36
150pts.

RUN

Enciclopedia Práctica del Spectrum

Nueva Lente/Ingelek

¡YA ESTA A LA VENTA!

semanal

la revista que esperaba el usuario de COMMODORE

novedad en España: «fuera errores»

UNA GRAN OBRA A SU ALCANCE

UNA OBRA COMPLETISIMA EN 30 VOLUMENES QUE TRATA TODOS LOS TEMAS , DESDE QUE ES UN ORDENADOR HASTA EL ESTUDIO DE LOS DIVERSOS LENGUAJES, PASANDO POR LOS LENGUAJES, METODOS DE PROGRAMACION, ELECCION DEL ORDENADOR ADECUADO, DICCIONARIO, ETC.

B.B.I.
INGELEK

30 EXTRAORDINARIOS VOLUMENES DE APARICION SEMANAL CON TODOS LOS CONCEPTOS DE LA INFORMATICA

GRAN OFERTA DE SUSCRIPCION
9.995 PTAS

AHORRE MAS DE 1.000 PTAS Y LLEVESE UNA MAGNIFICA CALCULADORA SOLAR VALORADA EN 2.500 PTAS.

OFERTA VALIDA UNICAMENTE PARA ESPAÑA

SUSCRIBASE POR TELEFONO

Todos los días, excepto sábados y festivos, de 8 a 6,30 atenderemos sus consultas en el

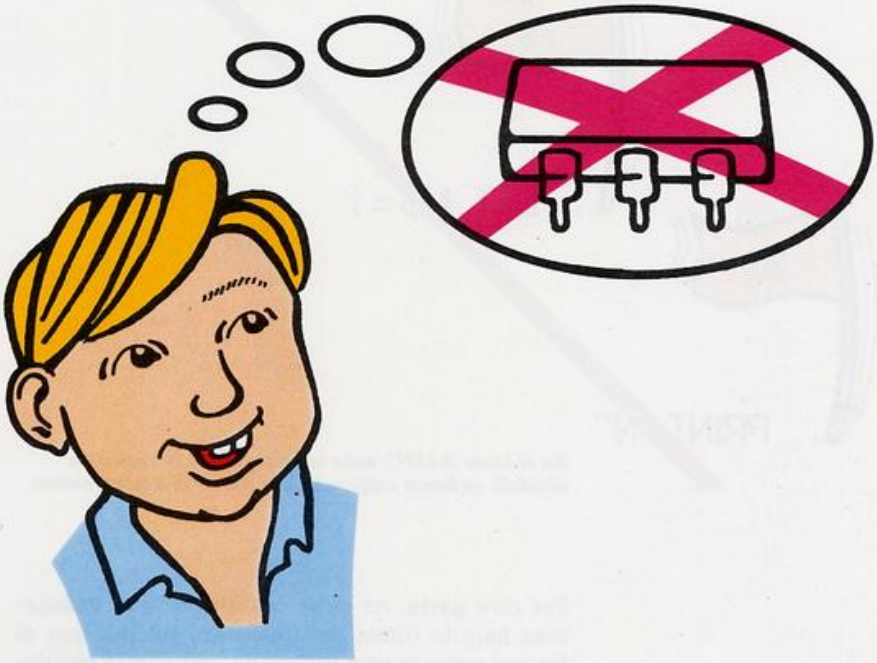
2505820

DENTRO DEL BASIC

QUIZÁ una de las grandes ventajas de los lenguajes de alto nivel sobre los ensambladores, sea la total despreocupación del programador por los «asuntos» internos de la máquina. Cuando programamos en BASIC, no nos importa la configuración interna de las variables, tanto numéricas como de cadena, ni tampoco la forma en que el Sistema organiza sus estructuras más complejas de información: las matrices.

Sin embargo, el conocimiento de la forma exacta en que el BASIC almacena la información puede sernos de gran utilidad para comprender ciertas situaciones. Puede aclararnos, por ejemplo, cómo se las ingenia el Sistema para almacenar datos numéricos con valores tan dispares como los que el BASIC puede admitir, recurriendo únicamente a un tipo de variable numérica.

Además de hablar sobre la estructura interna de las variables, trataremos también en este capítulo de la forma en que se almacenan internamente las líneas de texto BASIC, lo cual puede permitir a los más avanzados desarrollar algún tipo de programa de «ayuda» a la programación BASIC.



