TOSHIBA

MSX

ORDENADOR DOMESTICO MANUAL DE REFERENCIA DEL MSX BASIC

ORDENADOR DOMESTICO TOSHIBA MSX
MANUAL DE REFERENCIA DEL MSX BASIC
AVISO

1. Toshiba se reserva el derecho de introducir cambios en este manual sin notificación previa.

2. Al usar programas de aplicación especiales o procedimientos de cálculo en el MSX, se recomienda comprobar detenidamente la secuencia de ejecución y los resultados intermedios y finales.

3. Declinamos toda responsabilidad por pérdidas financieras, de beneficios o de otro tipo que pudieran resultar del uso del ordenador.

PREFACIO

En este Manual se describe detenidamente el MSX BASIC, el lenguaje de programación para el ordenador doméstico TOSHIBA MSX. Este manual está concebido como fácil referencia de las definiciones y descripciones de la sintaxis, comandos, instrucciones y funciones del MSX BASIC.

La operación del ordenador se explica en el "Manual del Usuario del Ordenador Doméstico TOSHIBA".

Para una exposición más detallada de las técnicas de programación y como información introductoria, lea los manuales escritos sobre el lenguaje MSX BASIC.

INDICE

CAPITULO 1

SINTAXIS

I INTRODUCCION AL MSX BASIC

1. EN QUE CONSISTE EL MSX BASIC

El MSX BASIC es un lenguaje estándar potente y versátil, concebido para ofrecer compatibilidad a nivel de programa entre diferentes ordenadores MSX. Con el MSX BASIC puede adquirirse una gran variedad de paquetes de software para uso con el ordenador MSX, y usted puede intercambiar programas con amigos que posean otros ordenadores MSX.
El MSX BASIC es una versión ampliada del BASIC Standard, versión 4.5, desarrollada por MICROSOFT CORPORATION.

2. EL LENGUAJE BASIC

El lenguaje BASIC está compuesto por comandos, instrucciones y funciones; sin embargo, no puede establecerse una delimitación rigurosa entre comandos e instrucciones.

- **Comandos** . . . . . Se usan para controlar la entrada, ejecución, edición y proceso de programas, principalmente en el modo Directo.
- **Instrucciones** . . . . . Son los elementos básicos de todo programa BASIC e indican al ordenador lo que tiene que hacer; normalmente van precedidas por un número de línea.
- **Funciones** . . . . . Permiten realizar operaciones aritméticas o manipular cadenas numéricas o de caracteres; se usan en las instrucciones.

3. PROGRAMAS BASIC

Todo programa BASIC es una serie de instrucciones y comandos, dispuestos en el orden correcto para dar solución a un problema específico o realizar una tarea determinada. El ordenador ejecuta las instrucciones y comandos, por regla general, en orden ascendente de números de línea.

(1) Línea de programa

Cada línea de programa consiste en un número de línea y una o varias instrucciones o comandos. Es el elemento descriptivo más simple de un programa BASIC.

n n n n instrucción **Ejemplo** 20 PRINT "a"
(n n n n = número de línea)

- Una línea de programa puede contener hasta 255 caracteres, incluyendo el número de línea.
- El número de línea ha de ser un entero, entre 0 y 65529.

(2) Multiinstruccione

Cada línea de programa está constituida normalmente por sólo una instrucción o comando; sin embargo, si usted desea reducir el número de líneas, puede incluir varias instrucciones o comandos en una misma línea, separándolos con dos puntos (:).
nnnn instrucción: instrucción: instrucción **Ejemplo** 20 INPUT A$: PRINT A$
Sin embargo, no es posible continuar de esta forma una línea de programa después de una de las siguientes instrucciones:

REM, END, RETURN, GOTO

Si, después de alguna de estas palabras clave, se especitica otra instrucción, la misma será ignorada.

4. MODOS DE EJECUCION

(1) Modo Comando

Cuando el ordenador está en modo Comando (en estado de espera), ejecutará los comandos e instrucciones o almacenará las líneas de programa que se mecanografíen en el teclado. El ordenador adopta el modo Comando (en la pantalla aparece "Ok") cuando se conecta y cuando se interrumpe o finaliza la ejecución de un programa.

(2) Modo Directo

Si usted escribe en el teclado un comando o una instrucción no precedida de número de línea cuando el ordenador está en modo Comando; el ordenador lo ejecutará inmediatamente después de que usted pulse la tecla [RETURN] . Este tipo de ejecución se llama modo Directo.

(3) Modo Programa

Si usted escribe instrucciones o comandos después de un número de línea, el ordenador los interpreta como líneas de programa y los almacena en su memoria interna para ulterior ejecución.

La ejecución de un programa cargado en la memoria interna puede iniciarse con el comando RUN o las instrucciones GOTO o GOSUB.

Este tipo de ejecución se denomina modo Programa.

(4) Paro o pausa de la ejecución

Pulsando simultáneamente las teclas [CTRL] y [STOP] se interrumpe la ejecución del programa, sea ésta en modo Directo o modo Programa, y el ordenador se restituye al modo Comando. Si se pulsa la tecla [STOP] durante la ejecución en modo Programa, se produce una pausa en la ejecución durante la cual sólo la tecla [STOP] permanece operativa. Para reanudar la ejecución, pulse dicha tecla una segunda vez; si, en cambio, desea terminar la ejecución y volver al modo comando, pulse simultáneamente [CTRL] [STOP] durante la pausa.

5. JUEGO DE CARACTERES UTILIZABLES EN BASIC

Los caracteres que pueden usarse en programas BASIC son: letras mayúsculas y minúsculas, símbolos alfabéticos, números y caracteres gráficos. Vea más detalles en la "Tabla de Código de Caracteres", en el Capítulo 3, Sección 1.

Además de su significado explicado al tratar las diversas instrucciones, los siguientes signos desempeñan las funciones descritas a continuación:

+	(más)	Operador aritmético para suma, o signo positivo.
−	(menos)	Operador aritmético para resta, signo negativo o símbolo indicativo de gama (en LIST y otras instrucciones).
*	(asterisco)	Operador aritmético de multiplicación.
/	(barra)	Operador aritmético de división.
\	(barra inversa)	Operador aritmético indicativo de cociente entero.
^	(exponenciación)	Operador aritmético de exponenciación.
=	(igual)	Operador relacional de igualdad, o símbolo de asignación.
>	(mayor que)	Operador relacional indicativo de "mayor que".
<	(menor que)	Operador relacional indicativo de "menor que".
(	(paréntesis izquierdo)	Símbolo que ayuda a especificar el orden de prioridad de operaciones aritméticas.
)	(paréntesis derecho)	Símbolo que ayuda a especificar el orden de prioridad de operaciones aritméticas.
%	(porcentaje)	Símbolo de declaración del tipo de variable, para tipo entero.
!	(signo de exclamación)	Símbolo de declaración del tipo de variable, para tipo real de simple precisión.
#	(N°)	Símbolo de declaración del tipo de variable, para tipo real de doble precisión.
$	(dólar)	Símbolo de declaración del tipo de variable, para tipo de caracteres.
&	(y comercial)	Usada en &H, &O, &B, para representar constantes hexadecimales, octales o binarias respectivamente.
.	(punto)	La última línea a ejecutar de BASIC (usado instrucciones AUTO, LIST o LLIST), o punto decimal.
:	(dos puntos)	Separador en líneas de multiinstrucciones.
;	(punto y coma)	Separador de variables o datos, en instrucciones (instrucciones PRINT, etc.)
,	(coma)	Separador de variables o datos, en instrucciones (instrucciones PRINT, DATA, etc.)
?	(signo de interrogación)	Usado en lugar de una instrucción PRINT.
'	(apóstrofo)	Denota un comentario (usado en lugar de REM).
"	(comillas)	Símbolo de especificación de una constante de caracteres.
	(espacio)	Espacio en blanco.

II PROGRAMACION

PROCEDIMIENTO DE PROGRAMACIÓN

Para escribir programas en MSX BASIC deberá ejecutar generalmente las siguientes operaciones (vea detalles del uso del teclado en el "Manual del Usuario del Ordenador Doméstico Toshiba"):

① Mecanografiado del programma
② Comprobación del programa
③ Edición del programa
④ Ejecución del programa
⑤ Corrección del programa
⑥ Almacenamiento del programa

[Ejemplo]
El siguiente programa muestra en pantalla el producto (C) de la multiplicación de dos valores numéricos A y B:

```
10 INPUT A
20 INPUT B
30 C=A*B
40 PRINT C
50 END
```

1. MECANOGRAFIADO DEL PROGRAMA

① Para entrar un nuevo programa en el ordenador, borre antes cualquier programa que pueda haber en la memoria interna, ejecutando el comando NEW.
② Mecanografíe el programa línea por línea, cada una precedida de su número de línea, por orden ascendente.
③ Al terminar de mecanografiar cada línea, pulse la tecla [RETURN] (si no pulsa dicha tecla, la línea de programa mecanografiada no se almacenará en la memoria interna).

[Ejemplo]
Las teclas que debe pulsar para entrar el anterior programa son:

[N][E][W][RETURN]
[1][0][ ][I][N][P][U][T][ ][A][RETURN]
[2][0][ ][I][N][P][U][T][ ][B][RETURN]
[3][0][ ][C][=][A][*][B][RETURN]
[4][0][ ][P][R][I][N][T][ ][C][RETURN]
[5][0][ ][E][N][D][RETURN]

En vez de mecanografiar los números de línea, puede generarlos automáticamente con el comando AUTO, ejecutándolo antes de empezar a escribir el programa. Puede ejecutar el comando AUTO inmediatemente después del comando NEW.

Escriba: [A][U][T][O][RETURN]
o: [F2][RETURN]

Para cancelar la función del comando AUTO, pulse simultáneamente las teclas [CTRL] y [STOP].

2. COMPROBACION DEL PROGRAMA

① Cuando haya completado la entrada del programa, use el comando LIST para que la secuencia de líneas aparezca en la pantalla, y compruebe que cada línea esté correctamente mecanografiada.
Si dispone de una impresora acoplada a su ordenador MSX, puede listar el programa con el comando LLIST.

Para visualizar el programa, pulse: [L][I][S][T][RETURN]
o: or [F4][RETURN]
Para imprimir el programa, pulse: [L][L][I][S][T][RETURN]

② Si la totalidad del programa no cabe en la pantalla, especifique una gama de líneas. Para más detalles, consulte la descripción de los comandos LIST y LLIST en el Capítulo 2, "DESCRIPCION DEL LENGUAJE".

[Ejemplo]
Pulse [F4][RETURN] para exhibir el programa y compruebe cada línea.

En la pantalla aparecerá:

```
LIST
10  INPUT A
20  INPUT B
30  C=A*B
40  PRINT C
50  END
Ok
■
```

3. EDICION DEL PROGRAMA

El contenido de cualquier línea de programa puede corregirse situando el cursor en el punto deseado y utilizando las funciones de edición.

① Use el comando LIST para poner en pantalla la parte del programa que contiene la línea a corregir.
② Sitúe el cursor en la posición en que se halla el carácter a modificar.
③ Realice la operación de edición deseada, como por ejemplo borrar, insertar o modificar.
④ Con el cursor todavía en la línea acabada de editar, pulse la tecla [RETURN].

Nota 1: Una operación de edición debe terminarse siempre pulsando [RETURN] en la línea acabada de editar.

Nota 2: Si usted cambia un número de línea y pulsa la tecla [RETURN], el nuevo número de línea se añade al programa, pero el viejo número de línea y los caracteres que le siguen continúan intactos en la memoria.

Objetivo	Operación de edición	Ejemplo de edición
Corrección de un carácter en una línea en pantalla. (Ejemplo) Cambiar PRIMT a PRINT)	① Use las teclas de movimiento del cursor ([⇨][⇦][⇧][⇩]) para situar el cursor en el carácter a corregir. ② Pulse el carácter correcto. ③ Pulse la tecla [RETURN].	PRIMT■ Usted quiere sustituir la M por una N (■ designa al cursor) PRIMT Sitúe el cursor en la M con [⇦]. PRINT En la pantalla aparece el carácter correcto (N) y el cursor pasa a la siguiente posición.
Borrado de un carácter en una línea en pantalla. (Ejemplo) Borrado de X de PRIXNT para obtener PRINT.	① Use las teclas de movimiento del cursor para situar éste en el carácter a borrar. ② Pulse la tecla [DEL]. ③ Pulse la tecla [RETURN].	PRIXNT■ Usted desea borrar la X. PRIXNT Sitúe el cursor en la X. PRINT Pulse la tecla [DEL].

Objetivo	Operación de edición	Ejemplo de edición
Inserción de un carácter en una línea en pantalla. (Ejemplo) Cambiar PRNT a PRINT insertando una I.	① Use las teclas de movimiento del cursor para situar éste en el punto en que desea insertar un carácter. ② Pulse la tecla [INS]. ③ Pulse el carácter a insertar. ④ Pulse de nuevo [INS]. ⑤ Pulse la tecla [RETURN].	PRNT■ Falta la letra I. PR**N**T Use [←] para situar el cursor. PRNT Pulse [INS] para seleccionar el modo inserción. PRINT Aparece la I y el cursor avanza una posición. PRI**N**T Pulse de nuevo [INS] para cancelar el modo de inserción.
Corrección de una línea de programa completa.	① Mecanografíe la línea de programa correcta usando el mismo número de línea que la línea a sustituir. ② Pulse la tecla [RETURN]. La nueva línea sustituye a la anterior.	Escriba la línea correcta como por ejemplo 40 PRINT C [RETURN].
Borrado de una línea completa.	① Escriba el número de línea de la línea a borrar. ② Pulse la tecla [RETURN].	Escriba, por ejemplo, 10 [RETURN], para borrar la línea nº 10.
Borrado de varias líneas de programa.	Ejecute el comando DELETE.	Para borrar las líneas de programa de la 20 a la 60, escriba: DELETE 20—60 [RETURN].
Inserción de una nueva línea entre otras dos (numeradas n y m).	Mecanografíe un número de línea, cuyo valor esté entre n y m, y el contenido de la línea.	
Renumeración de líneas de programa.	Ejecute el comando RENUM.	Para renumerar las líneas de un programa con incrementos de 10 a partir de 100, escriba: RENUM 100, 10 [RETURN].

4. EJECUCION DE UN PROGRAMA

① Completada la edición de un programa, use las teclas de movimiento del cursor ([⇨] [⇦] [⇧] [⇩]) para situar el cursor en una fila en que no hay ninguna línea de programa, o pulse simultáneamente las teclas [SHIFT] y [HOME] para borrar la pantalla; estas teclas dejan la pantalla en blanco, pero no afectan el contenido de la memoria interna.

② Use el comando RUN para empezar la ejecución del programa.

Escriba: [R] [U] [N] [RETURN]
o: [F5]

③ Para interrumpir la ejecución del programa, pulse simultáneamente las teclas [CTRL] y [STOP].

[Ejemplo]
Determine el producto de la multiplicación de dos números A y B, suponiendo que A=5 y B=4.

Pantalla	Teclas a pulsar
(programa para calcular el producto de dos números)	Pulse simultáneamente [SHIFT] y [HOME].
■(CURSOR)	Pulse [F5].
RUN ? ■	Pulse [5] y luego [RETURN].
? 5 ? ■	Pulse [4] y luego [RETURN].
? 4 20	(En la pantalla aparece el producto de A por B)

Cuando se complete la ejecución del programa representado como ejemplo, en la pantalla aparecerá:

```
RUN
? 5
? 4
 20
Ok
■
```

5. CORRECCION DE ERRORES

- **Error en modo Directo**

① Cuando ocurre un error en modo Directo, el ordenador se restituye al modo Comando, y en la pantalla aparece el pertinente mensaje de error.

Ejemplo: **Syntax error** (mensaje de error indicativo de sintaxis errónea)

② La mayoría de los errores que ocurren en el modo Directo se deben a una entrada de comando errónea.
③ Entre de nuevo el comando, con la sintaxis correcta.

- **Error en modo Programa**

① Si ocurre un error en modo Programa, la ejecución se interrumpe y el ordenador se restituye al modo Comando, con el pertinente mensaje en pantalla, seguido del número de línea en que se ha producido el error.

Ejemplo: **Syntax error** (Error de sintaxis) **in** **10** (Número de línea)

② Localice el error usando el mensaje de error como guía. El significado de los mensajes de error se explica en la "Lista de Códigos de Error", Capítulo 3, Sección 6.
③ Cuando haya localizado el error, corríjalo como se ha explicado en la Sección 3, "Edición del Programa".

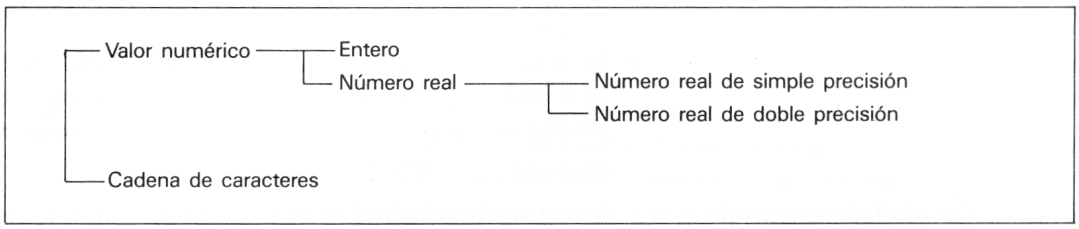
6. ALMACENAMIENTO DEL PROGRAMA

① Grabe el programa en una cinta cassette usando el comando CSAVE o SAVE antes de desconectar el ordenador, ya que al desconectarlo se borra el contenido de la memoria interna.
② Los programas almacenados en cinta cassette pueden volverse a cargar en el ordenador usando los comandos CLOAD, LOAD o MERGE.
③ La conexión del registrador de cassette al ordenador y el procedimiento de grabación de describen en el "Manual del Usuario" del Ordenador Doméstico Toshiba.

III CONSTANTES Y VARIABLES

1. TIPOS DE DATOS

BASIC manipula los valores numéricos y caracteres como datos. Los valores numéricos se clasifican en los siguientes tipos:



• Valores numéricos

Los valores numéricos que pueden usarse en programas BASIC son: enteros, números reales positivos o negativos y cero.

- **Enteros**
  Entre −32768 y +32767
- **Números reales**
  Pueden ser de simple o doble precisión:
  Número real de simple precisión:

  Un número real con seis cifras significativas, entre −9.99999E+62 y+9.99999+62. La gama válida de exponentes es desde E+62 a E−64. El valor positivo mínimo que puede expresarse con un número real de precisión simple es 1.0E−64.
  Número real de doble precisión:

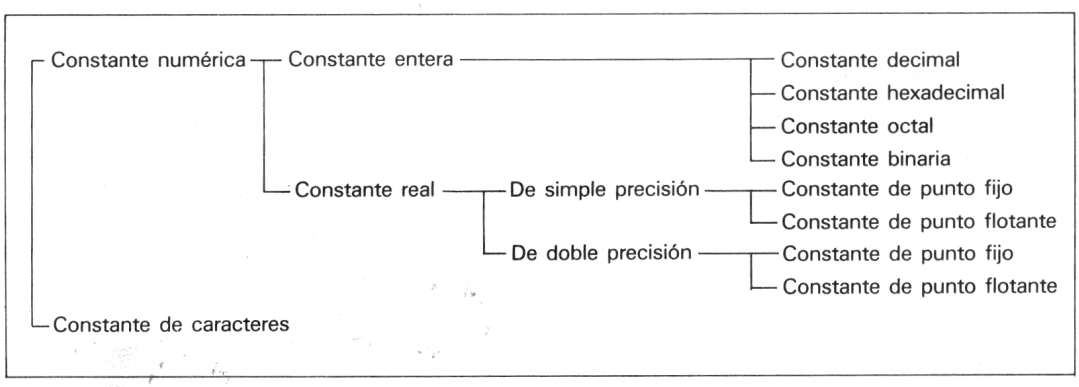
  Un número real con 14 cifras significativas, entre −9.9999999999999E+62 y +9.9999999999999E+62. La gama válida del exponente es desde E+62 hasta E−64. El valor positivo mínimo que puede expresarse con un número real de doble precisión es 1.0E−64.

• Cadena de caracteres:

Una cadena de hasta 255 caracteres.

2. CONSTANTES

Las constantes son valores o datos fijos e invariables usados en programas. Pueden clasificarse en los siguientes tipos:



(1) CONSTANTES NUMERICAS

Las constantes numéricas negativas han de llevar obligatoriamente el prefijo "−"; las constantes numéricas positivas no necesitan, en cambio, el prefijo "+".

① Constantes enteras

Las constantes enteras pueden expresarse en notación decimal, hexadecimal, octal o binaria.

Constante decimal:

- Un valor entero en notación decimal entre −32768 y +32767.
- Un entero o número real entre −32768 y +32767 que lleva el sufijo "%". En los números reales, los puestos de decimales se truncan, es decir se omiten sin redondeado.

(Ejemplo) 123 −567 12.7%

Constante hexadecimal:

- Un número hexadecimal se expresa con los caracteres 0−9 y A−F, con el prefijo &H, y puede oscilar entre &H0 y &HFFFF.
  Las letras mayúsculas A-F corresponden a los números 10 al 15 de la presentación decimal.
- Los valores hexadecimales &H0 a &H7FFF equivalen de 0 a 32767 en decimal; de la misma manera, &H8000 hasta &HFFFF equivalen de −32768 hasta −1 en decimal.

(Ejemplo) &HFF 255 en decimal
&HFFFE −2 en decimal

Constante octal:

- Un número octal se expresa con los caracteres O al 7, con el prefijo &O, y puede oscilar entre &O0 y &O177777.
- Los números octales &O0 hasta &O77777 corresponden de 0 a 32767 en decimal; los octales &O100000 hasta &O177777 corresponden de −32768 hasta −1 en decimal.

(Ejemplo) &O377 255 en decimal
&O177776 −2 en decimal

Constante binaria:

- Un número binario se expresa mediante ceros y unos con el prefijo &B, y puede oscilar entre &B0 y & B1111111111111111 (16 unos)
- Los números binarios &B0 hasta & B111111111111111 (15 unos) corresponden de 0 a 32767 en decimal; &B1000000000000000 hasta & B1111111111111111 (16 unos) corresponden de −32768 hasta −1 en decimal.

(Ejemplo) &B11111111 255 en decimal
&B1111111111111110 −2 en decimal

② Constantes reales de simple precisión

Constante real de punto fijo:

- Un número real compuesto de seis o menos cifras significativas. Un número real con más de seis cifras significativas constituye un número real de doble precisión.
- Un número real o un entero, que llevan como sufijo un signo de exclamación (!). Posee siete o más cifras significativas, y la séptima se redondea.

(Ejemplo) 9.87 0.0000345
44.449868691 (la séptima cifra significativa se redondea, resultando 44.4499).

Constante real de punto flotante:

- Consiste en una mantisa de seis o menos cifras significativas y un exponente representado por la letra E. La gama válida va desde −9.99999E+62 hasta +9.99999E+62 y la gama válida del exponente es desde+62 hasta −64. Cuando un valor de punto flotante posee más de seis cifras significativas en su mantisa, constituye una constante real de doble precisión con punto flotante.

  (Ejemplo) 1.234 E +23
  Mantisa Exponente

③ **Constantes reales de doble precisión**

Constante real de punto fijo:

- Un número real de 7 a 14 cifras significativas. Si un número real contiene 15 o más cifras significativas, la 15ª se redondea; el número real resultante contiene 14 cifras significativas.
- Un número real o un entero que lleven el sufijo "#". Si un número real contiene 15 o más cifras significativas, la 15ª se redondea; el número real resultante contiene 14 cifras significativas.

  (Ejemplo) 9.873333345
  23445555555500
  44.449#

Constante real de punto flotante:

- Consiste en una mantisa de 7 a 14 cifras significativas y un exponente representado con el uso de la letra E. La gama disponible va de −9.9999999999999E+62 a 9.9999999999999E+62. La gama válida del exponente va desde +62 a −64.

  (Ejemplo) 1.2345678E+23
  Mantisa Exponente

  Cuando el exponente es indicado por una "D" en vez de una "E", la constante real de punto flotante es un número real de doble precisión, aun cuando su mantisa posea 6 o menos cifras significativas. La gama válida es entre −9.9999999999999D+62 y +9.9999999999999D62. La gama válida del exponente va de +62 a −64.

  (Ejemplo) 1.23D+23
  Mantisa Exponente

(2) CONSTANTES DE CARACTERES

Una constante de caracteres consiste en una cadena de hasta 255 caracteres entrecomillados.

- Un par de comillas ("") sin nada dentro se considera una constante vacía y se trata como constante de caracteres.
- Un par de comillas con uno o más espacios en blanco dentro representa una constante de caracteres "espacio en blanco" y se diferencia de una constante vacía.
- Una cadena numérica entrecomillada, como por ejemplo "123", no es tratada como valor numérico, sino como constante de caracteres, por lo que no puede usarse en operaciones aritméticas.

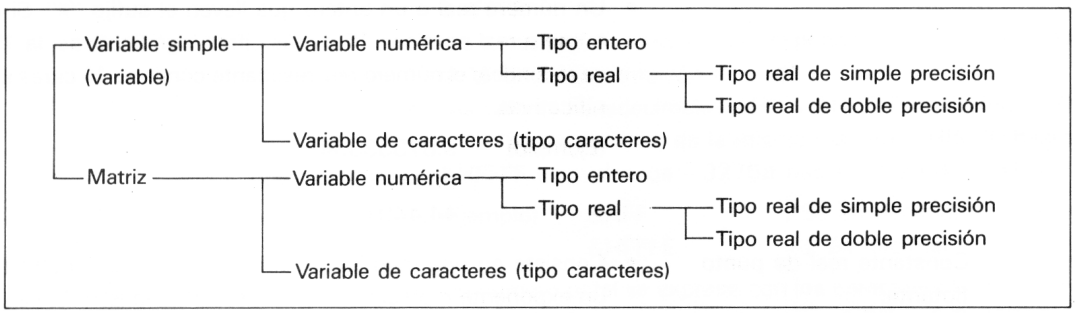
  (Ejemplo) "TOSHIBA"

3. VARIABLES

Una variable es una posición de memoria usada para alojar un dato que puede variar y que se denomina "valor de la variable". Cada variable posee su propio nombre, compuesto por un carácter o un grupo de caracteres, que sirve para hacer referencia a su valor. Usted puede asignar un valor a una variable ejecutando una instrucción LET, INPUT o READ (con ello se reserva para la variable una posición de memoria y se coloca en ella el valor especificado).
Si se hace referencia a una variable antes de que se le haya asignado un valor, éste será cero si es una variable numérica, y una constante vacía si se trata de una variable de caracteres.

(1) TIPOS DE VARIABLES

Las variables se clasifican en los siguientes tipos, según los datos que contienen. A cada tipo de variable puede asignarse sólo el mismo tipo específico de datos.
Una variable capaz de recibir sólo un valor, se denomina "variable" o "variable simple". Cuando puede asignársele más de un valor, se denomina "matriz" o "tabla de variables".



(2) NOMBRES DE VARIABLES

Cada nombre de variable se especifica con uno o dos caracteres alfanuméricos y un símbolo de declaración de tipo. A continuación se exponen algunas reglas para la elección de nombres de variables.

① El primer carácter de un nombre de variable ha de ser un carácter alfabético. El segundo puede ser un carácter alfabético o numérico.

Ejemplo: A3=2 El valor 2 se asigna a la variable A3.
3A=2 3A no es considerado como variable.

② Si se usan más de dos caracteres para designar una variable, sólo los dos primeros son válidos, ignorándose el resto.

Ejemplo: A3BC=2 El valor 2 es asignado a la variable A3.

③ Todos los símbolos gráficos o espacios en blanco usados en variables son ignorados.

Ejemplo: A☺3=2 El valor 2 es asignado a la variable A3.
A 3=2 El valor 2 es asignado a la variable A3.

④ Los caracteres alfabéticos mecanografiados en minúscula para designar una variable se convierten mayúsculas.

Ejemplo: a3=2 El valor 2 se asigna a la variable A3.

⑤ Las palabras reservadas (tales como nombres de comandos, instrucciones, funciones u operadores BASIC) no pueden usarse como nombres de variables

Ejemplo: AUTO3=2 El nombre "AUTO3" no puede usarse como nombre de variable, ya que contiene la palabra reservada "AUTO".

⑥ Para especificar un tipo de variable dado debe añadirse a su nombre uno de los siguientes símbolos de declaración de tipo:

% para variables tipo entero
! para variables tipo real de simple precisión
# o ningún símbolo para variables tipo real de doble precisión
$ para variables de caracteres

- Variables con idéntico nombre, pero símbolo de tipo distinto, son consideradas por el ordenador como variables diferentes.

  Ejemplo: A%, A! y A son tres variables diferentes.

- Cuando un nombre de variable no lleva ningún sufijo declarador de tipo, es considerada por el ordenador como variable real de doble precisión. Cuando en un programa se declara un tipo de variable con las instrucciones DEFINT (define integer), DEFSNG (define single), DEFDBL (define double) o DEFSTR (define string), la variable en cuestión se considera como perteneciente al tipo así especificado.

  Ejemplo: A# es la misma variable que A. Si en un programa se especifica "DEFINT A" dicha variable A será la misma que A%.

(3) TABLAS DE VARIABLES

Ciertas series de datos pueden manipularse más fácilmente usando variables con subíndices, dispuestas en una tabla, también llamada matriz.

Formato:		
	Nombre de variable (subíndice)	Tabla de variable unidimensional
	Nombre de variable (subíndice, subíndice)	Tabla de variables bidimensional
	Nombre de variable (subíndice, subíndice, subíndice...)	Tabla de variables de n dimensiones

(n números)

① La dimensión de una tabla de variables se representa con el número de subíndices entre paréntesis (y separados por comas) que siguen al nombre de la variable.
Se dispone de hasta 255 dimensiones.

② Los subíndices pueden oscilar entre cero y el máximo número de posiciones de memoria disponibles.

③ El tamaño (número de elementos) y la dimensión de una tabla de variables se especifica con una instrucción DIM (dimensión).

Ejemplo: 10 DIM A (5, 2)

Esta instrucción DIM declara una tabla de variables de dos dimensiones denominada A, en la que pueden usarse hasta (5+1)*(2+1)=18 variables con subíndice.

- Es posible usar tablas de variables de hasta tres dimensiones sin declararlas en instrucciones DIM. En tal caso, por cada dimensión se reservan automáticamente posiciones de memoria para once variables (0—10).

(4) AREA DE MEMORIA REQUERIDA POR CADA TIPO DE VARIABLE

Variable	Variable simple	Tabla de variables
Tipo entero	5	5+2*(número de elementos)+2*(dimensión)+1
Tipo real de simple precisión	7	5+4*(número de elementos)+2*(dimensión)+1
Tipo real de doble precisión	11	5+8*(número de elementos)+2*(dimensión)+1
Cadena de caracteres	6+(número de caracteres en la cadena)	5+3*(número de elementos)+2*(dimensión)+1+(número total de caracteres en las cadenas contenidas en los elementos)

(5) VARIABLES DEL SISTEMA

MSX BASIC incorpora las siguientes variables del sistema, reservadas para su propio uso:

TIME	Esta variable hace referencia al valor inicial de un temporizador de intervalos que se incrementan en "1" cada 1/50 de segundo. Este temporizador se ajusta al valor deseado asignando el valor en cuestión a esta variable.
SPRITE$(n)	Esta tabla de cadena de caracteres hace referencia a formas visualizables.
VDP(n)	Esta tabla hace referencia al valor del registro en el Procesador de Pantalla Video.

(6) CONVERSION DE TIPO

Un tipo de datos numéricos puede convertirse a otro tipo numérico siempre que sea necesario (para convertir un dato numérico a cadena de caracteres o viceversa, úsense las funciones STR$ y VAL).

① Si un valor numérico de un tipo dado se transfiere a una variable de otro tipo numérico, a ésta se asigna el valor convertido a su tipo de variable.
Ejemplo: A%=1.234 El entero "1" se asigna a la variable A%.

② Para operaciones lógicas, todos los valores se convierten al tipo enteros, y el resultado se da también en dicho tipo.
Ejemplo: A=NOT 1.234 El valor real 1.234 se convierte al entero 1, y el resultado de la operación NOT en 1 se asigna a la variable A.

- Cuando un valor real se convierte a entero, se truncan todos los decimales.
  Si el resultado de redondear un valor real excede de la gama de enteros (−32768 a +32767), se exhibe un mensaje de error.
- Cuando un número real de doble precisión se convierte a número real de precisión sencilla, el 7° dígito significativo se redondea para producir un número real de seis cifras.

Ⅳ OPERACIONES

1. EXPRESIONES

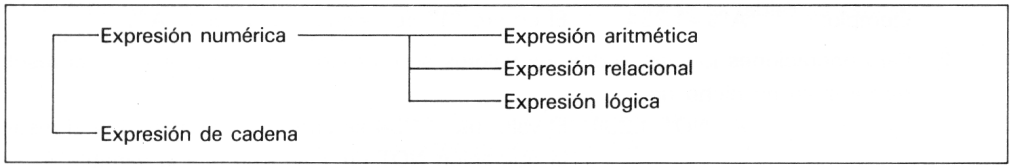
Por expresión se entiende una combinación de constantes, variables y/o funciones, conectadas entre sí por operadores; sin embargo, en ocasiones se asigna el nombre de expresión a una constante, variable o función, aún cuando carezcan de operador.

