caracteres de una línea demasiado complicada.

String too long (15)

- Se ha asignado a una variable alfanumérica más de 255 caracteres.

Subscript out of range (9)

- Se ha utilizado un subíndice mayor que 11 para una variable matricial no declarada por una instrucción DIM.

Syntax error (2)

- Hay una instrucción que no cumple las reglas gramaticales del BASIC-MSX2.

Too many files (67)

- El número de ficheros del disco es superior a 112.
- El número de ficheros de la RAM-Disk es superior a 32.

Type mismatch (13)

- Se ha intentado asignar un valor numérico a una variable alfanumérica, o al revés..
- Se ha intentado efectuar una operación lógica con datos de cadena.
- El tipo de dato especificado por una función es erróneo.

Underfined line number (8)

- Se ha especificado un número de línea no existente en una instrucción GOTO, GOSUB o RESUME.
- En la ejecución de una instrucción RENUM, en el programa se había especificado una instrucción GOTO con un número de línea inexistente, etc...

Undefined use function (18)

- Se ha intentado utilizar una función de usuario no definida por una instrucción DEFFN.

Unprintable error (23, 26-49, 60-255)

- Se ha producido un error que no tiene asignado número de error.
- Se ha producido un error debido a que en una instrucción ERROR ha sido especificado el número de un error indefinido.

Verify error (20)

- El programa en cinta de cassette es diferente al programa residente en memoria.

APENDICE

CARACTERES

CARACTERES DEL BASIC-MSX2

El BASIC-MSX2 puede visualizar los caracteres mostrados en la siguiente tabla de caracteres.

Código hexadecimal	00—1F		20—3F		40—5F		60—7F	
	Código	Caracter	Código	Caracter	Código	Caracter	Código	Caracter
0	0	(nulo)	32	(espacio)	64	@	96	
1	1	☺	33	!	65	A	97	a
2	2	☻	34	"	66	B	98	b
3	3	♥	35	#	67	C	99	c
4	4	♦	36	$	68	D	100	d
5	5	♣	37	%	69	E	101	e
6	6	♠	38	&	70	F	102	f
7	7	•	39	'	71	G	103	g
8	8	■	40	(	72	H	104	h
9	9	○	41	)	73	I	105	i
A	10	▣	42	*	74	J	106	j
B	11	♂	43	+	75	K	107	k
C	12	♀	44	,	76	L	108	l
D	13	♪	45	—	77	M	109	m
E	14	♫	46	.	78	N	110	n
F	15	☼	47	/	79	O	111	o
0	16	►	48	0	80	P	112	p
1	17	◄	49	1	81	Q	113	q
2	18	↕	50	2	82	R	114	r
3	19	‼	51	3	83	S	115	s
4	20	¶	52	4	84	T	116	t
5	21	§	53	5	85	U	117	u
6	22	▬	54	6	86	V	118	v
7	23	↨	55	7	87	W	119	w
8	24	↑	56	8	88	X	120	x
9	25	↓	57	9	89	Y	121	y
A	26	→	58	:	90	Z	122	z
B	27	←	59	;	91	[	123	{
C	28	∟	60	<	92	\	124	|
D	29	↔	61	=	93	]	125	}
E	30	▲	62	>	94	^	126	~
F	31	▼	63	?	95	_	127	

Código hexadecimal	80—9F		A0—BF		C0—DF		E0—FF	
	Código	Caracter	Código	Caracter	Código	Caracter	Código	Caracter
0	128	Ç	160	á	192	─	224	α
1	129	ü	161	í	193	┴	225	β
2	130	é	162	ó	194	┬	226	Γ
3	131	â	163	ú	195	├	227	Π
4	132	ä	164	ñ	196	─	228	Σ
5	133	à	165	Ñ	197	┼	229	σ
6	134	å	166		198	╞	230	μ
7	135	ç	167		199	╟	231	γ
8	136	ê	168	¿	200	╚	232	Φ
9	137	ë	169		201	╔	233	Θ
A	138	è	170		202	╩	234	Ω
B	139	ï	171	½	203	╦	235	δ
C	140	î	172	¼	204	╠	236	∞
D	141	ì	173		205	═	237	Ø
E	142	Ä	174	«	206	╬	238	∈
F	143	Å	175	»	207	╧	239	∩
0	144	É	176	Ã	208	╨	240	≡
1	145	æ	177	ã	209	╤	241	±
2	146	Æ	178	Ĩ	210	╥	242	≥
3	147	ô	179	ĩ	211	╙	243	≤
4	148	ö	180	Õ	212	╘	244	
5	149	ò	181	õ	213	╒	245	
6	150	û	182	Ũ	214	╓	246	÷
7	151	ù	183	ũ	215	╫	247	≈
8	152	ÿ	184		216	╪	248	°
9	153	Ö	185		217	┘	249	•
A	154	Ü	186	¾	218	┌	250	·
B	155	¢	187		219	█	251	√
C	156	£	188	◇	220	▄	252	ⁿ
D	157	¥	189	‰	221	▌	253	²
E	158		190		222	▐	254	■
F	159		191	§	223	▀	255	

Caracteres cuyo código está constituido por dos bytes

Los caracteres de los códigos 1 al 31 (decimal) de la tabla anterior, tienen códigos de caracter de 2 bytes. Sus códigos mostrados en la tabla deben ir precedidos por el código 1 y los códigos listados en la tabla se deben sumar a 64 (decimal).

