

 **ATARI®**

PERSONAL SOFTWARE  
**FirST BASIC**



De **HiSoft**

**Software y manual de FirST BASIC © Copyright de HiSoft, 1989**

Reservados todos los derechos en todo el mundo. Ninguna parte de este documento puede reproducirse ni transmitirse de ninguna forma y por ningún medio, incluyendo fotocopia y grabación, sin el permiso por escrito del titular del copyright. Dicho permiso debe obtenerse también para almacenar cualquier parte de esta publicación en un sistema de recuperación de cualquier tipo.

Se considera incumplimiento del copyright de **FirST BASIC** y su documentación asociada copiar, por cualquier medio, cualquier parte de **FirST BASIC** por cualquier razón que no sea hacer una copia de seguridad del código objeto según se describe en esta guía.

**FirST BASIC**, **Power BASIC** y **HiSoft BASIC** son marcas comerciales de HiSoft. Todas las demás marcas comerciales utilizadas se indican de forma apropiada.

Gracias a Stephan Somogyi, David Nutkins, Andy Pennell, Dave Howorth, Simon Goodwin, Sue, Julie, Natalie y ...la chica del perro... por su incalculable ayuda para la producción de **FirST BASIC**, **POWER BASIC** y **HiSoft BASIC**.

Diseño y composición de páginas de Bookmark Publishing Ltd., Greenfield, UK.

# Contenido

---

<b>Introducción a FirST BASIC</b>	<b>1</b>
Requisitos del Sistema	1
Haga siempre una copia de seguridad	2
<b>Ambito de esta Guía</b>	<b>2</b>
<b>Puesta en Marcha de FirST BASIC</b>	<b>2</b>
Introducción	2
Sus Primeros Programas	3
<b>Contenido del Disco</b>	<b>5</b>
<b>El Editor de FirST BASIC</b>	<b>5</b>
Introducción de Texto y Movimiento del Cursor	6
Cómo salir de FirST BASIC	8
Supresión de Texto	8
Cómo Salvar Texto	9
Carga de Texto	10
Búsqueda y Sustitución de Texto	10
Comandos de Bloque	11
Comandos Varios	11
Preferencias	12
Ejecución de Programas	13
El Modo Immediate	13
La Ventana del Editor GEM	14



<b>Referencia Rápida</b>	<b>14</b>
<b>Sentencias y Funciones</b>	<b>14</b>
<b>La Biblioteca Suministrada</b>	<b>27</b>
<b>Tipos de Variables</b>	<b>30</b>
<b>Juego de Caracteres</b>	<b>30</b>
<b>Operadores</b>	<b>31</b>
<b>Your FirST BASIC</b>	<b>32</b>
<b>Power BASIC y HiSoft BASIC</b>	<b>32</b>

## Introducción a FirST BASIC

Bienvenido a FirST BASIC de HiSoft para la gama de ordenadores Atari ST, una versión potente y moderna del lenguaje BASIC que constituye un entorno totalmente integrado e interactivo para la producción de sus programas.

FirST BASIC es uno de los compiladores de la gama HiSoft BASIC para el ST. Todos ellos son el resultado de muchos años de esfuerzo de diseño y programación durante los cuales nuestro objetivo ha sido producir un sistema BASIC con las siguientes características:

- dialecto moderno y estructurado del lenguaje BASIC
- ciclo interactivo de edición/compilación/ejecución, como un intérprete
- compila QuickBASIC 3™ de Microsoft haciendo unos pequeños cambios
- con pequeñas modificaciones, compila programas escritos en casi cualquier versión y variante del lenguaje BASIC
- tiempo de compilación rápido y tiempo de ejecución extremadamente rápido
- subprogramas y funciones recursivos
- numerosas sentencias estructuradas, como WHILE...END WHILE, DO...LOOP UNTIL, SELECT...CASE, etc.
- soporte completo del Atari ST y el GEM mediante el uso de bibliotecas
- informes y corrección de errores
- ningún límite en cuanto al tamaño de las variables

Gracias a la potencia y flexibilidad del Atari ST y su sistema operativo, hemos podido implantar todos estos objetivos de diseño y esperamos que disfrute de la potencia, flexibilidad y facilidad de transporte de FirST BASIC.

### Requisitos del Sistema

Los requisitos mínimos para ejecutar FirST BASIC en un Atari ST son los siguientes:

- 512 Kbytes de RAM
- TOS en la ROM
- una unidad de disco
- un ratón



## Haga siempre una copia de seguridad

---

Antes de utilizar FirST BASIC debe hacer una copia de seguridad del disco y guardar el original en un lugar seguro. El disco original no está protegido contra copia para que usted pueda hacer su copia de seguridad sin problemas. Debe copiar el disco utilizando Desktop o cualquier utilidad de copia de seguridad; antes de hacer una copia de seguridad, proteja siempre contra escritura el original para evitar que se borre accidentalmente.

## Ambito de esta Guía

---

Esta guía de FirST BASIC es ante todo una guía de consulta dirigida a los programadores con experiencia en BASIC para que se puedan empezar a utilizar rápidamente el paquete.

Si no tiene experiencia en programación en BASIC, o si desea sacar el máximo provecho del ST y del BASIC, le recomendamos que lea el libro **Your FirST BASIC**. Dicho libro contiene información completa sobre todos los aspectos del paquete, así como un cursillo que no sólo le enseña a utilizar un moderno lenguaje estructurado, sino que también desvela numerosos secretos del Atari ST. Al final de esta guía se incluyen detalles para obtener una copia de este libro de valor incalculable.

## Puesta en Marcha de FirST BASIC

---

### Introducción

---

En esta sección se explican las fases de escritura, compilación y ejecución de programas en FirST BASIC, centrándose en el uso del editor y el compilador, de forma que pueda aprender rápidamente a utilizar este entorno de programación rápido e interactivo.

En este capítulo no se pretende enseñarle a programar en BASIC, sino iniciarle en el uso del mismo; recuerde que puede comprar el libro **Your FirST BASIC** en tiendas especializadas o pedirlo directamente a HiSoft.

Ahora, encienda su Atari ST, inserte la copia de seguridad del disco de FirST BASIC y...

## Sus Primeros Programas

---

Desde la copia de seguridad, pulse dos veces el botón del ratón sobre 1STBASIC.PRG y espere a que aparezca la ventana del editor: una ventana normal del GEM con el control de aumento de tamaño (en la parte inferior derecha), el control de cierre (en la parte superior izquierda) y barras de desplazamiento horizontal y vertical. Ahora, escriba este programa de 8 líneas pulsando la tecla Return al final de cada línea:

```
t=TIMER
DO WHILE TIMER<t+20
  col=RND*17+1 : row=RND*16+2
  IF MOUSE(2) THEN
    CLS : BEEP : LOCATE row,col
    PRINT "HiSoft FirST BASIC"
  END IF
LOOP
```

Las palabras clave como TIMER, DO, LOOP, etc. pueden introducirse en mayúsculas, en minúsculas o en una combinación de ambas. Es una práctica común en programación escribirlas en mayúsculas, pero FirST BASIC acepta cualquier modelo.

Este es un pequeño programa muy sencillo que selecciona aleatoriamente una posición de la pantalla idónea para todas las resoluciones (col=RND\*17+1 : row=RND\*16+2), espera a que pulse el botón izquierdo del ratón (IF MOUSE(2)) y, luego, borra la ventana, emite un aviso acústico e imprime un mensaje en la posición aleatoria (CLS, etc.). Transcurridos 20 segundos, pasa el tiempo prefijado y termina (DO WHILE TIMER<+20 LOOP).

¿Desea ejecutarlo? Pulse Alternate-X, es decir, mantenga pulsada la tecla Alternate y pulse X (minúscula o mayúscula). Hay otra forma de ejecutar un programa: seleccione Run en el menú Program. Independientemente del método que siga para ejecutar el programa, verá una ventana con el título FirST BASIC is working.

Si hay algún error en el programa, el compilador mostrará un número de error y un mensaje de error, y volverá a llevarle a la pantalla del editor, situándole en la línea donde se ha producido el error, y mostrando el mensaje de error en la esquina superior derecha de la ventana, de forma que pueda corregir el error y ejecutar de nuevo el programa.

Una vez compilado satisfactoriamente el programa, éste se ejecutará ... pulse un botón del ratón para ver los resultados de la ejecución. Después de 20 segundos se le pedirá que pulse una tecla y volverá al editor.



¿Ha visto lo fácil que es escribir un programa GEM en una ventana? Esto se debe a que **FirST BASIC** abre automáticamente una ventana para el programa y la cierra cuando el programa termina.

Una última observación sobre este sencillo programa. ¿Se ha fijado en la sentencia **D0**? Es una estructura muy flexible que le permite realizar numerosos tipos de bucles con una única sentencia.