① **Expresiones numéricas y de cadena**

El resultado de una operación especificada en una expresión puede ser un valor numérico o una cadena de caracteres. Las expresiones que producen resultados numéricos se denominan expresiones numéricas, mientras que las que producen cadenas de caracteres se definen como expresiones de caracteres.

② **Tipos de expresiones numéricas**

Las expresiones numéricas se clasifican en aritméticas, relacionales y lógicas.



2. EXPRESIONES ARITMETICAS

Una expresión aritmética consiste en una o varias constantes, variables, funciones y/o expresiones, todas ellas de tipo numérico, conectadas entre sí por operadores aritméticos. Producen siempre un resultado numérico.

Operador aritmético	Operación	Formato de entrada
+	Suma	X+Y
−	Resta	X−Y
*	Multiplicación	X*Y
/	División	X/Y
^	Potencia	X^Y
−	Signo negativo	−X
\	División de enteros	X\Y
MOD	Módulo o resto	X MOD Y

(1) División entera

- Para la división entera (\) se efectúa el redondeado previo de la parte decimal de los números.
- El cociente de la división es igualmente un entero, con omisión de todos los decimales.

  Ejemplo: A=11.24\3 El entero 3 se asigna a la variable A.
- El resto de una división entera es un entero, que puede obtenerse con el operador MOD.

  Ejemplo: A=11.24 MOD 3 El entero 2 se asigna a la variable A.

(2) División por cero

Si se intenta dividir por cero, aparece el mensaje de error "División by zero", y el ordenador se restituye al modo Comando.

(3) Potencia de cero

La potencia de cero da normalmente cero.
La potencia cero del valor cero (0^0) es "1".
La potencia cero un número negativo da el mensaje de error "División by zero" y hace que el ordenador se restituya al modo Comando.

(4) Desbordamiento

Si el resultado de una asignación u operación aritmética excede de la gama numérica de la variable a que se transfiere el resultado, ocurre un desbordamiento: en la pantalla aparece el mensaje "Overflow" y el ordenador se restituye al modo Comando.

3. EXPRESIONES RELACIONALES

Una expresión relacional consiste en valores numéricos o cadenas de caracteres conectados entre sí por operadores relacionales. Produce siempre un resultado numérico:
−1 si la expresión es verdadera, y 0 si es falsa.
Las expresiones relacionales se usan principalmente para comparar dos datos en una instrucción IF.

Operadores relacionales	Significado	Formato de entrada
=	Igual	X=Y
< > o > <	Distinto	X< >Y
<	X es menor que Y	X<Y
>	X es mayor que Y	X>Y
< = o = <	X es igual o menor que Y	X< = Y
> = o = >	X es igual o mayor que Y	X> = Y

Ejemplos: IF X = Y THEN 100 ELSE 200
Si X es igual a Y, el control pasará a la línea nº 100; si no lo es, pasará a la nº 200.
A=X=Y
El primer signo "=" es el símbolo de asignación, mientras que el segundo es un operador relacional: si X es igual a Y, el valor −1 se asignará a la variable A; si X no es igual a Y, el valor asignado a A será 0.

(1) Comparación de cadenas de caracteres

Cuando en una expresión relacional se comparan dos cadenas de caracteres, se comprueba la concordancia de cada carácter, empezando con el primero de cada cadena.
Dos cadenas se consideran idénticas cuando cada carácter de una cadena coincide con el de la otra. Si las dos cadenas difieren, la que contiene el código de carácter más elevado se selecciona como la mayor.
Si las dos cadenas son de diferente longitud, la más larga se considera como mayor. Los espacios en blanco que pueda haber en las cadenas se cuentan también al calcular sus respectivas longitudes.

Ejemplos: "ABCDEF" es igual a "ABCDEF"
"AA" es menor que "AB"
"ABCDEF" es mayor que "ABCDE"
"AA " es mayor que "AA "
"A A" es menor que "AA"

(2) No es posible comparar un valor numérico con una cadena de caracteres. Los valores numéricos se comparan siempre con valores numéricos, y las cadenas de caracteres también sólamente entre sí.

4. EXPRESIONES LOGICAS

Las expresiones lógicas pueden contener una o varias constantes, variables, funciones y/o expresiones, todas ellas numéricas y conectadas entre sí mediante operadores lógicos. El resultado es siempre un entero.
Las expresiones lógicas se usan para comparar expresiones relacionales, principalmente en instrucciones IF, o para realizar manipulaciones de bits u operaciones de álgebra de Boole.

Operadores lógicos	Significado	Formato de entrada
NOT	Negación (no)	NOT X
AND	Producto lógico (y)	X AND Y
OR	Suma lógica (o)	X OR Y
XOR	OR exclusiva	X XOR Y
IMP	Implicación	X IMP Y
EQV	Equivalencia	X EQV Y

Ejemplo: 10 IF X > 10 AND X < 100 THEN 100
Si el valor de la variable X es superior a 10 e inferior a 100, el control pasará a la línea nº 100.
10 IF X > 10 OR Y < 100 THEN 100
Si el valor de la variable X es superior a 10 o el valor de la variable Y es inferior a 100, el control pasará a la línea nº 100.

(1) Tablas de verdad para operaciones lógicas

X	NOT X
1	0
0	1

X	Y	X AND Y	X OR Y	X XOR Y	X IMP Y	X EQV Y
1	1	1	1	0	1	1
1	0	0	1	1	0	0
0	1	0	1	1	1	0
0	0	0	0	0	1	1

(2) Operaciones lógicas

Las operaciones lógicas se realizan después de convertir todos los valores numéricos (entre −32768 y 32767) a su complemento a dos. Si se intenta una operación lógica con un valor fuera de esta gama, ocurrirá un error.
La operación lógica es realizada con cada bit de los operandos.

Ejemplo: A=7 OR 8
Los dos complementos a dos de los valores 7 y 8 son &B111 y &B1000, respectivamente. El resultado de una operación OR lógica en estos valores es &B1111 (15 en decimal). Por lo tanto, el valor 15 es asignado a la variable A.

5. EXPRESIONES DE CADENA

Una expresión de cadena consiste en dos o más cadenas de caracteres enlazadas con uno o más signos "+". El resultado de una expresión de cadena es siempre una cadena de caracteres.

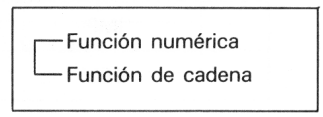
Ejemplo: A$="ABC":B$="DE":C$=A$+B$
La cadena de caracteres ABCDE se asigna a la variable C$.

6. FUNCIONES

El MSX BASIC incorpora una serie de funciones preprogramadas que simplifican la escritura de diversos tipos de programas. Dan el resultado de operaciones con funciones matemáticas especiales (por ejemplo raíz cuadrada, valor absoluto, funciones trigonométricas, etc.), en base a los datos (argumentos) especificados. Ciertas funciones pueden usarse también para manipular cadenas numéricas o de caracteres.

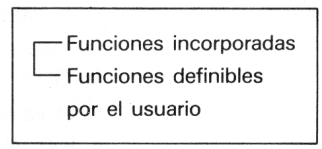
(1) Funciones numéricas y de cadena

Las funciones se clasifican en numéricas y de cadena, según den como resultado un valor numérico o una cadena de caracteres.



(2) Funciones incorporadas y funciones definidas por el usuario

Además de las funciones ya incorporadas en MSX BASIC, hay también funciones programables por el usuario, definibles con la instrucción DEF FN.



(3) Números reales usados en argumentos

En las operaciones de funciones, los argumentos especificados con enteros o números reales de simple precisión son tratados como números reales de doble precisión.

(4) Enteros usados en argumentos

En las operaciones de funciones, los argumentos especificados con números reales de doble o simple precisión se truncan en general a enteros (con todos los decimales redondeados).

7. ORDEN DE PRIORIDAD EN LAS OPERACIONES

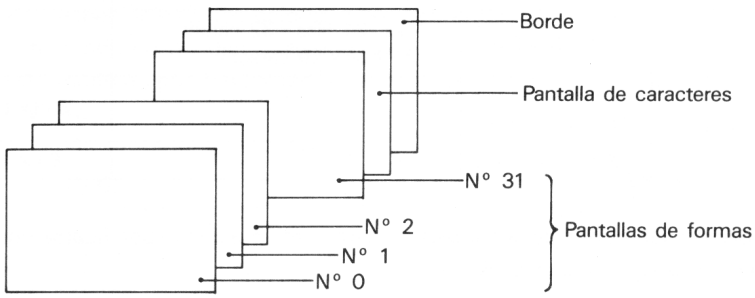
1. Operaciones entre paréntesis
2. Funciones
3. Potencia (^)
4. Signo negativo (−)
5. Multiplicación (*) y división (/)
6. División entera (\)
7. Resto (MOD)
8. Suma (+) y resta (−)
9. Operaciones relacionales (<, >, =, etc.)
10. Operación lógica (NOT)
11. Operación lógica (AND)
12. Operación lógica (OR)
13. Operación lógica (XOR)
14. Operación lógica (EQV)
15. Operación lógica (IMP)

Las operaciones de idéntica prioridad se ejecutan de izquierda a derecha.

V CONTROL DE LA PANTALLA

1. TIPOS DE PANTALLA

MSX BASIC permite trabajar con tres tipos de pantallas: una de caracteres, 32 de formas (sprite) y una de borde.



- **Pantalla de caracteres**
  Esta pantalla tiene dos modos: texto y gráficos. El modo texto muestra líneas de programa, datos, mensajes. El modo gráfico permite trazar figuras gráficas y fondos.

- **Pantallas de formas (sprite)**
  Estas pantallas se usan para visualizar gráficos preprogramados, en las posiciones especificadas de la pantalla, para producir efectos de movimiento.

- **Borde**
  Esta pantalla permite especificar colores. No puede usarse para texto ni gráficos.

- **Orden de prioridad de los tipos de pantalla**
  La prioridad más alta corresponde a las pantallas de formas; sigue la pantalla de pauta de caracteres, y la pantalla borde en último lugar.
  Entre las pantallas de formas, la nº 0 posee la prioridad más alta y sigue el resto, en orden de prioridad descendente.
  Cuando hay más de una pantalla en el monitor, la que tiene menor prioridad se oculta detrás de la que tiene mayor prioridad. Ello elimina el tratamiento de líneas invisibles en el programa.

2. PANTALLA DE CARACTERES

La pantalla de caracteres se usa para el modo texto y el modo gráfico, cada uno dividido en dos submodos. Estos modos y submodos se especifican con la instrucción SCREEN.

Modo	Submodo	Capacidad de caracteres (valor inicial)	Resolución	Formas
Modo Texto	Texto 40×24	Máx. 24 filas×40 caract. (24 filas×37 caract.)	—	No
	Texto 32×24	Máx. 24 filas×32 caract. (24 filas×29 caract.)	—	Sí
Modo Gráfico	Gráficos de alta resolución	——	256×192	Sí
	Multicolor	——	64×48	Sí

- Los modos y submodos de la pantalla de caracteres pueden especificarse con los enteros 0—3 después de la instrucción SCREEN:

  0 . . . . . Modo Texto 40×24 (SCREEN 0)
  1 . . . . . Modo Texto 32×24 (SCREEN 1)
  2 . . . . . Modo Gráfico de alta resolución (SCREEN 2)
  3 . . . . . Modo Multicolor (SCREEN 3)

- El modo texto se usa para visualizar caracteres y símbolos alfabéticos, y permite mostrar líneas de programas o datos. En el modo texto se pueden usar los siguientes comandos e instrucciones para controlar la pantalla:

  PRINT, PRINT USING, WIDTH, LOCATE, CSRLIN y POS.

- El modo gráfico permite trazar figuras o símbolos gráficos. En este modo pueden exhibirse también caracteres y símbolos alfabéticos.
  En el modo gráfico, pueden emplearse los siguientes comandos e instrucciones para controlar la pantalla:

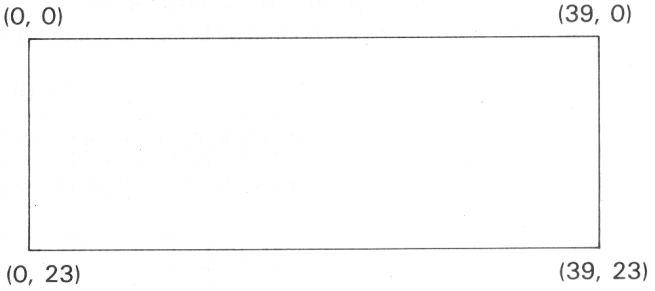
  CIRCLE, DRAW, LINE, PAINT, PSET, PRESET y POINT.

  Para visualizar caracteres o símbolos de texto en este modo, use la instrucción PRINT#. La instrucción INPUT no es válida en modo gráfico.

- Las instrucciones CLS y COLOR pueden usarse en ambos submodos.
- Cuando se selecciona el modo texto 40×24, no pueden utilizarse pantallas de formas.
- Al conectar el ordenador se selecciona automáticamente el modo texto 40×24, que permite visualizar 37 caracteres por fila.

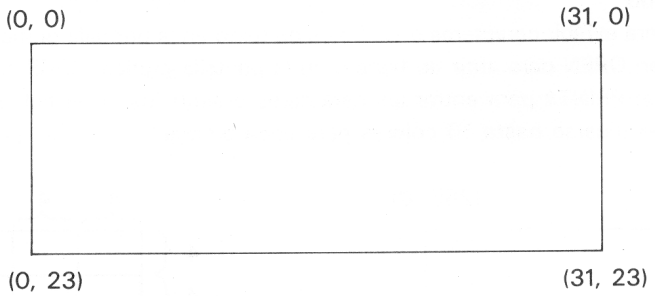
(1) MODO TEXTO 40×24 (SCREEN 0)

- La máxima capacidad de una pantalla es 24 filas de 40 caracteres, si bien la capacidad inicial es 24 filas de 37 caracteres. El número de caracteres por fila puede seleccionarse con la instrucción WIDTH.
- Cada carácter está formado por una matriz de 6×8 puntos.
  Por dicha razón, en el modo texto 40×24 a veces no aparecen ciertas partes de algunos símbolos gráficos, dado que para su completa representación se precisa una matriz de 8×8 puntos.
- Pueden especificarse dos colores: uno para el texto y el otro para el fondo de pantalla.
- En este modo no pueden usarse la pantalla de formas.
- En este modo no pueden usarse las instrucciones y comandos del modo gráfico (CIRCLE, DRAW, etc.).



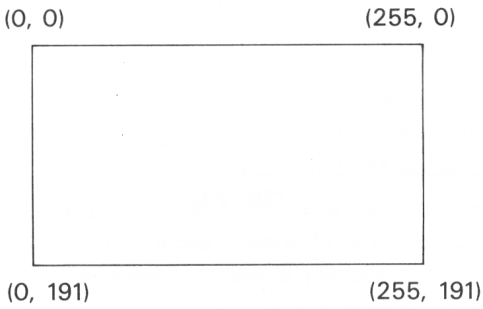
(2) MODO TEXTO 32×24 (SCREEN 1)

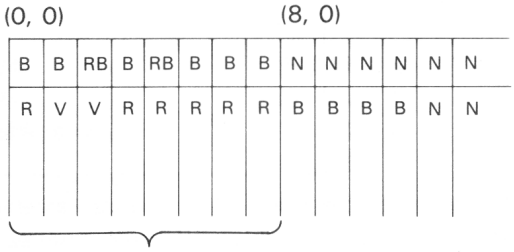
- La máxima capacidad por pantalla es 24 filas de 32 caracteres, si bien el valor inicial es 24 filas de 29 caracteres. El número de caracteres por fila puede especificarse con la instrucción WIDTH.
- Cada carácter es representado por una matriz de 8×8 puntos.
- Pueden especificarse hasta tres colores: para texto, el fondo y el borde.
- En este modo no pueden usarse las instrucciones o comandos del modo gráfico (CIRCLE, DRAW, etc.).



(3) MODO GRAFICO DE ALTA RESOLUCION (SCREEN 2)

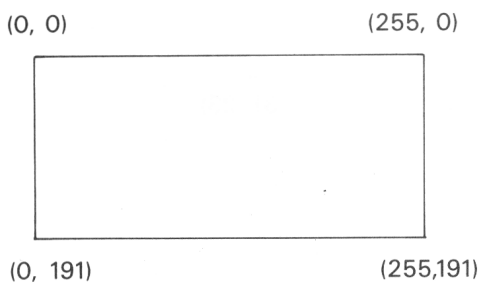
- Este modo permite trazar figuras gráficas usando una configuración de 256×192 puntos por pantalla.
- Cuando el ordenador se restituye al modo Comando, sea por completarse o interrumpirse la ejecución del programa, se selecciona automáticamente el modo Texto.
- En el modo Gráfico no pueden usarse las instrucciones del modo Texto (PRINT, etc.), como tampoco el comando INPUT.
- Cuando desee exhibir caracteres o símbolos de texto en la pantalla gráfica, use la instrucción OPEN para abrir un fichero en la pantalla gráfica ("GRP:"), luego emplee la instrucción PRINT# para visualizar los caracteres o símbolos deseados. En este caso, la esquina superior izquierda del primer carácter corresponde al último punto de referencia (LP) de la pantalla gráfica, que puede especificarse con la instrucción PRESET.
- Pueden seleccionarse hasta dos colores por cada área de 8 puntos. Ello significa que pueden especificarse sólo dos colores en el área comprendida entre (0,0) y (8,0). Si en esta área se especifica un tercer color, todos los puntos de la misma adoptarán este tercer color.

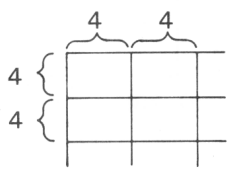




(4) MODO MULTICOLOR (SCREEN 3)

- Este modo permite dibujar gráficos multicolores usando una configuración de bloques 64×48 (4×4 puntos por bloque). Si bien la visualización es controlada bloque a bloque, la ubicación en la pantalla es especificada por punto.
- Cuando el ordenador se restituye al modo Comando al completarse o interrumpirse la ejecución del programa, se selecciona automáticamente el modo Texto.
- Las instrucciones de modo Texto (PRINT, etc.) y la instrucción INPUT no pueden utilizarse en este modo.
- Cuando quiera exhibir caracteres o símbolos de texto en la pantalla gráfica multicolor, use la instrucción OPEN para abrir un fichero en la pantalla gráfica ("GRP"), y luego emplee la instrucción PRINT# para entrar los caracteres o símbolos en el fichero.
- Pueden especificarse hasta 16 colores para cada bloque.





Hasta 16 colores por bloque de 4×4 puntos.

3. PANTALLA DE FORMAS

La pantalla de formas permite exhibir figuras y conferirles movimiento. Primero use la variable SPRITE$ para definir la figura deseada y luego la instrucción PUT SPRITE para visualizarla.

(1) TAMAÑO DE LAS FORMAS

Cada forma puede especificarse con una matriz de 8×8 ó 16×16 puntos. Cuando la forma está en pantalla, puede ampliarse doblando su altura y anchura. Las dimensiones iniciales y el atributo de ampliación se especifican con la instrucción SCREEN.

(2) NUMERO DE FORMAS DEFINIBLES

- El número de formas definibles depende de su tamaño:

Tamaño de formas	Formas definibles	Tamaño de cada forma visualizada
8×8	256 tipos	8×8 puntos 16×16 puntos
16×16	64 tipos	16×16 puntos 32×32 puntos

- Pueden visualizarse hasta 32 formas por pantalla y hasta 4 formas por fila. Si en una fila se especifican más formas, la quinta y sucesivas no aparecerán.

(3) INTERRUPCIONES POR LAS FORMAS

Si dos formas se solapan, causan una interrupción de la CPU. No es necesario investigar el solapamiento de las figuras en el programa.
La interrupción puede tratarse de la siguiente forma:

- Especificando la primera línea de la rutina de tratamiento de interrupciones a que se desea transferir el control cuando ocurre una interrupción por las formas:

```
ON SPRITE GOSUB
```

- Especificando si la interrupción Sprite debe ser habilitada, deshabilitada o retenida:

```
SPRITE ON      Habilita la interrupción
SPRITE OFF     Deshabilita la interrupción
SPRITE STOP    Retiene la interrupción
```

Si ocurre un solapamiento de formas después de ejecutar la instrucción SPRITE ON, se produce una interrupción y el control pasa a la rutina de tratamiento de interrupciones cuya primera línea se especificó en la instrucción ON SPRITE GOSUB.

4. CODIGO DE COLORES

Para especificar los colores de visualización se usan los números 0 al 15:

0	Transparente	8	Rojo
1	Negro	9	Rojo brillante
2	Verde	10	Amarillo
3	Verde claro	11	Amarillo claro
4	Azul oscuro	12	Verde oscuro
5	Azul claro	13	Púrpura
6	Rojo oscuro	14	Gris
7	Azul celeste	15	Blanco

5. COMO ESPECIFICAR COORDENADAS

Las coordenadas o posiciones de pantalla especificadas en instrucciones gráficas (CIRCLE, LINE, PAINT, PSET, PRESET y POINT) o en la instrucción PUT SPRITE, pueden ser de dos tipos:

Coordenadas absolutas

(x, y)
Especifican un punto absoluto de la pantalla, mediante los valores "x" e "y".

Coordenadas relativas

STEP (x, y)
Especifican un punto de la pantalla que dista del último punto de referencia (LP) los valores de longitud definidos por "x" e "y". LP es la última coordenada especificada en una instrucción gráfica, y posee un valor inicial de (0, 0).

VI SONIDO

1. GENERADOR DE SONIDO PROGRAMABLE (GSP)

El GSP es un sintetizador simplificado de música, que suministra tonos al altavoz interno cuando se ejecutan instrucciones PLAY o SOUND.

CARACTERISTICAS DEL GENERADOR DE SONIDO PROGRAMABLE

1) Contiene tres canales de audio independientes (A, B, C), que suministran simultáneamente tres tonos diferentes. Ello de permite tocar acordes de tres tonos.
2) Permite emitir un tono de ruido en cada canal, además de los tonos musicales, a fin de producir efectos sonoros para videojuegos.
3) La característica "envolvente" permite variar el volumen de los tonos de salida a lo largo del eje de tiempo. De este modo puede añadirse timbre o ritmo a los tonos emitidos.

2. MUSICA CON LA INSTRUCCION PLAY

La instrucción PLAY usa macrocomandos para generar música:

PLAY expresión cadena A, expresión cadena B, expresión cadena C

Las expresiones A, B y C representan macrocomandos para los canales A, B y C, respectivamente. El formato de la instrucción PLAY se explica detenidamente en el Capítulo 2, "Descripción del Lenguaje".

Se hallan disponibles los siguientes macrocomandos para música:

(1) MACROCOMANDOS PARA ESPECIFICAR NOTAS

A — G — Estos comandos especifican las siete notas completas en una escala de octavas en el orden C, D, E, F, G, A y B (Do, Re, Mi, Fa, Sol, La, Si, respectivamente).

+ − — Se usan para especificar la altura de una nota, para elevar o bajar su tono. Para hacer una nota más aguda, use "#" o "+", como por ejemplo A# o A+. Para bajar el tono de una nota, use "−", como por ejemplo A−.

Ejemplo: PLAY "F#", "G", "C"

O entero — Este comando especifica cuál de las 8 octavas disponibles debe usarse, mediante un entero entre 1 y 8. Se antepone a las notas (A — G), como por ej. O5. El valor estándar inicial es O4. Cuando este comando se especifica, permanece válido hasta que se entra otro comando de octavas. Este comando es válido sólo en el canal para el que se entra.

Ejemplo: PLAY "O4CDEFGABO5CDEFGAB".

N entero — Este comando especifica una altura dada en las ocho octavas completas, mediante un entero entre 0 y 96. N1 corresponde a O1C# y N95 especifica O8B. N0 designa una pausa. Cada incremento de 1 entero eleva la altura en medio tono.

Ejemplo: PLAY "N36N38N40N41N43N45N47N48"

(2) MACROCOMANDO PARA ESPECIFICAR DURACION

L entero Este comando se usa para especificar la duración (1/entero) de una nota y de todas las siguientes. La gama válida del entero es 1 — 64. El valor inicial es L4.

L1	Nota completa
L2	Media nota
L4	Cuarto de nota
L8	Octavo de nota
L64	64° de nota

Cuando se ha especificado un comando de duración, permanece válido hasta que se entra otro. El comando de duración es válido sólo en el canal para el que se especifica.

Ejemplo: PLAY "L1CDEFGAB"

Cuando usted quiera indicar una nota específica por una duración específica o pausa, escriba el correspondiente entero a continuación del comando de la nota, como por ejemplo A16.

Ejemplo: PLAY "C1DEFGAB"

Si un comando de duración o pausa va seguido de un punto (.), la nota en cuestión se alarga 1,5 veces. Si se especifican dos puntos consecutivos (..), la nota anterior se alarga 2,25 veces; si se especifican tres puntos consecutivos (...), la nota se toca 3,375 veces más larga.

Ejemplo: PLAY "L1C.", "C1.", "C1C2"

(3) MACROCOMANDO PARA ESPECIFICAR PAUSA

R entero Este comando especifica la duración (1/entero) de una pausa. Si se especifica sólo R, se selecciona automáticamente R4.

R1	Pausa completa
R2	Media pausa
R4	Cuarto de pausa
R8	Octavo de pausa
R64	64° de pausa

Ejemplo: PLAY "CDEF1FGABRCDEFGAB"

(4) MACROCOMANDO PARA ESPECIFICAR RITMO

T entero Este comando especifica el número de cuartos de nota que deben sonar por minuto en todas las notas que le siguen. La gama válida del entero va de 32 a 255. El valor por defecto es T120.
Cuando se ha especificado un ritmo, continúa válido hasta que se selecciona otro. Este comando es válido sólo para el canal que se especifica.

Ejemplo: PLAY "T240CDEFGABT60CDEFGAB"

(5) MACROCOMANDOS PARA ESPECIFICAR INTENSIDAD SONORA

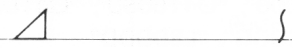
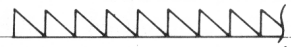
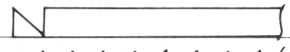
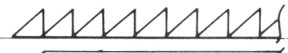
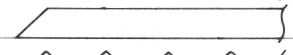
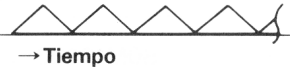
V entero Este comando especifica la intensidad sonora de los tonos que le siguen. La gama válida del entero es 0 — 15, con un valor por defecto de 8. Cuando este comando se ha especificado, continúa válido hasta que se entra otro. Este comando es válido sólo en el canal para el que se especifica, e invalida cualquier comando de envolvente que estuviera vigente.

Ejemplo: PLAY "V15CDEFGABV8CDEFGAB"

(6) MACROCOMANDOS PARA ESPECIFICAR TIMBRE

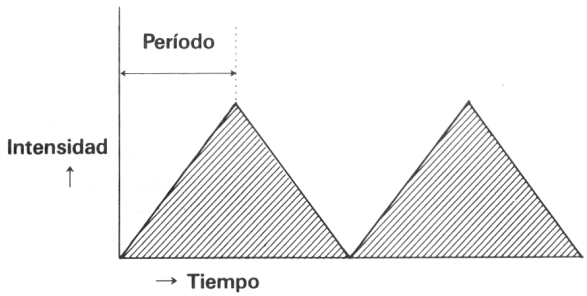
Los comandos Envolvente adoptan dos formas: el comando S y el comando M. El comando S se usa para especificar la forma de onda, o envolvente, con arreglo a la cual varía la intensidad sonora de la nota. El comando M se usa para especificar el período de la envolvente. Usted puede controlar el timbre tonal de su música combinando estos dos comandos. Cuando se especifica un comando Envolvente, se invalida el comando de intensidad sonora V. El comando Envolvente es válido en los tres canales de audio. Ello significa que en toda ocasión puede especificarse sólo una pauta de envolvente y período para los tres canales.

S entero Este comando especifica una pauta de envolvente. La gama válida del entero es 0 — 15, con los que pueden seleccionarse las ocho siguientes pautas de envolvente:

Valor de entero	Pauta de la envolvente
0~3,9	
4~7,15	
8	
10	
11	
12	
13	
14	

→Tiempo

M entero Este comando especifica el período de una envolvente. La gama válida del entero es de 1 a 65535.



Ejemplo: PLAY "S0M10000CDEFGAB"

(7) MACROMANDOS PARA MUSICA ASIGNADOS A VARIABLES

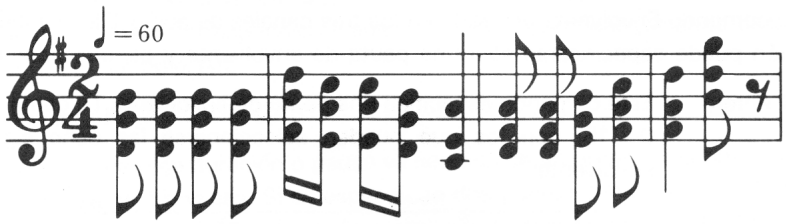
X Variable de caracteres; Esta variable hace que la instrucción PLAY genere música con arreglo a los macrocomandos asignados a la variable en cuestión; ha de ir seguida de punto y coma (;).

Ejemplo: A$="CDEFG":B$="AB" : PLAY"XA$;XB$;"

=variable numérica; Esta variable puede usarse en vez del entero a especificar en un macrocomando para música. Ha de ir seguida de punto y coma (;).

```
Ejemplo: 10  FOR J=3 TO 6
         20  PLAY"O=J;CDEFGAB"
         30  NEXT
         40  END
```

A título ilustrativo, he aquí las cuatro primeras líneas de "Rosas Silvestres" de Schubert:




```
10 PLAY "O4T60S0M10000", "O4T60S0", "O4T60S0"
20 PLAY "L8BBBB", "L8GGGG", "L8DDDD"                          1ª línea
30 PLAY "L16O5DCCO4BA4", "L16BAAGE4", "L16F+EEDC4"            2ª línea
40 PLAY "L8AABO5C", "L8F+F+GA", "L8DDEF+"                     4ª línea
50 PLAY "O5D4GR8", "A4O5DR8", "F+4BR8"                        3ª línea
60 END
```

10 PLAY "O4T60S0M10000"	O4	Especifica la 4ª octava.
	T60	Ajusta el ritmo a 60 cuartos de nota por minuto.
	S0M10000	Especifica la envolvente para simular un sonido de piano.
20 PLAY "L8BBBB"	L8	Especifica octavos de nota.
	BBBB	Genera la nota B cuatro veces.
30 PLAY "L16O5DCCCO4BA4"	L16	Especifica 16ª de nota.
	O5	Especifica la 5ª octava.
	DCCC	Genera las notas D, C, C y C.
	O4	Especifica la 4ª octava.
	A4	Genera un cuarto de nota de altura A.

3. EFECTOS SONOROS UTILIZANDO LA INSTRUCCION SOUND

La instrucción SOUND permite entrar valores en los registros del generador de sonido programable (GSP). Con las combinaciones correctas de varias instrucciones SOUND usted puede crear diversos efectos sonoros que no serían posibles con sólo instrucciones PLAY.

SOUND número de registro, expresión de enteros

Esta instrucción hace que la expresión de enteros sea cargada en el registro GSP especificado. El formato de esta instrucción se explica detenidamente en el Capítulo 2, "Descripción del lenguaje".

(1) REGISTROS GSP

GSP contiene 16 registros, 14 de ellos utilizables por el usuario.

N° de Registro	Función	Bit							
		b7	b6	b5	b4	b3	b2	b1	b0
0	Frecuencia del canal A	FT (A)							
1						CT (A)			
2	Frecuencia del canal B	FT (B)							
3						CT (B)			
4	Frecuencia del canal C	FT (C)							
5						CT (C)			
6	Frecuencia de ruido				NP				
7	Selección del canal de salida			Ruido			Tono		
		1	0	C	B	A	C	B	A
8	Sonoridad del canal A				M	L (A)			
9	Sonoridad del canal B				M	L (B)			
10	Sonoridad del canal C				M	L (C)			
11	Periodo de la envolvente	FT (E)							
12		CT (E)							
13	Pauta de la envolvente					EP			

(2) COMO USAR EL GENERADOR DE SONIDO PROGRAMABLE (GSP)

El GSP puede producir simultáneamente hasta tres señales de tonos y una de ruido. Estas señales se envían hacia el altavoz incorporado, a través de los tres canales audio (A, B, C), para crear diversos efectos sonoros.
El tono para cada canal se especifica cargando los datos requeridos de altura o frecuencia en los registros 0 — 5.
Los datos de frecuencia de ruido se entran en el registro 6.
El registro 7 se usa para seleccionar uno de los tres canales de tonos, y determina si dicho canal debe emitir un tono, un ruido o una combinación de ambos. Si bien hay sólo una fuente de ruido, éste puede darse de salida hacia uno cualquiera de los tres canales, junto con señales de tono.
Los registros, 8, 9 y 10 se usan para seleccionar los niveles de intensidad sonora de los tres canales.
Con estos registros puede especificarse una intensidad constante, o variables con arreglo a una pauta de envolvente específica.
Cuando se usa una envolvente, la misma se entra en el registro 13, mientras su período puede asignarse a los registros 11 y 12.