Entrada/salida de los códigos de caracter:

Entrada por el teclado:

Caracteres normales ... Entrada de un código de 1 byte.
Ejemplo: Código 65 para la "A".

Caracteres de códigos de 2 bytes ..... Se introduce 1 y el código correspondiente.
Ejemplo: Código 1 y 67 para el carácter " ♥"

Salida utilizando la función CHR$:

Caracteres normales ... Se utiliza como parámetro un código de 1 byte
Ejemplo: CHR$ (66) para la "B",

Caracteres de códigos de 2 bytes ..... Se utilizan 2 funciones CHR$; la primera es CHR$ (1) y la segunda es una función CHR$ que utiliza como parámetro uno de los códigos de la tabla anterior.
Ejemplo: CHR$(1); CHR$(68) para el " ♦"

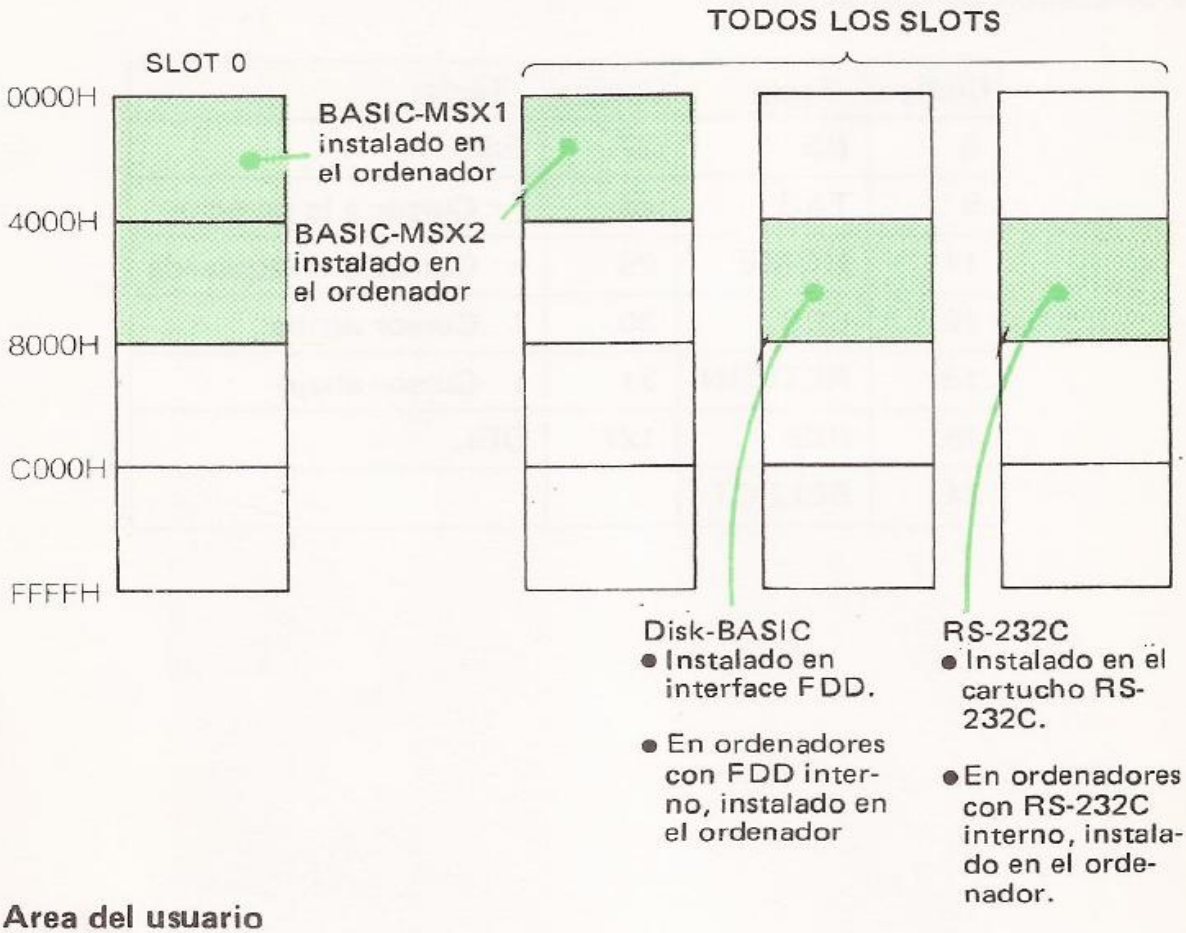
TABLA DE CODIGOS DE LAS TECLAS

En BASIC-MSX2 se pueden obtener los siguientes códigos al pulsar la tecla de edición correspondiente.