Veamos un programa más interesante. Seleccione **New** en el menú **File**. Aparecerá un recuadro con el mensaje **OK to lose changes?** Pulse el botón **OK** y escriba lo siguiente pulsando la tecla **Return** al final de cada línea:

```
medhigh=4-PEEKW(SYSTAB)      'resolución de la pantalla
lowmed=1-PEEKW(SYSTAB)
MOUSE -1                      'desactivar el ratón
Patterns:
  CLS
  RANDOMIZE TIMER
  IF medhigh THEN cx=300 ELSE cx=150
  IF lowmed THEN cy=95 ELSE cy=190
  pi=3.14159265#
  xs=cx/4+cx*RND/2 : ys=cy/4+cy*RND/2
  FOR k=0 TO 500 STEP 4*pi*RND+1
    tx=x : ty=y
    x=cx+xs*COS(k) : y=cy+ys*SIN(k)
    IF k=0 THEN PSET(x,y) ELSE LINEF tx,ty,x,y
  NEXT k
GOTO Pattern
```

Pulse **Alternate-X** para ejecutar el programa. Si lo ha escrito tal y como se muestra, **FirST BASIC** se detendrá y mostrará un error: **Pattern is not a label...** en la línea 16. ¿Ha detectado el error? Correcto, **Pattern** debe ser **Patterns** (en plural), ya que esta era la etiqueta definida en la línea 4 del programa. Entonces, cambie **Pattern** por **Patterns** escribiendo una **S** al final de la línea. Pulse **Alternate-X** para ejecutar de nuevo el programa; esta vez debe ejecutarse sin errores. ¿Verdad que es rápido?

El programa se repite indefinidamente y usted puede llegar a aburrirse. Pulse **Shift-Alternate-Help**; esta es una forma de interrumpir los programas **FirST BASIC** cuando no se está escribiendo texto en la pantalla. Aparecerá el mensaje:

```
BREAK pressed at line x
```

Al pulsar cualquier tecla volverá el editor.

Cuando se esté enviando texto a la pantalla, puede interrumpir un programa pulsando **Ctrl-C**. Esto tiene el mismo efecto que pulsar **Shift-Alternate-Help**. Este es un ejemplo de error en tiempo de ejecución y, como puede ver, se informa del mismo de manera precisa y es fácil de corregir.

## Contenido del Disco

**FirST BASIC** se suministra en un disco, en cuya copia de seguridad (ya la ha hecho, ¿no?) encontrará los siguientes ficheros como mínimo:

1STBASIC.PRG	Editor, compilador y biblioteca integrados
DEMO.BAS	Programa de demostración
JACK.SCR	Shot de pantalla de alta resolución para DEMO
JACKMED.SCR	Igual que el anterior, pero para resolución media
DUMP.BAS	Programa para el volcado de ficheros hexadecimales y ASCII
HANOI.BAS	Programa de las Torres de Hanoi
SIEVE.BAS	Banco de pruebas Sieve de BYTE

En sus discos de trabajo sólo es necesario que esté presente **1STBASIC.PRG**; los demás ficheros son opcionales.

## El Editor de FirST BASIC

El editor de **FirST BASIC** es un editor de pantalla que permite introducir y editar texto, así como salvar y cargar texto del disco. También permite imprimir todo el texto o parte del mismo, buscar y sustituir modelos de texto y utilizar cualquiera de los accesorios de la mesa de trabajo del **ST**. Este editor está basado en el **GEM**, lo que significa que utiliza todas las sencillas funciones de los programas **GEM** con las que usted ya está familiarizado, como ventanas, menús y ratones. Sin embargo, si es usted un entusiasta de la "informática dura", le agradecerá saber que puede hacer prácticamente todo lo que desee desde el teclado sin tener que usar ratón.

El editor está "basado en RAM", lo que significa que el fichero que está editando permanece todo el tiempo en la memoria, por lo que no tendrá que esperar a que el disco cargue distintas secciones del fichero a medida que lo edita. Puesto que la gama de ordenadores **ST** tiene mucha memoria, las limitaciones en cuanto a tamaño existentes con frecuencia en otros editores anteriores no existen en **FirST BASIC**; si dispone de suficiente memoria puede editar ficheros de más de 300 K (debe asegurarse de que el disco tiene capacidad suficiente para salvarlos). Puesto que todas las operaciones de edición, incluyendo la búsqueda, están basadas en la **RAM**, éstas se realizan rápidamente.



Si alguna vez no sabe cómo seguir, pulse la tecla **Help** para acceder a una pantalla que le mostrará las teclas necesarias para realizar funciones que no aparecen en los menús.

## Introducción de Texto y Movimiento del Cursor

Después de cargar **FirST BASIC**, aparecerá una ventana en blanco con una línea de estado en la parte superior y un recuadro negro intermitente, que es el cursor, en la esquina superior izquierda.

La línea de estado contiene información sobre la posición del cursor en forma de desplazamientos de Línea y Columna, así como el número de bytes de memoria que quedan disponibles para almacenar el texto. Inicialmente, se muestra 9980, ya que el tamaño del texto por defecto es de 10000 bytes. Si lo desea, puede cambiar este valor por defecto, así como otras opciones, seleccionando **Preferences** (que se describe más adelante). El resto de la línea de estado se utiliza para mostrar mensajes de error, que normalmente van acompañados de una señal acústica para avisarle del error. Todos los mensajes mostrados desaparecen de la pantalla al pulsar una tecla.

Para introducir texto, escríbalo desde el teclado. Cuando pulse una tecla, en la pantalla aparecerá el carácter correspondiente y el cursor avanzará por la línea. Si es un buen mecanógrafo, quizás escriba más rápido de lo que el editor tarda en mostrar de nuevo la línea. En tal caso, no se preocupe; el programa no pierde las pulsaciones realizadas y las muestra cuando se hace una pausa. Al final de la línea debe pulsar la tecla **Return** (o la tecla **Enter** del teclado numérico) para comenzar la línea siguiente. Puede corregir los errores pulsando la tecla **Backspace**, que borra el carácter situado a la izquierda del cursor, o la tecla **Delete**, que borra el carácter donde está situado el cursor.

### Teclas del Cursor

Para mover el cursor por el texto con el fin de corregir errores o introducir texto nuevo se utilizan las teclas del cursor, que tienen grabados los símbolos  $\leftarrow$ ,  $\rightarrow$ ,  $\uparrow$  y  $\downarrow$ . Si lleva el cursor más allá del extremo derecho de la línea, no se añadirá nada al texto; pero si intenta escribir texto en dicha posición, el editor añadirá el texto al final real de la línea. Si escribe líneas largas, se desplazará lateralmente la visualización de la ventana.

Si sube con el cursor hasta el principio de una ventana, la visualización se desplazará hacia abajo si hay una línea anterior, o se mostrará el mensaje **Top of file** en la línea de estado. Igualmente, si baja con el cursor hasta el final de la ventana, la visualización se desplazará hacia arriba si hay una línea a continuación, o se mostrará el mensaje **End of file**.

Puede mover el cursor de carácter en carácter pulsando el botón del ratón sobre los recuadros que contienen flechas en los extremos de las barras de desplazamiento horizontal y vertical.

Si está acostumbrado a utilizar **WordStar**, las teclas **Ctrl-S**, **Ctrl-D**, **Ctrl-E** y **Ctrl-X** funcionan de la misma manera que las teclas del cursor.

Pulse **Ctrl- $\leftarrow$**  para ir inmediatamente al principio de la línea actual, y **Ctrl- $\rightarrow$**  para ir al final de la línea actual.

Para mover el cursor una palabra hacia la izquierda, pulse **Shift- $\leftarrow$** , y **Shift- $\rightarrow$**  para llevar el cursor una palabra hacia la derecha. Se entiende por palabra cualquier elemento rodeado de un espacio, un tabulador o un principio o final de línea. Las teclas **Ctrl-A** y **Ctrl-F** también mueven el cursor una palabra hacia la izquierda y hacia la derecha, respectivamente.

Para mover el cursor una página hacia arriba, puede pulsar el botón del ratón sobre la parte gris superior de la barra de desplazamiento vertical, o bien **Ctrl-R** o **Shift- $\uparrow$** . Para mover el cursor una página hacia abajo, puede pulsar el botón del ratón sobre la parte gris inferior de la barra de desplazamiento, o bien **Ctrl-C** o **Shift- $\downarrow$** .

Si desea llevar el cursor a una posición determinada de la pantalla, mueva el puntero del ratón a dicho lugar y pulse el botón del mismo. (No hay ningún equivalente en **WordStar** para esta función.)

### Tecla Tab

La tecla **Tab** inserta un carácter especial (código 9 ASCII) en la memoria intermedia, que en la pantalla se presenta como una serie de espacios, aunque tiene otro significado. Al pulsar **Tab**, el cursor se alinea en la siguiente columna "que sea múltiplo de 4"; de esta forma, si pulsa esta tecla al principio de una línea (columna 1), el cursor pasará a la siguiente columna que sea múltiplo de 4, + 1, es decir, a la columna 5. Los tabuladores son muy útiles para alinear elementos verticalmente, y en **FirST BASIC** se utilizan principalmente para sangrar líneas de programa estructuradas.

### Tecla Backspace

La tecla **Backspace** borra el carácter situado a la izquierda del cursor. Si retrocede al principio de una línea, se borrará el carácter "invisible" de retorno de carro y la línea se unirá con el final de la línea anterior. Si se pulsa esta tecla cuando el cursor está situado más allá del final de la línea, se borrará el último carácter de la misma, a menos que la línea esté en blanco, en cuyo caso el cursor volverá a situarse en la parte izquierda de la pantalla.

### Tecla Delete

La tecla **Delete** borra el carácter situado debajo del cursor y no tiene ningún efecto si éste está situado más allá de la línea actual.



## Ir a una línea determinada

Para llevar el cursor a una línea de texto determinada, pulse el botón del ratón sobre la opción **Goto line ...** del menú **Options**, o pulse **Alternate-G**. Aparecerá un recuadro de diálogo donde puede introducir el número de línea deseado. Pulse **Return** o el botón del ratón sobre **OK** para ir a la línea, o pulse el ratón sobre **Cancel** para interrumpir la operación. Después de pulsar el ratón sobre **OK**, el cursor irá a la línea especificada, volviendo a mostrarla si es necesario, o mostrando el error **End of file** si dicha línea no existe.