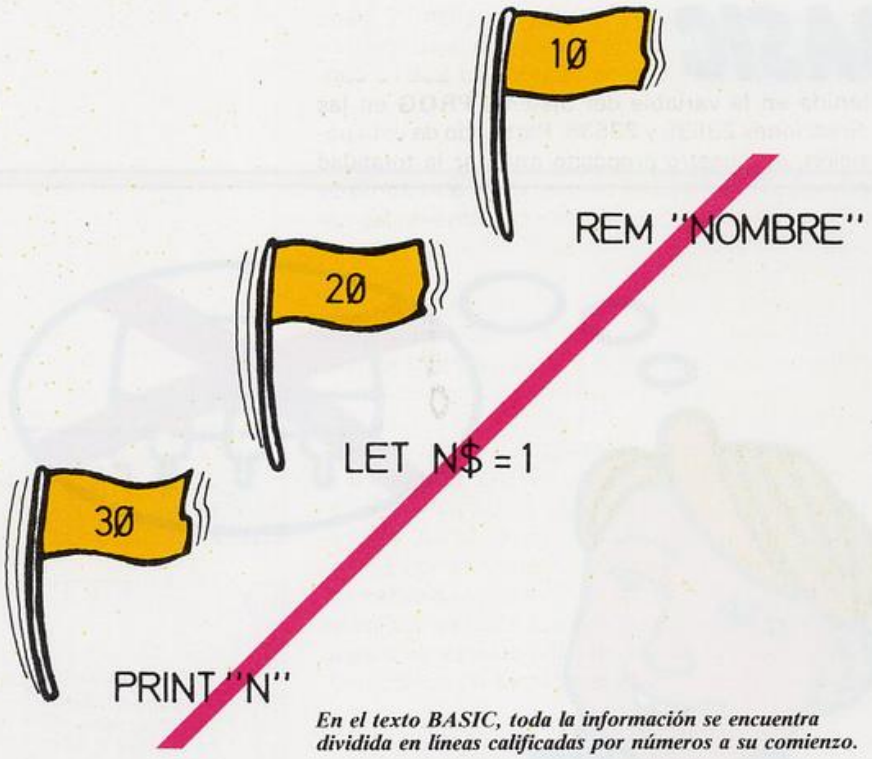
Una de las ventajas de los lenguajes de alto nivel sobre los ensambladores, es la despreocupación del programador en los asuntos internos de la máquina.

El texto BASIC se compone de palabras clave, símbolos, números, literales, nombres de variables y signos de puntuación.

EL TEXTO BASIC

Cuando listamos un programa, vemos aparecer una línea detrás de otra hasta completar una pantalla, y a continuación recibimos un mensaje del Sistema que nos interroga sobre si debe o no continuar con el listado *(scroll?)*.

Este listado nos es completamente inteligible, por lo cual podría parecer que está compuesto por los códigos ASCII de cada uno de los caracteres, como si de cualquier otro texto se tratara. Sin embargo, de ser así las cosas, el almacenamiento de programas BASIC requeriría gran cantidad de memoria.



En el texto BASIC, toda la información se encuentra dividida en líneas calificadas por números a su comienzo.

Por otra parte, no debemos olvidar que trabajamos bajo la tutela del intérprete BASIC, que al fin y al cabo es otro programa, por lo que requiere para el manejo de estas líneas una configuración adecuada a sus necesidades.
El texto BASIC se compone de: palabras clave, símbolos, números, literales, nombres de variables y signos de puntuación. Además, esto es evidente, toda la información se encuentra dividida en líneas calificadas por un número a su comienzo, el cual cumple a la vez la función de establecer la secuencia de ejecución de las mismas.
Internamente, cada línea de programa comienza con un número binario codificado en dos bytes. Este número es el encargado de mantener en orden la totalidad de líneas que componen el programa, correspondiendo el primer byte al más significativo y el segundo al menos significativo. Esta fórmula no es la habitual en la representación de variables enteras en Ensamblador, por lo cual podemos decir que se trata de una excepción. En cualquier caso, su interpretación la realizaremos multiplicando el valor decimal del primer byte por 256 y sumando al resultado el valor decimal del segundo byte.
Los dos siguientes bytes indican la longitud de la línea física, esta vez en el formato habitual de peso bajo y alto. Por último, inmediatamente después de la serie de bytes que componen el texto de la línea BASIC propiamente dicha, se encuentra un carácter 13 de retorno de carro (00001101 binario).