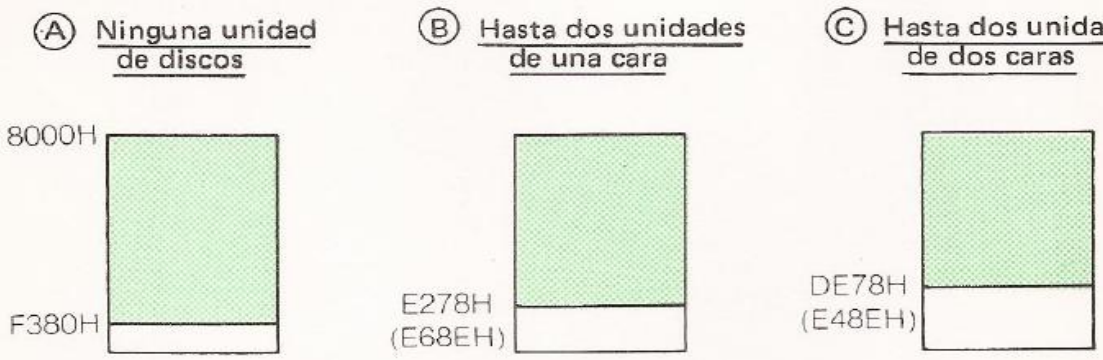
Código	Tecla	Código	Tecla
8	BS	27	ESC
9	TAB	28	→ Cursor à la derecha
11	HOME	29	← Cursor a la izquierda
12	CLS	30	↑ Cursor arriba
13	RETURN	31	↓ Cursor abajo
18	INS	127	DEL
24	SELECT		

MAPA DE MEMORIA

DIRECCIONES DE LA ROM-MSX2



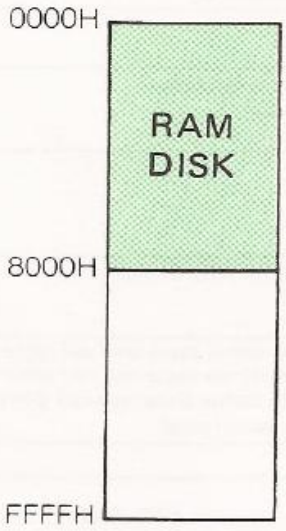
Area del usuario



- Ⓑ y Ⓒ se pueden utilizar como una unidad cuando se pulsa CTRL + RESET . En este caso, la dirección será la encerrada entre paréntesis.
- La dirección de Ⓑ y Ⓒ puede cambiar dependiendo del tipo de unidad de discos utilizada ó de una nueva versión del ordenador.
  La dirección del área de usuario se puede determinar así:
  J = &HFC4A:PRINT HEX$(PEEK(J + 1) * 256 + PEEK (J))
  La dirección de la posición inferior está almacenada en FC4AH y la superior en FC4BH.

DIRECCIONES DE LA RAM-DISK

En todos los SLOTS.



ASIGNACIONES DE LOS PORTS DE E/S

Utilización	No. Port (hex.)	Aplicación
VDP para adaptador MSX1	88 89 8A 8B	Lectura/escritura de datos Escritura dirección orden/lectura registro estado. Escritura registro de paleta de colores Escritura indirecta registro
Modem	8C 8D	— —
Impresora	90 91	Escritura: Bit 0: estroboscópico Lectura: Bit 1: estado Datos impresora
VDP	98 99 9A 9B	Igual que 88 Igual que 89 Igual que 8A Igual que 8B
PSG	A0 A1 A2	Registro de direcciones (escritura) Escritura de datos Lectura de datos
PPI	A8 A9 A8 A9	Lectura/escritura de datos para uso del port A (selección del slot de memoria) Lectura/escritura de datos para uso del port B (exploración del teclado) Lectura/escritura de datos para uso del port C Selección del modo (escritura)
MSX-ENGINE	AC:AF	—
Ampliación de memoria	B0 B1 B2 B3	Dirección A0-A7 Dirección A8-A10, control A13-A15 Dirección A11-A12, D0-D7 Selección del modo
Calendario/ reloj	B4 B5	Registro de direcciones Dato
Lapiz óptico	B8-BA BB	Lectura/escritura Solo escritura
VHD control	BC BD BE BF	Port A Port B Port C Selección del modo
MSX-Audio	C0-C1	—
Control del sistema	F5	b0 no utilizado b1 no utilizado b2 MSX-audio b3 superposición b4 MSX-interface b5 RS-232C b6 Lapiz óptico b7 Calendario/reloj (solo MSX1)
E/S bus color	F6	—
Control AV	F7	b0 Audio Derecho 0: mezcla b1 Audio Izquierdo 0: mezcla b2 Selección entrada vídeo 0: 21 pins b3 Captación entrada vídeo 0: sin entrada b4 Control AV 0: TV b5 Control Ym 0: TV b6 Control Ys 0: Super b7 Selección vídeo 0: TV
Mapeador de memoria	FC-FF	—

ORDENES DEL MSX–DOS

INTRODUCCION

El MSX-DOS es un sistema operativo de disco (DOS) para ordenadores MSX, que permite procesar ficheros, cargar y ejecutar programas y acceder a informaciones desde dispositivos periféricos como unidades de disco y teclados. Algunos programas llevan sus propias funciones de ejecución, pero la separación de las funciones con respecto a los programas ofrece dos ventajas importantes:
Una ventaja radica en el hecho de que utilizando un DOS para ejecutar programas no se necesita desperdiciar espacio de programa para listar funciones. Un DOS puede listar las funciones aparte, para que sean aplicables a cualquier programa.
Otra ventaja estriba en que un DOS es un interface que aporta compatibilidad entre diversos ordenadores y diferentes programas.
El MSX-DOS está constituido por los siguientes ficheros:

Nombre del fichero	Función del fichero
COMMAND.COM	Procesador de las órdenes del MSX-DOS
MSXDOS.SYS	Programa del sistema MSX-DOS

Una vez cargado, el MSX-DOS dispone la memoria de la siguiente forma:

- El sistema MSX-DOS necesita por lo menos 64K-bytes de memoria RAM.