① Selección de la frecuencia de los tonos

Para seleccionar una frecuencia de tonos, determine primero el valor TP (hasta 12 bits) calculado a partir de la siguiente fórmula; luego divida el valor TP en 4 bits de alto orden (CT) y 8 bits de bajo orden (FT), como se indica a continuación:

TP=1789772.5 (16*F)	
CT=TP/256	4 bits de alto orden de TP
FT=TP MOD 256	8 bits de bajo orden de TP

Los valores de FT y CT para los canales A, B y C se cargan en los registros 0 y 1; 2 y 3, y 4 y 5, respectivamente. En los registros 1, 3 y 5 son significativos sólo los 4 bits de bajo orden.

② **Selección de la frecuencia de ruido**

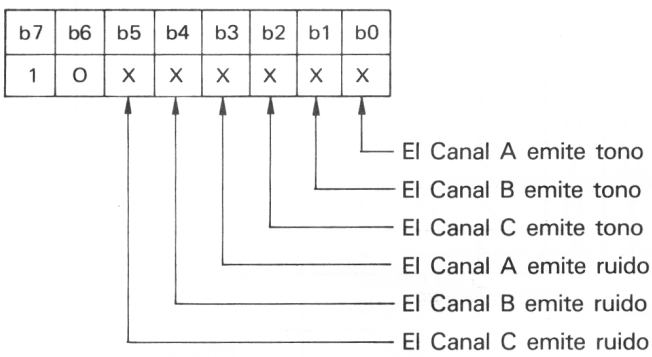
Para seleccionar una frecuencia de ruido (en Hz), cargue en el registro 6 un valor de cinco bits, NP, calculado a partir de la siguiente fórmula:

NP=1789772.5/(16*F)

Sólo los cinco bits de bajo orden del registro 6 son significativos; los tres bits de alto orden carecen de significado.

③ **Selección del canal**

Para seleccionar el canal de salida, entre un cero en el bit del registro 7 que corresponda al canal de salida.
Configuración de bits de registro 7: Ponga 0 en el bit que corresponda al canal de salida deseado:

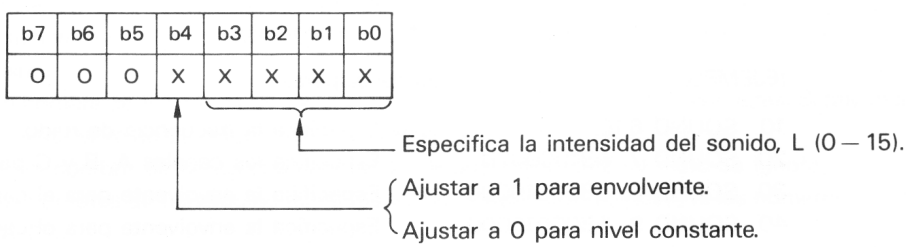


Los 2 bits más significativos de este registro se usan para especificar la dirección de entrada/salida de las vías de acceso (ports) de E/S general, completamente al margen de la función de salida de sonido.

④ **Selección de la intensidad sonora**

El nivel de sonoridad de los canales A, B y C se especifica en los registros 8, 9 y 10 respectivamente. Usando el bit nº 4 de cada uno de estos registros puede especificar una intensidad constante o variable (envolvente). Los 3 bits más significativos de cada registro carecen de significado.

Configuración de bits en los registros 8, 9 y 10:



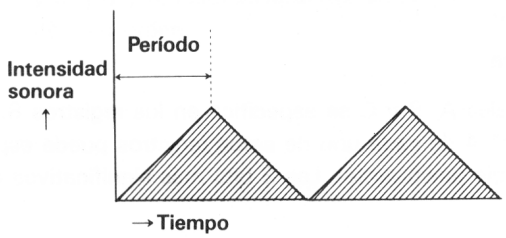
⑤ Especificación de la pauta de envolvente y período

El tono producido en el canal para el que se especifica una envolvente en los registros 8, 9 o 10 varían con arreglo al período especificado en los registros 11 y 12 y a la pauta de envolvente especificada en el registro 13. Las ocho pautas de envolvente siguientes pueden especificarse con los 4 bits menos significativos (BMS) del registro 13.

4 BMS				Pauta de envolvente
b3	b2	b1	b0	
0	0	×	×	
0	1	×	×	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

→Tiempo

El período de envolvente, T (en seg.), se especifica entrando el valor de TP (hasta 16 bits), calculado con la fórmula siguiente, en los registros 11 y 12: los 8 bits de bajo orden (FT) en el registro 11, y los 8 bits de alto orden (CT) en el registro 12.



TP=1789772.5*T/256
CT=TP＼256　　8 bits de orden alto
FT=TP MODE 256　　8 bits de orden bajo

[EJEMPLO] Cómo crear un sonido de disparos.

```
10 SOUND 6,15                  Especifica la frecuencia de ruido.
20 SOUND 7, &B10000111         Especifica los canales A, B y C para ruido.
30 SOUND 8, &B00010000         Especifica la envolvente para el canal A.
40 SOUND 9, &B00010000         Especifica la envolvente para el canal B.
50 SOUND 10, &B00010000        Especifica la envolvente para el canal C.
60 SOUND 11, 0: SOUND 12,16    Especifica el período de la envolvente.
70 SOUND 13, 0                 Especifica la pauta de la envolvente.
80 END
```

VII FICHEROS

Un fichero es una serie de registros, datos o programas, generalmente almacenados en un soporte de memoria (como por ejemplo en cinta cassette). Cada fichero posee su propio nombre, para poder diferenciarlo de otros ficheros. El concepto de fichero es a menudo aplicable a diversos dispositivos de entrada/salida.

1. ESPECIFICACIÓN DE UN FICHERO

Un fichero se especifica mediante un nombre de dispositivo y un nombre de fichero. Un programa o datos se almacenan o extraen del fichero especificado, en el dispositivo indicado, en el orden "NOMBRE DISPOSITIVO NOMBRE FICHERO".

- Un fichero puede especificarse también con variables de caracteres o con expresiones de caracteres, así como mediante cadenas de caracteres entrecomilladas.
- Un dispositivo o nombre de fichero puede también ser seleccionado automáticamente por el sistema.

(1) Nombre del dispositivo

El nombre del dispositivo ha de especificar el dispositivo de entrada/salida usado para acceso al fichero.

Dispositivo E/S	Nombre del dispositivo	Modo utilizable	
		Entrada	Salida
Registrador de cassette	CAS:	O	O
Pantalla de modo texto	CRT:	×	O
Pantalla de modo gráfico	GRP:	×	O
Impresora	LPT:	×	O

- La pantalla puede tratarse también como un dispositivo de salida, en cuyo caso se usa principalmente para visualizar caracteres o símbolos de texto en el modo gráfico.
- El nombre del dispositivo puede especificarse con mayúsculas o minúsculas.
- Las instrucciones CSAVE o CLOAD no requieren que se especifique el nombre del dispositivo.

(2) Nombre del fichero

El nombre del fichero debe especificarse cuando se precisa acceso a un fichero en cinta cassette.
Si el fichero no está en la cinta cassette, no se precisa indicar su nombre para tener accesso al mismo.

- Un nombre de fichero ha de ser una cadena de como máximo seis caracteres. Si hay menos de seis, el resto se rellena automáticamente con espacios en blanco.
  Si un nombre de fichero contiene más de seis caracteres, el resto se ignora.
- Los nombres de ficheros no deben contener el signo dos puntos(:) ni los números 0 ó 255 (&HFF).

2. FICHEROS DE PROGRAMAS

Los siguientes comandos se usan para almacenar o cargar ficheros de programas:

CSAVE	Almacena un programa en un fichero de cinta cassette.
CLOAD	Carga un programa en la memoria interna, almacenado antes en cinta cassette con el comando CSAVE.
SAVE	Almacena un programa formato ASCII en un fichero en un dispositivo especificado.
LOAD	Carga un programa ASCII desde un dispositivo especificado.
MERGE	Combina un fichero de programa formato ASCII con el programa actualmente en memoria.
BSAVE	Almacena un programa en código máquina en un dispositivo especificado.
BLOAD	Carga un programa en código máquina desde un dispositivo especificado.

3. FICHEROS DE DATOS

(1) Apertura de un fichero

Para tener accesso a un fichero de datos, el mismo debe abrirse antes con la instrucción OPEN. Con esta instrucción se especifica el nombre de dispositivo, el nombre del fichero a abrir, la operación de entrada/salida y el número del fichero. Interiormente, la instrucción OPEN hace que en la memoria se reserve un bloque de control del fichero (a usar para operaciones de entrada/salida).

(2) Número del fichero

El número del fichero representa los nombres del dispositivo y del fichero especificados en la instrucción OPEN. De este modo, las instrucciones PRINT# e INPUT# tienen acceso al fichero cuyo número se indica a continuación de estas palabras clave.
La gama disponible de números de fichero se selecciona con la instrucción MAXFILES. El valor inicial es "1".

(3) Cierre de un fichero

Cuando se ha completado el acceso de entrada/salida de un fichero, el mismo debe cerrarse usando la instrucción CLOSE. Si se dejara abierto, no se podría abrir otro fichero usando el mismo número de fichero.

VIII INTERRUPCIONES

El objetivo de una interrupción es indicar a la unidad central de proceso (CPU) que debe suspender lo que está haciendo y procesar el suceso o el dato entrado, para luego reanudar la operación suspendida. Si no se halla disponible ninguna señal de interrupción, el programa principal tiene que comprobar continuamente si hay algún acontecimiento o lógica externa que solicite tratamiento. Con una señal de interrupción, basta con especificar nombres de rutinas de tratamiento de interrupciones e instrucciones de habilitación de las interrupciones al principio del programa principal. Ello elimina la necesidad de un programa de comprobación de peticiones de tratamiento y acelera la ejecución del programa principal.

Causas posibles de interrupción	**Instrucciones de Trat. Inter.**
Interrupción por error	ON ERROR GOTO
Interrupción por tecla de función	ON KEY GOSUB
Interrupción por la tecla de STOP	ON STOP GOSUB
Interrupción por solapamiento de formas	ON SPRITE GOSUB
Interrupción por accionamiento de un mando de juegos (Interrupción por barra espaciadora)	ON STRIG GOSUB
Interrupción del temporizador de intervalos	ON INTERVAL GOSUB

El orden de prioridades de las distintas interrupciones es en el orden señalado en la lista anterior.

Ejemplo: En el siguiente programa, el control pasa a la línea 1000 cuando se pulsa la barra espaciadora.

Cuando se usan interrupciones (Programa)

```
10    ON STRIG GOSUB 1000
20    STRIG (0) ON
30    TIME=0
40    FOR K=1 TO 5000
50    PRINT T, TIME-T
60    T=TIME
70    NEXT
80    END
1000  PRINT "SPACE ON"
1010  RETURN
```

Cuando no se usan interrupciones (Programa)

```
10    TIME=0
20    FOR K=TO 5000
30    PRINT T, TIME-T
40    T=TIME
50    K$=INKEY$
60    IF K$="  " THEN GOSUB 1000
70    NEXT
80    END
1000  PRINT "SPACE ON"
1010  RETURN
```

IX LENGUAJE MAQUINA

(1) DESARROLLO DE UN PROGRAMA EN LENGUAJE MAQUINA

El ordenador MSX emplea un procesador Z80A para su CPU. Por consiguiente, usted debe usar el código máquina Z80 para sus programas en lenguaje máquina. Podrá encontrar información detallada sobre dicho lenguaje en los libros escritos sobre Lenguaje Máquina Z80.

① Use la instrucción CLEAR para reservar un área de memoria para el programa en código máquina.

② Cree su programa en lenguaje máquina empleando las instrucciones POKE y PEEK.

③ Para almacenar el programa en lenguaje máquina en cinta cassette, use la instrucción BSAVE. Para volverlo a cargar en la memoria interna, emplee la instrucción BLOAD.

④ El programa en lenguaje máquina puede ser ejecutado con la instrucción DEFUSR y la función USR.

Nota: Cualquier pequeño error en un programa en código máquina puede hacer que el sistema no pueda recuperarse de un estado de error y se precise desconectar el ordenador antes de poder continuar. Por dicha razón se recomienda que antes de ejecutar un programa en código máquina el mismo se grabe en cinta cassette.

(2) TRANSFERENCIA DE ARGUMENTOS MEDIANTE LA FUNCION USR

USR número, argumentos

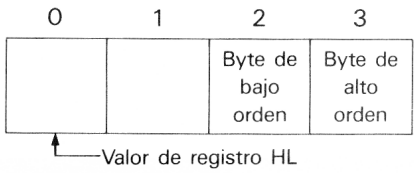
① Es posible transferir datos de un programa BASIC a otro en lenguaje máquina, mediante argumentos. El programa en lenguaje máquina usa los registros A y HL o DE para leer los datos contenidos en argumentos.

② El valor del registro A es 2, 3, 4 u 8, según el tipo de argumento.
Para transferir el resultado de la ejecución de un programa en lenguaje máquina a un programa BASIC, el resultado ha de tener el mismo formato que el argumento, y si es una cadena de caracteres debe tener la misma longitud de cadena que el argumento. Los datos resultantes han de tener la misma dirección de memoria que los datos del operando que fueron transferidos desde el programa BASIC.

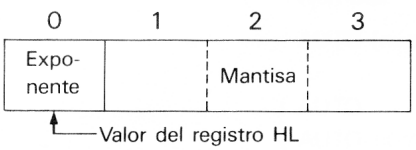
Valor del registro A	Tipo de argumento
2	Entero
3	Cadena de caracteres
4	Número real de simple precisión
8	Número real de doble precisión

③ El valor del registro HL o DE especifica la dirección de memoria de los datos a transferir. El formato con que debe especificarse la dirección difiere según el tipo de argumento:

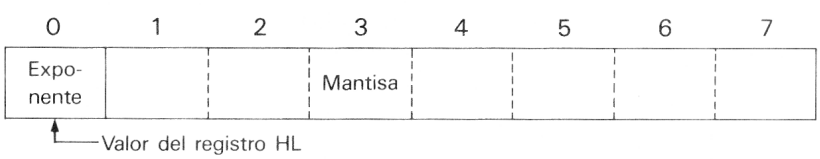
- **Entero**
  Un número entero se representa en forma binaria, con dos bytes, almacenándose en la memoria interna, primero el byte de orden bajo y luego el de orden alto, ambos precedidos por la dirección especificada por "valor del registro HL+2".



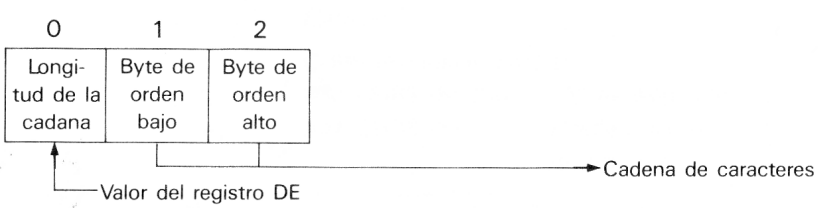
- **Número real de simple precisión**
  Un número real de simple precisión se representa mediante un solo byte para el exponente y tres bytes para la mantisa (4 bytes en total), y se almacena en la memoria en el orden de exponente primero y mantisa después, empezando con la dirección de memoria especificada por el valor del registro HL.
  El bit más significativo de la parte del exponente especifica el signo (0 para positivo, 1 para negativo) del número, y los siete bits restantes representan el exponente, que puede oscilar entre E+62 y E−64.
  La mantisa está representada por un número de notación decimal, codificado en binario, de 6 dígitos.



- **Número real de doble precisión**
  Un número real de doble precisión se representa mediante un solo byte para el exponente y siete bytes para la mantisa (8 bytes en total), y es almacenado en la memoria en el orden exponente y mantisa, empezando con la dirección de memoria especificada por el valor del registro HL.



- **Cadena de caracteres**
  Cuando se trata de una cadena de caracteres, su longitud y los bytes de orden bajo y alto de la dirección en que está alojada la cadena son almacenados en la memoria en el orden en que se escriben, empezando con la dirección, identificada por el valor del registro DE.



CAPITULO 2

DESCRIPCION DEL LENGUAJE

NOTAS EXPLICATIVAS SOBRE EL FORMATO

Corchetes [ ]

Designan que la expresión es opcional.

Ejemplo:

AUTO [número línea inicial] [, incremento]

AUTO
AUTO número línea inicial
AUTO , incremento
AUTO número línea inicial, incremento

...Repetición

Designa cualquier número de repeticiones dentro de una línea

Ejemplo:

Constante [, constante ...]

Constante
Constante, constante
Constante, constante, constante
etc.

Expresión entera

Cuando se indica "expresión entera" debe especificarse una expresión numérica (una variable numérica o una constante). Si en la expresión se halla un número real de simple o doble precisión, el valor de la expresión se convierte a un entero antes de ser ejecutada.

Ejemplo:

CHR$ (expresión entera)

A$=CHR$ (65.23) "A" es asignada a la variable A$.
A$=CHR$ (X) Pueden usarse también variables de tipo real de doble precisión.

ABS — Función

Función La función ABS suministra al programa el valor absoluto de una expresión.

Formato ABS (expresión numérica)

Descripción

1) La función ABS da el valor absoluto de la expresión numérica que la sigue.
2) El resultado se da siempre como número real de doble precisión, cualquiera que sea el tipo de expresión numérica.

Ejemplo

```
10 A=-1 : B%=1
20 AA=ABS(A) : BB=ABS (B%)
30 PRINT AA, BB
40 END
```

ASC — Función

Función La función ASC da el código ASCII del carácter que la sigue.

Formato ASC (expresión de cadena)

Descripción

1) La función ASC suministra al programa el código ASCII del primer carácter de la expresión de cadena que la sigue. El segundo y restantes caracteres no se tienen en cuenta.
2) Si la expresión de cadena está vacía (""), se exhibe un mensaje de error (illegal function call).
3) Si la expresión de cadena es un símbolo gráfico, se da el código ASCII que corresponde a la cabecera de identificación de carácter gráfico (&H01).

Ejemplo

```
10 A$="ABC"
20 PRINT ASC(A$), ASC("D")
30 END
```

ATN — Función

Función Esta función calcula el arcotangente de un número.

Formato ATN (expresión numérica)

Descripción

1) La función ATN da el arcotangente de la expresión numérica que la sigue.
2) El resultado es expresado en radianes, entre $-\pi/2$ y $\pi/2$.
3) El resultado se da siempre como número real de doble precisión, cualquiera que sea el tipo de la expresión numérica.

AUTO — Comando

Función Este comando genera una numeración automática de las líneas del programa.

Formato AUTO [número primera línea] [, incremento]

Descripción

1) Cuando se mecanografía este comando en el teclado y se pulsa la tecla [RETURN], en la pantalla aparece el primer número de línea (10 si se han omitido el número de la primera línea y el incremento). Cada vez que se termina de escribir una línea de programa y se pulsa [RETURN], aparece el siguiente número de línea con el incremento especificado. Evita el tener que mecanografiar un número de línea por cada línea del programa.
2) El número de línea ha de ser un entero entre 0 y 65529, y el incremento ha de ser un entero positivo.
3) Cuando se omite el número de la primera línea, se le da el valor cero. Cuando se omite el incremento, se selecciona como tal "10". Si se omiten ambas opciones, se selecciona automáticamente "10" para ambas.
4) Para cancelar la acción del comando AUTO, pulse simultáneamente las teclas [CTRL] y [STOP]: el sistema se restituye el modo Comando, con el aviso "Ok" en la pantalla.
5) Si se genera un número de línea que ya existe en el programa actualmente en memoria, a continuación de dicho número aparece un asterisco (*). Si usted escribe una línea de programa a continuación del asterisco y pulsa la tecla [RETURN], la anterior línea del programa será sustituida por la nueva acabada de mecanografiar. Si, en cambio, pulsa la tecla [RETURN] sin escribir ninguna línea de programa a continuación del asterisco, la vieja línea de programa en memoria se conservará tal como era.
6) Las funciones de edición utilizando el cursor se hallan disponibles también cuando el comando AUTO es activo.

Ejemplo AUTO 1000, 10

BASE — Función

Función La función BASE da la primera dirección de las tablas contenidas en la memoria de visualización (VRAM).

Formato BASE (expresión entera)

Descripción

1) Esta función da la primera dirección de la tabla (dentro de la memoria RAM de visualización) especificada por la expresión entera que la sigue.
2) La gama válida de la expresión entera va de 0 a 19. A continuación se relacionan las tablas que pueden especificarse:

Modo pantalla / Nombre de tabla	Texto 40×24	Texto 32×24	Gráfico de alta resolución	Multicolor
Tabla de nombres	0	5	10	15
Tabla de colores	No usada	6	11	No usada
Tabla de generador pauta	2	7	12	17
Tabla de atributos de las formas	No usada	8	13	18
Tabla de modelo de las formas	No usada	9	14	19

BEEP — Instrucción

Función La instrucción BEEP hace que suene el altavoz interno.

Formato BEEP

Descripción Esta instrucción genera un tono "bip" durante aprox. 0,04 seg. por el altavoz incorporado

BIN$ — Función

Función Esta función convierte un valor numérico a cadena de caracteres binarios.

Formato BIN$ (expresión entera)

Descripción La función BIN$ convierte la expresión entera que la sigue a una cadena de caracteres en notación binaria.

Ejemplo

```
10 A$ = BIN$ (16)   La cadena de caracteres "10000" es asignada a la variable A$.
20 PRINT A$
30 END
```

BLOAD — Comando

Función

Este comando carga un fichero de código máquina en la memoria interna del ordenador.

Formato

BLOAD "nombre dispositivo [nombre fichero]" [, R] [, desplazamiento]

Descripción

1) El comando BLOAD se usa para cargar en la memoria interna un fichero de programa o de datos en código máquina (que fue antes grabado con BSAVE).
2) Cuando se omite el nombre del fichero almacenado en cinta, se carga el primer fichero de la cinta.
3) Si se especifica la opción R, la ejecución del programa en código máquina empieza a partir de la dirección de inicio de la ejecución especificada en el comando BSAVE, inmediatamente después del proceso de carga.
4) Si se especifica "desplazamiento", el programa o los datos son cargados en un área de memoria cuya primera y última dirección son la suma del desplazamiento y de las direcciones antes especificadas con el comando BSAVE.
5) Dado que el comando BLOAD carga un programa o datos en código máquina en cualquier posición de memoria, puede provocar una ejecución sin fin del comando BLOAD o un desbocamiento ("runaway") del programa, si éste o los datos se cargan en el área de trabajo o en un bloque de control de ficheros. Por consiguiente, debe prestarse atención al valor del desplazamiento y a la direcciones especificadas en el comando BSAVE.

Ejemplo

BLOAD"CAS:CAJA",&H1000	Se especifica sólo el desplazamiento.
BLOAD"CAS:CAJA",R	Se especifica sólo la opción R.
BLOAD"CAS:CAJA",R,&H1000	Se especifican el desplazamiento y la opción R.

BSAVE — Comando

Función

El comando BSAVE almacena parte del contenido de la memoria interna del ordenador en un dispositivo especificado.

Formato

BSAVE "nombre dispositivo [nombre fichero]", primera dirección, última dirección [, dirección inicio ejecución].

Descripción

1) Este comando graba el contenido de la memoria ubicada entre la primera y la última dirección especificadas.
2) Si, luego, en el comando BLOAD se incluye la opción R(RUN), la ejecución de un programa cargado con BLOAD empieza automáticamente a partir de la dirección inicial de ejecución; de omitirse ésta, como tal se entiende la primera dirección.

Ejemplo

BSAVE"CAS:CAJA",&HD000,&HD1000
BSAVE"CAS:CAJA",&HD000,&HD1000,&HD00A

CALL — Instrucción

Función CALL se usa para llamar una instrucción contenida en un cartucho de ampliación de memoria ROM.

Formato
CALL nombre instrucción extendida
CALL nombre instrucción extendida (argumento, [, argumento...])
—nombre instrucción extendida [(argumento [, argumento...])]

Descripción
1) CALL llama a una instrucción contenida en un cartucho de ampliación ROM, escrita en lenguaje máquina.
2) En vez de la palabra clave CALL puede escribirse el guión de subrayado(—).
3) El uso de instrucciones almacenadas en un cartucho de ampliación ROM se describe en el manual de software MSX.

Ejemplos
CALL CAJA
CALL CAJA ("A", "123", "XX")

CDBL — Función

Función Esta función convierte el entero o número real de simple precisión que la sigue a número real de doble precisión.

Formato CDBL (expresión numérica)

Descripción
1) La función CDBL convierte el valor (entero o número real de simple precisión) de la expresión que la sigue a un número real de precisión doble.
2) El número de cifras significativas de los valores no cambia antes o después de la conversión.

Ejemplo
```
A#=CDBL(B%)
```

CHR$ — Función

Función Esta función convierte un código ASCII a su carácter equivalente.

Formato CHR$ (expresión entera)

Descripción
1) La función CHR$ da el carácter, símbolo o código de control que corresponde al código ASCII especificado por la expresión entera que la sigue.
2) Vea la lista de códigos ASCII en el Capítulo 3, Sección 1, "Tabla de Códigos de Caracteres".

Ejemplo
```
A$=CHR$(&H41)
```
El carácter "A" se asigna a la variable A$.

CINT — Función

Función Esta función convierte una expresión número a entera.

Formato CINT (expresión numérica)

Descripción

1) La función CINT convierte el valor de la expresión numérica que la sigue a un entero, truncando los decimales.
2) Si el resultado de la conversión cae fuera de la gama de −32768 a +32767, se exhibe un mensaje de error.

Ejemplo

```
A%=CINT(B#)
```

CIRCLE — Instrucción

Función Esta instrucción se usa para trazar circunferencias o elipses en una pantalla gráfica

Formato

CIRCLE (coordenada X, coordenada Y), Radio [, código color]
CIRCLE (coordenada X, coordenada Y), radio, [código color], ángulo inicial
CIRCLE (coordenada X, coordenada Y), radio, [código color], [ángulo inicial], ángulo final
CIRCLE (coordenada X, coordenada Y), radio, [código color], [ángulo inicial], [ángulo final], relación del radio Y al radio X
Las coordenadas (coordenada X, coordenada Y) pueden especificarse también de modo relativo con STEP (coordenada X, coordenada Y).

Descripción

1) La instrucción CIRCLE traza una circunferencia, arco o elipse en una pantalla gráfica, con su centro situado en las coordenadas especificadas y utilizando el color indicado en el código de color.
2) El código de color ha de ser un entero entre 0 y 15.
   Si se omite, para el dibujo se usa el color especificado con la instrucción COLOR.
3) Los ángulos inicial y final deben especificarse con una expresión numérica en radianes, entre -2π y 2π.
   Si se omiten los ángulos inicial y final, se seleccionan como tales 0 y 2π, respectivamente, y se traza la pertinente circunferencia.
   Si se especifican valores negativos para ambos ángulos, el valor absoluto de los ángulos se usa para el dibujo, y en la pantalla se traza un sector (con el centro del arco enlazado a cada extremo del arco por líneas rectas).
4) La relación del radio Y al radio X debe especificarse con una expresión numérica.
   Si se omite dicha relación, se selecciona 1,0 para la misma, con lo que en la pantalla se traza un arco con el radio especificado.
   Cuando se especifica la relación "radio a lo largo del eje Y/radio a lo largo del eje X", en la pantalla se traza una elipse con la excentricidad indicada.
   El radio especificado en la instrucción corresponde al mayor de los radios Y y X.
   Si la relación especificada es menor que "1", el radio especificado corresponde al radio Y.
   Cuando la relación especificada es mayor que "1", el radio indicado corresponde al radio X.
5) Si las coordenadas se especifican de modo relativo con STEP (coordenada X, coordenada Y), el centro de una circunferencia o arco será la distancia desde el último punto de referencia (LP).
   Cuando se ejecuta la instrucción CIRCLE, el punto LP se transfiere a las coordenadas del centro.

Ejemplo

```
10 COLOR 15, 5 : SCREEN 2
20 P=3.1415927
30 CIRCLE (20, 20), 20, 1                Traza una circunferencia
40 CIRCLE (60, 60), 20,, 0, P            Traza una semicircunferencia.
50 CIRCLE (100, 100), 20, 1, -P/2, -P    Traza un sector
60 CIRCLE (140, 140), 20,,,, 2           Traza una elipse
70 CIRCLE (180, 180), 20,,,, 1/2
80 GOTO 80
```

CLEAR

Instrucción

Función

Esta instrucción inicializa todas las variables y establece el tamaño del área de memoria a disposición del usuario.

Formato

CLEAR [tamaño área cadenas [, dirección del límite superior de memoria]]

Descripción

1) La instrucción CLEAR libera toda la memoria usada para datos, sin borrar el programa existente en la memoria.
   Restituye todas las variables numéricas a cero y todas las variables de cadena a un contenido nulo ("").
   Se cierran todos los ficheros abiertos.
   El contenido de las instrucciones que empiezan con DEF (DEF FN, DEF USR, DEFINT, DEFSNG, DEFDBL, DEFSTR, etc.) se invalida.
   Se cancelan todas las definiciones de tablas de variables.
   Se cancelan los bucles FOR NEXT.
   La instrucción RETURN cesa de devolver el control desde subrutinas.
2) El tamaño del área para cadenas (en la que se almacenan las cadenas de caracteres asignadas a variables de cadena) ha de ser especificado por una expresión entera, en bytes.
   El tamaño de selección automática es 200 bytes.
3) La dirección del límite superior del área de memoria del usuario debe ser especificada por una expresión entera, en bytes. Para utilizar la memoria, véase Capítulo 3, Sección 2, "Mapa de la Memoria".
   El área situada entre la dirección límite superior indicada y &HF380 no es accesible al programa BASIC como área de programa o de datos, y el programa en código máquina escrito en esta área no será destruído.
   El valor de selección automática del límite superior es &HF380, equivalente a la máxima dirección disponible.

Ejemplo

```
10 CLEAR 1000, &HE000
```

CLOAD

Comando

Función

Este comando permite al usuario cargar un fichero de programa desde una cinta cassette a la memoria interna.

Formato

CLOAD ["nombre fichero"]

Descripción

1) El comando CLOAD se usa para cargar el fichero de programa especificado desde una cinta cassette a la memoria interna del ordenador.
   Cuando se encuentra el fichero indicado, el ordenador exhibe el mensaje "Found: nombre fichero" e inicia la operación de carga. Al terminar ésta, en la pantalla aparece el aviso "Ok".
   Cada vez que se encuentra un fichero diferente del especificado, se exhibe el mensaje "Skip: nombre fichero".
2) El nombre de fichero debe estar compuesto de seis o menos caracteres alfabéticos. Si se utilizan siete o más caracteres, el séptimo y restantes se ignoran. Si no se indica el nombre del fichero a cargar, como tal se entiende el primero que se encuentre.
3) Al ejecutar el comando CLOAD, se borran de la memoria interna todos los programas y variables, y los ficheros abiertos se cierran.
4) La velocidad de transferencia de los datos con CLOAD no necesita especificarse.
   Se selecciona automáticamente la velocidad con que se realizó antes la grabación en cinta.

Ejemplos

```
CLOAD "CAJA"
CLOAD
```

CLOAD?

Comando

Función

Este comando permite al usuario comparar un programa en cinta cassette con otro en la memoria interna, para verificar su corrección.

Formato

CLOAD? ["nombre fichero"]

Descripción

1) El comando CLOAD? se usa para comprobar si el programa actualmente en memoria coincide con otro almacenado en cinta cassette; la comparación se realiza a medida que el ordenador lee el fichero en cinta cassette. Si ambos programas coinciden, en la pantalla aparece el aviso "Ok"; de lo contrario se exhibe el mensaje "Verify error".
2) Este comando se usa generalmente después de ejecutar un comando CSAVE, a fin de asegurarse de que el programa se haya grabado correctamente en la cinta cassette.

Ejemplos

```
CLOAD?
CLOAD? "CAJA"
```

CLOSE — Instrucción

Función Esta instrucción cierra los dispositivos y ficheros especificados, o todos los que estuvieren abiertos.

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

Descripción

1) La instrucción CLOSE cierra el fichero especificado por el número de fichero. El número de fichero usado para dicho cierre puede ser luego usado con una instrucción OPEN para abrir otro fichero.
2) Pueden cerrarse varios ficheros a la vez, especificando sus respectivos números de fichero en una misma instrucción CLOSE.
3) Si se omite el número de fichero, se cerrarán todos los que estén abiertos.
4) Si se ejecuta una instrucción CLOSE para un fichero que se abrió para salida de datos del ordenador, todos los datos que queden en la memoria intermedia se transferirán a dicho fichero.
   Para completar una operación de salida de datos hacia un fichero, éste debe cerrarse.
5) Las instrucciones END, RUN, NEW y CLEAR cierran también los ficheros que encuentran abiertos.

Ejemplos

```
CLOSE #1
CLOSE 1,3
```

CLS — Instrucción

Función Esta instrucción borra el contenido de la pantalla.

Formato CLS

Descripción

1) La instrucción CLS borra todos los caracteres y figuras gráficas de la pantalla, excepto las formas.
2) En una pantalla en modo texto, la instrucción CLS no borra los indicadores de funciones existentes en la parte inferior de la pantalla; el cursor se restituye a la esquina superior izquierda (0,0).
3) En una pantalla en modo gráfico, la instrucción CLS hace que el color de fondo cambie al color que se especificó con la instrucción COLOR.
   La ubicación del último punto de referencia (LP) no cambia con la ejecución de esta instrucción.