En realidad, el cometido de este último carácter de retorno de carro, aparentemente innecesario puesto que se especifica al comienzo de la línea la longitud en bytes de ésta, tiene la misión de facilitar los listados del programa.
Ya dentro del texto BASIC propiamente dicho debemos hacer algunas puntualizaciones. La primera de ellas es que las palabras clave BASIC son almacenadas en un solo carácter *(token)*, cuyos códigos ASCII corresponden a los números desde 165 a 255, aunque algunos de ellos ocupan códigos anteriores al 24.
De esta forma, además de conseguir un considerable ahorro de memoria, se facilita la labor del intérprete, el cual ha de evaluar un único byte para averiguar la palabra clave o símbolo de que se trata.
Otra puntualización consiste en la forma peculiar en que el BASIC trata los números contenidos dentro de la zona de texto. Estos números tienen una doble representación: una ASCII para mostrar en pantalla a través del listado, y otra codificada en 5 bytes binarios, manejada internamente por el intérprete.
Con este procedimiento se consigue una doble función: facilitar la acción del listado y la del intérprete. Sin embargo, como inconveniente debemos decir que estos valores numéricos tienen una ocupación de memoria elevada, puesto que además de los 5 bytes ocupados necesariamente por la representación binaria del número, con independencia de su magnitud, es necesario un carácter 14 como separador y tantos bytes extra como cifras tenga el número en cuestión. Por ejemplo, el número 5 ocupa 7 bytes y el 14516 11.

El texto BASIC tiene una dirección de comienzo común para los modelos de 16 y 48 Kbytes.

Los nombres de variables monocarácter, signos de puntuación, y literales alfanuméricos ocupan un byte de memoria.

Esta situación se produce tanto en el caso de asignaciones a una variable numérica por medio de **LET**, con o sin empleo de operadores matemáticos, como fruto de los argumentos de las sentencias **GO TO** y **GO SUB**. En cualquier caso, describiremos más adelante la composición de los 5 bytes que conforman la representación binaria del número.

El resto de los componentes del texto BASIC, tanto nombres de variables como signos de puntuación y literales alfanuméricos, ocupan un byte de memoria por cada carácter representado como si de un texto escrito se tratara.

Veremos todo esto más claro por medio de un ejemplo. Para ello, introduciremos el siguiente programa:

```
  10 REM - TEXTO BASIC (C) 1985 LOPEZ MA
RTINEZ
  20 DEF FN P(X)=PEEK X+256*PEEK (X+1)
  30 LET X=23635
  40 FOR I=FN P(X) TO FN P(X)+165
  50 PRINT PEEK I,(CHR$ PEEK I AND PEEK
I>31)
  60 NEXT I
```

Para empezar diremos que el texto BASIC tiene una dirección de comienzo común para los modelos de 16 y 48 Kbytes, la decimal 23813 contenida en la variable del Sistema **PROG** en las direcciones 23635 y 23636. Partiendo de esta posición, es nuestro propósito analizar la totalidad del texto. Para ello, al teclear **RUN** obtendremos en pantalla un listado a doble columna de los códigos y los caracteres representables de cada posición de memoria.

En cuanto a la forma de confeccionar el programa, debemos decir que la línea 20 incluye una definición de la función numérica **P(X)**, la cual permite interpretar el valor binario codificado en dos bytes consecutivos en el formato estándar.

En la línea 30 se asigna valor a **X** como la dirección decimal de la variable **PROG** apuntando al comienzo de la zona de texto BASIC; a continuación, en el bucle de las líneas 40 a 60, se produce la salida a pantalla del valor de los primeros bytes de texto con sus correspondientes representaciones como **CHR$** cuando su código es mayor que 31, para evitar los errores de *Invalid colour*, o impresión de caracteres de control (por debajo de 32).

Los primeros 43 bytes corresponden a la primera línea de programa: la sentencia **REM** de la línea 10.

Como podemos ver, los dos primeros bytes son un 0 y un 10, correspondientes a la representación en dos bytes binarios (peso alto-bajo) del número decimal 10, primera sentencia del programa.

Inmediatamente a continuación aparecen un 39 y un 0, indicativos del número de caracteres de que se compone la línea BASIC, codificados en binario esta vez en la forma estándar (peso bajo-alto). Esta cifra de caracteres es neta, es decir, no se incluyen ni los dos bytes del número de línea, ni los correspondientes a la longitud de ésta, aunque si se incluye el carácter 13 de final de línea.

El quinto byte de la línea es un *token*, precisamente el código decimal 234 correspondiente a

Las palabras clave BASIC son almacenadas en un solo carácter (token).

¡!