CARGA DEL MSX–DOS

1 Insertar el disco del sistema operativo en la unidad de discos.
2 Conectar la alimentación de la unidad de discos, si es externa.
3 Conectar la alimentación del ordenador y de la pantalla.

- Si el ordenador es un ordenador MSX (versión 1.0), saldrá en pantalla el siguiente mensaje:

  MSX-DOS versión X.XX
  Copyright 1984 by Microsoft

  COMMAND versión X.XX

  Current date is XXX XX-XX-XX
  Enter new date

  Caso de no querer cambiar la fecha, pulsar la tecla RETURN. Para cambiar la fecha, introducir
  mes-día-año (por ejemplo, 9-25-1985)
  y pulsar la tecla RETURN

- En un ordenador MSX2 aparecerá el siguiente mensaje:

  MSX-DOS versión x.xx
  Copyright 1984 by Microsoft

  COMMAND versión x.xx

Después saldrá el indicador A > del MSX-DOS. Este indicador recibe el nombre de "prompt" del sistema operativo e indica que el MSX-DOS está inicializado y en espera de comandos.

COPIAS DE SEGURIDAD DEL DISCO DEL SISTEMA

Conviene hacer una copia de seguridad del disco del sistema MSX-DOS, por si el original se daña o los ficheros se borran accidentalmente.

1 Inicializar el MSX-DOS.
2 Formatear un disco virgen.
   ① Teclear FORMAT y pulsar RETURN. Saldrá en pantalla el siguiente mensaje:

   Drive name? (A,B)

   ② Seleccionar la unidad del formateado y pulsar las teclas A o B. (Pulsar la tecla A cuando solo se trabaja con una unidad de discos).

   Si se utiliza una unidad de discos de una cara, saldrá en pantalla el siguiente mensaje:

   Strike a key when ready

   Si se utiliza una unidad de discos de dos caras, saldrá en pantalla el siguiente mensaje al introducir A o B:

   1 — Single sided, 9 sectors
   2 — Double sided, 9 sectors

   Seleccionar 1 o 2 en función del tipo de disco. Entonces saldrá el siguiente mensaje:

   Strike a key when ready

   ③ Insertar el disco a formatear en la unidad, y pulsar cualquier tecla.

   Cuando la operación de formateo haya llegado a su fin, se visualizará en pantalla el siguiente mensaje:

   Format complete

3 Copiar el fichero MSXDOS.SYS
   Si se trabaja con dos unidades de disco, insertar el disco del sistema en la unidad A y el disco virgen formateado en la B. Teclear luego COPY A:MSXDOS.SYS B: y pulsar la tecla RETURN.

Si se trabaja con una sola unidad de disco, teclear COPY A: MSXDOS.SYS B: y pulsar la tecla RETURN antes de cambiar el disco de la unidad. Sacar luego el disco del sistema e insertar el disco virgen formateado tras la aparición en pantalla del siguiente mensaje:

```
Insert diskette for drive B:
and strike a key when ready.
```

Una vez realizada la copia, se visualizará el mensaje:

```
1 File copied
A>
```

4 Copiar el fichero COMMAND.COM.
Copiar el fichero COMMAND.COM siguiendo el proceso del paso 3, con la diferencia siguiente: teclear COPY A:COMMAND.COM B: en vez de COPY A:MSXDOS.SYS B:

- En los pasos 3 y 4 se puede teclear la orden COPY A: *.* B: Esta orden copia todos los ficheros del disco.
(Ver el resumen de órdenes de la orden "COPY").

ERRORES DE DISCO

Si se produce algún error, se visualizará el siguiente mensaje en la pantalla:

```
XXX error XXXXX drive X
Abort, Retry, Ignore?
```

(Por ejemplo, Disk error reading drive A
Abort, Retry, Ignore?)