Ejemplo

```
CLS
```

COLOR

Instrucción

Función Esta instrucción especifica los colores de visualización

Formato

COLOR color imagen [, color fondo]
COLOR [color imagen], color fondo
COLOR [color imagen], [color fondo], color borde

Descripción

1) La instrucción COLOR se usa para especificar el color del la imagen (caracteres o figuras gráficas), el color del fondo de la pantalla y el color del borde.
2) Color de la imagen
   En una pantalla modo texto, el color de los caracteres en pantalla se selecciona mediante el color de la imagen. Cuando se ejecuta la instrucción COLOR, todos los caracteres visibles en la pantalla adoptan el color especificado por el color de la imagen.
   En una pantalla de modo gráfico, el color de la imagen especifica el color de cada figura gráfica. Esta especificación es válida cuando se omite el código de color en las instrucciones gráficas (PSET, LINE, CIRCLE, DRAW, PAINT, etc.)
3) Color de fondo
   En una pantalla modo texto, al ejecutar la instrucción COLOR la misma surte su efecto inmediatamente.
   En una pantalla modo gráfico, el color de fondo surte su efecto cuando se ejecuta una instrucción CLS después de la instrucción COLOR. La instrucción COLOR determina inmediatamente el color de fondo sólo si en la instrucción PRESET se omitió el código color.
4) Color del borde
   Por tal se entiende el color del área periférica de la pantalla, en la que no pueden exhibirse caracteres ni figuras gráficas.
5) Los códigos de color relacionados a continuación pueden seleccionarse para la imagen, el fondo y el borde de la pantalla. Los valores de selección automática son 15, 4 y 4, respectivamente.

Códigos de color

0	Transparente	8	Rojo
1	Negro	9	Rojo brillante
2	Verde	10	Amarillo
3	Verde claro	11	Amarillo claro
4	Azul oscuro	12	Verde oscuro
5	Azul claro	13	Púrpura
6	Rojo oscuro	14	Gris
7	Azul celeste	15	Blanco

Ejemplo

```
COLOR 10, 15, 1
```

CONT — Comando

Función Este comando se usa para reanudar la ejecución de un programa después de una interrupción.

Formato CONT

Descripción

1) El comando CONT se usa para reanudar la ejecución de un programa interrumpido por la pulsación simultánea de las teclas [CTRL] y [STOP] o bien por la ejecución de una instrucción STOP o END o por haber ocurrido un error.
2) En general, este comando se usa en la depuración de programas. Después de interrumpir la ejecución con una instrucción STOP, usted puede comprobar el valor de variables ejecutando un comando PRINT en modo Directo, o modificarlos utilizando la instrucción LET, antes de reanudar la ejecución con el comando CONT.
3) Si el contenido del programa se ha modificado durante una interrupción, el comando CONT no podrá reanudar la ejecución.

Ejemplo

```
CONT
```

COS — Función

Función Esta función calcula el coseno trigonométrico de un número.

Formato COS (expresión numérica)

Descripción

1) La función COS calcula el coseno trigonométrico de la expresión numérica que la sigue.
2) El valor de la expresión numérica debe expresarse en radianes.
3) El resultado es siempre un número real de doble precisión, cualquiera que sea el tipo de la expresión numérica.

Ejemplo

```
A=COS(1.73)
```

CSAVE — Comando

Función Este comando se usa para grabar un fichero de programa en cinta cassette.

Formato CSAVE "nombre fichero" [, velocidad en baudios]

Descripción

1) El comando CSAVE transfiere el programa actualmente en memoria a una cinta cassette.
   Los programas almacenados en cinta pueden cargarse en la memoria con el comando CLOAD.
   Inmediatamente después de grabar un programa, use el comando CLOAD? para asegurarse de que haya quedado correctamente almacenado.
2) El nombre del fichero ha de contener seis o menos caracteres alfanuméricos. Si excede de seis, el séptimo y restantes se ignoran.
3) La velocidad en baudios puede usarse para especificar la velocidad de transferencia de datos, indicando 1 o 2.
   Si se especifica 1: 1.200 baudios
   Si se especifica 2: 2.400 baudios
   Si no se utiliza la opción de velocidad, se selecciona automáticamente la anteriormente especificada con una instrucción SCREEN o con un comando CSAVE. EL valor estándar inicial es 1.200 baudios.

Ejemplos

```
CSAVE"GASTOS"
CSAVE"GASTOS",2
```

CSNG — Función

Función Esta función convierte un entero o un número real de doble precisión a número real de simple precisión.

Formato CSNG (expresión numérica)

Descripción

1) La función CSNG convierte la expresión numérica que la sigue a un número real de simple precisión.
2) Cuando el valor de la expresión numérica contiene siete o más cifras significativas, la séptima se redondea al número entero más próximo.

CSRLIN — Función

Función Esta función da el valor de la coordenada vertical del cursor en la pantalla.

Formato CRSLIN

Descripción

1) CSRLIN da el número de la fila de una pantalla en modo texto en que está situado el cursor en dicho momento.
2) El resultado es siempre un entero entre 0 y 23; 0 corresponde a la primera fila.

Ejemplo

```
PRINT CSRLIN
```

DATA — Instrucción

Función

Esta instrucción aloja constantes numéricas y de cadena dentro de un programa (de modo que sean accesibles a instrucciones READ).

Formato

DATA constante [, constante...]

Descripción

1) La instrucción DATA aloja constantes numéricas y de cadena dentro de un programa. Se tiene entonces acceso a estas constantes utilizando instrucciones READ dentro del mismo programa.
2) La instrucción DATA es de tipo no ejecutable y puede situarse en cualquier línea del programa.
3) En una instrucción DATA pueden especificarse varias constantes separándolas con comas (,); sin embargo, la línea de programa no puede exceder de 255 caracteres.
4) Las constantes especificadas en una instrucción DATA pueden ser numéricas o de cadena.
   Las constantes numéricas pueden ser de cualquier tipo.
   Cuando las constantes de cadena se escriben en una instrucción DATA, pueden omitirse sus comillas (""); sin embargo, si una cadena contiene coma (,), dos puntos (:) o punto y coma (;) en su interior o bien hay uno o varios espacios en blanco en primer o último lugar, su entrecomillado es obligatorio.
5) El tipo de constantes colocadas en una instrucción DATA debe coincidir con el tipo de variables incorporadas en la correspondiente instrucción READ.
   Si las constantes numéricas son leídas por variables de cadena, son consideradas como constantes de cadena.
   Si las constantes de cadena son leídas por variables numéricas, aparecerá un mensaje de error.
   El tipo de constantes numéricas (enteros, números reales de simple o doble precisión) no necesita coincidir con el de las variables numéricas a que se asignan. Cuando las constantes son leídas y asignadas a variables, su tipo se convierte al de éstas últimas.
6) En un mismo programa pueden usarse varias instrucciones DATA.
   La instrucción READ lee las constantes empezando por las contenidas en la instrucción DATA que posee el número de línea más bajo.
   La instrucción RESTORE permite leer constantes de una instrucción DATA situada en la línea especificada.

Ejemplo

```
Data 1.23, &HE9, GASTOS, "X:X"
```

DEF FN

Instrucción

Función Esta instrucción define y da nombre a una función creada por el usuario.

Formato DEF FN nombre función [(argumento [, argumento...])] = expresión de la función

Descripción

1) La función DEF FN asigna la expresión de la función, especificada en el lado derecho, al nombre de función indicado en el lado izquierdo.
   Cuando se llama "FN nombre función", la expresión definida en la función se utiliza para producir el resultado deseado.
2) El nombre de función y su tipo se especifican de la misma manera que un nombre de variable y su tipo.
   El tipo del nombre de función ha de coincidir con el tipo de expresión de la función.
3) Los nombres de variables asignados a argumentos corresponden a las variables de idéntico nombre empleadas en la expresión de la función.
   Estas variables no son afectadas por la manipulación de variables de idéntico nombre utilizadas en otras partes del mismo programa.
   Si en la expresión de la función hay un nombre de variable que no corresponde a una variable asignada a un argumento, el valor de la variable de idéntico nombre en otra parte del programa será asignada a la variable del argumento.
4) Esta función no puede usarse en modo Directo.

Ejemplo

```
10 DEF FNS (X, Y)=SQR (X*X+Y*Y)
20 A=FNS (4,3) :PRINT A
30 END
```

DEF USR

Instrucción

Función Esta instrucción define la primera dirección de una subrutina escrita en lenguaje máquina.

Formato DEF USR [número] = primera dirección

Descripción

1) La instrucción DEF USR define la primera dirección de una subrutina código máquina llamada por la función USR.
2) El número puede oscilar entre 0 y 9, lo que permite especificar hasta diez subrutinas en código máquina.
   Si se omite el número, se presupone que es cero.

Ejemplo DEF USR=&HF100

DEFDBL — Instrucción

Función Esta instrucción se usa para declarar que las variables que la siguen son de tipo real de doble precisión.

Formato
DEFDBL carácter alfabético-carácter alfabético [, carácter alfabético-carácter alfabético...]
DEFDBL carácter alfabético [, carácter alfabético...]

Descripción

1) La instrucción DEFDBL se usa para declarar que las variables cuyo nombre empieza con el carácter alfabético especificado o con un carácter de la gama especificada y no va seguido de símbolos de declaración de tipo, son variables de tipo real de doble precisión.
   Carácter alfabético-carácter alfabético:
   Las variables cuyo nombre empieza con un carácter comprendido en la gama especificada de caracteres alfabéticos son definidas como pertenecientes al tipo real de doble precisión.
   Carácter alfabético:
   Las variables cuyo nombre empieza con el carácter especificado son definidas como pertenecientes al tipo real de doble precisión.
2) Los nombres de variables con símbolos de declaración de tipo (%, 1, # o $) no son afectados y conservan el tipo definido por dichos símbolos.

Ejemplo

DEFDBL A, D — F Las variables AB, D, E1, etc, son definidas como de tipo real de precisión doble.

DEFINT — Instrucción

Función Esta instrucción se usa para declarar variables del tipo entero.

Formato
DEFINT carácter alfabético-carácter alfabético [, carácter alfabético-carácter alfabético...]
DEFINT carácter alfabético [, carácter alfabético...]

Descripción

1) La instrucción DEFINT se usa para declarar que las variables cuyo nombre empieza con el carácter especificado o con un carácter de la gama especificada y carecen de símbolos de declaración de tipo, son variables del tipo entero.
   Carácter alfabético-carácter alfabético:
   Las variables cuyo nombre empieza con un carácter en la gama especificada son definidos como pertenecientes al tipo entero.
   Carácter alfabético:
   Las variables cuyo nombre empieza con el carácter especificado son definidas como pertenecientes al tipo entero.
2) Los nombres de variables con símbolos de declaración de tipo (%, 1, # o $) no son afectados y conservan el tipo definido por dichos símbolos.

Ejemplos

DEFINT A, D — F Las variables AB, D, E1, etc; son definidas como de tipo entero.

DEFSNG

Instrucción

Función Esta instrucción se usa para declarar variables de tipo real de simple precisión.

Formato

DEFSNG carácter alfabético-carácter-alfabético [, carácter alfabético-carácter alfabético...]

DEFSNG carácter alfabético [, carácter alfabético...]

Descripción

1) La instrucción DEFSNG se usa para declarar que las variables cuyo nombre empieza con el carácter especificado o con uno de los caracteres comprendidos en la gama indicada y carecen de símbolos de declaración de tipo, pertenecen al tipo real de simple precisión.

   Carácter alfabético-carácter alfabético:

   Las variables cuyo nombre empieza con uno de los caracteres comprendidos en la gama especificada son definidas como tipo real de simple precisión.

   Carácter alfabético:

   Las variables cuyo nombre empieza con el carácter especificado son definidas como de tipo real de simple precisión.

2) Las variables con símbolos de declaración de tipo (%, 1, # o $) no son afectadas y conservan el tipo especificado por estos símbolos.

Ejemplo

DEFSNG A, D — F Las variables AB, D, E1, etc.,son definidas como pertenecientes al tipo real de simple precisión.

DEFSTR

Instrucción

Función Esta instrucción se usa para declarar variables de cadena.

Formato

DEFSTR carácter alfabético-carácter alfabético [, carácter alfabético-carácter alfabético...]

DEFSTR carácter alfabético [, carácter alfabético...]

Descripción

1) La instrucción DEFSTR se usa para declarar que las variables cuyo nombre empieza con el carácter especificado o con un carácter comprendido en la gama indicada y carecen de símbolos de declaración de tipo, son variables de cadena.

   Carácter alfabético-carácter alfabético:

   Las variables cuyo nombre empieza con un carácter de la gama especificada son definidas como de tipo de cadena.

   Carácter alfabético:

   Las variables cuyo nombre empieza con el carácter especificado son definidas como variables de cadena.

2) Las variables con símbolos de declaración de tipo (%, 1, # o $) no son afectadas y conservan el tipo especificado por estos símbolos.

Ejemplo

DEFSTR A, D — F Las variables AB, D, E1, etc.,son definidas como variables de cadena.

DELETE — Comando

Función

Este comando borra líneas de programa.

Formato

DELETE número línea
DELETE [número línea inicial] — número línea final

Descripción

1) El comando DELETE borra la línea o gama de líneas especificadas.
Cuando el comando va seguido de sólo un número de línea, se borra solamente ésta.
Si se especifican la línea inicial y final, se borra la gama especificada.
Si se indica sólo la línea final, se borran todas las líneas de programa hasta ésta última.

2) Si se usa un punto (.) en vez de un número de línea, ello se entiende como la última línea de ejecución, es decir la línea de programa última ejecutada por BASIC.
Cuando la ejecución de un programa se interrumpe a causa de una instrucción errónea, por última línea de ejecución se entiende la línea de programa en que se produjo el error.
Después de ejecutar un comando LIST o LLIST, por última línea de ejecución se entiende la última línea especificada.

3) Si el comando DELETE se usa en modo programa, el sistema vuelve al modo comando después de ejecutarlo.

Ejemplos

DELETE 10	Borra la línea de programa 10.
DELETE 10 — 50	Borra de la línea 10 a la 50.
DELETE — 50	Borra todas las líneas del programa de la primera hasta la nº 50.

DIM

Instrucción

Función

Esta instrucción especifica los valores máximos para subíndices de variables de tabla y reserva el pertinente espacio de memoria.

Formato

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

Descripción

1) La instrucción DIM especifica los valores máximos de subíndices de variables de tabla y de reserva de memoria para los datos asignados a dichas variables.
2) Las variables de tabla especificadas en una instrucción DIM pueden entonces recibir subíndices desde cero hasta los valores máximos de subíndices especificados en la instrucción.
   Cuando se usa una tabla de variables no definida en la instrucción DIM, los subíndices que pueden usarse van de 0 a 10.
3) El máximo valor de subíndices puede indicarse con una expresión entera desde 0 hasta la máxima ubicación disponible de memoria. Si el máximo valor de subíndices excede de la capacidad de memoria disponible, al ejecutar la instrucción DIM para tablas de variables numéricas se exhibirá un mensaje de error. En el caso de tablas de variables de cadena, el error aparecerá cuando se ejecute la instrucción DIM o cuando la cadena de caracteres transferida a variables de cadena exceda del área disponible para cadenas en la memoria. Las áreas de memoria disponibles para variables se explican en el Capítulo 1, Sección 3 "CONSTANTES Y VARIABLES".
   El tamaño del área de memoria no usada puede verse con la función FRE.
4) Pueden especificarse hasta 255 subíndices (dimensiones).
5) No es posible volver a definir una tabla de variables ya definida en una instrucción DIM.
6) Todas las variables, inmediatamente después de su definición en la instrucción DIM, tienen el valor de cero si son variables numéricas y el valor nulo ("") si se trata de variables de cadena.
7) Las tablas de variables pueden borrarse con una instrucción ERASE o CLEAR. Cuando se agota el área de memoria, las tablas de variables de las que pueda prescindirse pueden borrarse con dichas instrucciones. Al inicializar tablas de variables o actualizar el máximo valor de subíndices, puede borrarse temporalmente una tabla de variables en cuestión a fin de poder volverla a definir con la instrucción DIM.

Ejemplo

```
DIM A (25,3)
DIM A(3), B(5,8,7)
```

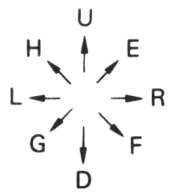
DRAW

Instrucción

Función Esta instrucción se usa para trazar figuras en una pantalla gráfica.

Formato DRAW expresión de cadena

Descripción



1) La instrucción DRAW permite dibujar diversas figuras gráficas en la pantalla usando macrocomandos especificados en la expresión de cadena. Los macrocomandos gráficos se usan principalmente para desplazar un punto de referencia, el cual opera como un pincel de pintor.
2) La expresión de cadena consiste en uno o más macrocomandos gráficos.
3) Lista de macrocomandos gráficos:

U distancia
- Desplaza el punto de referencia (PR) la distancia especificada hacia arriba.

D distancia
- Desplaza el PR la distancia especificada hacia abajo.

L distancia
- Desplaza el PR la distancia especificada hacia la izquierda.

R distancia
- Desplaza el PR la distancia especificada hacia la derecha.

E distancia
- Desplaza el PR la distancia especificada diagonalmente hacia la derecha y arriba.

F distancia
- Desplaza el PR la distancia especificada diagonalmente hacia la derecha y abajo.

G distancia
- Desplaza el PR diagonalmente la distancia especificada hacia la izquierda y abajo.

H distancia
- Desplaza el PR diagonalmente la distancia especificada hacia la izquierda y arriba.

M coordenada horizontal, coordenada vertical
- Desplaza el PR a la posición especificada.
- Si las coordenadas horizontal y vertical llevan el signo "+" o "–", especifican coordenadas relativas.

B
- Desplaza el PR sin trazar su trayectoria.

N
- Desplaza el PR mientras traza su trayectoria.
- Después de haber sido desplazado, el PR designa un punto inicial para otra trayectoria.

A ángulo
- Gira las figuras dibujadas con los comandos U, D, L, R, E, F, G, H o M (éste último sólo en el caso de coordenadas relativas); el giro se efectúa en incrementos de 90 grados.
- El ángulo se especifica con un entero desde 0 a 3: 0=0°, 1=90°, 2=180° y 3=270°.

C código color
- Traza figuras usando el color especificado por el código color.

S factor escala
- La distancia especificada en los comandos U, D, L, R, E, F, G, H y M es multiplicada por el valor del factor escala.
- El factor escala puede ser un entero entre 1 y 255.

X variable de cadena;
- Hace que el ordenador ejecute los macrocomandos gráficos asignados a la variable de cadena.

= variable numérica;
- Permite el uso de la variable numérica para distancia, ángulo, código color, factor escala, etc.

END — Instrucción

Función Esta instrucción se incorpora en un programa para terminar la ejecución

Formato END

Descripción La instrucción END cesa la ejecución del programa, cierra los ficheros abiertos y restituye el ordenador al modo Comando.

EOF — Función

Función Esta función da un valor que indica si se ha llegado o no al final de un fichero.

Formato EOF (número fichero)

Descripción

1) La función EOF da el valor -1 ó 0, para indicar si quedan o no datos a transferir del fichero a la memoria interna del ordenador.
   -1 se ha llegado al final del fichero.
   0 no se ha llegado todavía al final del fichero.
2) El fichero especificado por el número fichero debe abrirse con la instrucción OPEN en modo de entrada al ordenador.

ERASE — Instrucción

Función Esta instrucción se usa para borrar una o más tablas de variables de la memoria interna.

Formato ERASE nombre tabla [, nombre tabla...]

Descripción

1) La instrucción ERASE borra una o más tablas de variables de la memoria.
2) Esta instrucción puede usarse para borrar tablas de variables innecesarias, a fin de aumentar la memoria disponible.
3) Las tablas de variables borradas pueden luego volverse a definir con la instrucción DIM.

Ejemplo ERASE A,B

ERL — Función

Función Esta función da el número de la línea de programa en que se ha producido un error.

Formato ERL

Descripción

1) La función ERL se usa principalmente en una rutina de tratamiento de errores especificada en una instrucción ON ERROR GOTO.
2) En una instrucción IF, esta función debe situarse en el lado izquierdo de un operador relacional (con un número de línea en el lado derecho del operador). El número de línea escrito en la instrucción IF se cambia automáticamente cuando se ejecuta el comando RETURN, a condición de que dicho número de línea esté en el lado derecho del operador relacional.
3) Si se genera un error cuando el sistema está en modo Directo, la función ERL muestra el valor 65535.

Ejemplo

```
PRINT ERL
IF ERL=120 THEN PRINT "ERROR"
```

ERR — Función

Función Esta función muestra un código de error.

Formato ERR

Descripción

1) La función ERR se usa primordialmente en una rutina de tratamiento de errores, especificada en una instrucción ON ERROR GOTO.
2) Los códigos de error se relacionan en el Capítulo 3, Sección 5, "Lista de Códigos de Error".

Ejemplo

```
PRINT ERR
IF ERR=4 THEN PRINT "ERROR 4"
```

ERROR — Instrucción

Función Esta instrucción hace que el sistema genere deliberadamente un error.

Formato ERROR código error

Descripción La instrucción ERROR se usa para generar deliberadamente un error, para simulación de errores.

EXP — Función

Función Esta función calcula la potencia de "e", la base de logaritmos naturales.

Formato EXP (expresión numérica)

Descripción

1) La función EXP calcula la potencia de "e", la base de logaritmos naturales, usando el valor de la expresión numérica como exponente.
2) El valor de la expresión numérica puede oscilar entre -147.3654459516 y 145.0628605862.
3) El resultado es siempre un número real de doble precisión, cualquiera que sea el tipo de la expresión numérica.

Ejemplo

```
PRINT EXP (2.4)
```

FIX — Función

Función Esta función da la parte entera de un valor.

Formato FIX (expresión numérica)

Descripción

1) La función FIX da la parte entera del valor de la expresión numérica, truncando todas las cifras decimales.
2) Las funciones FIX e INT dan diferentes enteros cuando se trata de un valor negativo:

```
A = FIX(-1.23)      se asigna -1 a la variable A
A = INT(-1.23)      se asigna -2 a la variable A
```

FOR

Instrucción

Función

Esta instrucción ejecuta una serie de instrucciones dentro de un bucle formado por las instrucciones FOR y NEXT; dicha ejecución se repite el número de veces especificado.

Formato

FOR variable numérica=valor inicial TO valor final [STEP incremento]

Descripción

1) La instrucción FOR debe ir siempre seguida de una instrucción NEXT para completar un bucle.
2) El bucle de ejecución FOR-NEXT opera de la forma siguiente:
   Cuando se ejecuta la instrucción FOR, el valor inicial se asigna a la variable numérica, y cada vez que se encuentra en el bucle la instrucción NEXT, el incremento se añade al valor de la variable numérica.
   Cuando el valor de la variable numérica es igual o superior al valor final, el control pasa a la instrucción que sigue a la instrucción NEXT. Si no alcanza el valor final, el control se restituye a la instrucción que sigue a la instrucción FOR.

   Ejemplo
   ```
   10  FOR K=2 TO   10  STEP3
   20  PRINT K
   30  NEXT K
   40  END
   ```
3) El valor inicial, el valor final y el incremento se especifican mediante expresiones numéricas.
   El incremento puede ser un valor negativo. En dicho caso, el control pasa a la instrucción que sigue a NEXT cuando el valor de la variable numérica es igual o inferior al valor final.

   Ejemplo
   ```
   10  A=2
   20  FOR K=A TO   -3*A  STEP-3
   30  PRINT K
   40  NEXT K
   50  END
   ```
   Si se especifica cero como incremento, el bucle FOR-NEXT no tiene fin.
   Si se omite "STEP incremento", se le da el valor de+1.

   Ejemplo
   ```
   10  FOR K=1 TO 10
   20  PRINT K
   30  NEXT K
   40  END
   ```
4) El bucle FOR-NEXT se ejecuta una sola vez en los casos siguientes:
   El incremento es un valor positivo, y el valor inicial es mayor que el valor final.
   El incremento es negativo, y el valor inicial es menor que el valor final.
5) La variable numérica ha de ser simple; si se especifica una tabla de variables, aparece un mensaje de error.

6) Un bucle FOR-NEXT puede incluir otro bucle FOR-NEXT (anidamiento). El bucle mayor debe incluir completamente al menor; si parte del bucle menor está fuera del mayor, aparece un mensaje de error.

Ejemplo

Anidamiento completo

```
10 FOR K=1 TO 5
20 FOR J=1 TO 3
30 PRINT K, J
40 NEXT J
50 NEXT K
60 END
```

Bucle cruzado (error)

```
10 FOR K=1 TO 5
20 FOR J=1 TO 3
30 PRINT K, J
40 NEXT K
50 NEXT J
60 END
```

7) Si se ejecuta una instrucción CLEAR o MAXFILES dentro de un bucle FOR-NEXT, el bucle se interrumpe.

FRE — Función

Función Esta función muestra el tamaño del área de memoria no usada.

Formato
FRE (expresión numérica)
FRE (expresión de cadena)

Descripciones

1) La expresión numérica o de cadena es ficticia y puede tener cualquier valor. El resultado se da en bytes.
2) FRE (expresión numérica)
   Da el tamaño de memoria libre dentro del área del usuario.
   El área libre es una parte del área del usuario que no se usa para contener programas ni datos.
3) FRE (expresión de cadena)
   Muestra la magnitud de memoria no usada del área reservada a cadenas. Al ejecutar "FRE (expresión de cadena)", se borran las cadenas de caracteres no necesarias de dicha área, a fin de ampliar el área disponible para nuevas cadenas.
4) Para conocer las áreas asignadas a variables ver el Capítulo 3, Sección 2 "Mapa de Memoria".

Ejemplos

```
PRINT FRE (0).
PRINT FRE ( "" )
```

GOSUB

Instrucción

Función Esta instrucción llama a una subrutina.

Formato GOSUB número línea

Descripción Esta instrucción permite pasar a una subrutina cuya primera línea es especificada por el número de línea que la sigue.
Una instrucción RETURN colocada en la subrutina hace que la ejecución vuelva a la instrucción que sigue a GOSUB en el programa principal.

⟨Subrutina⟩

1) Una subrutina es una secuencia independiente de instrucciones que termina siempre con una instrucción RETURN.
2) Las subrutinas pueden llamarse desde cualquier punto del programa principal siempre que sea requerido para efectuar la misma operación.
3) Una subrutina puede llamar a otra. Ello se denomina "anidamiento de subrutinas".
   Cada vez que se anida una subrutina, se usa la correspondiente área de memoria de la pila (stack). El anidamiento está permitido en tanto haya disponible área de la pila.
4) En una subrutina puede haber más de una instrucción RETURN.
5) Si en una subrutina se ejecuta una instrucción CLEAR o MAXFILES, la instrucción RETURN existente en la subrutina no puede restituir el control al programa principal.

Ejemplo

```
10  GOSUB 100
20  K=0 :GOSUB 200
30  K=1 :GOSUB 200
40  END
100 'Subrutina 100
110 PRINT "SUB100"
120 RETURN
200 'Subrutina 200
210 PRINT "SUB200"
220 IF K=1 THEN RETURN
230 GOSUB 100
240 RETURN
```

GOTO — Instrucción

Función Esta instrucción crea una bifurcación a la línea especificada, en la que continúa la ejecución.

Formato GOTO número línea
GO TO número línea (se permite dejar 1 espacio entre GO y TO).

Descripción
1) La instrucción GOTO crea una bifurcación a la línea especificada, en la que continúa la ejecución.
2) Si la instrucción GOTO se ejecuta en modo Directo, la ejecución del programa empezará en la línea indicada.
A diferencia del comando RUN, la instrucción GOTO no inicializa variables ni cierra ficheros.

Ejemplo
```
10   GOTO 100
20   PRINT "20"
100  PRINT "100"
110  END
```

HEX$ — Función

Función Esta función convierte el valor que la sigue a una cadena de caracteres de notación hexadecimal.

Formato HEX$ (expresión entera)

Descripción La función HEX$ convierte el valor de la expresión entera a una cadena de caracteres en notacion hexadecimal.

Ejemplo
```
A$=HEX$(41)
```

IF — Instrucción

Función Esta instrucción se usa para tomar decisiones condicionales.

Formato
IF expresión numérica THEN instrucción [:instrucción...] [ELSE instrucción [:Instrucción...]]
IF expresión numérica THEN instrucción [:instrucción...] [ELSE número línea]
IF expresión numérica THEN número línea [ELSE instrucción [:instrucción...]
IF expresión numérica THEN número línea [ELSE número línea]
IF expresión numérica GOTO número línea [ELSE instrucción [:instrucción...]]
IF expresión numérica GOTO número línea [ELSE número línea]

Descripción
1) La expresión IF decide la ruta a seguir en la ejecución del programa según el resultado de una expresión numérica; si ésta es relacional o lógica, la decisión se adopta según sea verdadera o falsa.

2) Si el valor de la expresión numérica no es cero o la condición es verdadera, se ejecuta la instrucción que sigue a THEN, o el programa pasa al número de línea que sigue a THEN o GOTO.

Ejemplo

```
10   A=1 : B=-1
20   IF A=-1 THEN PRINT"A=1"
30   IF B THEN PRINT "B<>0"
40   END
```

3) Si el valor de la expresión numérica es cero o la condición contenida en la expresión relacional no se satisface (es falsa), se ejecuta la instrucción que sigue a ELSE, o el control pasa al número de línea especificado en la cláusula ELSE.
Si no hay cláusula ELSE, la ejecución continúa en la línea siguiente.

Ejemplos

```
10   A=1 : B=0
20   IF A=0 THEN PRINT"A=0"ELSE PRINT"A<>0"
30   IF B THEN PRINT "B<>0"
40   PRINT "40"
50   END
```

4) La cláusula THEN o ELSE en una instrucción IF puede contener otra instrucción IF (anidamiento).
El número de cláusulas THEN puede diferir del número de cláusulas ELSE. Cada cláusula ELSE forma pareja con la cláusula THEN más próxima.

Ejemplo

```
IF A$="X" THEN IF B$="Y"THEN PRINT"Y"ELSE PRINT"X"ELSE 100
```

5) En expresiones relacionales se usan operadores relacionales (=, <>, <, >, =<, ó =>) para comparar las magnitudes de valores numéricos o cadenas de caracteres. En expresiones lógicas, se usan operadores lógicos (AND, OR, NOT, etc.) para definir la correlación lógica entre diferentes expresiones relacionales.
Las expresiones relacionales y lógicas se tratan detenidamente en el Capítulo 1, Sección 4, OPERACIONES.

Ejemplo

```
10   INPUT "Y/N/E?";A$
20   IF A$="E" THEN 40
30   IF A$<>"Y" OR A$<>"N" THEN PRINT "Input Miss="
     ;A$:GOTO 10 ELSE 100
40   PRINT "End"
50   END
100  IF A$="Y" THEN PRINT "Yes":GOTO 10
110  IF A$="N" THEN PRINT "No":GOTO 10
```

INKEY$ — Función

Función Esta función se usa para leer un carácter desde el teclado.

Formato INKEY$

Descripción

1) La función INKEY$ da el carácter de la tecla que se ha pulsado en el teclado. Si no se ha pulsado ninguna tecla, da una cadena vacía ("").
Cada vez que se pulsa una tecla, el carácter de la tecla se transfiere al almacenamiento intermedio (buffer) del teclado. Cuando se ejecuta la función INKEY$, lee el último carácter transferido desde el teclado al buffer.
Por consiguiente, la función INKEY$ puede leer un carácter que fué transferido al buffer antes de que la misma fuera ejecutada.
2) Si no se ha pulsado ninguna tecla, la función INKEY$ al ser ejcutada no espera por ninguna pulsación de tecla, sino que pasa el control a la siguiente instrucción.
Si usted desea que el ordenador espere a que le produzca una pulsación de tecla, debe confeccionar un programa que compruebe cuando se efectúa una pulsación.
3) La función INKEY$ no exhibe en la pantalla la tecla pulsada.

Ejemplo

```
10  IF INKEY$<>"" THEN 10      Borra el buffer del teclado.
20  K$=INKEY$
30  IF K$="" THEN 20           Espera a que se pulse una tecla.
40  PRINT " KEY=";K$
50  IF K$="E" THEN 100
60  GOTO 20
100 END
```

INP — Función

Función Esta función se usa para leer datos desde una vía de acceso (port) de E/S.

Formato INP (dirección del port)

Descripción

1) La función INP lee 1 byte de datos desde la vía de acceso especificada en la dirección.
2) La dirección ha de ser un entero entre 0 y 255.
3) Las direcciones de vías de acceso se explican en el Capítulo 3, Sección 4, "Mapa de entrada/salida".

Ejemplo

```
10 A=INP(&HA8)
20 A$=BIN$(A)
30 PRINT RIGHT$("0000000"+A$, 8)
40 END
```

INPUT

Instrucción

Función

Esta instrucción se usa para entrar números o cadenas por el teclado y asignarlos a variables.