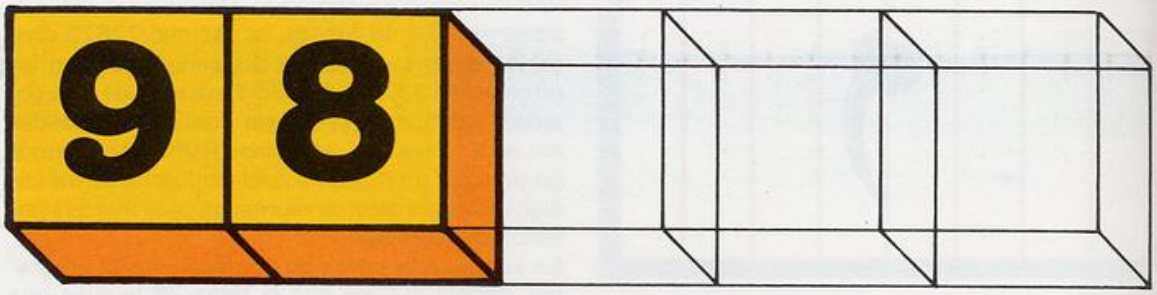
Para la decodificación del área de variables, debemos tener en cuenta que el BASIC de Spectrum permite la utilización de nombres multicaracteres para variables numéricas.

*

El *stack* de subrutinas se encuentra ubicada al final de la zona RAM utilizable por el BASIC, y tiene la peculiaridad de crecer en sentido inverso al texto BASIC o área de variables.

*

El *stack* de subrutinas es conocido también bajo el nombre de PILA DE RETORNOS, y contiene la información relativa a los retornos pendientes de subrutinas BASIC.



En el tratamiento BASIC de las variables numéricas de valor entero, sólo se emplean dos de los cinco bytes disponibles a tal fin.

la palabra clave **REM**. A partir de este punto, la secuencia de bytes coincide con la representación en pantalla de cada uno de los caracteres componentes de la línea de comentario; concluyendo ésta con un carácter 13 indicativo del final físico.

Los siguientes 42 bytes corresponden a la segunda línea de programa: la sentencia **DEF FN** de la línea 20.

De forma similar al caso anterior, los dos primeros bytes contienen un 0 y un 20 indicativos del número de línea, y los dos siguientes un 38 y un 0 correspondientes a la longitud de la línea física. Inmediatamente a continuación se encuentra un *token,* el de la palabra clave **DEF FN** (206) seguido por la transcripción literal de **P(X)**. Sin embargo, la representación de la variable **X** dentro del texto BASIC se hace ocupando 7 bytes: 88, 14, 0, 0, 83, 92 y 0.

El primer byte corresponde a la representación de la variable (88 de **X** ó 120 de x). El segundo es la marca de final de representación y comienzo de valor binario. De los cinco bytes que vienen a continuación, no se utilizan ni los dos primeros ni el último (son siempre 0), por darse el caso de que el valor de la variable a representar **(X)** es entero (en el rango –32768 a 32767); siendo el tercer y cuarto bytes en peso bajo-alto el número 23635.

A continuación se incluyen los caracteres = y el *token* de la palabra clave **PEEK** seguido por una **X** y el símbolo +. Inmediatamente la representación del entero 256 con los caracteres 2, 5, 6, 14 de marca de separación, y el valor codificado en binario en cinco bytes del entero en la forma: 0, 0, 0, 1, 0.

Como curiosidad, debemos notar que la representación del **(X+1)** del final de la sentencia 20 se hace con la X como un carácter normal y con el 1 en representación de número (siete bytes en este caso). Esto es debido a que este dato será evaluado en el momento de la interpretación de la sentencia.

La línea 30 está representada en la zona de texto BASIC por 13 bytes. Los cuatro primeros son: 0, 30, 15 y 0. Con ello, sabemos que nos encontramos en la línea 30 y que ocupa un total de 19 posiciones.

A continuación aparece un 241 (el *token* de la palabra clave **LET**) y la representación de los caracteres **X**= y del número entero 23635 (en once posiciones totales), para concluir la línea física con un carácter 13 como es habitual.

El resto del texto hasta la línea 60, no ofrecerá ya ninguna dificultad para su interpretación.

¡!

En el texto BASIC, la información se encuentra dividida en líneas calificadas por un número a su comienzo.

*

Conociendo la forma en que se almacena el texto BASIC, podemos acometer la confección de algún tipo de programa de «ayuda» a la programación BASIC.