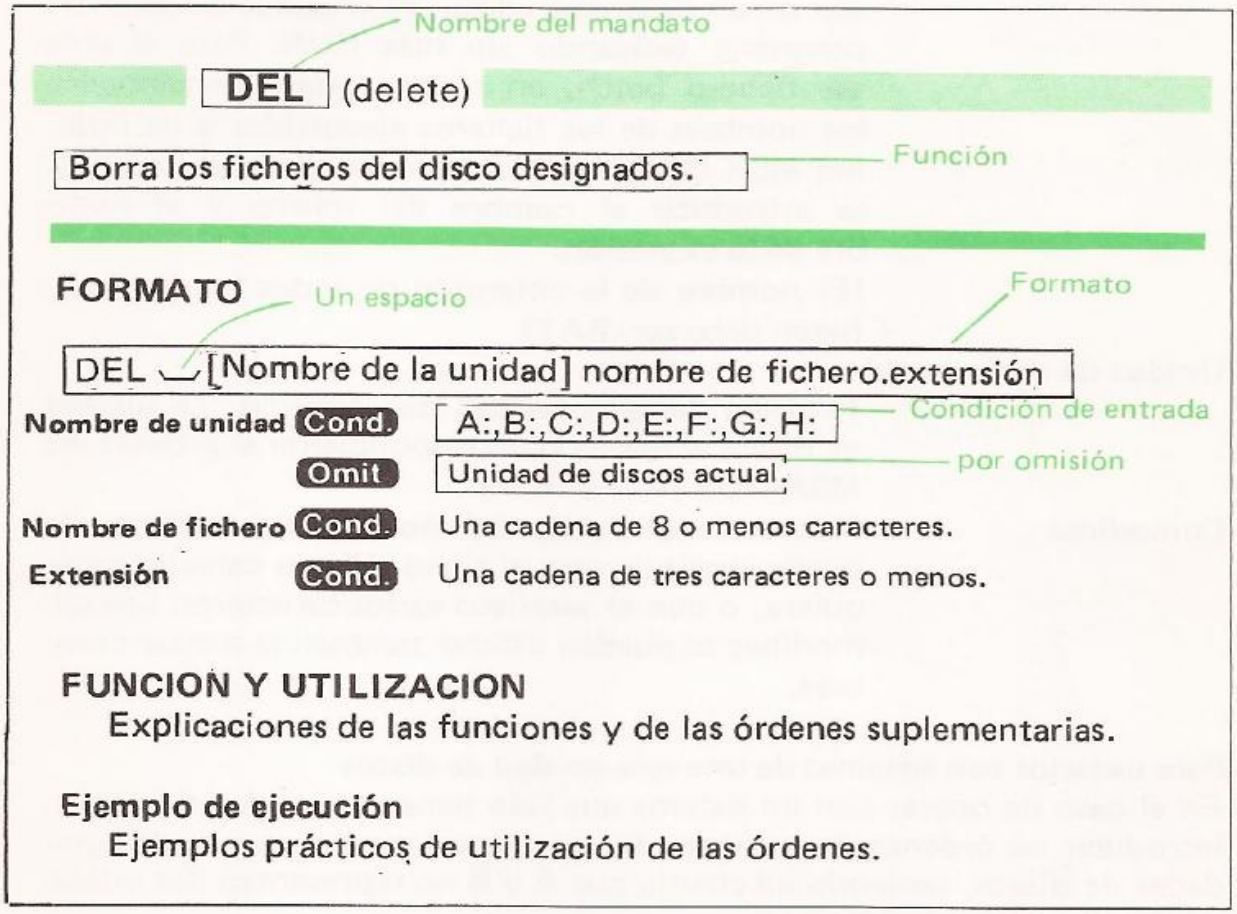
Pulsar A, R o I en respuesta a este mensaje. Normalmente se realizará otro intento pulsando R (hacer otro intento) o se pulsará A (poner fin al programa e intentarlo con otro disco). Si se introduce I, el programa ignorará el error.

DESCONEXION DEL SISTEMA

1 Sacar los discos de las unidades de disco.
2 Desconectar la alimentación del ordenador y de los periféricos.

ORDENES

El sistema MSX-DOS procesa 13 órdenes con el siguiente formato:



ORDEN ␣ [términos opcionales entre corchetes]

Hay que separar las órdenes de las opciones pulsando la tecla de espacios, la coma, el punto y coma, el signo igual o la tecla tab.

Nombre de la unidad	Es la designación de la unidad de disco.
Nombre del fichero	Es cualquier nombre válido para un fichero de disco, con exclusión de la extensión.
.Extensión	Es un nombre de la extensión constituido por un punto y un máximo de 3 caracteres

Ténganse en cuenta también los siguientes términos:

Fichero Batch — Un programa ejecutable MSX-DOS es diferente de un programa BASIC. No se puede escribir o editar un programa ejecutable con MSX-DOS sin hacer uso de un programa editor, ni se puede ejecutar un programa tecleando sin más RUN. Pero si crea un fichero batch, en el que están mencionados los nombres de los ficheros ejecutables y las órdenes MSX-DOS, se puede ejecutar el fichero con solo introducir el nombre del mismo y el nombre de la extensión.
(El nombre de la extensión de todos los ficheros-batch debe ser .BAT).

Unidad de disco actual — También llamada unidad por omisión. La unidad de discos actual es la correspondiente al prompt del MSX-DOS (A > a H >).

Comodines — Caracteres generales del nombre del fichero. Se puede sustituir con el signo "?" un caracter cualquiera, o con el asterisco varios caracteres. Los comodines se pueden utilizar también al teclear nombres.