Formato

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

Descripción

1) La instrucción INPUT, al ser ejecutada, muestra una frase, el signo de interrogación (?) y un espacio en blanco, y espera a que usted escriba en el teclado los datos (números o cadenas).
2) Los datos que se escriben en el teclado aparecen en la pantalla a continuación del espacio en blanco. Al pulsar la tecla [RETURN], la información se transfiere a la variable especificada.
   La información mecanografiada puede corregirse con las teclas de edición de pantalla antes de pulsar [RETURN].
3) Si no se escribe ningún dato y se pulsa simplemente la tecla [RETURN], el valor de la variable sigue siendo el mismo que antes de ejecutarse la instrucción INPUT.
   Sin embargo, si la variable en cuestión se usa por primera vez, al pulsar [RETURN] se asignará cero a las variables numéricas y una cadena vacía ("") a las variables de caracteres.
4) Los espacios en blanco antepuestos o postpuestos a la información mecanografiada se omiten.
5) Si en la instrucción INPUT se omite la frase y las comillas, en la pantalla se exhibirá sólo el interrogante (?) y un espacio en blanco.
6) Cuando usted desee escribir en el teclado varios datos para su asignación a las respectivas variables, cada dato debe ir separado por una coma (,).
   Si el número de datos mecanografiados no coincide con el de las variables especificadas, ocurrirá lo siguiente:
   Si el número de datos es inferior al de variables:
   En la pantalla aparecerán dos interrogantes (??), indicando que usted debe continuar escribiendo datos.
   Si el número de datos excede al de variables:
   En la pantalla aparecerá el mensaje "Extra ignored", con el significado de que los datos sobrantes se omitirán y que la ejecución continuará.
7) Si mecanografía una cadena de caracteres para una variable numérica, aparecerá el mensaje "Redo from start" (volver a empezar), y el sistema esperará a que usted vuelva a escribir datos.
8) Cuando se escriba una cadena de caracteres con destino a una variable de cadena, pueden omitirse las comillas (").
   Sin embargo, si la cadena de caracteres contiene una o más comas o lleva uno o más espacios en blanco antepuestos o postpuestos a la cadena, la misma debe entrecomillarse.
9) Cuando se ejecuta una instrucción INPUT en el modo Gráfico, el sistema se restituye automáticamente al modo Texto.

Ejemplo

```
10 INPUT "ABC";A$
20 PRINT A$
30 INPUT B
40 PRINT B
50 INPUT C, D, E$
60 PRINT C;D;E$
70 END
```

INPUT# — Instrucción

Función Esta instrucción lee datos de un fichero y los asigna a variables del programa.

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

Descripción

1) La instrucción INPUT# lee datos (valores numéricos o cadenas de caracteres) del fichero especificado por el número de fichero y los asigna a las variables indicadas.
2) El fichero especificado por el número de fichero debe abrirse primero con una instrucción OPEN, para modo de entrada.
3) Los datos legibles con la instrucción INPUT# son los grabados en el fichero con la instrucción PRINT#.
4) Los tipos de variables deben coincidir con los tipos de los correspondientes datos.

Ejemplo:

```
INPUT# 1,A,B
```

INPUT$ — Función

Función Esta función da como resultado una cadena de un número especificado de caracteres, leídos desde el teclado o del fichero indicado.

Formato

INPUT$ (expresión entera, [#] número fichero)
INPUT$ (expresión entera)

Descripción

1) Cuando se especifica INPUT$ (expresión entera, [#] número fichero):
   La función da una cadena de caracteres cuya longitud corresponde a la expresión entera y que es leída del fichero especificado por el número de fichero.
2) Cuando se especifica INPUT$ (expresión entera):
   La función da una cadena de caracteres cuya longitud corresponde a la expresión entera y que es leída del teclado.
   Cuando se ha dado la longitud de cadena exigida, la ejecución continúa en la siguiente instrucción, sin que sea necesario pulsar la tecla [RETURN].
   La cadena de caracteres mecanografiada no aparece en la pantalla.
3) Esta función puede leer cualquier código de caracteres excepto los de las teclas [CTRL] y [STOP]. Entre los códigos de caracteres legibles se encuentran los de CR (&H0D) y LF (&H0A).

Ejemplo

```
10 A$=INPUT$(2)
20 PRINT A$
30 END
```

INSTR — Función

Función Esta función da la posición de un carácter especificado dentro de una cadena especificada.

Formato INSTR ([expresión entera,] expresión de cadena 1, expresión de cadena 2).

Descripción

1) La función INSTR busca en la cadena indicada en la expresión 1 la cadena especificada en la expresión 2, y da la posición del primer carácter de la cadena 1.
2) La expresión entera indica la primera posición de caracteres a partir de la cual empezara la búsqueda.
   El valor de la expresión entera ha de ser un entero en la gama 1 — 255. Si se omite, la búsqueda empieza en la posición del primer carácter de la cadena 1.
3) Si en la cadena 1 no se encuentra la cadena 2, o el valor de la expresión entera excede del número de caracteres existentes en la cadena 1, la función dará como resultado cero.

Ejemplo

```
10 A$="ABCDEFGABC"
20 B$="FG"
30 A=INSTR (AS,B$)
40 B=INSTR (3,A$,"ABC")
50 PRITN A, B
60 END
```

INT — Función

Función Esta función da un entero que no excede del valor de una expresión numérica.

Formato INT (expresión numérica)

Descripción

1) La función INT da un entero que no excede del valor de la expresión numérica que la sigue, truncando todos los decimales del valor de la expresión.
2) Si se trata de valores negativos, las funciones INT y FIX dan diferentes enteros; en cambio, con valores positivos, su resultado es idéntico:

```
A=INT(-1.23)        se asigna -2 a la variable A
A=FIX(-1.23)        se asigna -1 a la variable A
```

INTERVAL ON

Instrucción

Función Esta instrucción habilita interrupciones determinadas por el temporizador de intervalos.

Formato INTERVAL ON

Descripción Cuando la instrucción INTERVAL ON ha sido ejecutada, la rutina de tratamiento de interrupciones especificada en la instrucción ON INTERVAL GOSUB se ejecutará cada vez que transcurra el intervalo especificado en la misma instrucción.

INTERVAL OFF

Instrucción

Función Esta instrucción deshabilita las interrupciones determinadas por el temporizador de intervalos.

Formato INTERVAL OFF

Descripción Cuando la instrucción INTERVAL OFF ha sido ejecutada, quedan deshabilitadas las interrupciones determinadas por el temporizador de intervalos (la rutina de tratamiento de interrupciones especificada en la instrucción ON INTERVAL GOSUB no será ejecutada cuando haya transcurrido el intervalo especificado en la misma).

INTERVAL STOP

Instrucción

Función Esta instrucción retiene una interrupción determinada por el temporizador de intervalos.

Formato INTERVAL STOP

Descripción

1) La instrucción INTERVAL STOP retiene interrupciones del temporizador de intervalos (el intervalo es especificado en la instrucción ON INTERVAL GOSUB) hasta que se ejecuta la instrucción INTERVAL ON.
   Después de haberse ejecutado una instrucción INTERVAL STOP, cualquier intervalo especificado que transcurra determinará la ejecución de la rutina de tratamiento de interrupciones cuando se ejecute ulteriormente una instrucción INTERVAL ON.
2) La instrucción INTERVAL STOP retiene la ejecución de la instrucción INTERVAL ON. Si no se ha ejecutado ninguna instrucción INTERVAL ON, las interrupciones no son retenidas, sino simplemente ignoradas.

KEY — Comando

Función Este comando asigna una cadena de caracteres a cualquiera de las teclas de funciones.

Formato KEY expresión entera, expresión de cadena

Descripción

1) El comando KEY asigna el contenido de la expresión de cadena de caracteres a la tecla de funciones especificada en la expresión entera.
2) El valor de la expresión entera ha de ser un entero en la gama 1 — 10, que corresponda al número de la tecla de funciones.
3) En la cadena pueden usarse hasta 15 carácteres; si su número es superior, el resto es ignorado.
   Si en la expresión de cadena se incluye algún código de control, deberá ir precedido de un signo más (+) una función CHR$.

Ejemplo KEY 2, "SCREEN 0"+CHR$ (&H0D)

KEY LIST — Comando

Función Este comando muestra en la pantalla una lista de todas las designaciones de las teclas de funciones.

Formato KEY LIST

Descripción El comando KEY LIST exhibe en pantalla una lista de todas las cadenas de caracteres asignadas a las teclas de funciones; la pantalla ha de estar en modo texto.

KEY ON — Instrucción

Función Esta instrucción muestra las designaciones de las teclas de funciones en la parte inferior de la pantalla.

Formato KEY ON

Descripción

1) La instrucción KEY ON se usa para exhibir en la parte inferior de la pantalla los primeros 5 caracteres de las cadenas asignadas a las teclas de funciones.
2) Normalmente en la pantalla aparecen las designaciones de las teclas de funciones 1 — 5; para exhibir las de las teclas 6 — 10 debe pulsarse la tecla [SHIFT].

KEY OFF — Instrucción

Función Esta instrucción desactiva la visualización de las designaciones de las teclas de funciones en la parte inferior de la pantalla.

Formato KEY OFF

Descripción La instrucción KEY OFF se usa para que desaparezcan las designaciones de teclas de funciones visibles en la parte inferior de la pantalla.

KEY (n) ON

Instrucción

Función Esta instrucción se usa para habilitar interrupciones por medio de las teclas de funciones.

Formato KEY (expresión entera) ON

Descripción

1) Cuando se pulsa la tecla de funciones especificada por la expresión entera después de que se haya ejecutado la instrucción KEY (n) ON, el programa pasa a la rutina de tratamiento de interrupciones especificada en la instrucción ON KEY GOSUB.
2) Cuando KEY (n) ON es válida, todas las designaciones de las teclas de función son ignoradas.

KEY (n) OFF

Instrucción

Función Esta instrucción se usa para desactivar interrupciones por las teclas de funciones.

Formato KEY (expresión entera) OFF

Descripción Cuando se ha ejecutado la instrucción KEY (n) OFF no se producirá ninguna interrupción cuando se pulse la tecla de función especificada en la expresión entera.

KEY (n) STOP

Instrucción

Función Esta instrucción se usa para retener interrupciones de una tecla de funciones.

Formato KEY (expresión entera) STOP

Descripción

1) La instrucción KEY (n) STOP se usa para retener una interrupción causada por la pulsación de la tecla de función especificada por la expresión entera, hasta que se ejecute una instrucción KEY (n) ON.
   Si la tecla de función especificada se pulsa después de que se haya ejecutado la instrucción KEY (n) STOP, la rutina de tratamiento de interrupciones será ejecutada cuando se ejecute la instrucción KEY (n) ON.
2) La instrucción KEY (n) STOP se usa para retener la ejecución de la instrucción KEY (n) ON; si ésta no ha sido ejecutada, las interrupciones no serán retenidas, sino simplemente no se tendrán en cuenta.
3) Cuando se ha ejecutado al instrucción KEY (n) STOP, queda sin efecto la designación de la tecla de funciones especificada.

LEFT$ — Función

Función Esta función da un número especificado de caracteres de una cadena de caracteres, empezando con el carácter más a la izquierda.

Formato LEFTS (expresión de cadena, número caracteres)

Descripción

1) La función LEFT$ da el número indicado de caracteres de la cadena especificada en la expresión, empezando con el carácter más a la izquierda.
2) Cuando el número indicado de caracteres es igual o mayor que el número de caracteres existentes en la cadena, se dan como resultado todos los caracteres de la cadena.
3) Si se especifica cero como número de caracteres, el resultado es una cadena vacía ("")
4) La cabecera de identificación de signos gráficos (&H01) cuenta como un carácter, es decir un símbolo gráfico se contará como dos caracteres.

Ejemplo

```
A$=LEFT$("ABCDEF",3)      "ABC" se asigna a la variable A$.
```

LEN — Función

Función Esta función da la longitud de una cadena.

Formato LEN (expresión de cadena)

Descripción

1) La función LEN da el número total de caracteres existentes en la cadena especificada por la expresión de cadena que la sigue.
2) Un código de control o un espacio en blanco cuentan también como caracteres.
3) La cabecera de identificación de signos gráficos (&H01) cuenta también como un carácter. Ello significa que cada símbolo gráfico se cuenta como dos caracteres.

Ejemplo

```
10 A$="ABC"+CHR$(&H0D)
20 A=LEN(A$) : B=LEN(" ☺ ")
30 PRINT A, B
40 END
```

LET — Instrucción

Función Esta instrucción se usa para asignar el valor de una expresión a una variable.

Formato

```
LET variable=expresión
Variable=expresión
```

Descripción

1) La instrucción LET asigna el valor de la expresión a la variable.
2) La palabra LET puede omitirse.
3) No es posible asignar el valor de una expresión de cadena a una variable numérica, ni viceversa.

Ejemplo

```
10 LET A=10
20 LET A$="ABC"
30 B=20
40 PRINT A, A$, B
50 END
```

LINE — Instrucción

Función

Esta instrucción se usa para trazar líneas rectas o rectángulos en una pantalla gráfica.

Formato

LINE [(coordenada X_1, coordenada Y_1)]-(coordenada X_2, coordenada Y_2) [, código color]

LINE [(coordenada X_1, coordenada Y_1)], (coordenada X_2, coordenada Y_2), [código color], B

LINE [(coordenada X_1, coordenada Y_1)]-(coordenada X_2, coordenada Y_2), [código color], BF

Pueden usarse especificaciones de coordenadas relativas, en la forma STEP (X_1,Y_1) y STEP (X_2,Y_2), en vez de las coordenadas (X_1,Y_1) y (X_2,Y_2), respectivamente.

Descripción

1) La instrucción LINE permite trazar rectas o rectángulos en una pantalla gráfica, usando el color especificado por el código de color.

2)

Si se omite B o BF:	Se traza una recta entre los dos puntos de coordenadas (X_1,Y_1) y (X_2,Y_2).
Si se especifica B:	Se traza un rectángulo con sus vértices opuestos situados en los puntos de las coordenadas (X_1,Y_1) y (X_2,Y_2).
Si se especifica BF:	Se traza un rectángulo, con sus vértices opuestos en los puntos de coordenadas (X_1,Y_1) y (X_2,Y_2), y el área interior es coloreada con el color especificado por el código de color.

3) Si se omite el código de color, se selecciona el color especificado en la instrucción COLOR.

4) Si se omite el punto inicial (X_1,Y_1) se selecciona como tal el último punto de referencia (LP).
Si el punto inicial (X_1,Y_1) se especifica con coordenadas relativas, éstas se refieren al LP.
Si el punto final (X_2,Y_2) se especifica con coordenadas relativas estas hacen referencia al LP.

6) El punto de la coordenada X puede especificarse con un entero entre 0 y 255 y el punto de la coordenada Y como un entero entre 0 y 191.
Si un punto de coordenadas se especifica fuera de dichas gamas, como punto de coordenada X se selecciona 0 ó 255, y como punto de coordenada Y se selecciona 0 ó 191.

Ejemplo

```
10 SCREEN                                   Traza una recta
20 LINE (10, 10)-(50, 50), 1                Traza un rectángulo
30 LINE (60, 60)-(100, 100),, B             Traza un rectángulo y lo colorea
40 LINE (110, 110)-(150, 150), 1, BF        Traza un rectángulo y lo colorea.
50 LINE STEP (10,-40)-STEP (40, 40), 1, BF
                                            Especificado con coordenadas relativas.
60 GOTO 60
```

LINE INPUT

Instrucción

Función Esta instrucción lee una cadena de caracteres del teclado y la asigna a una variable de cadena.

Formato LINE INPUT ["comentario";] variable de cadena

Descripción

1) La instrucción LINE INPUT muestra el comentario programado y espera a que usted escriba en el teclado una cadena de caracteres.
2) La cadena que se mecanografía aparece después del punto y coma (;). Al pulsar la tecla [RETURN] , los caracteres se asignan a la variable de cadena. La cadena mecanografiada puede editarse con las teclas de edición en pantalla antes de pulsar la tecla [RETURN] .
3) Si se pulsa la tecla [RETURN] sin haber escrito ningún carácter, se asigna a la variable una cadena vacía ("").
4) A diferencia de la instrucción INPUT, la instrucción LINE INPUT no exhibe el interrogante después del comentario, y transfiere comas (,), y comillas (") que puedan escribirse a la variable de cadena.
5) Si la instrucción LINE INPUT se ejecuta en una pantalla gráfica, los datos no se asignarán a la variable y la pantalla no se restituirá al modo texto.

Ejemplo

```
10 LINE INPUT"ABC";A$
20 PRINT A$
30 END
```

LINE INPUT#

Instrucción

Función Esta instrucción lee una cadena de caracteres de un fichero y la asigna a una variable de cadena.

Formato LINE INPUT# número fichero, variable de cadena

Descripción

1) La instrucción LINE INPUT# lee una cadena de caracteres de un fichero especificado por el número de fichero. Todos los caracteres hasta los códigos CR(&H0D) y LF(&H0A) o sólo el código CR se asignan a la variable.
2) Sólo una combinación de códigos CR y LF, en este orden, o un código CR, se consideran como delimitadores. Una combinación de códigos LF y CR en este orden no se considera como delimitador.
3) Cuando la cadena de caracteres leída de un fichero excede de 254 caracteres, los primeros 254 se asignan a la primera variable de cadena, y el resto se asigna a una segunda variable.
4) Los datos almacenados en un fichero con la instrucción PRINT# usan códigos CR y LF como delimitadores, por lo que pueden ser leídos con la instrucción LINE INPUT#.
5) Los ficheros ASCII creados con el comando SAVE contienen códigos CR y LF como delimitadores al final de cada línea, por lo que pueden leerse como variables de cadena.

LIST — Comando

Función

Este comando exhibe en la pantalla las líneas del programa existente en la memoria interna.

Formato

LIST [número línea]
LIST [número línea comienzo]-número línea fin
LIST número línea comienzo-[número línea fin]

Descripción

1) El comando LIST exhibe todas las líneas del programa actualmente en memoria, desde la línea de comienzo a la línea fin, en una pantalla en modo texto.
2) Si se usa sólo LIST: Se exhiben todas las líneas del programa.
   Si se usa LIST número línea: Se muestra sólo la línea especificada.
   Si se usa LIST número línea fin: Aparecen todas las líneas, desde la primera del programa, hasta la indicada.
   Si se usa LIST número línea comienzo: Se exhiben las líneas que van desde la línea de comienzo hasta la última del programa.
3) Si se usa un punto (.) en vez de número de línea, en pantalla aparecerá la última línea ejecutada por BASIC. Cuando la ejecución del programa se interrumpe a causa de una instrucción de error, por "última línea ejecutada" se entiende la línea en que ocurrió el error. Cuando se ha ejecutado antes otra instrucción LIST o LLIST, por última línea ejecutada se entiende la última línea especificada.
4) Para detener temporalmente el listado de líneas, pulse la tecla [STOP]. Para reanudar el listado, vuelva a pulsar [STOP].
5) Para cancelar una operación de listado con LIST y volver al modo Comando, pulse simultáneamente [CTRL] y [STOP].

Ejemplo

LIST
LIST 100
LIST 100 — 200

LLIST — Comando

Función

Este comando lista líneas de programa en una impresora.

Formato

LLIST [número línea]
LLIST [número línea comienzo]-número línea fin
LLIST número línea comienzo-[número línea fin]

Descripción

El comando LLIST es idéntico al comando LLIST, con la diferencia de que las líneas especificadas se listan en la impresora acoplada.

Ejemplo

LLIST 100 — 200

LOAD — Comando

Función Este comando carga en memoria un fichero de programa en código ASCII.

Formato LOAD "nombre dispositivo [nombre fichero]" [, R]

Descripción

1) El comando LOAD carga en memoria un programa almacenado como fichero ASCII con el comando SAVE.
2) Al ejecutarse, el comando LOAD borra los programas o variables que pudiera haber en la memoria interna y cierra los ficheros abiertos.
3) Si se especifica la opción R, el programa es ejecutado inmediatamente que se carga, en cuyo caso los ficheros abiertos no se cierran.
4) Si se omite el nombre de fichero, se carga el primer fichero de programa que se encuentra.

Ejemplo

```
LOAD"CAS:CAJA"
LOAD"CAS:CAJA", R
```

LOCATE — Instrucción

Función Esta instrucción sitúa el cursor en el punto especificado de la pantalla.

Formato

```
LOCATE posición columna
LOCATE [posición columna], posición fila
LOCATE [posición columna], [posición fila], interruptor cursor
```

Descripción

1) La instrucción LOCATE sitúa el cursor en la posición especificada de la pantalla en modo texto.
2) Las posiciones de columna pueden oscilar entre 0 y 39, correspondiendo 0 a la columna más a la izquierda.
   Si se especifica una posición columna que exceda de la máxima anchura de pantalla seleccionada con una instrucción SCREEN o WIDTH, se selecciona la posición que corresponde a dicha máxima anchura.
   Si se omite la posición columna, la anterior es tomada como válida.
3) Las posiciones de fila pueden oscilar entre 0 y 23, correspondiendo 0 a la fila superior.
   Si se especifica un valor superior a la última posición de fila, se selecciona ésta última.
   La última posición de fila es 22 cuando en la parte inferior de la pantalla se muestran las designaciones de las teclas de funciones; cuando dichas designaciones no se visualizan, la última posición de fila es 23.
   Si la posición fila se omite, se selecciona la posición de fila actual.
4) El interruptor del cursor se especifica con una expresión numérica, cuyo valor puede ser 1 ó 0.
   Cuando el valor del interruptor es cero, el cursor no aparece en la pantalla excepto cuando el sistema espera una entrada en el teclado. Si su valor es "1", el cursor es visible en la pantalla continuamente.
   El valor inicial de selección automática es 0.

Ejemplo

```
10 LOCATE 10, 10 : INPUT A$
20 LOCATE 10 : PRINT A$
30 END
```

LOG — Función

Función Esta función calcula el logaritmo natural de una expresión numérica.

Formato LOG (expresión numérica)

Descripción

1) La función LOG determina el logaritmo natural (usando "e" como base) de la expresión numérica que la sigue.
2) El valor de la expresión numérica ha de ser mayor que cero.
3) El resultado es siempre un número real de doble precisión, cualquiera que sea el tipo de expresión numérica.

Ejemplo

```
A=LOG (1.23)
```

LPOS — Función

Función Esta función da la posición de la cabeza impresora.

Formato LPOS (expresión)

Descripción

1) La función LPOS da la posición de la cabeza de impresión en el almacenamiento intermedio (buffer) de la impresora en la memoria. No se refiere a la posición física de la cabeza impresora.
2) La expresión es un número ficticio, que puede tener cualquier valor.

Ejemplo

```
A=LPOS (X)
```

LPRINT — Instrucción

Función Esta instrucción envía datos (valores numéricos o cadenas de caracteres) hacia la impresora.

Formato

```
LPRINT
LPRINT expresión [;expresión...][;]
LPRINT expresión [, expresión...][,]
```

Descripción

1) La instrucción LPRINT envía hacia la impresora los números o cadenas derivados de las expresiones.
2) Cuando se especifican varias expresiones, deben separarse con comas (,) o punto y coma (;).
3) Si LPRINT se usa sin ninguna expresión, su ejecución determina tan sólo un avance de línea en la impresora.
4) Si la instrucción LPRINT no va seguida de coma o punto y coma, los datos de salida irán seguidos de un código CR (&H0D) y LF (&H0A).
5) El manejo de comas, punto y coma y datos es idéntico al de la instrucción PRINT, excepto en que los datos se envían a la impresora.

Ejemplo

```
10 LPRINT "ABC",123;
20 LPRINT "CDE"
30 END
```

LPRINT USING

Instrucción

Función

Esta instrucción se usa para enviar a la impresora números o cadenas, usando un formato especificado.

Formato

LPRINT USING cadena de formato; expresión [;expresión...][;]
LPRINT USING cadena de formato, expresión [, expresión...][,]

Descripción

1) La instrucción LPRINT USING envía a la impresora los números o cadenas derivados de las expresiones, usando un formato específico.
2) Cuando se indica más de una expresión, deben separarse con una coma (,) o punto y coma (;).
3) Si la instrucción LPRINT USING no va seguida de una coma o punto y coma, los datos enviados terminan con un código de retorno del carro CR (&H0D) y avance de línea LF (&H0A).
4) El tratamiento de comas, punto y coma y datos es idéntico al de la instrucción PRINT, excepto en que los datos se dan de salida a la impresora.
5) La especificación de formato es idéntica a la de la instrucción PRINT USING. Véanse más detalles en la instrucción PRINT USING.

Ejemplo

```
10 LPRINT USING "\\####";"ABC";123;
20 LPRINT USING "\\";"CDE"
30 END
```

MAXFILES — Instrucción

Función Esta instrucción especifica el número máximo de ficheros.

Formato MAXFILES=expresión entera

Descripción

1) La instrucción MAXFILES se usa para especificar el número máximo de ficheros que pueden usarse en una instrucción OPEN, definiéndolo con una expresión entera.
   El número de ficheros en cuestión puede entonces abrirse simultáneamente con OPEN.
2) El número que puede especificarse va de 0 a 15.
   El valor inicial de la expresión de enteros es "1". Si el valor de la expresión es cero, no puede abrirse ningún fichero con OPEN.
3) Al igual que la instrucción CLEAR, esta instrucción inicializa todas las variables:
   Las variables numéricas se inicializan a cero; las variables de cadena, a una cadena vacía ("").
   Todos los ficheros abiertos se cierran.
   El contenido de las variables definido en las instrucciones que empiezan con DEF (DEF FN, DEF USR, DEFINT, DEFSNG, DEFDBL y DEFSTR) es invalidado.
   Todas las tablas de variables se borran.
   Los bucles FOR NEXT quedan sin continuidad.
   El control no se restituye de una subrutina por acción de la instrucción RETURN.
4) Cuando se ejecuta la instrucción MAXFILES, se reservan tantos bloques de control de ficheros como el valor de la expresión de enteros+1.
   Por cada bloque de control de ficheros se reserva un espacio de 267 bytes en la memoria.

Ejemplo MAXFILES=10

MERGE — Comando

Función Este comando lee un fichero de programa ASCII y lo combina con el programa actualmente en memoria.

Formato MERGE "nombre dispositivo [nombre fichero]"

Descripción

1) El comando MERGE carga un fichero de programa ASCII almacenado con el comando SAVE y lo combina con el programa actualmente en memoria.
2) Si se omite el nombre de fichero, se cargará el primer fichero de programa ASCII que se encuentre.
3) Si el programa del fichero contiene un mismo número de línea que el programa actual, la línea de programa del fichero sustituirá a la existente en memoria.

Ejemplo MERGE "CAS:SAMPLE"

MID$ Función

Función Esta función obtiene la parte especificada de la cadena indicada.

Formato MID$ (expresión de cadena, posición [, número de caracteres])

Descripción

1) La función MID$ obtiene el número especificada de caracteres, empezando por la posición de carácter indicada, de la cadena definida en la expresión de cadena.
2) La posición es especificada por una expresión entera cuyo valor puede oscilar entre 1 y 255.
   Si la posición especificada excede del número de caracteres se mostrará una cadena vacía ("").
3) El número de caracteres es especificado con una expresión entera, cuyo valor oscila entre 0 y 255.
   Si se omite el número de caracteres, se obtendrá el carácter situado en la posición especificada y todos los demás existentes a su derecha.
   Si como número de caracteres se escribe cero, se obtendrá una cadena vacía("").
4) La cabecera de identificación de carácter gráfico (&H01) en un símbolo gráfico cuenta como un carácter. Ello significa que cada símbolo gráfico ocupa dos posiciones de caracteres.

Ejemplo

```
10 A$="012345689ABCDEF"
20 B$=MID$ (A$, 2, 1) : C$=MID$ (A$, 4)
30 PRINT B$, C$
40 END
```

MID$ Instrucción

Función Esta instrucción sustituye la parte especificada de la cadena indicada con otra cadena.

Formato MID$ (variable de cadena, posición [, número caracteres]) = expresión de cadena

Descripción

1) La instrucción MID$ se usa para sustituir un número dado de caracteres de una cadena por una variable de cadena que posee el mismo número de caracteres que se especificó en la expresión de cadena, empezando por la posición de carácter indicada.
   El número de caracteres asignado a la variable de cadena no debe cambiarse antes ni después de la sustitución.
2) La posición del caracter inicial se especifica mediante una expresión entera que puede oscilar entre 1 y 255.
   La posición no debe exceder del número de caracteres asignado a la variable de cadena.
3) El número de caracteres a sustituir se indica mediante una expresión entera, cuyo valor es 0 — 255.
   Si el número de caracteres se omite o bien excede del número existente en la expresión de cadena, el número de caracteres que se sustituirá será igual al número existente en la expresión de cadena.
   Cuando el número de caracteres a sustituir es menor que el número existente la expresión, los caracteres de la variable serán sustituidos por el correspondiente número de caracteres existente en la expresión de cadena, empezando con el carácter más a la izquierda de la expresión.
4) Si el valor (posición + número de caracteres-1) excede del número de caracteres de la variable de cadena, los sobrantes se omitirán.
5) La cabecera de identificación de carácter gráfico (&H01) de un signo gráfico cuenta como carácter. Ello significa que cada signo gráfico ocupa dos posiciones de caracteres.

Ejemplo

```
10 A$="012345689"
20 MID$ (A$, 2, 4)="AAAAAAA"
30 PRINT A$
40 END
```

MOTOR Instrucción

Función Esta instrucción se usa para conectar y desconectar el motor del registrador cassette.

Formato

```
MOTOR ON
MOTOR OFF
MOTOR
```

Descripción

1) La instrucción MOTOR controla el funcionamiento del motor de un registrador cassette conectado al ordenador y que se ha ajustado al modo grabación o lectura.
2)

MOTOR ON	Pone en marcha el motor.
MOTOR OFF	Para el motor.
MOTOR	Para el motor, si está funcionando, y lo arranca si está parado.

NEXT — Instrucción

Función

Esta instrucción se usa con una instrucción FOR para constituir un bucle de ejecución en un programa.

Formato

NEXT [variable numérica[, variable numérica...]]

Descripción

1) La instrucción NEXT se usa con la instrucción FOR en la cual se coloca la misma variable numérica.
2) Cuando se coloca más de una variable numérica en esta instrucción, forma bucles de ejecución con el correspondiente número de instrucciones FOR. En tal caso, las variables numéricas de la instrucción NEXT deben disponerse de modo que la primera variable corresponda a la instrucción FOR más próxima, la segunda variable corresponda a la siguiente instrucción FOR más próxima, y así sucesivamente.
3) Cuando se omiten variables numéricas, la instrucción NEXT se empareja con la instrucción FOR más próxima.

Ejemplo

```
10 FOR M=1 TO 5
20 FOR J=1 TO 3
30 FOR K=1 TO 2
40 PRINT M, J, K : NEXT K, J, M
50 FOR X=1 TO 4 : PRINT X
60 NEXT
80 END
```

Ningún bucle cruzado

NEW — Comando

Función

Este comando borra el programa actualmente en memoria, así como todas las variables.

Formato

NEW

Descripción

1) El comando NEW borra el programa actualmente en memoria y todas las variables.
   Se cierran los ficheros abiertos.
   Se invalida el contenido de las instrucciones que empiezan con DEF (DEF FN, DEF USR, DEFINT, DEFSNG, DEFDBL, DEFSTR).
   Se borran todas las tablas de variables (arrays).
2) El comando NEW se usa corrientemente antes de empezar un programa.

OCT$ — Función

Función Esta función convierte un valor numérico a una cadena de caracteres en notación octal.

Formato OCT$ (expresión entera)

Descripción La función OCT$ devuelve una cadena que es igual al valor octal de la expresión entera en notación decimal que la sigue.

Ejemplo

```
10 A$=OCT$ (16)        Se asigna "20" a la variable A$.
20 PRINT A$
30 END
```

ON ERROR GOTO — Instrucción

Función Esta instrucción se usa para especificar el primer número de línea de una rutina de tratamiento de errores.

Formato ON ERROR GOTO número línea

Descripción

1) La instrucción ON ERROR GOTO habilita interrupciones causadas por errores. Cuando ocurre un error, desvía la ejecución a la rutina de tratamiento de errores cuyo primer número de línea especifica.
   Cuando esta instrucción ha sido ejecutada, el sistema BASIC deja de manejar el error (en la pantalla no aparece el mensaje de error y el sistema no se restituye al modo comando).
2) Una rutina de tratamiento de errores contiene las funciones ERR o ERL para procesar el error, y termina con una instrucción RESUME.
   Si ocurre un error dentro de una rutina de tratamiento de errores, no se produce ninguna interrupción, pero en la pantalla aparece un mensaje de error y el sistema vuelve al modo comando.
3) Si se ejecuta una instrucción ON ERROR GOTO O, las interrupciones causadas por errores se deshabilitan y todos los errores son manejados por el sistema BASIC.
4) Cuando se ha ejecutado esta instrucción, las interrupciones de error permanecen habilitadas incluso después de completarse la ejecución del programa. Esto significa que una interrupción causada por un error que ocurrió después de que el sistema volviera al modo comando, hará que se ejecute la rutina de tratamiento de errores.
   Las interrupciones habilitadas por una instrucción ON ERROR GOTO no se deshabilitan hasta que se ejecuta otra instrucción ON ERROR GOTO o bien un comando RUN o una instrucción CLEAR. Para deshabilitar interrupciones causadas por errores dentro del programa actual, debe ejecutarse una instrucción ON ERROR GOTO O antes de que el programa termine.

Ejemplo ON ERROR GOTO 1000

ON GOSUB

Instrucción

Función

Esta instrucción hace que la ejecución pase a una subrutina especificada.