Otra forma rápida de moverse por el fichero es desplazarse por la barra de desplazamiento vertical, que funciona como en el GEM.

## Ir al principio del fichero

Para ir al principio del texto, pulse el botón del ratón sobre la opción **Goto Top** del menú **Options**, o pulse **Alternate-T**. Se volverá a mostrar la pantalla desde la línea 1.

## Ir al final del fichero

Para llevar el curso al principio de la última línea de texto, pulse el botón del ratón sobre **Goto Bottom**, o pulse **Alternate-B**.

## Cómo Salir de FirST BASIC

---

Para salir de **FirST BASIC**, pulse el botón del ratón sobre **Quit** (en el menú **File**), o pulse **Alternate-Q**. Si se han hecho cambios en el texto pero no se han salvado en el disco, aparecerá un recuadro de aviso pidiendo confirmación para salir. Si pulsa el botón del ratón sobre **Cancel** volverá al editor, mientras que si lo pulsa sobre **OK** se anularán los cambios realizados y volverá a la Mesa de trabajo.

## Supresión de Texto

---

### Suprimir línea

Es posible suprimir la línea actual de texto pulsando **Ctrl-Y**.

### Suprimir hasta el final de la línea

Es posible suprimir el texto existente desde la posición del cursor hasta el final de la línea actual pulsando **Ctrl-Q**. (Esto equivale a pulsar **Ctrl-Q Y** en **WordStar**.)

## Recuperar línea suprimida

Cuando se suprime una línea utilizando uno de los comandos anteriores, ésta se almacena en una memoria intermedia interna y puede volver a insertarse en el texto pulsando **Ctrl-U** o la tecla **Undo**. Esta operación puede hacerse las veces que sea necesario, y resulta especialmente útil para repetir líneas similares o intercambiar líneas situadas en distintas posiciones.

## Suprimir todo el texto

Para borrar el texto actual, pulse el botón del ratón sobre **New** en el menú **File**. Si ha hecho algún cambio en el texto pero no lo ha salvado en el disco, aparecerá un recuadro de aviso y tendrá que dar su confirmación. Si pulsa el botón del ratón sobre **OK** se suprimirá el texto; si pulsa sobre **Cancel** se interrumpirá la operación.

## Cómo Salvar Texto

---

Para salvar el texto que está editando en este momento, pulse el botón del ratón sobre **Save As** del menú **File**, o pulse **Alternate-S**. Aparecerá el selector de ficheros (**File Selector**) estándar del GEM, que le permite seleccionar un disco y un nombre de fichero apropiados. Si pulsa el botón del ratón sobre **OK** o pulsa **Return**, el fichero se salvará en el disco. Si se produce algún error, aparecerá un recuadro mostrando un número de error **TOS**.

Si pulsa el botón del ratón sobre **Cancel**, el texto no se salvará. Normalmente, si existe un fichero con ese nombre, éste se borrará y se sustituirá por la nueva versión, pero si se ha seleccionado **Backups** en las opciones **Preferences**, el fichero existente adoptará la extensión **.BAK** (suprimiéndose cualquier fichero **.BAK** existente) antes de salvar la nueva versión.

### Save

Si ya ha seleccionado alguna vez **Save As** (o **Load**), **FirST BASIC** recordará el nombre del fichero y lo mostrará en la barra de título de la ventana. Si desea salvarlo sin utilizar el selector de ficheros, pulse el botón del ratón sobre **Save** (del menú **File**) y utilice el nombre anterior y sálvelo según se ha explicado. Si selecciona **Save** sin haber especificado previamente un nombre de fichero, aparecerá el selector de ficheros (como ocurre con **Save As**).



## Carga de Texto

---

Para cargar un nuevo fichero de texto, pulse el botón del ratón sobre Load (del menú File) o pulse Alternate-L. Si ha hecho algún cambio en el texto pero no lo ha salvado, tendrá que dar su confirmación. Aparecerá el selector de ficheros del GEM, que le permite especificar el disco y el nombre de fichero. Suponiendo que no seleccione Cancel, el editor intentará cargar el fichero. Si éste cabe, se cargará en la memoria y se volverá a dibujar la ventana. Si no cabe, aparecerá un recuadro de aviso y deberá utilizar Preferences para aumentar el tamaño de la memoria intermedia de edición y, luego, intentar cargar de nuevo el fichero.

## Búsqueda y Sustitución de Texto

---

Para encontrar una sección de texto determinada, pulse el botón del ratón sobre Find (del menú Search) o pulse Alternate-F. Aparecerá un recuadro de diálogo donde puede introducir las cadenas de búsqueda (Find) y de sustitución (Replace). Si pulsa el ratón sobre Cancel, no se tomará ninguna acción; si lo pulsa sobre Next (o pulsa Return), se iniciará la búsqueda hacia adelante, mientras que si lo pulsa sobre Previous la búsqueda comenzará hacia atrás. Si no desea sustituir el texto buscado, no indique nada en la cadena Replace. Si la búsqueda ha sido satisfactoria, se volverá a dibujar la pantalla con el cursor situado al principio de la cadena. Si no se ha encontrado la cadena, aparecerá el mensaje Not found en el área de estado y el cursor no se moverá. Por defecto, en la búsqueda no se hace distinción entre letras mayúsculas y minúsculas; si pulsa el botón del ratón sobre UPPER & lower case Different, sí se tendrán en cuenta las mayúsculas y minúsculas en la búsqueda.

Para encontrar la siguiente aparición de la cadena, pulse el botón del ratón sobre Find Next del menú Search, o pulse Alternate-N. La búsqueda comenzará en la posición siguiente a la del cursor.

Para buscar la aparición anterior de la cadena, pulse el botón del ratón sobre Find Previous del menú Search, o pulse Alternate-P. La búsqueda comenzará en la posición anterior a la del cursor.

Si se ha encontrado el texto buscado, puede sustituirse por la cadena indicada en Replace pulsando el botón del ratón sobre Replace del menú Search, o pulsando Alternate-R. Si lo sustituye, el editor buscará la siguiente aparición.

Si desea sustituir todas las apariciones de la cadena indicada en Find por la especificada en Replace desde la posición del cursor en adelante, pulse el botón del ratón sobre Replace All del menú Search. Durante la sustitución global puede utilizarse la tecla Esc para interrumpir la operación, en cuyo caso se mostrará en el área de estado el número de sustituciones realizadas.

## Comandos de Bloque

---

Un Bloque es una sección marcada de texto que se puede copiar en otra sección, suprimir, imprimir o salvar en disco. Las teclas de función se utilizan para controlar los bloques.

### Cómo marcar un bloque

El principio del bloque se marca llevando el cursor a la posición deseada y pulsando la tecla F1. El final del bloque se marca llevando el cursor a dicha posición y pulsando F2. No es necesario marcar el principio y el final de un bloque en un orden específico, aunque es mejor que marque primero el principio del bloque.

### Cómo salvar un bloque

Una vez marcado un bloque, éste se puede salvar pulsando la tecla F3. Suponiendo que se haya marcado un bloque válido, aparecerá el selector de ficheros del GEM donde puede seleccionar un disco y un nombre de fichero apropiados. Si salva el bloque con un nombre que ya existe, se sustituirá la versión anterior. Con este comando no se hacen copias de seguridad.

### Copia de un bloque

Un bloque marcado puede copiarse en otro lugar del texto, si la memoria lo permite, llevando el cursor al lugar donde desea copiar el bloque y pulsando la tecla F4. Si intenta copiar un bloque sobre sí mismo, aparecerá el mensaje Invalid block! y se interrumpirá la operación.

### Supresión de un bloque

Es posible suprimir del texto un bloque marcado pulsando Shift-F5. Es necesario pulsar la tecla Shift para evitar el borrado accidental al pulsar F5.

Las marcas de bloque permanecen durante todos los comandos de edición, trasladándose si es necesario, y sólo se restauran con los comandos New, Delete block y Load. Si edita una línea que contiene marcas de bloque, la nueva posición de las marcas no estará definida.

## Comandos Varios

---

### Sobre FirST BASIC

Si pulsa el botón del ratón sobre FirST BASIC... en el menú Desk, aparecerá un recuadro de diálogo con detalles sobre FirST BASIC, incluyendo el número de versión. Al pulsar Return o al pulsar el botón del ratón sobre OK volverá al editor.



## Pantalla de Ayuda

Es posible ver las teclas equivalentes a los comandos que no figuran en los menús pulsando la tecla **Help** o **Alternate-H**. Aparecerá un recuadro que muestra las teclas de WordStar y las teclas de función, así como la memoria disponible para el sistema.

## Preferences

---

Al seleccionar Preferences en el menú **Options** aparece un recuadro de diálogo que permite cambiar varios ajustes del editor:

### Tabs (Tabuladores)

Por defecto, el ajuste de tabulación es 4, pero puede cambiarse a cualquier valor comprendido entre 2 y 16.

### Text Buffer Size (Tamaño de la Memoria Intermedia de Texto)

Por defecto, el tamaño de la memoria intermedia de texto es de 10000 bytes, pero puede cambiarse a un valor comprendido entre 4000 y 999000 bytes. Esto determina el mayor tamaño que puede tener un fichero para que se pueda cargar y editar. Debe tener cuidado de dejar suficiente espacio en la memoria para las compilaciones; pulsando la tecla **Help** podrá ver la memoria libre del sistema, que para las compilaciones debe ser siempre 100 Kbytes como mínimo. Si cambia el tamaño del espacio de trabajo del editor, se perderá el texto que esté editando, por lo que se requiere una confirmación si dicho texto no se ha salvado en el disco.