Para usuarios con sistemas de una sola unidad de discos

En el caso de operar con un sistema que solo tiene una unidad de discos, introducir las órdenes de la misma forma que en los sistemas de dos unidades de discos, teniendo en cuenta que A y B no representan dos unidades de disco sino dos discos.
Si se designa la unidad B cuando el último disco utilizado era el "disco unidad A", el ordenador pedirá la inserción de un disco en la unidad B.
Si se designa la unidad A cuando el último disco utilizado era el "disco unidad B", el ordenador pedirá la inserción de un disco en la unidad A.

Durante la operación de copiado o ejecución de un fichero-batch, se visualizará en pantalla el siguiente mensaje:

Insert diskette for drive A (o B):
and strike a key when ready

Insertar el disco correspondiente en la unidad y pulsar cualquier tecla.

BASIC

Inicializa el MSX Disk BASIC a partir del MSX-DOS, carga el fichero BASIC designado, y lo ejecuta.

FORMATO

BASIC ␣ [[nombre de unidad]nombre de fichero [.extensión]]

Nombre de unidad	Cond.	A:, B:, C:, D:, E:, F:, G:, H:
	Omit	Unidad de disco actual.
Nombre de fichero	Cond.	Una cadena de 8 o menos caracteres.
	Omit	Inicializa el MSX Disk BASIC.
.Extensión	Cond.	Una cadena de 3 o menos caracteres.

FUNCION Y UTILIZACION

Cuando se designa un programa BASIC, carga y ejecuta automáticamente el programa.

- Esta orden activa la ROM del Disk BASIC. Los mapas de memoria del MSX-DOS y del MSX Disk BASIC son distintos.
- Para volver desde el Disk BASIC al MSX-DOS, teclear CALL SYSTEM y pulsar la tecla RETURN.

Ejemplo de ejecución

```
BASIC B:PROG4.BAS
```

Inicializa el MSX Disk BASIC, carga el fichero PROG4.BAS de la unidad B y ejecuta ese programa.

COPY (copy)

Copia uno o más ficheros en el mismo disco o en otro distinto. Se pueden asignar nuevos nombres a las copias.

FORMATO

COPY ␣ [nombre de unidad] nombre de fichero.extensión [+nombre de fichero.extensión...] ␣ [nombre de unidad] [nuevo nombre de fichero.extensión]

Nombre de unidad	Cond.	A:, B:, C:, D:, E:, F:, G:, H:
	Omit	Unidad de discos actual.
Nombre de fichero	Cond.	Una cadena de 8 o menos caracteres.
	Omit	El nombre de fichero original.
.Extensión	Cond.	Una cadena de 3 o menos caracteres.

FUNCION Y UTILIZACION

- Es imposible omitir a la vez el nombre de la segunda unidad y el nuevo nombre de fichero.
- Si no se introduce un nombre de la segunda unidad, copiará el fichero en el disco de la unidad actual.
- Si no se introduce un nuevo nombre de fichero, el fichero copiado tendrá el mismo nombre que el original.
- Se pueden utilizar comodines en vez del nombre de fichero y de la extensión.
- Se pueden unir dos o más ficheros en uno utilizando el signo + .

Ejemplos de ejecución

```
COPY ABC.DAT XYZ.PRS
```

Copia el fichero ABC.DAT del disco de la unidad actual en otro sitio del mismo disco, con el nombre XYZ.PRS.

```
COPY A:ABC.C+BCD.C B:ABCD.C
```

Crea un nuevo fichero formado por la unión de los ficheros ABC.C y BCD.C del disco de la unidad A, y lo graba en el disco de la unidad B con el nombre ABCD.C.

```
COPY A:*.COM B:
```

Copia en el disco de la unidad B todos los ficheros del disco de la unidad A que tengan la extensión, COM, y les da el mismo nombre que tenían.

DATE (date)

Visualiza y modifica la fecha del calendario interno.

FORMATO

DATE ␣ [mes-día-año]

Mes-día-año **Cond.** Mes: Enteros del 1 al 12.
Día: Enteros del 1 al 31.
Año: Enteros del 80 al 99 (no presenta el 19)
00 al 79 (no presenta el 20)
o de 1980 a 2079.

Omit Visualiza en pantalla la fecha del reloj del ordenador. Se puede cambiar introduciendo la nueva fecha.

FUNCION Y UTILIZACION

- El MSX-DOS está programado para cambiar correctamente la fecha, con independencia de que el año sea bisiesto o de que los meses tengan, 28, 30 o 31 días siempre que el ordenador incorpore un reloj alimentado por baterias.
- Las entradas del mes, el día y el año deben estar separadas por guiones, barras o puntos. Si no se hace así, saldrá en pantalla el siguiente mensaje.

```
Invalid date
Enter new date:
```

Este mensaje saldrá una y otra vez hasta que se introduzca la fecha correctamente.

- Si no quiere cambiar la fecha pulse RETURN.

Ejemplo de ejecución

```
DATE 9-25-1985
```

DEL (delete)

Borra del disco los ficheros designados.

FORMATO

DEL ␣ [nombre de unidad] nombre de fichero.extensión

Nombre de unidad	Cond.	A:, B:, C:, D:, E:, F:, G:, H:
	Omit	Unidad de disco actual.
Nombre de fichero	Cond.	Una cadena de 8 o menos caracteres.
.Extensión	Cond.	Una cadena de 3 o menos caracteres.