Formato

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

Descripción

1) La instrucción ON GOSUB hace que la ejecución pase a la subrutina que posee el número de línea cuya posición en la instrucción, es indicada por la expresión entera.
2) Si el valor de la expresión entera es cero o el número de línea en la posición indicada por dicha expresión ha sido omitido, la ejecución no pasa a subrutina alguna, sino que continúa con la instrucción existente después de la instrucción ON GOSUB.

Ejemplo

ON X GOSUB 100,,55
Cuando X=1, se ejecutará la subrutina que empieza con el número de línea 100.
Cuando X=2, la ejecución pasará a la siguiente instrucción.
Cuando X=3, se ejecutará la subrutina que empieza con el número de línea 55.
Cuando X=4, la ejecución pasa a la siguiente instrucción.

ON GOTO

Instrucción

Función

Esta instrucción crea una bifurcación a la línea especificada.

Formato

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

Descripción

1) La instrucción ON GOTO hace que la ejecución se bifurque a la línea especificada cuya posición corresponde a la expresión entera.
2) Si el valor de la expresión entera es cero o el número de línea en la posición indicada por dicha expresión ha sido omitido, la ejecución continúa con la instrucción que sigue a la instrucción ON GOTO.

Ejemplo

ON X GOTO 100,,55
Cuando X=1, la ejecución se bifurca a la línea 100.
Cuando X=2, la ejecución pasa a la siguiente instrucción
Cuando X=3, la ejecución se bifurca a la línea 55.
Cuando X=4, la ejecución pasa a la siguiente instrucción

ON INTERVAL GOSUB

Instrucción

Función

Esta instrucción especifica la primera línea de una rutina de tratamiento de las interrupciones generadas por el temporizador interno.

Formato

ON INTERVAL=tiempo intervalo GOSUB número línea.

Descripción

1) La instrucción ON INTERVAL GOSUB se usa para especificar el intervalo de tiempo con que se piden interrupciones al temporizador y para indicar el número de la primera línea de una rutina de tratamiento de las interrupciones generadas por el temporizador. Cuando la interrupción del temporizador es habilitada por la instrucción INTERVAL ON, el temporizador solicita una interrupción a los intervalos de tiempo especificados, haciendo que la ejecución pase a la rutina de tratamiento de interrupciones cuyo primer número de línea se especifica en la instrucción.
2) El intervalo de tiempo puede ajustarse en incrementos de 1/50 segundo y se especifica con una expresión entera que puede oscilar entre 1 y 65535. La cuenta atrás del intervalo empieza cuando se ejecuta la instrucción ON INTERVAL GOSUB.
3) Para salir de la subrutina, en la misma se incorpora una instrucción RETURN. Mientras se ejecuta la rutina de tratamiento de interrupciones, el sistema está en INTERVAL STOP (estado de retención de la interrupción). Al ejecutar la instrucción RETURN, el sistema vuelve al estado de habilitación de interrupción o INTERVAL ON.
4) Mientras se está ejecutando la rutina de tratamiento de interrupciones especificada por una cualquiera de las siguientes instrucciones, el sistema permanece en estado INTERVAL STOP (retención de interrupción), hasta que concluye la ejecución de la subrutina.

ON ERROR GOTO	ON KEY GOSUB
ON STOP GOSUB	ON SPRITE GOSUB
ON STRING GOSUB	

Ejemplo

```
10  ON INTERVAL=50 GOSUB 100
20  TIME=0
30  INTERVAL ON
40  GOTO 40
100 PRINT TIME/50
110 RETURN
```

ON KEY GOSUB

Instrucción

Función

Esta instrucción especifica el primer número de línea de una rutina de tratamiento de interrupciones cuya ejecución es indicada por la pulsación de une tecla de funciones.

Formato

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

Descripción

1) La instrucción ON KEY GOSUB se usa para especificar el primer número de línea de una rutina de tratamiento de interrupciones cuya ejecución es iniciada por la pulsación de una tecla de función después de que se haya ejecutado una instrucción KEY(n) ON en el mismo programa. La ejecución pasa a la rutina que posee el primer número de línea especificado.
2) Los números de línea en la instrucción se escriben en el mismo orden que los números de teclas de funciones, lo que permite especificar diferentes números de línea para las distintas teclas. Pueden indicarse hasta diez números de línea esta con instrucción, separándolos con comas.
   Cuando se omite un número de línea, la correspondiente tecla de funciones no provocará una interrupción al ser pulsada.
3) Para salir de una rutina de tratamiento de interrupciones, en la misma se incorpora una instrucción RETURN. Mientras se está ejecutando una rutina de tratamiento de interrupciones, el sistema permanece en estado KEY(n) STOP (retención de interrupción). Cuando se ejecuta la instrucción RETURN, el sistema vuelve al estado KEY(n) ON (habilitación de interrupción).
4) Mientras se está ejecutando una rutina de tratamiento de interrupciones como resultado de una instrucción ON ERROR GOTO, el sistema permanece en el estado KEY(n) STOP (retención de interrupción) hasta que se completa la ejecución de la subrutina.

Ejemplo

```
10  ON KEY GOSUB 100,,,200
20  KEY (1) ON
30  KEY (4) ON
40  GOTO 40
100 PRINT "F1"
110 RETURN
200 PRINT "F4"
210 RETURN
```

ON SPRITE GOSUB

Instrucción

Función

Esta instrucción especifica el primer número de línea de una rutina de tratamiento de interrupciones de las formas.

Formato

ON SPRITE GOSUB número línea

Descripción

1) La instrucción ON SPRITE GOSUB se usa para especificar el primer número de línea de una rutina de tratamiento de interrupciones cuya ejecución se inicia cuando dos formas se solapan en la pantalla.
   Una interrupción por forma ocurre cuando una forma trazada con la instrucción PUT SPRITE (después de haberse ejecutado la instrucción SPRITE ON) colisiona con otra forma en la pantalla, haciendo que la ejecución pase a la rutina de tratamiento de interrupciones especificada.
2) Para salir de una rutina de tratamiento de interrupciones, en la misma se incorpora una instrucción RETURN. Mientras la rutina de tratamiento de interrupciones está siendo ejecutada, el sistema permanece en estado SPRITE STOP (retención de interrupción). Al ejecutarse la instrucción RETURN, el sistema vuelve al estado SPRITE ON (habilitación de interrupciones).
3) Mientras se está ejecutando la rutina de tratamiento de interrupciones especificada por una cualquiera de las siguientes instrucciones, el sistema permanece en estado SPRITE STOP (retención de interrupción), hasta que se completa la ejecución de la subrutina.
   ON ERROR GOTO ON KEY GOSUB
   ON STOP GOSUB

Ejemplo

```
10  SCREEN 2
20  ON SPRITE GOSUB 100
30  SPRITE ON
40  SPRITE$(0)=STRING$(8,255)
50  FOR K=0 TO 300
60  PUT SPRITE 0, (K,200-K), 1
70  PUT SPRITE 1, (200-K, K), 8, 0
80  NEXT
90  END
100 RETURN 50
```

ON STOP GOSUB

Instrucción

Función

Esta instrucción especifica la primera línea de una rutina de tratamiento de interrupciones cuya ejecución es iniciada por una interrupción causada por la pulsación de las teclas [CTRL] y [STOP]

Formato

ON STOP GOSUB número línea

Descripción

1) La instrucción ON STOP GOSUB se usa para especificar el primer número de línea de una rutina de tratamiento de interrupciones causada por la pulsación simultánea de las teclas [CTRL] y [STOP]
Si las teclas [CTRL] y [STOP] se pulsan simultáneamente después de que se haya ejecutado una instrucción STOP ON, ocurre una interrupción, haciendo que la ejecución pase a una rutina de tratamiento de interrupciones que posee el número de línea especificado.
2) Para salir de la rutina de tratamiento de interrupciones, en ésta se incorpora una instrucción RETURN.
Mientras se está ejecutando una rutina de tratamiento de interrupciones, el sistema permanece en estado STOP STOP (retención de interrupción). Al ejecutar la instrución RETURN, el sistema vuelve al estado STOP ON (habilitación de interrupcción).
3) Mientras se esta ejecutando la rutina de tratamiento de interrupciones especificada por una de las siguientes instrucciones, el sistema permanece en el estado STOP STOP (retención de interrupción), hasta que se completa la ejecución de la rutina:
ON ERROR GOTO ON KEY GOSUB
4) La ejecución errónea de esta instrucción puede provocar un desbocamiento del programa que puede pararse sólo desconectando el ordenador. Por consiguiente, asegúrese de que esta instrucción esté correctamente programada.

Ejemplo

```
10  ON STOP GOSUB 100
20  STOP ON
30  KS=INKEY$
40  IF K$="E" THEN 200
50  GOTO 30
100 PRINT "100"
110 RETURN
200 STOP OFF
210 END
```

ON STRIG GOSUB

Instrucción

Función

Esta instrucción especifica la primera línea de una rutina de tratamiento de interrupciones a la que pasa la ejecución cuando ocurre una interrupción causada por la pulsación de la barra espaciadora del teclado o un pulsador de un mando de juegos.

Formato

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

Descripción

1) La instrucción ON STRIG GOSUB se usa para especificar el primer número de línea de una rutina de tratamiento de interrupciones a la que pasa la ejecución cuando se produce una interrupción causada por el accionamiento de la barra espaciadora del teclado o de algún botón de un mando de juego. Si la barra espaciadora o el botón de un mando de juego son pulsados después de que se haya ejecutado una instrucción STRIG (n) ON, ocurre una interrupción, que hace que la ejecución pase a la rutina de tratamiento de interrupciones cuya primera línea se especifica en esta instrucción.

2) En esta instrucción pueden especificarse hasta cinco números de línea. Los números de activadores 0 — 4 corresponden a números de línea en el orden en que se escriben en la instrucción:

   0 Barra espaciadora del teclado.
   1 1er botón activador del mando de juego conectado al conector JOYSTICK 1.
   2 1er botón activador del mando de juego conectado al conector JOYSTICK 2.
   3 2° botón activador del mando de juego conectado al conector JOYSTICK 1.
   4 2° botón activador del mando de juego conectado al conector JOYSTICK 2.

   Los números de línea se separan con comas (,).
   Si se omite algún número de línea, el correspondiente accionamiento no causará ninguna interrupción.

3) Para salir de la rutina de tratamiento de interrupciones se incorpora una instrucción RETURN en la misma. Mientras dura la ejecución de la rutina de tratamiento de interrupciones el sistema permanece en el estado STRIG(n) STOP (retención de interrupción). Cuando se ejecuta una instrucción RETURN, el sistema vuelve al estado STRIG(n) ON (habilitación de interrupción).

4) Mientras se está ejecutando la rutina de tratamiento de interrupciones especificada en una cualquiera de las siguientes instrucciones, el sistema permanece en el estado STRIG(n) STOP (retención de interrupción), hasta que se ha completado la ejecución de la rutina.

ON ERROR GOTO	ON KEY GOSUB
ON STOP GOSUB	ON SPRITE GOSUB

Ejemplo

```
10  ON STRIG GOSUB 100, 200, 300
20  STRIG (0) ON : STRIG (1) ON : STRIG (2) ON
30  GOTO 30
100 PRINT "SPACE" : RETURN
200 PRINT "JOY 1" : RETURN
300 PRINT "JOY 2" : RETURN
```

OPEN

Instrucción

Función

Esta instrucción se usa para abrir un fichero para E/S de datos.

Formato

OPEN "nombre dispositivo [nombre fichero]" [FOR modo] AS[#] número fichero

Descripción

1) La instrucción OPEN abre un fichero con el "nombre fichero" especificado y le asigna el "número fichero" indicado. Una vez abierto, el fichero puede usarse para entrada o salida de datos utilizando instrucciones INPUT# o PRINT#. Cuando se ha completado el acceso al fichero, deberá cerrarse con la instrucción CLOSE.
2) La cláusula FOR se usa para especificar entrada o bien salida de datos:

FOR INPUT	Modo de entrada al ordenador
FOR OUTPUT	Modo de salida del ordenador

3) El "número fichero" puede ser un entero en la gama 1 — 15.
No debe exceder el valor límite especificado en la instrucción MAXFILES.
No está permitido usar el número fichero de un fichero ya abierto.
4) La instrucción OPEN se usa para entrada/salida de datos de ficheros en cinta cassette o para exhibir caracteres o símbolos de texto en una pantalla gráfica.

Ejemplo

```
10  SCREEN 3
20  OPEN "GRP:" AS 1
30  PRESET (10, 10)
40  PRINT# 1, "SAMPLE"
50  CLOSE
60  GOTO 60
```

OUT — Instrucción

Función Esta instrucción se usa para enviar 1 byte de datos hacia una puerta (port) de entrada/salida.

Formato OUT dirección puerta, expresión entera

Descripción

1) La instrucción OUT se usa para enviar 1 byte de datos, especificado por la expresión entera, hacia la puerta de E/S con la dirección de puerta especificada.
2) La dirección de puerta se indica con una expresión entera que puede oscilar entre 0 y 255.
3) Las direcciones de puerta se explican en el Capítulo 3, Sección 4 "Mapa de Entrada/Salida".
4) La ejecución errónea de una instrucción OUT puede provocar un desbocamiento del programa que solo podrá detenerse desconectando el sistema.

PAD — Función

Función Esta instrucción da el estado de una tabla sensora.

Formato PAD (expresión entera)

Descripción

1) La función PAD comunica el estado de una tabla sensora (touch pad) conectada a uno de los conectores JOYSTICK y especificado por la expresión entera.
2) El valor de la expresión entera especifica el conector JOYSTICK al que está enchufada la tabla, así como el tipo de estado que se desea:

0~3	Especifica el conector JOYSTICK 1
4~7	Especifica el conector JOYSTICK 2
0 ó 4	Da -1 cuando la tabla ha sido tocada Da 0 cuando la tabla no ha sido tocada
1 ó 5	Da la coordenada X del punto tocado en la tabla.
2 ó 6	Da la coordenada Y del punto tocado en la tabla.
3 ó 7	Da -1 cuando el interruptor de la tabla está pulsado. Da 0 cuando el interruptor no está pulsado.

PAINT — Instrucción

Función Esta instrucción se usa para colorear un área de la pantalla.

Formato
PAINT (coordenada X, coordenada Y) [,color área[,color contorno]]
PAINT (coordenada X, coordenada Y), [color área], color contorno
Pueden usarse también coordenadas relativas STEP (X, Y), en vez de coordenadas absolutas (X, Y).

Descripción

1) La instrucción PAINT se usa para colorear toda el área de una figura (dibujada en el color de contorno y en la cual se halla el punto de coordenada especificado) utilizando el "color área" especificado.
   En el modo gráfico de alta resolución no puede especificarse un color de contorno: el color indicado para el área interna se usa también para el contorno.
2) Ambos colores se especifican mediante sus pertinentes códigos. Si el color contorno se omite, como tal se usa el color seleccionado para el área interna de la figura.
   Si se omite el color área, como tal se usa el color especificado en la instrucción COLOR.
3) Cuando se usan coordenadas relativas con STEP (X, Y), el punto de coordenadas es relativo al último punto de referencia (LP).
4) Si el punto de coordenada especificado está situado en el borde de la pantalla o en un punto en que se ha especificado el mismo color que para el contorno, el área especificada no se coloreará.

Ejemplo

```
10  SCREEN 2
20  CIRCLE (100, 100), 50, 8
30  PAINT (70, 70), 8
40  GOTO 40
```

PDL — Función

Función Esta función comunica el estado actual de la tabla sensora.

Formato PDL (expresión entera)

Descripción

1) La función PDL da el estado de la tabla sensora cuyo número se especifica en la expresión entera.
2) Pueden conectarse hasta 12 tablas en los conectores JOYSTICK del ordenador.
   El conector a usar se especifica con el valor de la expresión entera:
   Si el valor es 1, 3, 5, 7, 9, ó 11: receptáculo JOYSTICK 1.
   Si el valor es 2, 4, 6, 8, 10 ó 12: receptáculo JOYSTICK 2.
3) El resultado comunicado por la función puede oscilar entre 0 y 255.

PEEK — Función

Función Esta función da el contenido de una posición específica de memoria.

Formato PEEK (dirección)

Descripción

1) La función PEEK da el contenido de la dirección de memoria especificada.
2) La dirección ha de especificarse con una expresión de enteros, cuyo valor oscile entre &H0 y &HFFFF (−32768 a+32767).
3) Véanse detalles de la memoria en Capítulo 3, Sección 2, "Mapa de la Memoria".

Ejemplo X=PEEK (&HFFFF)

PLAY — Instrucción

Función Esta instrucción se usa para tocar música.

Formato PLAY expresión de cadena A[, expresión de cadena B[, expresión de cadena C]]

Descripción

1) La instrucción PLAY se usa para tocar música de acuerdo con los macrocomandos especificados en las expresiones de cadena.
2) Las expresiones de cadena A, B, y C especifican macrocomandos de música para los canales de audio A, B y C, respectivamente.
   Ejemplo: PLAY "C", "E", "G"
3) Cada expresión de cadena consiste en una cadena de uno o varios macrocomandos de música.
   Cuando el valor de una expresión de cadena es nulo (""), no se emitirá ningún tono en el correspondiente canal.
4) Los macrocomandos de música son:

Comandos para especificar altura	A~G, #, +, −, O, N
Comando para especificar notas	L
Comando para especificar pausa	R
Comando para especificar ritmo	T
Comando para especificar intensidad	V
Comandos para especificar timbre	S, M

Véanse más detalles en el Capítulo 1, Sección 6, CARACTERISTICAS DE SONIDO.

PLAY — Función

Función Esta función da valores que indican si se toca música o no.

Formato PLAY (expresión entera)

Descripción

1) La función PLAY da valores que indican si el canal de audio especificado está tocando música con la instrucción PLAY.

Cuando el valor es −1:	Toca
Cuando el valor es 0:	No toca

2) El valor de la expresión entera selecciona el canal de audio a comprobar:

0	Comprueba si uno cualquiera de los canales A, B, o C está tocando.
1	Comprueba el canal A
2	Comprueba el canal B
3	Comprueba el canal C

Ejemplo

```
10  PLAY "C"
20  PRINT PLAY (1)
30  GOTO 10
```

POINT — Función

Función Esta función da el código de color de un punto de coordenadas especificado.

Formato

POINT (coordenada X, coordenada Y)
POINT STEP (coordenada X, coordenada Y)

Descripción

1) La función POINT da el código de color del punto de coordenadas especificado en una pantalla gráfica.
2) Si el punto de coordenadas se especifica relativo, STEP (coordenada X, coordenada Y), el punto de coordenadas es relativo al último punto de referencia (LP).
   El punto LP permanece invariable una vez que se ha usado la función POINT.

Ejemplo

```
10  SCREEN 2
20  PSET (10, 10), 1
30  A=POINT (10, 10)
40  CIRCLE (10, 10), 50, A
50  GOTO 50
```

POKE — Instrucción

Función Esta instrucción asigna un byte de datos a una dirección de memoria especificada.

Formato POKE dirección, expresión entera

Descripción

1) La instrucción POKE se usa para asignar datos, contenidos en la expresión entera a una dirección de memoria especificada.
2) La dirección debe especificarse mediante una expresión entera que puede oscilar entre &H0 y &HFFFF (−32768 a +32767).
3) Vea detalles del mapa de memoria en el Capítulo 3, Sección 2, "Mapa de la Memoria."
4) La ejecución errónea de esta instrucción puede provocar un desbocamiento del programa, del cual el sistema sólo puede recuperarse desconectando el ordenador.

Ejemplo POKE &HC100, &H1F

POS — Función

Función Esta función da la posición de columna actual del cursor de la pantalla.

Formato POS (expresión)

Descripción

1) La función POS devuelve la posición de columna del cursor en la pantalla en modo texto.
   Cuando el cursor está en la posición completamente a la izquierda de la pantalla, da cero.
2) La expresión es ficticia y puede tener cualquier valor.

Ejemplo PRINT POS (0)

PRESET — Instrucción

Función Esta instrucción se usa para actualizar el color del punto cuyas coordenadas se especifican.

Formato
PRESET (X, Y) [, código color]
PRESET STEP (X, Y) [, código color]

Descripción

1) La instrucción PRESET se usa para sustituir el color del punto cuyas coordenadas se especifican con el color indicado por el código de color, en una pantalla gráfica.
2) Cuando se omite el código de color, se selecciona como tal el antes indicado en la instrucción COLOR.
3) Si el punto de coordenadas se especifica relativo, con STEP (X, Y), como origen se toma el último punto de referencia (LP).
   Cuando se ha ejecutado la instrucción PRESET, el punto LP queda situado en el punto de coordenadas especificado.

Ejemplo

```
10  SCREEN 2
20  LINE (10, 10)−(100, 100), 1, BF
30  FOR K=0 TO 150
40  PRESET (K, 30)
50  NEXT
60  GOTO 60
```

PRINT — Instrucción

Función

Esta instrucción se usa para exhibir datos en la pantalla.

Formato

PRINT
PRINT expresión [; expresión...] [;]
PRINT expresión [, expresión...] [,]
En vez de PRINT puede usarse el signo de interrogación (?)

Descripción

1) La instrucción PRINT se usa para exhibir datos (valores numéricos o cadenas) especificados por la expresión, en una pantalla en modo texto.
2) Los datos se ponen en pantalla en los siguientes formatos:
   Expresiones numéricas: Cada número visualizado va precedido de un espacio en blanco (si es positivo) o el signo (-), y va seguido de otro espacio en blanco.
   Expresiones de cadena: Los caracteres especificados se muestran alineados por la izquierda.
3) Si se especifica sólo la instrucción PRINT, sin expresión, aparecerá una fila de espacios en blanco.
4) En una instrucción PRINT pueden especificarse varias expresiones, separándolas con coma (,) o punto y coma (;).
   El formato de visualización depende del delimitador usado:
   Cuando se usa la coma como delimitador:
   Las filas se dividen en campos de visualización, cada uno de 14 columnas (es decir 14 espacios de caracteres), y cada dato se exhibe justificado por la izquierda en un campo separado. Cuando un dato contiene más de 14 caracteres (incluyendo un espacio en blanco al final, si es un número), se continúa en el siguiente campo de visualización.
   - Cuando se usa punto y coma como delimitador:
     Un dato va seguido inmediatamente de otro; si es un número, va precedido de un espacio en blanco o el signo menos (si es negativo), y va seguido de un espacio en blanco.
5) Si una instrucción PRINT termina con una coma o punto y coma, o no va seguida de nada, afecta diferentemente a la siguiente instrucción PRINT o PRINT USING:
   - Cuando no va seguida de nada:
     Al final de la primera instrucción PRINT, ocurre un avance de línea y los datos de la siguiente instrucción PRINT se exhiben en la fila siguiente.
     Cuando termina con coma:
     No ocurre avance de línea, y los datos de la nueva instrucción PRINT aparecen en el siguiente campo de visualización, en la misma fila.
   - Cuando termina con punto y coma:
     No ocurre avance de línea, y los datos de la siguiente instrucción PRINT siguen inmediatamente a los de la anterior.

Ejemplo

```
10  PRINT 123, "ABC"
20  PRINT 123; "ABC", 123,4; "A", "B"
30  PRINT 123; 4, 5, 6, 78;
40  PRINT "ABC"
50  END
```

PRINT USING

Instrucción

Función

Esta instrucción se usa para exhibir datos en una pantalla de modo texto usando formatos específicos.

Formato

PRINT USING cadena control formato; expresión [; expresión...] [;]
PRINT USING cadena control formato; expresión [, expresión...] [,]
En vez de la palabra clave PRINT puede usarse el signo de interrogación (?).

Descripción

1) La instrucción PRINT USING se usa para exhibir en la pantalla de modo texto los datos (valores numéricos o cadenas) indicados por la expresión, utilizando el formato de visualización especificado en la cadena de control formato.
2) Pueden especificarse varias expresiones, separándolas con una coma (,) o punto y coma (;). Sin embargo, la cadena de control formato ha de separarse de la expresión siempre con punto y coma.
   Los campos de visualización se especifican con la cadena de control formato y no dependen del tipo de delimitador (coma o punto y coma).
3) Según la instrucción se termine con una coma, punto y coma o nada, ello afecta la ubicación de los datos exhibidos en una subsiguiente instrucción PRINT o PRINT USING:
   - Si no va seguida de nada: Se produce un avance de línea, y los datos de la siguiente instrucción PRINT o PRINT USING se muestran en la siguiente fila.
   - Si va seguida de coma o punto y coma: No ocurre avance de línea y los datos de subsiguientes instrucciones PRINT USING o PRINT siguen inmediatamente a los anteriores.
4) Si bien la cadena de control formato puede especificarse como variable de cadena o constantes de cadena, los símbolos de control a usar difieren según el tipo de datos a visualizar.
   Símbolos de control para datos numéricos:

#(nº)	. (punto)
+ (más)	– (menos)
* (asterisco)	$ (dólar)
^ (exponenciación)	

   Símbolos de control para datos de cadena:

! (signo de exclamación)	\ barra inversa
& (Y comercial)	

Si en una cadena de control formato se usa algún carácter que no sea los acabados de relacionar, el mismo aparecerá antes o después de los datos visualizados.

Ejemplo

```
PRINT USING "X=###Y=###"; 12;456
```

5) Cuando un dato numérico a visualizar contiene una parte entera cuyo número de dígitos excede del especificado por la cadena de control formato, los datos exhibidos irán precedidos por el signo porcentaje (%).

Ejemplo: PRINT USING "##"; 123
En pantalla aparece: %123

Formato

Cadena para datos numéricos	Descripción
#···#	Cada símbolo # corresponde a un dígito del número a visualizar. Cuando el número de esos símbolos excede del de dígitos, la cantidad se exhibirá alineada por la derecha. Ejemplo: PRINT USING "####"; 123, 456
#···#.#···#	Para especificar la posición del punto decimal en los datos visualizados se usa un punto (.). El número de decimales a visualizar se selecciona con el número de signos # a continuación del punto (.); si el número de dichos signos excede del de decimales, el resto de puestos de decimales se rellenará con ceros. Ejemplo: PRINT USING "##.##"; 1.234; 5.6
#···#, #···#,.#···#	Cuando en una posición cualquiera de la cadena de signos # se inserta una coma en la parte entera, éstos serán representados con coma de mil, es decir a intervalos de tres dígitos. Ejemplo: PRINT USING "##########,#"; 123,456 (coma de mil, con significado de punto de mil)
+#···# +#···#.#···#	Con este formato, los datos se muestran precedidos del signo más o menos (+ o −). Ejemplo: PRINT USING "+###.##"; 12.3; −4,567
#···#+ #···#.#···#+	Los datos se visualizarán seguidos de un signo (+ o −). Ejemplo: PRINT USING "###.##+"; 12.3, −4,567
#···#− #···#.#···#−	Los números negativos aparecerán seguidos del signo menos (−). Ejemplo: PRINT USING "###.##−"; 12.3; −4,567
**#···# **#···#.#···#	Cuando los dígitos enteros a visualizar son menos que el número de signos # especificados en la parte de enteros, el resto de dígitos se rellenan con asteriscos. Ejemplo: PRINT USING "**####"; 12; 123456

$$#···#
$$#···#.#···#

Cuando una cadena de control formato va precedida de dos signos dólar ($$), los datos se visualizan precedidos de un solo signo dólar ($).

Ejemplo: PRINT USING "$$#####"; 12; 12345

**$#···#
**$#···#.#···#

Cuando una cadena de control formato va precedida por dos asteriscos y un signo dólar (**$), los datos aparecen precedidos del signo dólar ($), y los primeros puestos vacantes de la parte entera, de haberlos, se rellenan con asteriscos.

Ejemplo: PRINT USING "**$#####"; 12; 12345

#···#^^^^
#···#.#···#^^^^

Cuando una cadena de control formato va seguida de cuatro signos de exponenciación (^), los datos se visualizan como número de punto flotante usando el símbolo E.

Ejemplo: PRINT USING "###.###^^^^"; 1. 12345678

Cadena para datos de cadena

!

Exhibe el carácter más a la izquierda de los **datos de cadena.**

Ejemplo: PRINT USING "!"; "ABCDE"

\ ——— \

n blancos

Muestra caracteres cuyo número es igual al número de espacios en blanco contenidos entre las dos barras invertidas (\). Cuando la longitud de la cadena a visualizar excede del número de espacios especificados, los caracteres sobrantes no se visualizarán. Si, en cambio, el número de caracteres es inferior al indicado, el resto se rellenará con espacios en blanco.

Ejemplo: PRINT USING "\ \"; "AB"; "ABCDEFG "ABC"

&

Cuando se usa el signo Y comercial ( ), se visualizan todas las cadenas especificadas.

Ejemplo: PRINT USING "&###&###"; "ABC="; 1;" -X+";456

PRINT# — Instrucción

Función

Esta instrucción se usa para enviar datos hacia un fichero específico.

Formato

PRINT# número fichero
PRINT# número fichero, expresión [; expresión...] [;]
PRINT# número ficheor, expresión [, expresión...] [,]
En vez de la palabra clave PRINT puede usarse el signo de interrogación (?).

Descripción

1) La instrucción PRINT# se usa para transferir datos (números o cadenas) especificados por expresiones a un fichero indicado por el número fichero.
2) El número de fichero ha de referirse a un fichero que haya sido abierto con la instrucción OPEN para el modo de salida del ordenador.
3) Los formatos de salida de los datos son idénticos a los descritos para la instrucción PRINT:
   Cada dato numérico va precedido de un espacio en blanco (si es un valor positivo) o el signo menos (-) (si es un valor negativo), y los números se convienrten a cadena, seguida de un espacio en blanco.
   Cuando se trata de una cadena, se envían todos sus caracteres.
4) En una instrucción PRINT# pueden especificarse varias expresiones, separadas con una coma (,) o punto y coma (;).
   El formato de salida de los datos depende del tipo de delimitador usado y es idéntico al empleado para la instrucción PRINT, excepto en que los datos se transfieren a un fichero.
   Cuando los datos numéricos se separan con comas, se ponen en campos, y cada dato puede ir seguido de espacios en blanco.
   Cuando las cadenas se separan con punto y coma, se envian al fichero de modo continuo. Cuando, luego, el fichero es leído con una instrucción INPUT# o LINE INPUT#, las cadenas se ponen en una cadena única.
   Si usted desea que las cadenas de datos sean leídas separadamente cuando se ejecuta una instrucción INPUT#, debe separarlas con comas en la instrucción PRINT#.
   Si quiere que sean leídas separadamente también cuando se ejecuta una instrucción LINE INPUT#, debe separar cada cadena con los códigos retorno del carro CR (&H0D) y avance de línea LF (&H0A) en la instrucción PRINT#.
5) Cuando la instrucción PRINT# no termina con coma ni punto y coma, los datos de salida irán seguidos de CR (&H0D) y LF (&H0A).
   Si al final de la instrucción se pone coma o punto y coma, no se envían dichos códigos de avance de carro y avance de línea. El formato de salida es idéntico al de la instrucción PRINT, excepto en que con PRINT# los datos no se exhiben en la pantalla, sino que se almacenan en un fichero.

Ejemplo

Salida de datos hacia un fichero de cinta cassette:

```
10  OPEN "CAS:SAMPEL" FOR OUTPUT AS 1
20  PRINT#1, 12.3;4.56
30  PRINT#1, "ABC"; "DEF"; ","; "GH";
40  PRINT#1, "IJK"
50  CLOSE
60  END
```

Entrada de datos desde un fichero de cinta cassette:

```
10  OPEN "CAS:SAMPLE "FOR INPUT AS 1
20  INPUT#1, A, B:PRINT A, B
30  INPUT#1, C$:PRINT C$
40  LINE INPUT#1, D$:PRINT D$
50  CLOSE
60  END
```

PRINT# USING

Instrucción

Función

Esta instrucción envía datos (números o cadenas) a un fichero específico, usando un formato dado.

Formato

PRINT# número fichero, USING cadena de control formato;
expresión [; expresión...] [;]

PRINT# número fichero, USING cadena de control formato;
expresión [, expresión...] [,]

En vez de la palabra clave PRINT# puede usarse el signo de interrogación (?).

Descripción

1) La instrucción PRINT# USING se usa para enviar los datos (números o cadenas) especificados por las expresiones y transferirlos al fichero indicado por el número de fichero, usando un formato especificado.
2) La forma de salida de los datos es idéntica a la de la instrucción PRINT#, excepto que exige que se especifique el formato.
3) Las especificaciones de formato son idénticas a las de la instrucción PRINT USING.

Ejemplo

```
PRINT#1, USING" \\####"; "ABC"; 123
```

PSET

Instrucción

Función

Esta instrucción se usa para marcar un punto en una posición especificada de la pantalla gráfica.

Formato

PSET (coordenada X, coordenada Y) [, código color]
PSET STEP (coordenada X, coordenada Y) [, código color]

Descripción

1) La instrucción PSET se usa para marcar un punto en una intersección de coordenadas de la pantalla gráfica, utilizando el color indicado en el código de color.
2) Cuando se omite el código de color, se emplea el antes indicado en la instrucción COLOR.
3) Cuando se usa la especificación relativa de coordenadas con STEP (X,Y), el origen es el último punto de referencia (LP).
   Cuando se ha ejecutado la instrucción PSET, el LP pasa a ser el punto de coordenadas especificado.

Ejemplo

```
10  SCREEN 2
20  FOR K=0 TO 255 STEP 2
30  PSET (K, 50), 1
40  NEXT
50  GOTO 50
```

PUT SPRITE

Instrucción

Función

Esta instrucción se usa para dibujar una forma en la pantalla.