### Maximum Size (Tamaño Máximo)

Su valor por defecto es 30 K y es el tamaño de una memoria intermedia que crea **FirST BASIC** para compilar su programa. Sólo debe aumentar este tamaño si aparece el error **Code generation failed** al intentar ejecutar el programa.

### Numeric Pad (Teclado Numérico)

La opción **Numeric Pad** permite utilizar el teclado numérico de un IBM-PC, pudiendo utilizar las funciones de cursor del mismo. El valor por defecto es **Cursor**.

### Backups (Copias de Seguridad)

Por defecto, el editor no hace copia de seguridad de los programas al salvarlos. Pero puede hacerse copia de seguridad pulsando el ratón sobre el botón **Backups**.

## Auto Indenting (Sangrado Automático)

Puede utilizarse para sangrar automáticamente líneas respecto del margen izquierdo; el editor admite un modo de sangrado automático. Cuando está activado, al pulsar **Return** se añade un sangrado al principio de cada línea nueva que se crea. El contenido del sangrado de la nueva línea se toma del espacio en blanco (tabuladores y/o espacios) que hay al principio de la línea anterior.

## Cómo salvar los valores de Preferences

Si pulsa el botón del ratón sobre **Cancel**, se ignorarán los cambios realizados. Si lo pulsa sobre el botón **OK**, los cambios realizados seguirán vigentes hasta que salga del editor. Si desea que la configuración definida sea permanente, pulse el ratón sobre el botón **Save**, lo que creará el fichero **1STBASIC.INF** en el disco. La próxima vez que ejecute **FirST BASIC**, se leerá la configuración desde dicho fichero.

## Ejecución de Programas

---

Si pulsa el botón del ratón sobre **Run** en el menú **Program**, o pulsa **Alternate-X**, ejecutará el programa de la ventana principal. Cuando termine el programa, normalmente esperará a que pulse una tecla para llevarle de nuevo al editor. Si se produce algún error, no se podrá ejecutar el programa.

## El Modo Immediate

---

**FirST BASIC** dispone de un modo inmediato especial que permite probar pequeñas secciones de un programa sin que se vea afectado el programa principal. Para ello, pulse **Alternate-0** (el número 0, no la letra O mayúscula) o pulse el botón del ratón sobre **Immediate window** en el menú **Program**. De esta forma, el editor pasará al modo **Immediate** (inmediato), que funciona de manera similar al modo normal (conocido como **Program Window**, Ventana de Programa). Puede escribir algunas líneas de programa y ejecutarlas de la forma habitual pulsando **Alternate-X**. También puede borrar el contenido de la ventana **Immediate** pulsando **Alternate-C** o pulsando el botón del ratón sobre **Clear** en el menú **File**. En modo inmediato, la mayoría de las opciones de los menús están desactivadas, así como los comandos de bloque. Para volver a la ventana de programa (**Program window**), pulse **Alternate-0** o pulse el ratón sobre **Immediate Window** en el menú **Program**, o bien pulse el ratón sobre el recuadro **Close** de la ventana.

No escriba demasiado texto en la ventana **Immediate**, ya que no puede salvarla; sólo puede salvar el contenido de la ventana **Program**.



## La Ventana del Editor GEM

La ventana que utiliza el editor funciona como todas las demás ventanas del GEM, de forma que puede moverse por ella utilizando la barra Move de la parte superior, cambiar su tamaño "arrastrando" el recuadro de la ventana y ampliarla a su tamaño máximo (y devolverla a su tamaño original) pulsando el botón del ratón sobre el recuadro Full. Pulsarlo sobre el recuadro Close equivale a seleccionar Quit en el menú File.

## Referencia Rápida

### Sentencias y Funciones

<b>ABS</b> ( <i>expresión numérica</i> ) muestra el valor absoluto de <i>expresión numérica</i>	Función
<b>ASC</b> ( <i>expresión cadena</i> ) muestra el código ASCII del primer carácter de <i>expresión cadena</i>	Función
<b>ATN</b> ( <i>expresión numérica</i> ) muestra el arcotangente de <i>expresión numérica</i>	Función
<b>BAR</b> <i>princ_x, princ_y, anchura, altura</i> dibuja un recuadro en la ventana de salida actual	Sentencia
<b>BEEP</b> envía un aviso acústico al altavoz del monitor	Sentencia
<b>BLOAD</b> <i>nombfich, dirección</i> carga un fichero binario en <i>dirección</i>	Sentencia
<b>BSAVE</b> <i>nombfich, dirección, longitud</i> salva en <i>nombfich</i> los bytes indicados en <i>longitud</i> a partir de <i>dirección</i>	Sentencia
<b>[CALL]</b> <i>nombre_subprograma</i> [( <i>parámetro</i> , <i>parámetro</i> )...] llama a un subprograma o a una rutina de la biblioteca	Sentencia
<b>CALL LOC</b> <i>dirección</i> [, <i>parámetro</i> ]... llama a una rutina en código máquina con parámetros opcionales	Sentencia
<b>CALLS</b> <i>variable_subprograma</i> llama indirectamente a un subprograma utilizando un puntero de variable	Sentencia
<b>CDBL</b> ( <i>expresión numérica</i> ) convierte <i>expresión numérica</i> en un número de doble precisión	Función

<b>CHDIR</b> <i>nombcamino</i> establece el directorio actual	Sentencia
<b>CHR\$</b> ( <i>código_ASCII</i> ) muestra una cadena de un carácter que corresponde a <i>código_ASCII</i>	Función
<b>CINT</b> ( <i>expresión numérica</i> ) convierte <i>expresión numérica</i> en un entero mediante redondeo	Función
<b>CIRCLE</b> <i>centro_x, centro_y, radio</i> [, <i>ángulo_princ, ángulo_final</i> ] dibuja un círculo o un arco en la ventana actual	Sentencia
<b>CLEAR</b> borra todas las variables y cierra todos los canales	Sentencia
<b>CLOSE</b> [[ <i>#</i> ] <i>número_canal</i> [, [ <i>#</i> ] <i>número_canal</i> ]...] termina la E/S del fichero o dispositivo especificado	Sentencia
<b>CLS</b> borra el contenido de la pantalla y lleva el cursor a la esquina superior izquierda de la misma	Sentencia
<b>COLOR</b> <i>color_texto</i> [, <i>color_relleno</i> ] [, <i>color_línea</i> ] [, <i>índice</i> ] [, <i>estilo</i> ] establece los atributos de color actual y de dibujo de líneas	Sentencia
<b>CONST</b> <i>nombre = constante_entera</i> [, <i>nombre = constante_entera</i> ]... define constantes enteras simbólicas	Sentencia
<b>COS</b> ( <i>expresión numérica</i> ) muestra el coseno de <i>expresión numérica</i> , que debe expresarse en radianes	Función
<b>CSNG</b> ( <i>expresión numérica</i> ) convierte <i>expresión numérica</i> en un número de precisión simple	Función
<b>CSRLIN</b> muestra la posición de línea actual del cursor	Función
<b>CVD</b> ( <i>cadena de 8 bytes de un flotante de doble precisión</i> )	Función
<b>CVI</b> ( <i>cadena de 2 bytes de un entero</i> )	Función
<b>CVL</b> ( <i>cadena de 4 bytes de un entero largo</i> )	Función
<b>CVS</b> ( <i>cadena de 4 bytes de un flotante de precisión simple</i> ) estas funciones muestran los valores numéricos internos de sus argumentos de cadena	Función
<b>DATA</b> <i>constante</i> [, <i>constante</i> ]... define los datos que hay que utilizar junto con READ	Sentencia



<b>DATES</b>	Sentencia/Función
muestra (función) o establece (sentencia) la fecha actual	
<b>DECR</b> <i>variable_numérica</i> resta 1 de <i>variable_numérica</i>	Sentencia
<b>DEF FN</b> <i>nombre_función</i> [( <i>lista_parámetros</i> )] = <i>expresión</i>	Sentencia
<b>DEF FN</b> <i>nombre_función</i> [( <i>lista_parámetros</i> )] [LOCAL <i>lista_variables</i> ] [STATIC <i>lista_variables</i> ] [SHARED <i>lista_variables</i> ] <i>sentencias</i> [EXIT DEF] END DEF	Sentencia
define funciones de usuario de una o varias líneas con parámetros	
[ <b>DEF FN</b> <i>nombre_función</i> = <i>expresión</i> ] define funciones de usuario de una sola línea sin parámetros	
<b>DEF SEG</b> = <i>expresión_numérica</i>	Sentencia
define el modo de funcionamiento de PEEK y POKE	
<b>DEFDBL</b> <i>rango_letras</i> [, <i>rango_letras</i> ]...	Sentencia
<b>DEFINT</b> <i>rango_letras</i> [, <i>rango_letras</i> ]...	Sentencia
<b>DEFLNG</b> <i>rango_letras</i> [, <i>rango_letras</i> ]...	Sentencia
<b>DEFSNG</b> <i>rango_letras</i> [, <i>rango_letras</i> ]...	Sentencia
<b>DEFSTR</b> <i>rango_letras</i> [, <i>rango_letras</i> ]...	Sentencia
declara variables como de precisión doble, entera, entera larga, simple o cadena, respectivamente.	
<b>DIM</b> [SHARED] <i>variable</i> [( <i>subíndices...</i> )] [, <i>variable</i> [( <i>subíndices...</i> )]]...	Sentencia
define los valores máximos para los subíndices de una matriz y asigna la memoria	
<b>DO</b> [{WHILE   UNTIL} <i>expresión_booleana</i> ] <i>sentencias</i> [EXIT {LOOP   DO}] <i>sentencias</i> {WEND   LOOP} [{WHILE   UNTIL} <i>expresión_booleana</i> ]	Sentencia
repite las sentencias que hay dentro de DO...LOOP mientras la condición o condiciones sean verdaderas (WHILE) o falsas (UNTIL)	
<b>ELLIPSE</b> <i>centro_x</i> , <i>centro_y</i> , <i>radio_x</i> , <i>radio_y</i> [, <i>ángulo_princ</i> , <i>ángulo_final</i> ]	Sentencia
dibuja una elipse o un arco elíptico en la ventana actual	