FUNCION Y UTILIZACION

Borra todos los ficheros designados. Se pueden utilizar comodines en el nombre de fichero y en la extensión.
Este mandato realiza la misma función que el mandato ERASE.

Ejemplo de ejecución

```
DEL A:TEST.DAT
```

Borra el fichero TEST.DAT del disco de la unidad A.

```
DEL A:TEST.*
```

Borra todos los ficheros de nombre TEST del disco de la unidad A con independencia de su extensión.

```
DEL A:*.*
```

Presenta el siguiente mensaje.

Are you sure? (Y/N)

Si se introduce Y, borrará todos los ficheros del disco de la unidad A.

DIR (directory)

Visualiza el directorio del disco.

FORMATO

DIR ␣ [nombre de unidad] [nombre de fichero.extensión] [/P] [/W]

Nombre de la unidad	Cond.	A:, B:, C:, D:, E:, F:, G:, H:
	Omit	Unidad de disco actual.
Nombre de fichero	Cond.	Una cadena de 8 o menos caracteres.
	Omit	Todos los ficheros del disco de la unidad designada.
.Extensión	Cond.	Una cadena de 3 o menos caracteres.
/P		Selecciona el modo Página.
/W		Selecciona Pantalla total.

FUNCION Y UTILIZACION

La orden DIR presenta en pantalla el nombre del fichero y su tamaño (No. de bytes), y la fecha de creación o última modificación del fichero. Si el ordenador dispone de reloj y está regulado a 36 o más caracteres por línea, presentará también la hora de creación o última alteración del fichero.
Se pueden utilizar comodines en los nombres de ficheros y en las extensiones.
En el modo página, la generación del listado se detiene cuando llena una pantalla. Para seguir, pulsar una tecla.
En el modo pantalla total, solo lista los nombres de fichero y las extensiones, ninguna información más. Presenta en cada línea todos los nombres de fichero.extensiones posibles.

Ejemplos de ejecución

`DIR` Lista todos los ficheros del disco de la unidad actual.

`DIR A:` Lista todos los ficheros del disco de la unidad A.

`DIR A:TEST.*`

Lista todos los ficheros del disco de la unidad A de nombre TEST, con independencia de su extensión.

`DIR A:TEST.DAT`

Busca el fichero de nombre TEST.DATA en el disco de la unidad A y presenta en pantalla la información.

ERASE (erase)

Borra del disco el fichero designado.

FORMATO

ERASE ␣ [nombre de unidad] nombre de fichero.extensión

Nombre de unidad	Cond.	A:, B:, C:, D:, E:, F:, G:, H:
	Omit	Unidad de disco actual.
Nombre de fichero	Cond.	Una cadena de 8 o menos caracteres.
.Extensión	Cond.	Una cadena de 3 o menos caracteres.

FUNCION Y UTILIZACION

Borra todos los ficheros designados. Se pueden utilizar comodines en el nombre de fichero y en la extensión. Este mandato realiza la misma función que el mandato DEL.

Ejemplos de ejecución

```
ERASE A:TEST.DAT
```

Borra el fichero de nombre TEST.DATA del disco de la unidad A.

```
ERASE A:TEST.*
```

Borra todos los ficheros de nombre TEST del disco de la unidad A, con independencia de su extensión.

```
ERASE A:*.*
```

Presenta el siguiente mensaje:

Are you sure? (Y/N)

Si pulsa Y, borrará todos los ficheros del disco de la unidad A.

FORMAT (format)

Formatea un disco preparándolo para aceptar ficheros MSX-DOS o MSX Disk BASIC.

FORMATO

FORMAT

FUNCION Y UTILIZACION

Los discos vírgenes se deben formatear antes de usarlos. Los discos formateados se pueden utilizar indistintamente para MSX-DOS o para MSX Disk BASIC, porque los formatos son iguales.

Con unidades de disco de una cara

① Ejecutar
FORMAT
Saldrá en pantalla el mensaje:
Drive name? (A,B)

② Seleccionar la unidad de discos y pulsar A o B. (Si se trabaja con una sola unidad, pulsar la tecla A).
Saldrá en pantalla el mensaje:
Strike a key when ready

③ Insertar el disco a formatear en la unidad A o B y pulsar cualquier tecla.

Con unidades de discos de dos caras

① Tras introducir el nombre de unidad, sale en pantalla el mensaje:

1 — Single sided, 9 sectors
2 — Double sided, 9 sectors

② Introducir un 1 para formatear discos de una cara y un 2 para formatear discos de dos caras. Se visualizará entonces el siguiente mensaje:

Strike a key when ready

③ Insertar el disco a formatear en la unidad A o B y pulsar cualquier tecla.

Una vez formateado el diskette, aparecerá el siguiente mensaje:

Format complete

- Cuando se formatea un disco se borra todo su contenido.
- Con anterioridad al comienzo de la operación de formateado, cabe la posibilidad de cancelar la orden FORMAT pulsando simultáneamente las teclas CTRL y STOP. Esta acción no tiene ningún efecto una vez comenzada la operación de formateado.

MODE (mode)

Selecciona el número de caracteres por línea.