Formato

PUT SPRITE número pantalla de formas, (X,Y) [, código color] [, número forma]
PUT SPRITE número pantalla de formas, (X,Y) [código color], número forma.
Puede usarse la especificación de coordenadas relativas, STEP (X,Y), en lugar de la de coordenadas absolutas, (X,Y).

Descripción

1) La instrucción PUT SPRITE se usa para trazar una forma definida en la instrucción SPRITE$, en un punto de coordenadas especificado de la pantalla de formas, con el número de pantalla de formas indicado.
2) El número de pantalla de formas se especifica con un entero, que puede oscilar entre 0 y 31.

   Hay disponibles hasta 32 pantallas de formas, cada una definida con un entero desde 0 a 31.

   En cada pantalla de formas puede dibujarse sólo una forma; luego, las distintas pantallas así preparadas pueden visualizarse simultáneamente.

   Sin embargo, en cada fila puede haber como máximo cuatro figuras.

   Una pantalla de formas con un número más bajo tiene una mayor prioridad; la pantalla de formas de número más alto quedará oculta detrás de otra de número menor si ambas se solapan.
3) La coordenada X se especifica con una expresión de enteros cuyo valor puede oscilar entre −32 y 255; la coordenada Y se indica en la gama −32 a 191.

   Cuando el punto de coordenadas es especificado con coordenadas relativas, utilizando STEP (X,Y), el origen es siempre el último punto de referencia (LP).

   Cuando se ha ejecutado la instrucción PUT SPRITE el LP se traslada al punto de coordenadas especificado.
4) Para borrar las formas de la pantalla:

   Especifique 208 como coordenada Y. Con ello desaparecerá la forma con el número de pantalla indicado, así como todas las formas de las pantallas de número más alto.

   Si especifica 209 para la coordenada Y, se borrará sólo la forma de la pantalla especificada.
5) Si se omite el código de color, se selecciona el color de visualización actual.
6) Los números de formas corresponden a los definidos en la instrucción SPRITE$.

   Si se omite el número de forma, se selecciona el de la forma actualmente en pantalla. El valor inicial del número de forma es igual al número de la pantalla de formas especificado.

Ejemplo

```
10  SPRITE$ (0)=STRING$ (8, 255)
20  PUT SPRITE 1, (100, 100), 1, 0
30  PUT SPRITE 0, (105, 105), ,8
40  END
```

READ

Instrucción

Función

Esta instrucción se usa para leer un valor de una instrucción DATA y asignarlo a una variable.

Formato

READ variable [, variable...]

Descripción

1) La instrucción READ se usa para leer constantes de una instrucción DATA en orden secuencial y asignarlas a una o más variables.
2) El tipo de variable en la instrucción READ debe coincidir con el tipo de constante en la instrucción DATA de la que se leen los datos.
3) Cuando el número de variables en la instrucción READ excede del número de constantes en la instrucción DATA, las constantes de la siguiente instrucción DATA se asignan a las variables adicionales de la instrucción READ.
4) Cuando el número de variables en la instrucción READ es inferior al número de constantes en la instrucción DATA, las constantes adicionales de ésta última se asignan a variables de la siguiente instrucción READ.
5) La primera instrucción READ de un programa lee datos de la primera instrucción DATA que encuentra (la que tiene el menor número de línea).
   Con la instrucción RESTORE puede especificarse la instrucción DATA en que la instrucción READ empezará a leer datos.

Ejemplo

```
10  DATA ABC, 123, 4.56
20  DATA 7.89, DE, "G Y"
30  READ X$, X:PRINT X$;X
40  READ Y, Z, Y$, Z$:PRINT Y;X;Y$;Z$
50  END
```

REM

Instrucción

Función

Esta instrucción se usa para explicar la razón de un segmento de programa.

Formato

REM [cadena]
' [cadena]

Descripción

1) La instrucción REM no es ejecutada, es decir el comentario que contiene no es exhibido en la pantalla ni surte ningún otro efecto durante la ejecución del programa.
2) Esta instrucción se usa para documentar con comentarios un programa.

Ejemplo

```
10  REM *** SAMPLE PROGRAM
20  GOTO 100
100 'CHECK ROUTINE
110 END
```

RENUM — Comando

Función Este comando se usa para cambiar el número de líneas de programa.

Formato
RENUM [nuevo número línea[, viejo número línea[, incremento]]]
RENUM [nuevo número línea],, incremento

Descripción

1) El comando RENUM se usa para cambiar el número de una línea de programa y todas las subsiguientes, aplicando una nueva secuencia de numeración, a partir del nuevo número de línea, y usando el incremento especificado.
2) Cuando se usa sólo la palabra clave RENUM:
   Renumera automáticamente todas las líneas en incrementos de 10, empezando con la primera línea del programa.
   Cuando se omite el nuevo número de línea:
   La renumeración empieza asignando el número 10 a la primera línea del programa.
   Cuando se omite el viejo número de línea:
   La renumeración empieza en la primera línea del programa.
   Cuando se omite el incremento:
   Las líneas se renumeran en incrementos de 10.
3) Este comando no puede efectuar bifurcaciones en el programa.
4) Al ser ejecutado, cambia también los números de línea especificados en instrucciones GOTO, GOSUB, ON GOTO, ON GOSUB y ERL.

Ejemplo

```
RENUM
RENUM 1000, , 100
```

RESTORE — Instrucción

Función Esta instrucción se usa para especificar el número de línea de la instrucción DATA en que la instrucción READ deberá empezar a leer datos.

Formato
RESTORE [número línea]

Descripción

1) Una instrucción READ existente despúes de una instrucción RESTORE leerá datos de la instrucción DATA cuyo número de línea se especifica en la instrucción RESTORE.
2) Si se omite el número de línea, la instrucción READ empezará a leer datos de la instrucción DATA que posea el número de línea más bajo.

Ejemplo

```
10   RESTORE 130: READ A, B:PRINT A;B
20   READ C, D, E:PRINT C;D;E
30   RESTORE :READ A$:PRINT A$
40   END
100  DATA LINE 100
120  DATA 1.23
130  DATA 130
140  DATA 140, 10, 20, 30, 40, 50
```

RESUME

Instrucción

Función Esta instrucción se incorpora en una rutina de tratamiento de errores para restituir la ejecución al programa principal.

Formato

RESUME [0]
RESUME NEXT
RESUME número línea

Descripción

1) La instrucción RESUME se usa en las rutinas de tratamiento de errores especificadas en instrucciónes ON ERROR GOTO.
   Al ser ejecutada, el control se restituye al programa principal.
2) Cuando se usa sólo RESUME o RESUME 0:
   La ejecución se reanuda en la línea en que ocurrió el error y se intenta ejecutar de nuevo la misma instrucción.
   Cuando se usa RESUME NEXT:
   La ejecución se reanuda en la instrucción que sigue a la que causó error.
   Cuando se especifica "RESUME número línea":
   La ejecución se reanuda a la línea indicada.

RETURN

Instrucción

Función Esta instrucción se usa para salir de una subrutina.

Formato RETURN [número línea]

Descripción

1) La instrucción RETURN se usa en una subrutina especificada en una instrucción GOSUB u ON GOSUB, o en una rutina de tratamiento de interrupciones, para restituir la ejecución a la línea de programa especificada.
2) Cuando se omite el número de línea, la ejecución se restituye a la instrucción que sigue a la instrucción GOSUB.
3) En una misma subrutina puede haber más de una instrucción RETURN.

Ejemplo

```
10  X=1 :GOSUB 100
20  X=2 :GOSUB 100
30  END
100 PRINT X
110 IF X=2 THEN RETURN
120 RETURN 20
```

RIGHT$ Función

Función Esta función devuelve el número de caracteres deseado de una cadena, empezando por el situado más a la derecha.

Formato RIGHT$ (expresión de cadena, número de caracteres)

Descripción

1) La función RIGHT$ devuelve el número de caracteres deseado de la cadena especificada, comenzando por el extremo derecho.
2) Cuando el número de caracteres especificado excede del número de caracteres de la cadena, se devuelve la cadena entera.
3) Si se especifican 0 caracteres se devolverá una cadena nula ("").
4) La cabecera de un caracter gráfico (&H01) se cuenta como un carácter. Por ello un símbolo gráfico ocupa realmente dos posiciones.

Ejemplo

```
10  A$="ABCDF"
20  B$=RIGHT$ (A$, 3)
30  PRINT B$
40  END
```

RND Función

Función Esta función devuelve un número aleatorio entre 0 y 1.

Formato RND (expresión numérica)

Descripción

1) La función RND devuelve un número aleatorio entre 0 y 1 (exclusivos).
2) Cuando el valor de la expresión numérica es positivo:
   Cada vez que se ponga en marcha el programa se generará la misma secuencia de números aleatorios.
   Cuando el valor de la expresión numérica es 0:
   Se devuelve el número aleatorio anterior.
   Cuando el valor de la expresión numérica es negativo:
   Se generan diferentes secuencias de números aleatorios, dependiendo del valor de la expresión.
3) Normalmente se usa un valor positivo para la expresión. Por lo tanto cada vez que se ponga en marcha el programa se generará la misma secuencia, a menos que el generador de números aleatorios se reinicialice.
   Para cambiar la secuencia de números aleatorios ejecute X=RND (−TIME) para seleccionar otra secuencia y a continuación ejecute RND (valor positivo). La variable X es una variable ficticia y puede ser cualquier número.

Ejemplo

```
10  X=RND (-TIME)
20  A=RND (1) :A=INT (20*A)+5
30  PRINT A; :LOCATE A:PRINT "*"
40  GOTO 20
```

RUN — Comando

Función Este comando empieza la ejecución del programa.

Formato RUN [número línea]

Descripción

1) El comando RUN se usa para empezar la ejecución del programa desde el número de línea especificado.
   Cuando este comando es ejecutado, se borra el contenido de todas las variables y se cierran todos los ficheros abiertos.
2) Cuando se omite el número de línea, la ejecución empieza en la primera línea del programa.

Ejemplos

```
RUN
RUN 1000
```

SAVE — Comando

Función Este comando se usa para grabar un programa en código ASCII en un fichero especificado.

Formato SAVE "nombre dispositivo [nombre fichero]"

Descripción

1) El comando SAVE se usa para almacenar un programa en código ASCII en el fichero especificado por el nombre de fichero, en el dispositivo indicado en el nombre dispositivo.
2) Un programa en código ASCII es un programa imagen de la memoria en el que las líneas del programa están representadas por códigos de caracteres ASCII. Tiene las siguientes características:
   Ocupa un espacio de fichero mayor que el requerido por el comando CSAVE. Los programas que deseen combinar con el comando MERGE deben estar ambos en código ASCII.
   Dado que un fichero de programa ASCII puede ser tratado como un fichero de datos, cada línea de programa puede ser asignada a una variable usando la instrucción LINE INPUT#.
   Al final de cada línea de un programa ASCII hay un código de retorno de carro CR (&H0D) y de avance de línea LF (&H0A).
   Al final del fichero hay un código de fin (&H1A).
3) Al grabar un fichero en cinta cassette, la velocidad de transferencia de datos puede especificarse con la instrucción SCREEN.

Ejemplo

```
SAVE "CAS:SAMPLE"
```

SCREEN

Instrucción

Función

Esta instrucción se usa para especificar modos de pantalla, el tamaño de las formas, el sonido de las teclas, la velocidad de transferencia de datos para grabar o cargar ficheros y las opciones de impresora.

Formato

SCREEN modo pantalla
SCREEN [modo pantalla], tamaño de la forma
SCREEN [modo pantalla], [tamaño de la forma], interruptor sonido tecla
SCREEN [modo pantalla], [tamaño de la forma], [interruptor sonido teclas], velocidad.
SCREEN [modo pantalla], [tamaño de la forma], [interruptor sonido teclas], [velocidad] conmutador impresora

Descripción

1) Modo pantalla
Especifica los modos de visualización (vea una descripción de éstos en el Capítulo 1, Sección 5, "Control de la Pantalla")
El modo pantalla es especificado con un entero, cuyo valor va del 0 al 3:
   - 0 Modo Texto de 24 filas×40 columnas (valor inicial: 24 filas×37 columnas).
   - 1 Modo Texto de 24 filas×32 columnas (valor inicial: 24 filas×29 columnas).
   - 2 Modo gráfico de alta resolución
   - 3 Modo multicolor

2) Tamaño de las formas
Especifica el tamaño de las formas en la variable SPRITE$, y la escala de las figuras exhibidas por la instrucción PUT SPRITE. El tamaño de las figuras se especifica con un entero cuyo valor va de 0 al 3:
   - 0 Exhibe las formas con una configuración de 8×8 puntos cuando poseen dicha configuración.
   - 1 Aumenta las figuras de 8×8 puntos a 16×16 puntos, con lo que dobla su tamaño en la pantalla.
   - 2 Exhibe las formas con una configuración de 16×16 puntos cuando poseen dicha configuración.
   - 3 Aumenta las figuras de 16×16 puntos a 32×32 puntos, es decir dobla su tamaño en la visualización.

3) Interruptor de sonido de las teclas
Conecta y desconecta el sonido de las teclas y puede especificarse con un entero cuyo valor puede ser 0 ó 1:
   - 0 El interruptor del sonido de las teclas queda desconectado.
   - 1 El interruptor del sonido de las teclas es conectado.

4) Velocidad de transferencia de datos al cassette.
Especifica la velocidad con que se graban los datos con comandos CSAVE, BSAVE o SAVE, así como en la instrucción PRINT#.
Cuando un fichero es leído de un cassette, la velocidad de transferencia se ajusta automáticamente, por lo que no necesita ser estipulada.
La velocidad puede seleccionarse con un entero, cuyo valor puede ser 1 ó 2:
   - 1 1200 baudios
   - 2 2400 baudios

6) Conmutador de la impresora

Especifica si la impresora acoplada al ordenador es un modelo MSX o no. Se selecciona con un entero, cuyo valor puede ser 0 ó 1:

0 Está acoplada una impresora tipo MSX.

1 La impresora no pertenece al tipo MSX.

SGN — Función

Función Esta función da el signo de un valor.

Formato SGN (expresión numérica)

Descripción

1) La función SGN da el signo del valor de la expresión numérica que la sigue, con los enteros -1, 0 y 1:
   - −1 El valor es negativo.
   - 0 El valor es cero.
   - 1 El valor es positivo.

Ejemplo

```
10  A=SGN (−1.34)
20  B=SGN (0)
30  C=SGN (4.56)
40  PRINT A;B;C
50  END
```

SIN — Función

Función Esta función calcula el seno trigonométrico de un número.

Formato SIN (expresión numérica)

Descripción

1) La función SIN calcula el seno trigonométrico del valor de la expresión numérica que la sigue.
2) El valor de la expresión numérica debe ser en radianes.
3) El resultado se da siempre como número real de doble precisión, cualquiera que sea el tipo de la expresión numérica.

SOUND

Instrucción

Función

Esta instrucción se usa para entrar valores en los registros del generador de sonido programable (GSP).

Formato

SOUND número registro, expresión entera

Descripción

1) La instrucción SOUND se usa para entrar el valor de la expresión numérica en el registro GSP cuyo número se especifica.
   La combinación de varias instrucciones SOUND permite crear efectos sonoros que no serían posibles con sólo instrucciones PLAY.
2) El número de registro se especifica con un entero de 0 — 13.
   El GSP tiene 16 registros, de los cuales 14 pueden emplearse para entrar valores.

N° de Registro	Función	Bit							
		b7	b6	b5	b4	b3	b2	b1	b0
0	Frecuencia del canal A	FT (A)							
1						CT (A)			
2	Frecuencia del canal B	FT (B)							
3						CT (B)			
4	Frecuencia del canal C	FT (C)							
5						CT (C)			
6	Frecuencia del ruido				NP				
7	Selección del canal			Ruido			Tono		
		1	0	C	B	A	C	B	A
8	Sonoridad del canal A				M	L (A)			
9	Sonoridad del canal B				M	L (B)			
10	Sonoridad del canal C				M	L (C)			
11	Periodo de la envolvente	FT (E)							
12		CT (E)							
13	Pauta de la envolvente					EP			

SPACE$ — Función

Función Esta función da una longitud especificada de espacios en blanco.

Formato SPACE$ (expresión entera)

Descripción

1) La función SPACE$ da una cadena vacía cuya longitud es indicada por la expresión entera que la sigue.
2) El valor de la expresión entera puede ser un entero entre 0 y 255.

Ejemplo

```
A$=SPACE$ (5)
PRINT "AB"; SPACE$ (5); "CD"
```

SPC — Función

Función Esta función envía una cadena vacía de la longitud indicada, sea hacia la pantalla o hacia la impresora.

Formato SPC (expresión entera)

Descripción

1) La función SPC se usa en instrucciones de salida, tales como LPRINT o PRINT, para emitir una cadena vacía cuya longitud es especificada en la expresión entera que la sigue.
2) Esta función se usa sólo en instrucciones de salida y no puede emplearse en instrucciones de asignación, tales como LET.
3) El valor de la expresión entera ha de ser un entero en la gama 0 — 255.

Ejemplo

```
PRINT "AB"; SPC (5); "CD"
```

SPRITE ON

Instrucción

Función Esta instrucción se usa para habilitar interrupciones de las formas.

Formato SPRITE ON

Descripción La instrucción SPRITE ON se usa para habilitar interrupciones causadas por colisiones de formas en la pantalla.
Si ocurre una colisión de formas en la pantalla después de haberse ejecutado una instrucción SPRITE ON, se produce una interrupción, que hace que la ejecución pase a la rutina de tratamiento de interrupciones especificada en la instrucción ON SPRITE GOSUB.

SPRITE OFF

Instrucción

Función Esta instrucción se usa para deshabilitar interrupciones de las formas.

Formato SPRITE OFF

Descripción Cuando se ha ejecutado la instrucción SPRITE OFF, no ocurren interrupciones cuando colisionan formas en la pantalla.

SPRITE STOP

Instrucción

Función Esta instrucción se usa para retener interrupciones de las formas.

Formato SPRITE STOP

Descripción

1) La instrucción SPRITE STOP se usa para retener interrupciones causadas por colisiones de formas, hasta que se ejecuta la instrucción SPRITE ON. Si se produce una colisión de figuras en la pantalla después de que se haya ejecutado la instrucción SPRITE STOP, ocurre una interrupción cuando se ejecuta luego la instrucción SPRITE ON, haciendo que el control pase a la rutina de tratamiento de interrupciones especificada en la instrucción ON SPRITE GOSUB.
2) Si no se ha ejecutado ninguna instrucción SPRITE ON antes de SPRITE STOP, las interrupciones no son retenidas, sino que no se tienen en cuenta.

SPRITE$ — Variable del Sistema

Función Esta variable del sistema se usa para definir una forma.

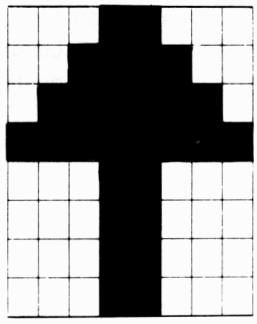
Formato SPRITE$ (expresión entera)=expresión cadena

Descripción

1) La variable SPRITE$ se usa para definir la forma especificada por la expresión de cadena para la forma cuyo número se indica en la expresión entera.
2) La gama de la expresión entera difiere según el tamaño de figura especificado en la instrucción SCREEN:

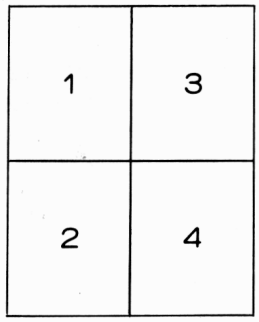
Tamaño de figura según instrucción SCREEN	Tamaño de figura en la pantalla	Gama de expresión de enteros
0 ó 1	8×8 puntos	0~255
2 ó 3	16×16 puntos	0~32

3) La expresión cadena usa 8 caracteres para especificar una forma de 8×8 puntos, y 32 caracteres cuando el tamaño es de 16×16 puntos.
   - Expresión cadena para 8×8 puntos:
     Cada fila (constituida por 8 puntos) de una matriz de forma está representada por una pauta de bits.
     Es decir, un punto de pantalla se conecta con un bit de activación y se desconecta con un bit de desactivación.
     Para representar una matriz de forma completa se usan ocho caracteres.



CHR$ (&B00011000)
CHR$ (&B00111100)
CHR$ (&B01111110)
CHR$ (&B11111111)
CHR$ (&B00011000)
CHR$ (&B00011000)
CHR$ (&B00011000)
CHR$ (&B00011000)

   - Expresión cadena 16×16 puntos:
     Una forma se divide en cuatro secciones de 8×8 puntos cada una, y estas secciones de disponen en la forma ilustrada en la siguiente figura. Dado que la matriz en cada sección está representada por 8 caracteres, para representar las cuatro secciones de la matriz se requieren 32 caracteres.



Ejemplo

```
10  SCREEN 1, 1
20  P$=""
30  FOR K=0 TO 7
40  READ D$
50  P$=P$+CHR$ (VAL ("&B" + D$))
60  NEXT
70  SPRITE$ (0)=P$
80  PUT SPRITE 0, (50, 50), 1
90  END
100 DATA 00011000
110 DATA 00111100
120 DATA 01111110
130 DATA 11111111
140 DATA 00011000
150 DATA 00011000
160 DATA 00011000
170 DATA 00011000
```

SQR — Función

Función Esta función calcula la raíz cuadrada de un número.

Formato SQR (expresión numérica)

Descripción

1) La función SQR calcula la raíz cuadrada del valor de la expresión numérica que la sigue.
2) El valor de la expresión numérica debe ser mayor que cero.
3) El resultado se da siempre como número real de doble precisión, cualquiera que sea el tipo de la expresión numérica.

Ejemplo

```
A=SQR(4)
```

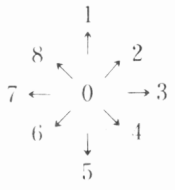
STICK — Función

Función Esta función comunica la dirección de la operación de un mando de juego.

Formato STICK (expresión entera)

Descripción

1) La función STICK comunica la dirección de operación del mando de juego (joystick) especificado por la expresión entera, o bien la tecla de control del cursor que se ha pulsado.
2) El valor de la expresión entera es 0, 1 ó 2, con el siguiente significado:
   0 Tecla o teclas de movimiento del cursor en el teclado.
   1 Mando de juego conectado al conector JOYSTICK 1.
   2 Mando de juego conectado al conector JOYSTICK 2.
3) El resultado dado es un entero de 1 a 8, que representa la dirección de operación.
   Si el mando de juego no es operado, la función da cero.



Cuando se pulsan simultáneamente teclas de movimiento del cursor con direcciones perpendiculares, la función da un valor que indica una dirección diagonal de operación.
Por ejemplo, la pulsación simultánea de las teclas de flechas [⇧] y [⇨] hace que la función dé 2.

Ejemplo

```
10  A=STICK (0) :B=STICK (1)
20  PRINT A; B
30  GOTO 10
```

STOP

Instrucción

Función

Esta instrucción detiene la ejecución del programa y restituye el sistema al modo Comando.

Formato

STOP

Descripción

1) La instrucción STOP se usa para detener la ejecución del programa y restituye el sistema al modo Comando, con el siguiente mensaje en pantalla.

   BREAK in nnnn

   (número de línea)

2) A diferencia de la instrucción END, la instrucción STOP no cierra los ficheros que pueda haber abiertos.
3) La ejecución de un programa interrumpida con una instrucción STOP puede reanudarse con la instrucción CONT.

Ejemplo

```
10  PRINT "A"
20  STOP
```

STOP ON

Instrucción

Función Esta instrucción se usa para habilitar una interrupción causada por la pulsación simultánea de las teclas [CTRL] y [STOP]

Formato STOP ON

Descripción Cuando se ha ejecutado una instrucción STOP ON, se produce una interrupción si se pulsan simultáneamente las teclas [CTRL] y [STOP], con lo que la ejecución pasa a la rutina de tratamiento de interrupciones especificada en la instrucción ON STOP GOSUB.

STOP OFF

Instrucción

Función Esta instrucción se usa para deshabilitar una interrupción causada por la pulsación simultánea de las teclas [CTRL] y [STOP]

Formato STOP OFF

Descripción Cuando se ha ejecutado la instrucción STOP OFF, la pulsación simultánea de las teclas [CTRL] y [STOP] no causa una interrupción.

STOP STOP

Instrucción

Función Esta instrucción se usa para retener una interrupción causada por la pulsación simultánea de las teclas [CTRL] y [STOP]

Formato STOP STOP

Descripción

1) La instrucción STOP STOP se usa para retener una interrupción causada por la pulsación simultánea de las teclas [CTRL] y [STOP], hasta que se usa la instrucción STOP ON.
   Si las teclas [CTRL] y [STOP] se pulsan simultáneamente después de que se haya ejecutado una instrucción STOP STOP, se producirá una interrupción cuando se ejecute la instrucción STOP ON, haciendo que la ejecución pase a la rutina de tratamiento de interrupciones especificada en la instrucción ON STOP GOSUB.
2) Si no se ha ejecutado ninguna instrucción STOP ON antes de STOP STOP, las interrupciones no son retenidas, sino que se ignoran.

STRIG — Función

Función

Esta función da valores que indican si el botón accionador del mando de juego es operado o no.

Formato

STRIG (expresión entera).

Descripción

1) La función STRIG da valores indicativos de si el botón accionador del mando de juegos o bien la barra espaciadora del teclado han sido pulsados.
2) El valor de la expresión entera puede ser sólo de 0 a 4 y especifica la barra espaciadora del teclado o uno de los botones accionadores de un mando de juegos conectado a uno cualquiera de los conectores JOYSTICK del ordenador.
   - 0 Barra espaciadora del teclado
   - 1 1er botón del mando juegos conectado a JOY 1
   - 2 1er botón del mando juegos conectado a JOY 2
   - 3 2° botón del mando juegos conectado a JOY 1
   - 4 2° botón del mando juegos conectado a JOY 2
3) El resultado puede ser el entero -1 ó 0.
   -1 significa que el botón ha sido pulsado.
   0 significa que el botón no ha sido pulsado.

Ejemplo

```
10  A=STRIG (0) :B=STRIG (1)
20  PRINT A;B
30  GOTO 10
```

STRIG (n) ON

Instrucción

Función Esta instrucción se usa para habilitar interrupciones causadas por operaciones del botón accionador de un mando de juegos.

Formato STRIG (expresión entera) ON

Descripción

1) Si uno cualquiera de los botones accionadores del mando de juegos especificado por la expresión entera o bien la barra espaciadora se pulsan después de que se haya ejecutado una instrucción STRIG (n) ON, se producirá una interrupción que causará el paso de la ejecución a una rutina de tratamiento de interrupciones especificada por la instrucción ON STRIG GOSUB.
2) El significado de los valores de la expresión entera es idéntico al descrito para la función STRIG.

STRIG (n) OFF

Instrucción

Función Esta instrucción se usa para deshabilitar interrupciones causadas por la operación del botón accionador de un mando de juegos.

Formato STRIG (expresión entera) OFF

Descripción

1) La instrucción STRIG (n) OFF se usa para deshabilitar interrupciones causadas por la pulsación de un botón accionador de un mando de juegos o bien de la barra espaciadora del teclado.

STRIG (n) STOP

Instrucción

Función Esta instrucción se usa para retener una interrupción causada por el accionamiento de un botón de mando de juegos.

Formato STRIG (expresión entera) STOP

Descripción

1) La instrucción STRIG (n) STOP se usa para retener una interrupción causada por la pulsación de un botón accionador del mando de juegos especificado por la expresión entera, o de la barra espaciadora del teclado, hasta que se ejecuta una instrucción STRIG (n) ON.
   Si el botón del mando de juegos especificado o la barra espaciadora se pulsan después de que se haya ejecutado la instrucción STRIG (n) STOP, se producirá una interrupción cuando se ejecute luego la instrucción STRIG (n) ON, con lo que la ejecución pasará a la rutina de tratamiento de interrupciones.
2) Si no se ejecutó ninguna instrucción STRIG (n) ON antes de la ejecución de STRIG (n) STOP, las interrupciones no serán retenidas sino que no se tendrán en cuenta.
3) El significado de los valores de la expresión entera es el mismo que el descrito en la función STRIG.

STR$ — Función

Función Esta función convierte un número a una cadena.

Formato STR$ (expresión numérica)

Descripción
1) La función STR$ de una cadena que representa el valor decimal de la expresión numérica que la sigue.
2) La expresión numérica puede ser de cualquier tipo.

Ejemplo
```
A$=STR$ (123.45)
B$=STR$ (&HFF)
```

STRING$ — Función

Función Esta función da una cadena de la longitud especificada del carácter que se indica.

Formato
STRING$ (expresión entera, expresión cadena)
STRING$ (expresión entera, código carácter)

Descripción
1) La función STRING$ da una cadena (cuya longitud es especificada por la expresión entera) del primer carácter de la expresión de cadena o del carácter dado por el código de carácter.
2) El valor de la expresión entera ha de ser entre 0 y 255.
3) Sólo el primer carácter de la expresión de cadena es significativo, el resto no se tiene en cuenta.
   Una cabecera de identificación de carácter gráfico (&H01) es un símbolo gráfico que se cuenta como carácter. Por consiguiente, si se coloca un símbolo gráfico en la primera posición de la expresión de cadena, el resultado será una cadena cuyo código de carácter es siempre &H01.

Ejemplo
```
A$=STRING$ (10, "ABC")
A$=STRING$ (10, &H41)
```

SWAP — Instrucción

Función Esta instrucción intercambia el valor de dos variables.

Formato SWAP variable 1, variable 2

Descripción
1) La instrucción SWAP intercambia el valor de la variable 1 con el de la variable 2.
2) El tipo de ambas variables ha de ser idéntico.
   No es posible intercambiar valores entre una variable numérica y otra de cadena.
   En el caso de variables numéricas, los valores pueden intercambiarse también sólo si ambas variables son del mismo tipo: tipo entera, tipo real de simple precisión o tipo real de doble precisión.

Ejemplo
```
SWAP A, B
SWAP A%, B%
SWAP A$, B$
SWAP A(5), B
```

TAB — Función

Función Esta función inserta espacios en blanco en la posición de columna especificada, sea en la pantalla o en la impresora.

Formato TAB (expresión entera)

Descripción

1) La función TAB se usa en instrucciones de salida, tales como PRINT o LPRINT, para situar espacios en blanco en la pantalla o en la impresora, empezando con la posición actual del cursor hasta la posición de columna especificada.
2) Esta función no puede usarse en instrucciones de asignación, tales como LET.

Ejemplo

```
PRINT "ABC"; TAB (10); "CDE"
```

TAN — Función

Función Esta función calcula la tangente trigonométrica de un número.

Formato TAN (expresión numérica)

Descripción

1) La función TAN calcula la tangente trigonométrica del valor de la expresión numérica que la sigue.
2) El valor de la expresión numérica ha de estar en radianes.
3) El resultado se da como número real de doble precisión, cualquiera que sea el tipo de expresión numérica.

TIME — Variable del Sistema

Función Esta variable se usa para obtener o ajustar el valor del reloj interno (cronómetro de intervalos).

Formato TIME

Descripción

1) La variable TIME se usa para asignar el valor de un cronómetro de intervalos interno que se incrementa en 1 a intervalos de aprox. 1/50 seg.
2) El cronómentro de intervalos puede ajustarse asignando el valor deseado a la variable TIME.
3) El cronómetro se para mientras se entran datos de un fichero cassette o se graban en él. Dado que el cronómetro cuenta interrupciones en intervalos de 1/50 segundo desde el procesador de visualización (VDP), se para cuando se realiza una operación de E/S a un fichero cassette, ya que durante dicha operación las interrupciones del procesador de visualización están deshabilitadas.

Ejemplo

```
A=TIME
TIME=0
```

TRON — Comando

Función Este comando se usa para rastrear la ejecución de un programa paso a paso.

Formato TRON

Descripción
1) El comando TRON hace que la línea de programa que se acaba de ejecutar se exhiba en la pantalla de modo texto, con su número de línea entre corchetes[ ].
2) Para cancelar el modo de rastreo, ejecute el comando TROFF o NEW.

Ejemplo TRON

TROFF — Comando

Función Este comando se usa para cancelar el modo de rastreo entrado con TRON.

Formato TROFF

Descripción El comando TROFF se usa para terminar el modo de rastreo seleccionado con el comando TRON.

USR — Función

Función Esta función hace que se ejecute una subrutina escrita en lenguaje máquina.

Formato USR [número] (argumento)

Descripción
1) La función USR hace que una subrutina en lenguaje máquina sea ejecutada a partir de la dirección definida en la instrucción DEFUSR.
2) El número es un entero entre 0 y 9, que corresponde al número definido en la instrucción DEFUSR. Cuando no se especifica ningún número, se selecciona cero.
3) El argumento debe especificarse. Si no se necesita transferencia de argumento, deberá usarse un argumento ficticio.
4) La ejecución errónea de esta función puede provocar un desbocamiento del programa del que puede salirse solamente desconectando el ordenador.

Ejemplos
A=USR ("ABC")
B=USR2 (0)

VAL — Función

Función Esta función da el valor numérico de una cadena.