<b>END</b> [{DEF   IF   REPEAT <i>nombre</i>  SELECT   SUB}]	Sentencias
termina un programa BASIC, definición de función, bloque IF...THEN...ELSE, sentencia SELECT, bucle REPEAT o subprograma, respectivamente	
<b>EOF</b> ( <i>número_canal</i> )	Función
muestra la condición de fin-de-fichero de <i>número_canal</i>	
<b>ERASE</b> <i>nombre_matriz</i> [, <i>nombre_matriz</i> ]...	Sentencia
desasigna la matriz definida anteriormente con DIM	
<b>ERL</b>	Función
Muestra el número de línea donde se ha producido el último error	
<b>ERR</b>	Función
muestra el número del último error	
<b>ERROR</b> <i>expresión_entera</i>	Sentencia
simula la aparición de un error en tiempo de ejecución BASIC	
<b>EXIT</b> {DEF   DO   IF   FOR   LOOP   SELECT   SUB   <i>identificador</i> }	Sentencias
sale de una definición de función, subprograma BASIC o sentencia estructurada, respectivamente	
<b>EXP</b> ( <i>expresión_numérica</i> )	Función
muestra el valor exponencial de <i>expresión_numérica</i>	
<b>FEXISTS</b> ( <i>nombfich</i> )	Función
determina si existe o no el fichero especificado	
<b>FIELD</b> [#] <i>número_canal</i> , <i>anchura_campo</i> AS <i>variable_cadena</i> ...	Sentencia
asigna espacio para las variables dentro del archivo de acceso aleatorio	
<b>FILES</b> [ <i>espec_fich</i> ]	Sentencia
muestra los nombres de los ficheros del directorio especificado	
<b>FILL</b> <i>princ_x</i> , <i>princ_y</i>	Sentencia
rellena las formas cerradas que se han dibujado alrededor de <i>princ_x</i> , <i>princ_y</i>	
<b>FIX</b> ( <i>expresión_numérica</i> )	Función
muestra la parte entera truncada de <i>expresión_numérica</i>	
<b>FOR</b> <i>contador</i> = <i>comienzo</i> TO <i>final</i> [STEP <i>incremento</i> ]	Sentencia
<i>sentencias</i> NEXT [ <i>contador</i> ] [, <i>contador</i> ]...	
ejecuta una serie de instrucciones en bucle durante un número determinado de veces	



<b>FRE</b> {( <i>expresión numérica</i> ) ( <i>expresión cadena</i> )}	Funciones
muestra el espacio "heap" que queda libre o la memoria GEMDOS libre	
<b>GET</b> [#] <i>número canal</i> [, <i>número registro</i> ]	Sentencia de E/S de ficheros
lee en una memoria intermedia un registro de un fichero de disco de acceso aleatorio ya abierto	
<b>GET</b> ( <i>x1,y1</i> )-( <i>x2,y2</i> ), <i>nombre matriz</i>	Sentencia gráfica
captura imágenes binarias de cualquier parte de la pantalla	
<b>GOSUB</b> { <i>número1 línea</i>   <i>etiqueta1 línea</i> }	Sentencias
<i>sentencias</i>	
<b>RETURN</b> { <i>número2 línea</i>   <i>etiqueta2 línea</i> }	
bifurca a una subrutina y vuelve desde la misma	
<b>GOTO</b> { <i>número línea</i>   <i>etiqueta línea</i> }	Sentencias
transfiere la ejecución del programa a la línea/etiqueta especificada	
<b>HEX\$</b> ( <i>expresión numérica</i> )	Función
muestra una cadena que representa el valor hexadecimal de <i>expresión numérica</i>	
<b>IF</b> <i>expresión booleana</i> THEN <i>sentencia 1</i> [ <i>sentencia 2</i> ]...	Sentencia
[ELSE IF <i>expresión booleana</i> THEN <i>sentencia 3</i> [ <i>sentencia 4</i> ]...	
[ELSE <i>sentencia 5</i> [ <i>sentencia 6</i> ]...] END IF	
permite la ejecución condicional o la bifurcación basándose en el valor de una expresión booleana	
<b>INCR</b> <i>variable numérica</i>	Sentencia
suma 1 a <i>variable numérica</i>	
<b>INKEY\$</b>	Función
lee un carácter del teclado, si hay alguno, sin mostrarlo en la pantalla	
<b>INP</b> ( <i>gestión BIOS</i> )	Función
muestra un byte del dispositivo especificado por <i>gestión BIOS</i>	
<b>INPUT</b> [;] [" <i>indicador</i> " {;},] <i>lista variables</i>	Sentencia
solicita una entrada desde el teclado y, luego, lee dicha entrada en <i>lista variables</i>	
<b>INPUT</b> # <i>número canal</i> , <i>lista variables</i>	Sentencia
lee datos de un fichero o dispositivo en <i>lista variables</i>	
<b>INPUT\$</b> ( <i>n</i> , [#] <i>número canal</i> )	Sentencia
lee <i>n</i> caracteres del canal especificado	

<b>INSTR</b> ([ <i>princ</i> ,] <i>1ª cadena</i> , <i>2ª cadena</i> )	Función
muestra la posición de la primera aparición, después de <i>princ</i> , de <i>2ª cadena</i> dentro de <i>1ª cadena</i>	
<b>INT</b> ( <i>expresión numérica</i> )	Función
muestra el mayor entero menor o igual que <i>expresión numérica</i>	
<b>KILL</b> <i>espec fich</i>	Sentencia
suprime del disco actual todos los ficheros que cumplen la <i>espec fich</i>	
<b>LBOUND</b> ( <i>matriz</i> [, <i>dimensión</i> ])	Función
muestra el menor subíndice de la matriz especificada	
<b>LCASE\$</b> ( <i>expresión cadena</i> )	Función
convierte en minúsculas las letras de <i>expresión cadena</i>	
<b>LEFT\$</b> ( <i>expresión cadena</i> , <i>n</i> )	Función
muestra una cadena formada por los <i>n</i> caracteres situados más a la izquierda de <i>expresión cadena</i>	
<b>LEN</b> ( <i>expresión cadena</i> )	Función
muestra el número de caracteres de <i>expresión cadena</i>	
[LET] <i>variable</i> = <i>expresión</i>	Sentencia
asigna <i>expresión</i> a <i>variable</i>	
<b>LIBRARY</b> <i>nombre biblioteca</i> [, <i>nombre biblioteca</i> ]...	Sentencia
define las bibliotecas que va a utilizar el programa	
<b>LINEF</b> <i>princ x</i> , <i>princ y</i> , <i>final x</i> , <i>final y</i>	Sentencia
dibuja una línea entre dos puntos de la ventana actual	
<b>LINE INPUT</b> [;] [" <i>indicador</i> "] <i>variable cadena</i>	Sentencia
asigna toda una línea de entrada a <i>variable cadena</i> , ignorando delimitadores tales como comas	
<b>LINE INPUT</b> # <i>número canal</i> , <i>variable cadena</i>	Sentencia
lee una secuencia de caracteres terminados por un CRLF en <i>variable cadena</i> desde el dispositivo/fichero especificado por <i>número canal</i>	
<b>LOC</b> ( <i>número canal</i> )	Función
muestra la posición actual dentro de un fichero abierto	
<b>LOCAL</b> <i>lista variables</i>	Sentencia
declara la lista de variables como local para una función o sub_programa	



<b>LOCATE</b> <i>fila[,columna[,cursor]]</i> sitúa el cursor en la posición especificada y lo activa/desactiva	Sentencia
<b>LOF</b> ( <i>número_canal</i> ) muestra la longitud de un fichero determinado	Función
<b>{LOG   LOG10   LOG2}</b> ( <i>expresión_numérica</i> ) muestra los logaritmos, en varias bases, de <i>expresión_numérica</i>	Funciones
<b>LPOS</b> ( <i>argumento</i> ) muestra la posición actual de la cabeza de impresión	Función
<b>LPRINT</b> [ <i>lista_expresiones</i> ][{; ,}]	Sentencia
<b>LPRINT USING</b> <i>cadena_formato</i> ; <i>lista_expresiones</i> {; ,}] imprime datos a través de la puerta de impresora actual; ver PRINT, PRINT USING	Sentencia
<b>LSET</b> <i>variable_cadena</i> = <i>expresión_cadena</i> justifica a la izquierda una variable de cadena, que normalmente se utiliza para variables de campo	Sentencia
<b>MID\$</b> ( <i>expresión_cadena</i> , <i>n</i> [, <i>longitud</i> ]) muestra <i>longitud</i> caracteres de <i>expresión_cadena</i> , empezando en la <i>n</i> -sima posición de la <i>cadena</i>	Función
<b>MID\$</b> ( <i>variable_cadena</i> , <i>n</i> [, <i>longitud</i> ]) = <i>expresión_cadena</i> modifica parte de una variable de cadena	Sentencia
<b>MKDIR</b> <i>nombcamino</i> crea el subdirectorio especificado por <i>nombcamino</i>	Sentencia
<b>MKI\$</b> ( <i>expresión_entera</i> )	Función
<b>MKL\$</b> ( <i>expresión_entera_larga</i> )	Función
<b>MKSS\$</b> ( <i>expresión_precisión_simple</i> )	Función
<b>MKDS\$</b> ( <i>expresión_precisión_doble</i> ) convierte en cadenas los datos numéricos de las expresiones	Función
<b>MOUSE</b> ( <i>atributo</i> ) lee la posición actual del ratón, el estado de los botones y de las teclas Shift (mayúsculas) del teclado	Función
<b>MOUSE</b> <i>tipo_ratón</i> ajusta el puntero del ratón a una forma predefinida o lo borra de la pantalla	Sentencia
<b>NAME</b> <i>nombfich_ant</i> AS <i>nombfich_nue</i> renombra un fichero	Sentencia