FORMATO

MODE ␣ número

Número Cond. Enteros del 1 al 40 en ordenadores MSX.
Enteros del 1 al 80 en ordenadores MSX2.

FUNCION Y UTILIZACION

El formato inicial es de 29 caracteres por línea en el modo de pantalla 1 con ordenadores MSX, y de 80 caracteres por línea en el modo pantalla 0 con ordenadores MSX2. Si el número está comprendido entre 1 y 32, selecciona el modo de pantalla 1, y si no selecciona el modo de pantalla 0.

Ejemplo de ejecución

MODE 35

PAUSE (pause)

Suspende la ejecución del fichero-batch.

FORMATO

PAUSE ␣ [comentario]

FUNCION Y UTILIZACION

PAUSE suspende la ejecución de un fichero-batch hasta que se pulse una tecla cualquiera, y así da tiempo a cambiar de disco o a realizar cualquier otra operación. Cuando el procesador de comandos lee PAUSE, visualiza en pantalla el siguiente mensaje:

Strike a key when ready

Si pulsa la tecla C mientras se mantiene pulsada la tecla CTRL, saldrá en pantalla el siguiente mensaje:

Terminate batch file (Y/N)?

Si se introduce Y queda abortada la ejecución del fichero-batch y el control vuelve al sistema operativo. Si se introduce N, se reanuda la ejecución del fichero-batch.

Si pulsa cualquier otra tecla a parte de CTRL y C, se reasume la ejecución del fichero.

- Al hacer la programación del fichero-batch puede introducir un comentario en la línea PAUSE para dar las instrucciones correspondientes al usuario. El comentario saldrá en pantalla con anterioridad al mensaje "Strike a key when ready".

REM (remark)

Visualiza un comentario durante la ejecución de un fichero batch.

FORMATO

REM␣[comentario]

FUNCION Y UTILIZACION

El comentario de la orden REM se visualiza en la pantalla durante la ejecución de un fichero batch.

- Separar REM del comentario propiamente dicho pulsando la barra de espacios, la tecla TAB o la tecla de la coma.

REN (rename)

Cambiar el nombre de fichero y/o la extensión por otro distinto.

FORMATO

REN ␣ [nombre de unidad] nombre de fichero.extensión ␣ nuevo nombre de fichero.extensión

Nombre de unidad	Cond.	A:, B:, C:, D:, E:, F:, G:, H:
	Omit	Unidad de disco actual.
Nombre de fichero	Cond.	Una cadena de 8 o menos caracteres.
.Extensión	Cond.	Una cadena de 3 o menos caracteres.

FUNCION Y UTILIZACION

Se pueden utilizar comodines en el nombre de fichero y en la extensión. Si se utilizan comodines en el nuevo nombre de fichero y/o en la nueva extensión, no cambian los caracteres originales que deberían sustituir.

Ejemplos de ejecución

```
REN *.LST *.PRN
```
Cambia todas las extensiones .LST por .PRN.

```
REN B:SUN ?O?
```
Cambia el nombre de fichero SUN de la unidad B por SON. El fichero sigue en la unidad B.

TIME (time)

Visualiza y modifica la hora del reloj interno.

FORMATO

TIME ␣ [hora [:minutos [; segundos]]]

Hora; Minutos: Segundos Cond. Hora: Enteros del 0 al 23.
Minutos Enteros del 0 al 59.
Segundos Enteros del 0 al 59.

Omit Visualiza en pantalla la hora que tiene el ordenador. Se puede cambiar introduciendo la nueva hora

FUNCION Y UTILIZACION

- Las entradas de la hora, los minutos y los segundos deben ir separadas por el signo dos puntos. Si no se hace así, saldrá en pantalla el mensaie:

  Invalid time
  Enter new time:

- Este mensaje aparecerá repetidamente hasta que se introduzca la hora correctamente.
- Si se introduce solo la hora o la hora y los minutos, en los términos omitidos queda, por omisión 00.
- Si no se quiere cambiar la hora pulse RETURN .
  El reloj del ordenador comenzará a contar cuando pulse . RETURN .
- La orden TIME es inefectiva si el ordenador no tiene reloj interno alimentado por baterias.

Ejemplos de ejecución

```
TIME 3:06:22
```

TYPE (type)

Visualiza en pantalla el contenido de los ficheros designados.

FORMATO

TYPE ␣ [nombre de unidad] nombre de fichero.extensión

Nombre de unidad	Cond.	A:, B:, C:, D:, E:, F:, G:, H:
	Omit	Unidad de disco actual.
Nombre de fichero	Cond.	Una cadena de 8 o menos caracteres.
.Extensión	Cond.	Una cadena de 3 o menos caracteres.

FUNCION Y UTILIZACION

Utilice la orden DIR para localizar un fichero y la orden TYPE para examinar el contenido del fichero sin hacer modificaciones.

- Esta orden no se puede utilizar con ficheros binarios como xxxx.COM, pues están escritos en lenguaje máquina.

Ejemplo de ejecución

```
TYPE A:ROME.C
```

MANUAL DE REFERENCIA PARA PROGRAMACION BASIC-MSX Versión 2.0

Sony Corporation
Editado por Sony España, S.A. 1986