Formato VAL (expresión cadena)

Descripción

1) La función VAL da el valor numérico de una cadena especificada por la expresión de cadena que la sigue.
2) La cadena puede ser binaria (&B), octal (&O) o hexadecimal (&H), a condición de que represente un valor numérico.
3) Si la cadena no es numérica, el resultado es cero.

Ejemplos

```
A=VAL ("-1.234")
B=VAL ("&B1010")
C=VAL ("&H" + "1F")
```

VARPTR — Función

Función Esta función da la primera dirección de un área de variables o de un bloque de control de fichero.

Formato

```
VARPTR (variable)
VARPTR (# número fichero)
```

Descripción

1) Cuando se especifica una variable, la función VARPTR da la primera dirección del área de variables en que está almacenado el valor asignado a dicha variable.
2) Cuando se especifica "# número fichero", la función VARPTR da la primera dirección del bloque de control de fichero que se ha especificado con el número de de fichero.

Ejemplos

```
A=VARPTR (X)
B=VARPTR (#1)
```

VDP — Variable del Sistema

Función Esta variable se usa para cargar valores en los registros del procesador de visualización VDP (Video Display Procesador), así como para obtener valores de dichos registros.

Formato VDP (expresión entera)

Descripción

1) La variable VDP se usa para cargar un valor en el registro VDP cuyo nombre se especifica en la expresión entera, así como para obtener el valor de dicho registro.
2) El valor de la expresión entera puede ser de 0 a 8 y se usa para especificar uno de los registros VDP.
3) El uso erróneo de esta variable puede provocar una visualización anormal. Antes de usar esta variable consulte la descripción del hardware del sistema MSX, para comprender plenamente el funcionamiento del procesador VDP.

Ejemplos

```
A=VDP (0)
VDP(7)=1
```

VPEEK — Función

Función Esta función da el contenido de la memoria RAM de visualización (VRAM).

Formato VPEEK (dirección)

Descripción

1) La función VPEEK da el contenido de la dirección de memoria VRAM especificada.
2) La dirección se especifica con una expresión entera en la gama &H0 — &H3FFF.

Ejemplo

```
A=VPEEK (&H1000)
```

VPOKE — Instrucción

Función Esta instrucción se usa para escribir 1 byte de datos en una posición específica de la memoria de visualización (VRAM).

Formato VPOKE dirección, expresión entera

Descripción

1) La instrucción VPOKE se usa para escribir 1 byte de datos, especificado en la expresión entera, en la dirección VRAM indicada.
2) La dirección se especifica con una expresión entera que puede oscilar entre &H0 y &H3FFF.
3) El valor de la expresión entera debe estar dentro de la gama 0 — 255.
4) El uso erróneo de esta instrucción puede provocar un funcionamiento anómalo de la pantalla. Antes de emplear esta instrucción consulte la descripción del hardware del sistema MSX, para comprender plenamente el funcionamiento de VDP.

Ejemplo

```
VPOKE &H1803, &H41
```

WAIT — Instrucción

Función Esta instrucción hace que el sistema espere hasta que una vía de acceso (port) de entrada posea un valor especificado.

Formato WAIT dirección port, expresión entera 1 (, expresión entera 2)

Descripción

1) La instrucción WAIT hace que el sistema espere a que los datos cuya pauta de bits corresponda a la especificada en las expresiones 1 y 2 entre en el port indicado por la dirección.
2) El sistema lee el valor del port de entrada especificado y lo compara con el de la expresión 2, aplicando XOR (OR exclusiva bit a bit). El resultado de XOR es comparado entonces con el valor de la expresión 1, aplicando AND. Si el resultado de la operación AND es -1 (verdadero), la ejecución continúa con la siguiente instrucción. Si el resultado es 0 (falso), el sistema vuelve a leer un valor desde el mismo port de entrada.
3) Si se omite la expresión 2, se le da el valor de cero.
4) El uso erróneo de esta instrucción puede hacer que el sistema no tenga en cuenta los datos procedentes del resto de los ports de entrada, de cuyo estado puede restituirse sólo mediante la desconexión del ordenador.
   El direccionamiento de las vías de acceso se explica en el Capítulo 3, Sección 4, "Mapa de Entrada/Salida".

WIDTH — Instrucción

Función Esta instrucción se usa para especificar el número de caracteres por fila a visualizar en una pantalla de modo texto.

Formato WIDTH expresión entera

Descripción

1) La instrucción WIDTH se usa para seleccionar la anchura de visualización en la pantalla de modo texto.
2) La gama especificable varía según el modo de pantalla:
   Modo texto 40×24 ..... de 1 a 40
   Modo texto 32×24 ..... de 1 a 32
3) Si la imagen visualizada rebasara uno u otro extremo de la pantalla física, use la instrucción WIDTH para reducir la anchura de visualización.
4) La anchura inicial de pantalla es:
   Modo texto 40×24: 37 columnas
   Modo texto 32×24: 29 columnas

Ejemplo WIDTH 20

CAPITULO 3

REFERENCIA

1. TABLA DE CODIGOS DE CARACTERES

Los códigos de caracteres están representados por los enteros hexadecimales del &H00 al &HFF, a los cuales se asignan los caracteres y símbolos reproducidos en la siguiente tabla.
El código de un carácter es representado por un número hexadecimal de dos dígitos, compuesto por los 4 bits más significativos y los 4 bits menos significativos.

4 bits más significativos →

4 bits menos significativos →

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0			Blank (Space)	ø	@	P	`	p	C	É	á	Ã	▄	◢	α	☰
1			!	1	A	Q	a	q	ü	æ	í	ã	▚	▓	β	±
2		INS	"	2	B	R	b	r	é	Æ	ó	Ĩ	▀	▓	Γ	≥
3			#	3	C	S	c	s	â	ô	ú	Ĩ	▛	▟	Π	≤
4			$	4	D	T	d	t	ä	ö	ñ	Õ	▬	▟	Σ	∫
5			%	5	E	U	e	u	à	ò	Ñ	õ	▙	▟	σ	⌠
6			&	6	F	V	f	v	å	û	a	Ũ	▌	▟	μ	÷
7	BL		´	7	G	W	g	w	ç	ù	o	ũ	▚	▞	τ	≈
8	BS	SELECT	(	8	H	X	h	x	ê	ÿ	¿	∏	▌	△	Φ	°
9	TAB		)	9	I	Y	i	y	ë	Ö	⌐	ij	▐	╪	Θ	•
A	LF		*	:	J	Z	j	z	è	Ü	¬	¾	▐	ω	Ω	•
B	HOME	ESC	+	;	K	[	k	{	ï	¢	½	~	▨	▀	δ	√
C	CLS	→	,	<	L	\	l	¦	î	£	¼	◇	▨	▂	∞	n
D	CR	←	−	=	M	]	m	}	Ì	¥	¡	‰	▼	▗	φ	²
E		↑	.	>	N	^	n	~	Ä	Pt	≪	¶T	▲	▗	∈	▮
F		↓	/	?	O	_	o	Blank (DEL)	Å	ƒ	≫	§	▶	▂	∩	Blank (FF)

↑ Números hexadecimales

Números hexadecimales ←

Con la cabecera de identificación de carácter gráfico

	4	5
0	Blank (NULL)	┼
1	☺	┤
2	☻	├
3	♥	┬
4	♦	┴
5	♣	┼
6	♠	│
7	•	─
8	◘	┐
9	○	└
A	◙	┘
B	♂	┌
C	♀	╳
D	♪	╱
E	♫	╲
F	☼	┼

Cabecera de carácter gráfico (hexadecimal &H01)

Los caracteres precedidos por una cabecera de identificación de carácter gráfico se usan para representar símbolos gráficos.

(Ejemplo): Al ejecutarse PRINT CHR$(1); CHR$(&H41), se exhibe ☺
Al ejecutar PRINT CHR$(&H41) se exhibe "A"

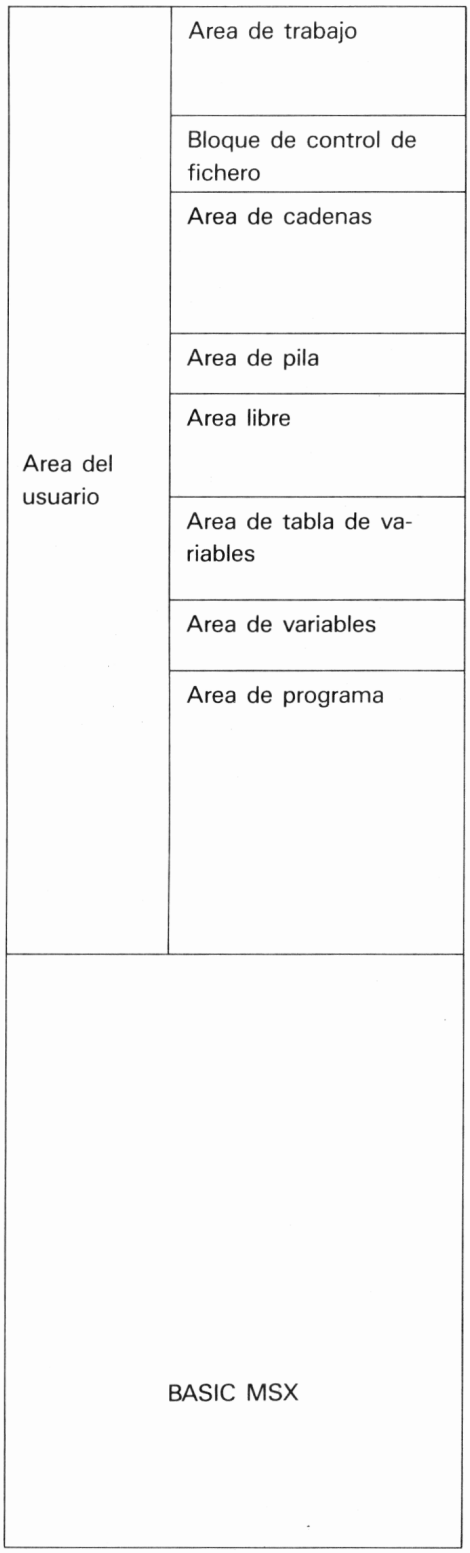
Cuando un símbolo gráfico se pulsa en el teclado, va precedido automáticamente de una cabecera de identificación de carácter gráfico (header), por lo que el código de carácter resultante consiste en dos bytes.

Códigos de control

Los códigos de caracteres cuyos 4 bits de orden alto representan un valor de 0 ó 1 se usan para controlar el cursor y la pantalla, y sus caracteres equivalentes no se muestran en la pantalla:
(Ejemplo): La ejecución de PRINT CHR$(&H0C) borra la pantalla.
Se generan códigos de control también cuando se pulse una tecla de datos más la tecla [CTRL], o una tecla de control.

Código de control	Abrev.	Teclas	Descripción
01	—	[CTRL] y [A]	• Genera una cabecera de carácter gráfico.
02	—	[CTRL] y [B]	• Retrocede el cursor a la primera posición del dato anterior.
03	—	[CTRL] y [C] , o [CTRL] y [STOP]	• Detiene la ejecución del programa.
05	—	[CTRL] y [E]	• Borra los caracteres desde la posición actual del cursor hasta el final de la línea.
06	—	[CTRL] y [F]	• Avanza el cursor a la primera posición del dato siguiente.
07	BL	[CTRL] y [G]	• Hace sonar el zumbador.
08	BS	[CTRL] y [H] o [BS]	• Borra el carácter anterior a la posición actual del cursor.
09	TAB	[CTRL] y [I] o [TAB]	• Tabula el cursor a intervalos de ocho posiciones.
0A	LF	[CTRL] y [J]	• Avance de línea (sitúa el cursor en la primera posición de la fila siguiente).
0B	HOME	[CTRL] y [K] o [HOME]	• Restituye el cursor a la primera posición de la primera fila.
0C	CLS	[CTRL] y [L] o [SHIFT] y [HOME]	• Borra la pantalla.
0D	CR	[CTRL] y [M] o [RETURN]	• Retorno del carro (completa la operación de entrada).
0E	—	[CTRL] y [N]	• Sitúa el cursor a la fila inferior de la pantalla.
12	INS	[CTRL] y [R] o [INS]	• Inserta un carácter.
15	—	[CTRL] y [U]	• Borra una línea.
1C	→	[CTRL] y [\] o [⇨]	• Desplaza el cursor hacia la derecha.
1D	←	[CTRL] y [ ] o [⇦]	• Desplaza el cursor hacia la izquierda.
1E	↑	[CTRL] y [^] o [⇧]	• Desplaza el cursor hacia arriba.
1F	↓	[CTRL] y [SHIFT] y [_] o [⇩]	• Desplaza el cursor hacia abajo.

2. MAPA DE LA MEMORIA



- Area de trabajo (Work area)
  Area de trabajo del lenguaje BASIC.

- Area del usuario (User area)
  En este área el usuario puede almacenar sus propios programas. Las direcciones del área del usuario, pueden seleccionarse en &HF380 e inferiores mediante la instrucción CLEAR.

- Bloque de control de fichero (File control block)
  Este área se usa para ficheros de entrada/salida. El tamaño de este bloque corresponde al número de ficheros especificado en la instrucción MAXFILES.

- Area de cadenas (String area)
  Este área se usa para almacenar cadenas asignadas a variables de cadena. El tamaño de este área se especifica en la instrucción CLEAR. El tamaño inicial es 200 bytes.

- Area de pila (Stack area)
  Este área almacena las direcciones de retorno para FOR — NEXT, GOSUB y otras instrucciones de bifurcación.

- Area libre (Free area)
  Area no usada, cuyo tamaño puede obtenerse con la funcción FRE.

- Area de tablas de variables (Array variable area)
  Este área almacena datos asignados a variables tipo array. Este área contiene también los punteros que señalan datos de cadena almacenados en el área de cadenas, para tablas de variables de cadena. Este área se reserva cuando se ejecuta la instrucción DIM o cuando se usa una tabla de variables con 10 o menos subíndices.

- Area de variables (Variable area)
  Este área almacena datos asignados a variables. Cuando se trata de variables de cadena, este área contiene los punteros que señalan los datos de cadena almacenados en el área de cadenas.

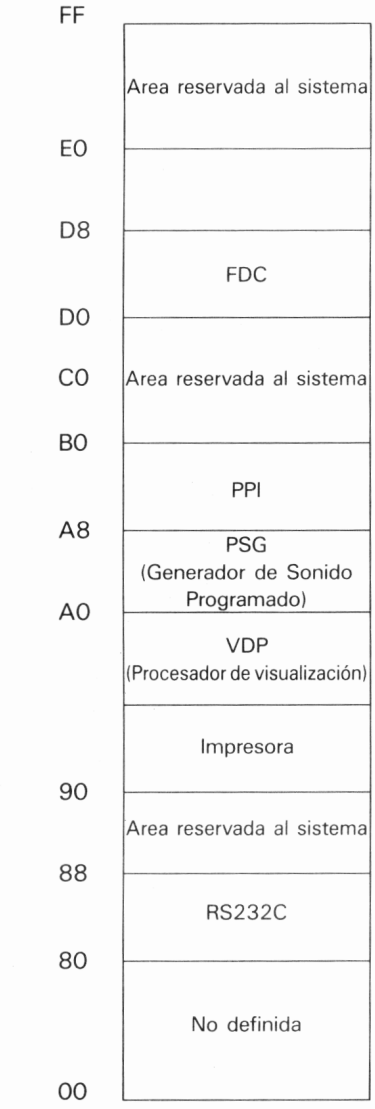
- Area de programa (Program area)
  Este área almacena los programas en BASIC.

3. ZOCALOS DE AMPLIACION

En los sistemas MSX, los 64 Kbytes iniciales pueden ampliarse añadiendo módulos de extensión. En el sistema MSX hay cuatro zócalos normalizados, en cada uno de los cuales puede acoplarse hasta cuatro módulos.

El espacio de memoria se divide en páginas de 16K bytes, y a cada página se asigna un zócalo. Los zócalos se seleccionan automáticamente al conectar el ordenador. Para seleccionar un zócalo especial se precisa un programa en lenguaje máquina. Vea más detalles en los manuales sobre software MSX.

4. MAPA DE ENTRADA/SALIDA



Dirección E/S	E/L	Contenido	Observaciones
AB	E	Selección modo	PPI
AA	E	Escritura datos a port C	
	L	Lectura datos del port C	
A9	E	Escritura datos a port B	
	L	Lectura datos del port B	
A8	E	Escritura datos a port A	
	L	Lectura datos a port A	
A2	L	Lectura datos	PSG
A1	E	Escritura datos	
A0	E	Selección registro	
99	E	Comando y dirección	VDP
	L	Lectura estado	
98	E	Escritura datos en VRAM	
	L	Lectura datos de VRAM	
91	E	Salida datos	
90	E	Salida estrobo (bit 0)	
	L	Entrada estado (bit 1)	

E: Escritura L: Lectura

Vea más detalles en los manuales de hardware y software MSX.

5. LISTA DE CODIGOS DE ERROR

Mensaje de error	N° de error	Descripción
Bad file name (Nombre de fichero ilegal)	56	(Nombre fichero erróneo) El nombre de fichero es defectuoso. • Especificación errónea en la instrucción OPEN.
Bad file number (Número de fichero ilegal)	52	(Número fichero erróneo) El número de fichero es defectuoso. • Número de fichero mayor que la especificación en MAXFILES. • El número de fichero no se abrió con OPEN.
Can't continue (No puede seguir)	17	(No puede continuar) No puede reanudarse la ejecución del programa. • Continuación después de interrupción causada por un error. • Continuación después de modificación del programa.
Device I/O error (Error en el mecanismo I/O)	19	(Error E/S dispositivo) Error de transferencia de datos en comunicación con un dispositivo E/S. • Error de lectura de fichero cassette. • Interrupción de la comunicación con un dispositivo E/S por haberse pulsado las teclas CTRL y STOP.
Direct statement in file (Sentencia directa en el fichero)	57	(Comando directo en fichero) En un fichero ASCII que se está cargando hay un comando directo (sin número de línea).
Division by zero (División por cero)	11	(Se intentó una división por cero) • El divisor es cero • El divisor es una variable no definida.
File already open (Fichero ya abierto)	54	(Fichero ya abierto) El fichero ya está abierto
File not found (Fichero no hallado)	53	(Fichero no encontrado)
File not open (Fichero no abierto)	59	(Fichero no abierto)
Illegal direct (Sentencia ilegal)	12	Se intentó ejecutar una instrucción que no puede usarse en modo directo.
Illegal función call (Llamada una función ilegal)	5	(Llamada ilegal a función) • El argumento en una instrucción o función excede de la gama especificada. • Una tabla de variables lleva subíndices negativos o números demasiado grandes.
Internal error (Error interno)	51	Ha ocurrido un error dentro de BASIC. • Este tipo de error no puede producirse normalmente. De ocurrir, desconectar temporalmente el ordenador.

Mensaje de error	N° de error	Descripción
Input past end (Fin de entrada)	55	(Entrada más allá del final) Se intentó leer datos de un fichero con una instrucción INPUT# después de que todos los datos habían sido leídos. • El número de variables en la instrucción INPUT# excede del número de datos. • Se intentó leer un fichero que no contenía datos. (Este error puede recuperarse usando la función EOF).
Missing operand (Operador perdido)	24	(Falta operando) Falta un operando que es necesario. • El número de operandos es erróneo. • Se han usado puntos (.) como separadores, en vez de comas (,).
NEXT without FOR (NEXT sin FOR)	1	El número de instrucción NEXT no concuerda con el de instrucciones FOR. • Un bucle FOR — NEXT contiene parte de otro bucle FOR — NEXT.
No RESUME (No "RESUME")	21	En una rutina de tratamiento de errores falta la instrucción RESUME. • La salida de la rutina se efectuó con la instrucción GOTO.
Out of DATA (Fuera de datos)	4	(Datos agotados) No quedan datos en la instrucción READ. • El número de expresiones de datos es insuficiente. • Número de línea erróneo en la instrucción RESTORE. • Delimitadores no válidos en la instrucción DATA.
Out of memory (Fuera de memoria)	7	(Memoria agotada) Insuficiente capacidad de memoria. • El programa es demasiado grande. • Demasiadas variables. • Tablas de variables demasiado largas. • Las tablas de variables no necesarias pueden borrarse con instrucciones ERASE.
Out of string space (Fuera de espacio de la cadena de caracteres.)	14	(Espacio cadenas agotado) El espacio para cadenas es insuficiente. • El espacio para cadenas seleccionado con la instrucción CLEAR es insuficiente.
Overflow (Sobrecapacidad)	6	(Desbordamiento) El valor numérico excede de la gama. • El resultado de la operación aritmética es demasiado grande o demasiado pequeño.
Re-dimensioned array (Redimensionar la matrlz)	10	Un array se dimensionó dos veces. • Con la instrucción DIM se intentó definir una tabla de variables ya definida. • Se usó un array con subíndices de 10 o menos sin dimensionarlo previamente, y luego se intentó dimensionarlo con DIM.
RESUME without error (Error sin "RESUME")	22	Se ejecutó una instrucción RESUME en una parte de programa que no es una rutina de tratamiento de errores.

Mensaje de error	N° de error	Descripción
RETURN without error (Error sin "RETURN")	3	Se ejecutó una instrucción RETURN antes de una instrucción GOSUB. • La ejecución pasó a una subrutina a causa de una instrucción GOTO. • Al final del programa principal se omitió la instrucción END, y se ejecutó la siguiente subrutina.
String formula too complex (Fórmula de cadena de carácteres)	16	(Fórmula de cadena demasiado compleja) Una cadena es demasiado compleja. • Las operaciones de la cadena escrita en una línea son demasiado complejas (demasiados paréntesis).
String too long (Cadena de caracteres demasiaso larga)	15	(Cadena demasiado larga) • Se intentó asignar más de 256 caracteres a una variable de cadena.
Subscript out of range (Subíndice fuera de rango)	9	El subíndice de una tabla de variables excede de la gama permitida. • El subíndice es demasiado grande. • Una tabla de variables no dimensionada lleva un subíndice superior a 10.
Syntax error (Error de sintaxis)	2	La sintaxis no concuerda con las reglas de MSX BASIC. • Entrada errónea a causa de un error de mecanografía. • Delimitador no válido (coma, punto, punto y coma, etc.) • Los paréntesis no concuerdan. • Nombre de variable que empieza con un carácter no alfabético.
Type mismatch (Tipos no emparejados)	13	(Discrepancia de tipo) Los tipos de variables discrepan. • Se intentó asignar una cadena a una variable numérica. • Un tipo de argumento en una función no concuerda.
Undefined line number (Número de línea indefinido)	8	(Número línea no definido) Designación errónea de número línea. • El número de línea especificado en una instrucción GOTO, GOSUB, RESTORE o RESUME no existe.
Undefined user function (Función no definida)	18	(Función del usuario no definida) Una función usuario no es definida. • Nombre de función erróneo en la instrucción DEF FN. • No se ejecutó la instrucción DEF FN.
Verify error	20	Error de verificación. • Se encontró una discrepancia entre el programa en memoria y el leído del cassette durante la ejecución del comando CLOAD? • Un programa grabado desde un sistema con diferente capacidad RAM fue verificado comparándolo con el programa en memoria (se producirá un mensaje de error aún cuando el contenido del programa sea normal).

6. INDICE DE LOS COMANDOS Y FUNCIONES BASIC

Función	Descripción	Comando	Pág.
Programación	Borre el programa	NEW	88
	Genera automáticamente números de línea	AUTO	45
	Renumera todas las líneas	RENUM	111
	Borra una línea	DELETE	60
	Lista el programa en la pantalla	LIST	81
	Lista el programa en la impresora	LLIST	81
	Ejecuta el programa	RUN	114
	Reanuda la ejecución del programa	CONT	54
	Inicia el rastreo	TRON	130
	Cancela el rastreo	TROFF	130
Registrador cassette	Graba el programa	CSAVE	55
		SAVE	114
	Carga el programa	CLOAD	51
		LOAD	82
	Comprueba el programa	CLOAD?	51
	Combina programas	MERGE	85
	Graba el programa en lenguaje máquina	BSAVE	47
	Carga el programa en lenguaje máquina	BLOAD	47
	Controla el motor	MOTOR ON	87
		MOTOR OFF	87
	Especifica la velocidad de transferencia	SCREEN	115
	Especifica el número de ficheros	MAXFILES	85
	Abre un fichero	OPEN	96
	Envía datos	PRINT#	106
		PRINT# USING	107
	Entra datos	INPUT#	73
		LINE INPUT#	80
		INPUT$	73
	Determina si se ha alcanzado el fin del fichero	EOF	63
	Cierra el fichero	CLOSE	52
	Determina la dirección del bloque de control de fichero	VARPTR	131
Teclado	Entra datos	INPUT	72
		LINE INPUT	80
		INKEY$	71
		INPUT$	73
	Controla el sonido de las teclas	SCREEN	115

Función	Descripción	Comando	Pág.
Teclado	Define una tecla de funciones	KEY	76
	Lista el contenido de las teclas de funciones	KEY LIST	76
	Exhibe el contenido de las teclas de funciones	KEY ON	76
	Cancela la exhibición de las teclas de función	KEY OFF	76
	Controla la interrupción generada por teclas de funciones	KEY (n) ON	77
		KEY (n) OFF	77
		KEY (n) STOP	77
		ON KEY GOSUB	92
	Controla la interrupción generada por las teclas CTRL y STOP	STOP ON	125
		STOP OFF	125
		STOP STOP	125
		ON STOP GOSUB	94
	Determina qué tecla de movim. cursor se pulsó	STICK	123
	Determina si la barra espaciadora se pulsó	STRIG	126
	Controla la interrupción generada por la barra espaciadora	STRIG (n) ON	127
		STRIG (n) OFF	127
		STRIG (n) STOP	127
		ON STRIG GOSUB	95
Control de pantalla	Selecciona el modo pantalla	SCREEN	115
	Especifica el color	COLOR	53
	Borra la pantalla	CLS	52
	Escribe datos en el registro VDP	VDP	132
	Determina el contenido del registro VDP	VDP	132
	Determina la dirección inicial de la tabla VRAM	BVASE	46
	Determina el contenido de la memoria VRAM	VPEEK	132
	Almacena datos en la memoria VRAM	VPOKE	132
Modo de pantalla texto	Visualiza datos	PRINT	102
	Visualiza datos formateados	PRINT USING	103
	Lista el programa en la pantalla	LIST	81
	Especifica la anchura de la línea de pantalla	WIDTH	133
Modo de pantalla texto	Genera espacios en blanco de tabulación	TAB	129
	Genera espacios en blanco	SPC	119
	Mueve el cursor	LOCATE	82
	Determina la posición vertical (línea) del cursor	CSRLIN	55
	Determina la posición horizontal (columna) del cursor	POS	101
Modo de pantalla gráfica	Traza circunferencias y elipses	CIRCLE	49
	Traza rectas y rectángulos	LINE	79
	Dibuja gráficos	DRAW	62
	Rellena con color	PAINT	98
	Dibuja puntos	PSET	108
	Cambia el color de puntos	PRESET	101
	Determina el código de color del punto especificado	POINT	100

Función	Descripción	Comando	Pág.
Modo de pantalla gráfica	Exhibe caracteres y números	MAXFILES	85
		OPEN	96
		PRINT#	106
		PRINT# USING	107
		CLOSE	52
Pantalla de formas	Define la pauta de la forma	SPRITE$	121
	Exhibe la forma	PUT SPRITE	109
	Controla la interrupción por formas	SPRITE ON	120
		SPRITE OFF	120
		SPRITE STOP	120
		ON SPRITE GOSUB	93
Impresora	Imprime datos	LPRINT	83
	Imprime datos formateados	LPRINT USING	84
	Lista programa	LLIST	81
	Genera espacios en blanco de tabulación	TAB	129
	Genera espacios en blanco	SPC	119
	Especifica si se usa o no la impresora MSX	SCREEN	115
	Determina la posición de la cabeza impresora	LPOS	83
Sonido	Toca música	PLAY	99
	Determina si se toca música	PLAY	100
	Escribe datos en el registro GSP	SOUND	118
	Hace sonar el altavoz	BEEP	46
Mando de juego, etc.	Determina la dirección del mando de juego	STICK	123
	Determina si se pulsa el botón accionador	STRIG	126
	Controla la interrupción por pulsación de botón	STRIG ON	127
		STRIG OFF	127
		STRIG STOP	127
		ON STRIG GOSUB	93
	Determina la condición de la tabla sensora	PAD	97
	Determina la condición de la tabla sensora	PDL	98
Port E/S	Envía datos	OUT	97
	Determina el valor del port de entrada	INP	71
	Espera a que se entre el valor especificado	WAIT	133
Ficheros	Especifica el número de ficheros	MAXFILES	85
	Abre el fichero	OPEN	96
	Envía datos	PRINT#	106
		PRINT# USING	107

Función	Descripción	Comando	Pág.
	Entrada de datos	INPUT#	73
		LINE INPUT#	80
		INPUT$	73
	Determina si se la llegado al fin del fichero	EOF	63
	Cierra el fichero	CLOSE	52
	Determina la dirección del bloque de control de fichero	VARPTR	131
Bifurcación	El programa pasa a la línea especificada	GOTO	69
	Determina la condición	IF	69
	El programa pasa a diferentes líneas según las condiciones	ON GOTO	90
Subrutina	Ejecuta una subrutina	GOSUB	68
	Ejecuta una subrutina dada según las condiciones	ON GOSUB	90
	El programa vuelve al programa principal	RETURN	112
Repetición	Repite la ejecución el número indicado de veces	FOR	66
		NEXT	88
Error	Genera intencionadamente un error	ERROR	64
	Define la línea inicial de una rutina de recuperación de errores	ON ERROR GOTO	89
	Restituye la ejecución al programa principal, desde la rutina de tratamiento de errores	RESUME	112
	Determina el número de línea en que se generó el error	ERL	64
	Determina el número de error	ERR	64
Paro	Detiene la ejecución del programa	STOP	124
Fin	Termina la ejecución del programa	END	63
Comentarios	Inserta comentarios en el programa	REM	110
Manejo de cadenas de caracteres	Sustituye parte de una cadena de caracteres	MID$	87
	Determina parte de una cadena de caracteres	LEFT$	78
		MID$	86
		RIGHT$	113
	Determina la longitud especificada de una cadena de espacios en blanco	SPACE$	119
	Determina la cadena de caracteres especificada	STRING$	128
	Determina la posición de una cadena de caracteres dentro de otra cadena de caracteres	INSTR	74
	Determina la longitud de una cadena de caracteres	LEN	78

Función	Descripción	Comando	Pág.
Conversión de tipo	Convierte un valor numérico a un número real de doble precisión	CDBL	48
	Convierte un valor numérico a entero	CINT	48
	Convierte un valor numérico a simple precisión	CSNG	55
	Determina el código del carácter	ASC	44
	Convierte una cadena de caracteres a su valor numérico	VAL	131
	Convierte un código de carácter a su correspondiente carácter	CHR$	48
	Convierte un decimal a cadena binaria	BIN$	46
	Convierte un decimal a cadena octal	OCT$	89
	Convierte un decimal a cadena hexadecimal	HEX$	69
	Convierte un valor numérico a cadena de caracteres	STR$	128
Operaciones numéricas	Determina el arcotangente	ATN	44
	Determina el coseno	COS	54
	Determina el seno	SIN	117
	Determina la tangente	TAN	129
	Determina la función exponencial	EXP	65
	Determina el logaritmo	LOG	83
	Determina la raíz cuadrada	SQR	123
	Determina el valor absoluto	ABS	44
	Determina la parte entera	FIX	65
	Determina el entero máximo por debajo del número especificado	INT	74
	Determina el signo	SGN	117
Variables	Asigna un valor a una variable	LET	78
	Almacena constantes para ser leídas por READ	DATA	110
	Lee las constantes especificadas en DATA	READ	56
	Especifica la instrucción DATA a ser leída por READ	RESTORE	111
	Define dimensiones de una tabla	DIM	61
	Borra una tabla de variables (array)	ERASE	62
	Inicializa todas las variables	CLEAR	50
	Intercambia los valores de dos variables	SWAP	128
	Determina la dirección de memoria de una variable	VARPTR	131
	Define una variable entera	DEFSNG	59
	Define una variable de simple precisión	DEFINT	59
	Define una variable de doble precisión	DEFDBL	58
	Define una variable de cadena	DEFSTR	59
Números aleatorios	Determina el valor de un número aleatorio	RND	113
Funciones del usuario	Define una función del usuario	DEF FN	57

Función	Descripción	Comando	Pág.
Memoria	Determina la cantidad de memoria no usada	FRE	67
	Determina el contenido de la dirección especificada	PEEK	99
	Escribe datos en la dirección especificada	POKE	101
	Define el tamaño del área de memoria	CLEAR	50
Lenguaje máquina	Define la dirección inicial de una subrutina en lenguaje máquina	DEFUSR	57
	Ejecuta una subrutina en lenguaje máquina	USR	130
Cronómetro intervalos (reloj interno)	Ajusta el valor del reloj interno	TIME	129
	Determina el valor del reloj interno	TIME	129
	Controla las interrupciones del temporizador de intervalos	INTERVAL ON	75
		INTERVAL OFF	75
		INTERVAL STOP	75
		ON INTERVAL GOSUB	91
Instrucción de ampliación	Llama la instrucción contenida en un cartucho de ampliación	CALL	8

NOTES

NOTES

TOSHIBA CORPORATION
TOKYO JAPAN
PRINTED IN JAPAN
E0104-2