<b>OCT\$</b> ( <i>expresión_numérica</i> ) muestra una cadena que es la representación octal de <i>expresión_numérica</i>	Función
<b>ON ERROR GOTO</b> { <i>número_línea</i>   <i>etiqueta_línea</i> } activa la gestión de errores y especifica la rutina de la misma	Sentencia
<b>ON n GOSUB</b> { <i>número_línea</i>   <i>etiqueta_línea</i> },{ <i>número_línea</i>   <i>etiqueta_línea</i> }...] llama a una de una lista de subrutinas, dependiendo del valor de <i>n</i>	Sentencia
<b>ON n GOTO</b> { <i>número_línea</i>   <i>etiqueta_línea</i> },{ <i>número_línea</i>   <i>etiqueta_línea</i> }...] hace que la ejecución del programa bifurque a una de una lista de líneas/etiquetas de programa, dependiendo del valor de <i>n</i>	Sentencia
<b>OPEN</b> <i>espec_fich</i> [FOR <i>modo</i> ] AS [#] <i>número_canal</i> [LEN = <i>tamaño_registro</i> ]	Sentencia
<b>OPEN</b> <i>cadena_modo</i> ,[#] <i>número_canal</i> , <i>espec_fich</i> [, <i>tamaño_registro</i> ] abre un fichero para su lectura o escritura	Sentencia
<b>OPTION BASE</b> {0   1} define el menor valor de subíndice para matrices	Sentencia
<b>OUT</b> <i>gestión_bios</i> , <i>expresión_entera</i> envía un byte a las rutinas de salida del BIOS	Sentencia
<b>PALETTE</b> <i>número_color</i> , <i>color_físico</i> permite cambiar la paleta de colores de la pantalla	Sentencia
<b>PALETTE USING</b> <i>elemento_matriz</i> modifica la paleta utilizando los elementos de la matriz	Sentencia
<b>PCIRCLE</b> <i>centro_x</i> , <i>centro_y</i> , <i>radio</i> [, <i>ángulo_princ</i> , <i>ángulo_final</i> ] dibuja un círculo sólido o un arco en la ventana actual	Sentencia
<b>PCOPY</b> vuelca la pantalla actual en la impresora	Sentencia
<b>PEEK</b> ( <i>dirección</i> ) muestra el byte, palabra o valor largo (dependiendo de la sentencia DEF SEG) del contenido de la posición <i>dirección</i>	Función
<b>PEEKB</b> ( <i>dirección</i> )	Función
<b>PEEKL</b> ( <i>dirección_par</i> )	Función
<b>PEEKW</b> ( <i>dirección_par</i> ) muestra el contenido (de varios tipos) de la memoria especificada	Función



<b>PELLIPSE</b> <i>centro_x,centro_y,radio_x,radio_y</i> [, <i>ángulo_princ,ángulo_final</i> ]	Sentencia
dibuja una elipse sólida o un arco elíptico en la ventana actual	
<b>POINT</b> ( <i>punto_x,punto_y</i> )	Función
muestra el color de un punto determinado	
<b>POKE</b> <i>dirección,datos</i>	Sentencia
escribe datos directamente en la <i>dirección</i> de la memoria	
<b>POKEB</b> <i>dirección,valor_byte</i>	Sentencia
<b>POKEL</b> <i>dirección_par,valor_largo</i>	Sentencia
<b>POKEW</b> <i>dirección_par,valor_palabra</i>	Sentencia
escribe datos de distintos tipos directamente en la memoria	
<b>POS</b> ( <i>x</i> )	Función
muestra el número de columna de la posición actual del cursor	
<b>PRESET</b> [STEP] ( <i>pos_x,pos_y</i> ) [, <i>color</i> ]	Sentencia
restaura o define un punto como de un color determinado dentro de la ventana actual	
<b>PRINT</b> [ <i>expresión_1</i> ] [{ ;     , } [ <i>expresión_2</i> ] [;]...	Sentencia
imprime datos en la pantalla	
<b>PRINT #</b> <i>número_canal</i> ,[ <b>USING</b> <i>cadena_formato</i> ]	Sentencia
<i>lista_expresiones</i> [;]	
escribe datos formateados en un fichero o dispositivo	
<b>PRINT USING</b> <i>cadena_formato</i> ; <i>lista_expresiones</i> [{ .   ; }]	Sentencia
imprime datos formateados en la pantalla	
<b>PSET</b> [STEP] ( <i>pos_x,pos_y</i> ) [, <i>color</i> ]	Sentencia
traza un punto de un color determinado en la ventana actual	
<b>PUT</b> [#] <i>número_canal</i> ,[ <i>número_registro</i> ]...	Sentencia de E/S de ficheros
escribe un registro desde la memoria intermedia de ficheros al fichero de acceso aleatorio	
<b>PUT</b> ( <i>x,y</i> ), <i>nombre_matriz</i> [, <i>verbo</i> ]	Sentencia gráfica
copia en la pantalla la imagen rectangular salvada con GET	
<b>RANDOMIZE</b> [ <i>expresión</i> ]	Sentencia
inicializa el generador aleatorio para una nueva secuencia	
<b>READ</b> <i>lista_variables</i>	Sentencia
asigna variables con valores especificados por sentencias DATA	
<b>REDIM</b> [APPEND] <i>matriz(subíndices)</i> [, <i>matriz(subíndices)</i> ]...	Sentencia
cambia el tamaño de una matriz ya definida	

<b>REM</b> <i>comentarios...</i>	Sentencia
permite añadir comentarios a un programa (también puede utilizar ')	
<b>REPEAT</b> <i>nombre</i>	Sentencia
<i>sentencias</i>	
[EXIT <i>nombre</i> ]	
<i>sentencias</i>	
END REPEAT <i>nombre</i>	
las sentencias del bucle REPEAT se ejecutan hasta que se encuentra una sentencia EXIT o END REPEAT para el bucle	
<b>RESET</b>	Sentencia
cierra todos los ficheros abiertos del disco	
<b>RESTORE</b> [{ <i>número_línea</i>   <i>etiqueta</i> }]	Sentencia
permite a una sentencia READ acceder a una sentencia DATA que ya se ha leído	
<b>RESUME</b> { <i>número_línea</i>   <i>etiqueta</i> }	Sentencia
reanuda la ejecución del programa desde dentro de una rutina de gestión de errores en la línea/etiqueta especificada	
<b>RETURN</b> [{ <i>número_línea</i>   <i>etiqueta</i> }]	Sentencia
devuelve la ejecución de un programa desde una subrutina	
<b>RIGHT\$</b> ( <i>expresión_cadena</i> , <i>n</i> )	Función
muestra una cadena empieza en el carácter número <i>n</i> desde la derecha de <i>expresión_cadena</i>	
<b>RMDIR</b> <i>nombrcamino</i>	Sentencia
borra un subdirectorio vacío	
<b>RND</b> [( <i>n</i> )]	Función
muestra un número de precisión simple pseudo-aleatorio comprendido entre 0 y 1	
<b>RSET</b> <i>variable_cadena</i> = <i>expresión_cadena</i>	Sentencia
lleva datos a una memoria intermedia de ficheros de acceso aleatorio. También puede utilizarse para justificar a la derecha la cadena de <i>variable_cadena</i>	
<b>RUN</b> [{ <i>número_línea</i> }]	Sentencia
reinicia el programa actual	
<b>SADD</b> ( <i>expresión_cadena</i> )	Función
muestra la dirección de memoria de <i>expresión_cadena</i>	
<b>SCREEN</b> <i>modo</i>	Sentencia
cambia el modo de la pantalla	



<b>SELECT</b> [CASE   ON] <i>variable</i> [CASE   =] <i>lista_caso</i> <i>sentencias</i> [CASE   =] <i>lista_caso</i> <i>sentencias</i> [CASE   =] <i>lista_caso</i> <i>sentencias</i> END SELECT	Sentencia
se ejecuta una serie de sentencias dependiendo del valor de <i>variable</i>	
<b>SGN</b> ( <i>expresión numérica</i> )	Función
muestra el signo de <i>expresión numérica</i>	
<b>SHARED</b> <i>lista variables</i>	Sentencia
permite que un subprograma acceda a variables del programa principal sin pasarlas como parámetros	
<b>SIN</b> ( <i>expresión numérica</i> )	Función
muestra el seno de <i>expresión numérica</i> , que debe estar en radianes	
<b>SOUND</b> <i>voz, volumen</i> [, <i>nota</i> ][, <i>octava</i> ][, <i>duración</i> ]	Sentencia
permite controlar los 3 canales de sonido	
<b>SPACE\$</b> ( <i>n</i> )	Función
muestra una cadena de <i>n</i> espacios	
<b>SPC</b> ( <i>n</i> )	Función
hace que se ignoren <i>n</i> columnas en una sentencia PRINT	
<b>SQR</b> ( <i>expresión numérica</i> )	Función
muestra la raíz cuadrada de <i>expresión numérica</i>	
<b>STATIC</b> <i>lista variables</i>	Sentencia
declara variables como locales en una función o subprograma, pero conserva constantes sus valores en múltiples llamadas y no crea una nueva variable para cada llamada	
<b>STICK</b> ( <i>n</i> )	Función
muestra las posiciones de los dos joysticks	
<b>STOP</b> [ <i>código proceso</i> ]	Sentencia
hace que termine el programa; se cierran todos los ficheros y el control vuelve al sistema operativo	
<b>STR\$</b> ( <i>expresión numérica</i> )	Función
muestra la representación de cadena de <i>expresión numérica</i>	
<b>STRIG</b> ( <i>n</i> )	Función
muestra el estado del botón de disparo del joystick especificado	

<b>STRING\$</b> ( <i>m, n</i> )	Función
muestra una cadena del carácter ASCII <i>n</i> de longitud <i>m</i>	
<b>STRING\$</b> ( <i>m, expresión cadena</i> )	Sentencia
muestra una cadena del carácter ASCII que es el primer carácter de <i>expresión cadena</i> , de longitud <i>m</i>	
<b>SUB</b> <i>nombre global</i> [( <i>lista parámetros</i> )] [STATIC] <i>sentencias</i> [EXIT SUB] <i>sentencias</i> END SUB	Sentencia
define un subprograma	
<b>SWAP</b> <i>1ª variable, 2ª variable</i>	Sentencia
intercambia el valor de dos variables	
<b>SYSTAB</b>	Función
muestra la dirección de una tabla interna del sistema	
<b>SYSTEM</b>	Sentencia
termina la ejecución de un programa, cierra todos los ficheros y vuelve al sistema operativo	
<b>TAB</b> ( <i>n</i> )	Función
lleva la posición de impresión a la <i>n-sima</i> columna	
<b>TAN</b> ( <i>expresión numérica</i> )	Función
muestra la tangente de <i>expresión numérica</i> , que debe estar en radianes	
<b>TIMES</b>	Sentencia/Función
muestra o define la hora del sistema	
<b>TIMER</b>	Función
muestra los segundos del temporizador interno como un número en punto flotante de precisión simple	
<b>UBOUND</b> ( <i>nombre matriz</i> [, <i>dimensión</i> ])	Función
muestra el mayor subíndice de la matriz especificada	
<b>UCASE\$</b> ( <i>expresión cadena</i> )	Función
pone en mayúsculas todos los caracteres alfabéticos de <i>expresión cadena</i>	
<b>VAL</b> ( <i>expresión cadena</i> )	Función
muestra el valor numérico de <i>expresión cadena</i>	



<b>VARPTR</b> ( <i>nombre_variable</i> ) muestra la dirección de memoria de la variable <i>nombre_variable</i>	Función
<b>VARPTR</b> ( <i>núm_canal</i> ) muestra la dirección de memoria de la memoria intermedia de entrada/salida asociada al canal especificado	Función
<b>VARPTRS</b> ( <i>subnombre</i> ) muestra la dirección de memoria del subprograma especificado	Función
<b>WAVE</b> <i>activar</i> [, <i>sobre</i> [, <i>forma</i> [, <i>tiempo</i> [, <i>demora</i> ]]]] controla las formas de onda que utiliza la sentencia SOUND	Sentencia
<b>WHILE</b> <i>condición</i> <i>sentencias</i> <b>WEND</b> se ejecuta una serie de sentencias en un bucle hasta que condición sea falsa	Sentencia
<b>WIDTH</b> [# <i>número canal</i> ,] <i>anchura</i> [, <i>tabulación</i> ]	Sentencia
<b>WIDTH LPRINT</b> <i>anchura</i> [, <i>tabulación</i> ] asigna una anchura de línea a un fichero, pantalla o impresora determinado	Sentencia
<b>WINDOW CLOSE</b> [ <i>id</i> ]	Sentencia
<b>WINDOW CONTRL</b> <i>id</i> , <i>control</i> , <i>valor</i>	Sentencia
<b>WINDOW FULLW</b> [ <i>id</i> ]	Sentencia
<b>WINDOW GET</b> <i>id</i> , <i>rectángulo</i> , <i>posx</i> , <i>posy</i> , <i>anchura</i> , <i>altura</i>	Sentencia
<b>WINDOW LOCATE</b> <i>id</i> , <i>posx</i> , <i>posy</i> , <i>anchura</i> , <i>altura</i>	Sentencia
<b>WINDOW NAME</b> <i>id</i> [, <i>cadena_título</i> ] [, <i>cadena_estado</i> ]	Sentencia
<b>WINDOW OFF</b>	Sentencia
<b>WINDOW ON</b>	Sentencia
<b>WINDOW OPEN</b> <i>id</i> , <i>cadena_título</i> , <i>posx</i> , <i>posy</i> , <i>anchura</i> , <i>altura</i> , <i>tipo</i>	Sentencia
<b>WINDOW OUTPUT</b> [ <i>id</i> ]	Sentencia
<b>WINDOW READ</b> <i>id</i> , <i>control</i> , <i>resultado</i> estas sentencias controlan la salida en pantalla de varias ventanas	Sentencia
<b>WRITE</b> [ <i>lista_expresiones</i> ] imprime en la pantalla los datos especificados en <i>lista_expresiones</i>	Sentencia
<b>WRITE</b> # <i>número_canal</i> , <i>lista_expresiones</i> imprime en un fichero secuencial los datos especificados en <i>lista_expresiones</i>	Sentencia

## La Biblioteca Suministrada

### GEMVDI: Control

fnGDOS ... fnHANDLE ... CHANGE\_HANDLE gestión ... fnRESOLUTION...  
v\_opnwk env(), varptr gestión, salv() ... v\_clswk ...  
v\_opnvwk env(), varptr gestión, salv() ... v\_clsvwk ... v\_updwbk ...  
v\_clrwrk ...  
FNvst\_load\_fonts ... vst\_unload\_fonts ...  
vs\_clip señal,x1,y1,x2,y2

### GEMVDI: Dibujo de Primitivas

v\_pline n,pts() ... v\_pmarker n,pts() ... v\_gtext x,y,texto\$ ... v\_fillarea  
n,pts() ... v\_contourfill x,y,color ... vr\_recfl x1,y1,x2,y2 ... v\_bar  
x1,y1,x2,y2 ... v\_arc x,y,r,ángulo\_princ,ángulo\_final ... v\_pieslice  
x,y,r,ángulo\_princ,ángulo\_final ... v\_circle x,y,r ... v\_ellarc  
x,y,xr,yr,ángulo\_princ,ángulo\_final ... v\_ellpie  
x,y,xr,yr,ángulo\_princ,ángulo\_final ... v\_ellipse x,y,xr,yr ... v\_rbox  
x1,y1,x2,y2 ... v\_rfbox x1,y1,x2,y2 ... v\_justified  
x,y,texto\$,longitud,señalpal,señalgráf

### GEMVDI: Atributos

vswr\_mode modo ... vs\_color índice,r,g,b ... vsl\_type tipo ... vsl\_udsty  
trama vsl\_width anchura ... vsl\_color color ... vsl\_ends  
estiloprinc,estilofinal ... vsm\_type tipo ... vsm\_height altura ... vsm\_color  
color ... vst\_height altura ... vst\_point puntos ... vst\_rotation ángulo ...  
vst\_font tipo ... vst\_color color ... vst\_effects efectos ... vst\_alignment  
horizontal,vertical ... vsf\_interior relleno\_interior vsf\_style índice\_estilo  
... vsf\_color color ... vsf\_perimeter señal ... vsf\_updat plano(),planos

### GEMVDI: Funciones de Barrido

vro\_cpyfm modo,xy(), fuente&, dest& ...  
vrt\_cpyfm modo,xy(), fuente&, dest&, principal, fondo...  
v\_get\_pixel x,y,varptr valor, varptr índice

### GEMVDI: Funciones de Estado del Ratón y del Teclado

v\_show\_c señal ... v\_hide\_c ...  
vq\_mouse varptr botón, varptr x, varptr y ... vq\_key\_s varptr kstado



## GEMVDI: Consultas

vq\_extnd señal,info() ... vq\_color índice\_color,señal,rgb() ...  
vql\_attributes info()  
vqm\_attributes info() ... vql\_attributes info() ... vqt\_attributes info() ...  
vqt\_extent texto\$,info() ... vqt\_width car,varptr anchuracelda, varptr  
izquierda, varptr derecha  
FNvqt\_name (número, varptr nombre\$) ...  
vqt\_fontinfo varptr primercar, varptr últcar, distancias(), varptr  
anchuramáx, efectos()

## GEMAES: Paso de Mensajes

appl\_read id\_ap,longitud,mensaje& ...  
appl\_write id\_ap,longitud,mensaje& ... FNappl\_find nombre\$

## GEMAES: Sucesos

FNevnt\_keybd ...  
FNevnt\_button (pulsaciones,máscara,estado,varptr xsal, varptr ysal,  
varptr botón, varptr kestado) ...  
evnt\_mouse señal,x,y,w,h, varptr xsal, varptr ysal, varptr botón, varptr  
kestado ...  
envt\_mesag mensaje& ... envt\_timer hora&  
FNevnt\_multi (eseñales,pulsaciones,máscara,estado,señal1,x1,y1,  
anch1,alt1,señal2,x2,y2,anch2,alt2,mensaje&,hora&, varptr xsal, varptr  
ysal, varptr botón, varptr kestado, varptr tecla\_pulsada, varptr clicks) ...  
FNevnt\_dclick valornue,serorget

## GEMAES: Menús

FNmenu&(cadena\_menú\$) ... menu\_bar árbol&,señal ...  
menu\_ichack árbol&,opción,señal ... menu\_ienable árbol&,opción,señal  
menu\_tnormal árbol&,título,señal ... menu\_text árbol&,opción,texto\$

## GEMAES: Objetos

FNobjc\_add(árbol&,padre,hijo) ... FNobjc\_delete(árbol&,objeto) ...  
FNobjc\_draw(árbol&,objeto,fondo,x,y,anch,alt) ...  
FNobjc\_find(árbol&,objeto,fondo,x,y) ...  
FNobjc\_offset(árbol&,objeto,varptr x,varptr y) ...  
FNobjc\_order(árbol&,objeto,posnue) ...  
FNobjc\_edit(árbol&,objeto,car,índice,hijo,varptr indicenue) ...  
FNobjc\_change(árbol&,objeto,x,y,anch,alt,estadonue,señal)

## GEMAES: Formatos

FNform\_do(árbol&,objeto) ...  
form\_dial señal,x1,y1,anch1,alt1,x2,y2,anch2,alt2 ...  
FNform\_alert(defecto,alerta\$) ...  
FNform\_error(códigoerror) ... form\_center árbol&,varptr x,varptr  
y,varptr anch,varptr alt

## GEMAES: Gráficos

graf\_rubberbox x,y,anchmín,altmín,varptr anchsal,varptr altsal ...  
graf\_dragbox anch,alt,xprinc,yprinc,xrec,yrec,anchrec,altrec,varptr  
xsal,varptr ysal)  
graf\_movebox anch,alt,x1,y1,x2,y2 ...  
graf\_growbox x1,y1,anch1,alt1,x2,y2,anch2,alt2 ...  
graf\_shrinkbox x1,y1,anch1,alt1,x2,y2,anch2,alt2  
FNgraf\_watchbox(árbol&,objeto,estadoen,estadosal)  
FNgraf\_slidebox(árbol&,padre,objeto,vertical)  
FNgraf\_handle(varptr anchcelda,varptr altcelda,varptr anchrec,varptr  
altrec) ...  
graf\_mouseflag, formato\_usuario& ...  
graf\_mkstate varptr x,varptr y,varptr estado,varptr kestado

## GEMAES: Funciones de Directorio

FNscrp\_read(varptr nombre\_directorio\$) ... FNscrp\_write  
(nombre\_directorio\$)

## GEMAES: Selector de Ficheros

fsel\_input varptr camino\$, varptr nombre\$,varptr ok

## GEMAES: Ventanas

FNwind\_create clase,x,y,anch,alt ... FNwind\_open(gestión,x,y,anch,  
alt) ...  
FNwind\_close (gestión) ... FNwind\_delete(gestión) ...  
FNwind\_get (gestión,clasif,varptr x,varptr y,varptr anch,varptr alt) ...  
FNwind\_set(gestión,clasif,x,y,anch,alt) ... FNwind\_find(x,y)  
FNwind\_update(clasif) ... FNwind\_calc(tipo,clase,x,y,anch,alt, varptr xsal,  
varptr ysal, varptr anchsal, varptr altsal)

## GEMAES: Ficheros de Recursos

FNrsrc\_load(nombre\_fich\$) ... FNrsrc\_free ...  
FNrsrc\_gaddr(tipo,índice,varptr direc&) ... FNrsrc\_saddr(tipo,  
índice,direc&) ...  
rsrc\_obfix árbol&, objeto



## GEMAES: Rutinas del Entorno

```
FNshel_read(varptr cmd$, varptr final$) ... FNshel_find(varptr fich$)
shel_envrn varptr env$, nombre$
```

## Tipos de Variables

Tipo	Identificador	Rango
Entero (16 bits)	%	-32768 a 32767
Entero largo (32 bits)	&	-2147483648 a 2147483647
FFP precisión simple	!	-9.2E18 a +9.2E18
IEEE precisión doble	#	-1.8D308 a 1.8D308
Cadena	\$	0 a 16.777.215 caracteres

## Juego de Caracteres

FirST BASIC utiliza caracteres ASCII en sus ficheros de entrada. Los siguientes caracteres tienen significados especiales:

a-z, A-Z	Las letras, que se utilizan en palabras reservadas y en los nombres de las variables de usuario, así como en los nombres de etiquetas y subprogramas. En las definiciones de variables y palabras reservadas no se diferencia entre minúsculas y mayúsculas.
E, e, D, d	también se utilizan para exponentes en los números.
0-9	Los dígitos, que se utilizan en números y también pueden usarse en nombres, siempre y cuando no sean el primer carácter.
.	El punto, que se utiliza como punto decimal en números y también puede usarse en nombres, siempre y cuando no sea el primer carácter.
%	El signo de tanto por ciento, que se utiliza para indicar que una variable es un entero de 16 bits; es decir, cuyos valores deben estar comprendidos entre -32768 y 32767.
&	Se utiliza para indicar que las variables son enteros largos; es decir, sus valores deben estar comprendidos entre $-2^{31}$ y $2^{31}-1$ . También se utiliza para introducir constantes hexadecimales, octales y binarias.
!	El signo de admiración, que se utiliza para indicar que una variable es un número de punto flotante de precisión simple.
#	El signo de número, que se utiliza para indicar que una variable es un número de punto flotante de doble precisión, así como para indicar que ciertas operaciones de entrada/salida deben direccionarse a canales en lugar de a la pantalla (por ejemplo, PRINT #).
\$	se utiliza para indicar variables de cadena.

-	El carácter de subrayado, que puede utilizarse en variables después del primer carácter. Si aparece al principio de un símbolo, indica que se ignorará el resto de la línea y que la línea siguiente se va a considerar parte de la línea actual.
"	Las comillas se utilizan para delimitar literales de cadena.
'	El apóstrofo, o comilla simple, se utiliza para indicar que el resto de la línea se va a considerar un comentario.
( )	Los paréntesis, que se utilizan alrededor de argumentos de funciones, para marcar la prioridad de los operadores y para indicar matrices.
+ - * /	los operadores aritméticos básicos.
=	operador de asignación y de igualdad.
< >	operadores de comparación menor que y mayor que.
^	operador de exponenciación.
\	El carácter de barra invertida, que se utiliza como operador de división de enteros.
[CTRL]-Z	Si en un fichero se encuentra el carácter ASCII [CTRL]-Z (chr\$(26)), se considerará como el final de fichero.

Otros caracteres con valores ASCII inferiores a 32 se consideran espacios en blanco y se ignoran. Otros caracteres pueden utilizarse en cadenas, pero de lo contrario generarán un aviso y se ignorarán.

## Operadores

Las expresiones están formadas por constantes, variables, variables de matriz, llamadas a funciones y operadores. A continuación se muestra el orden de prioridad, siendo la primera la mayor prioridad:

1. Exponenciación (elevar a la potencia de) (^)
2. Menos unitario (-)
3. Multiplicación (\*) y División en Punto Flotante (/)
4. División de Enteros (\)
5. Módulo (MOD)
6. Adición (+) y Substracción (-)
7. Comparaciones (=, <, >, <=, >=, =)
8. NOT
9. AND
10. OR y XOR (o exclusivo)
11. EQV
12. IMP

La única excepción es que  $x^{-y}$  se evalúa como  $x^{(-y)}$ . Utilice paréntesis para cambiar el orden de evaluación.



## Your FirST BASIC

---

Este libro sobre **FirST BASIC** contiene valiosa información sobre cómo sacar el máximo provecho del paquete, incluyendo una explicación detallada de todos los comandos y bibliotecas, así como un curso que no sólo le enseñará de manera sencilla a utilizar **FirST BASIC**, sino que también le introducirá en los secretos de la programación del Atari ST y del GEM para producir programas fascinantes.

**Your FirST BASIC** contiene más de 300 páginas de información y puede adquirirse en cualquier tienda especializada o directamente en HiSoft.

## Power BASIC y HiSoft BASIC

---

HiSoft dispone de otros muchos lenguajes de programación y utilidades para el Atari ST, incluyendo dos compiladores más de BASIC, **Power BASIC** y **HiSoft BASIC**. Ambos paquetes permiten compilar en disco, de forma que pueda producir programas autónomos para su uso propio o para regalar o vender sin pagar royalties. Ambos incluyen un conjunto completo de opciones de compilación que le permiten controlar por completo el proceso de compilación, lo cual acelerará notablemente sus programas. También incluyen bibliotecas (BIOS, XBIOS, sonido y gráficos) y manuales que cubren todos los aspectos de los paquetes.

**HiSoft BASIC** va dirigido al programador profesional e incluye numerosas características avanzadas y útiles, como el ajuste de programas para mejorar la velocidad de los mismos, bibliotecas ampliables por el usuario de forma que pueda escribir sus propias extensiones del BASIC, la posibilidad de incluir símbolos de sus programas para depuración a bajo nivel y una versión autónoma del compilador, de forma que pueda utilizar su propio entorno de edición si lo desea. **HiSoft BASIC** también permite crear accesorios de mesa de trabajo desde dentro del BASIC.