*

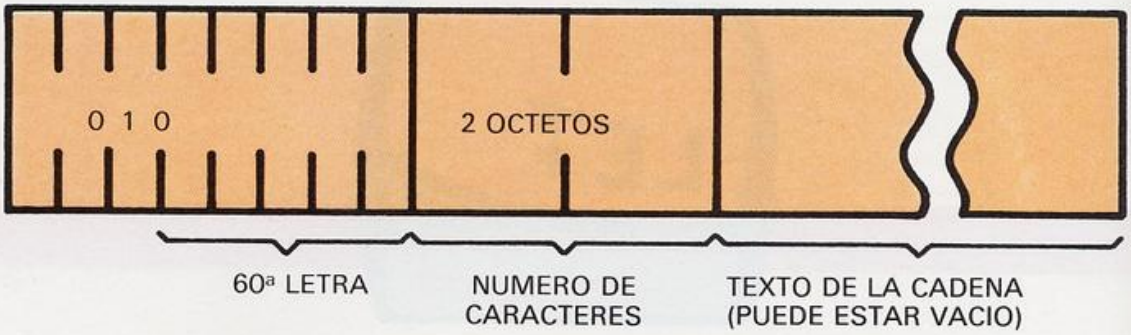
El área de variables se sitúa inmediatamente después de la zona de texto BASIC.

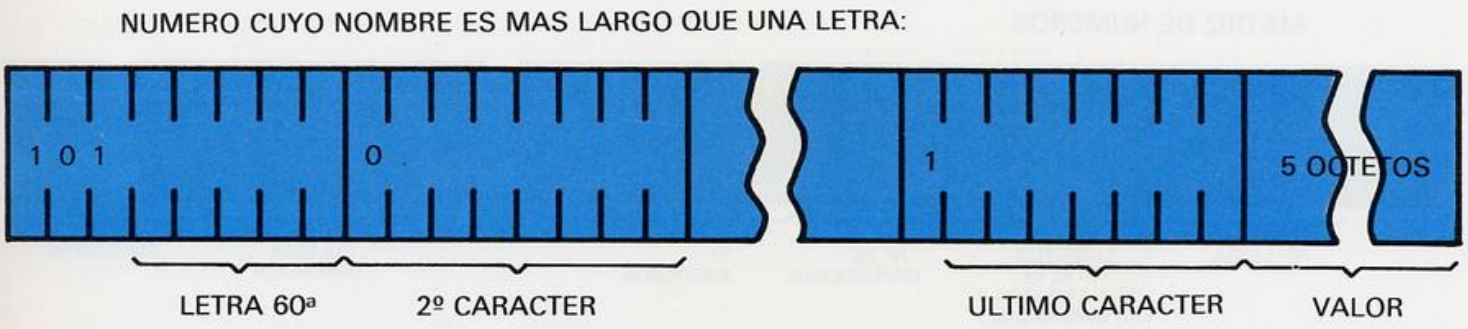
*

El conocimiento de la forma exacta en que el BASIC almacena su información (programa, variables, etc...) puede sernos de gran utilidad para el uso del código máquina.

CADENAS

Modelo de variable de cadena (string).

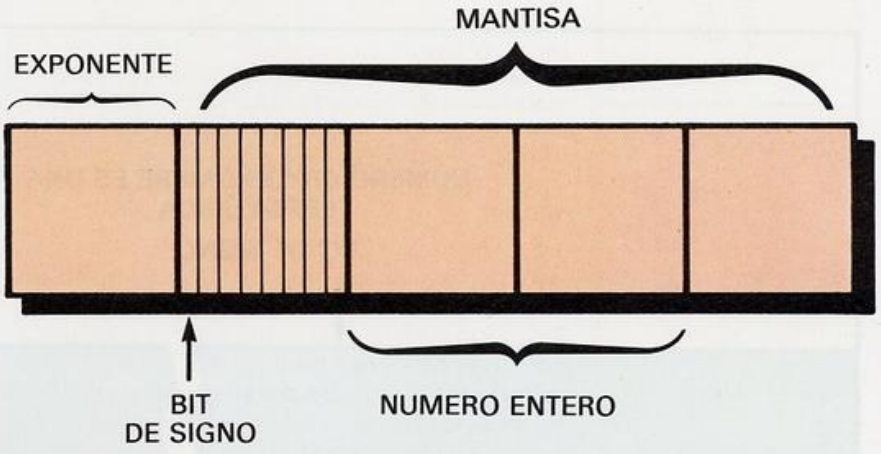




Modelo de variable numérica multicarácter.

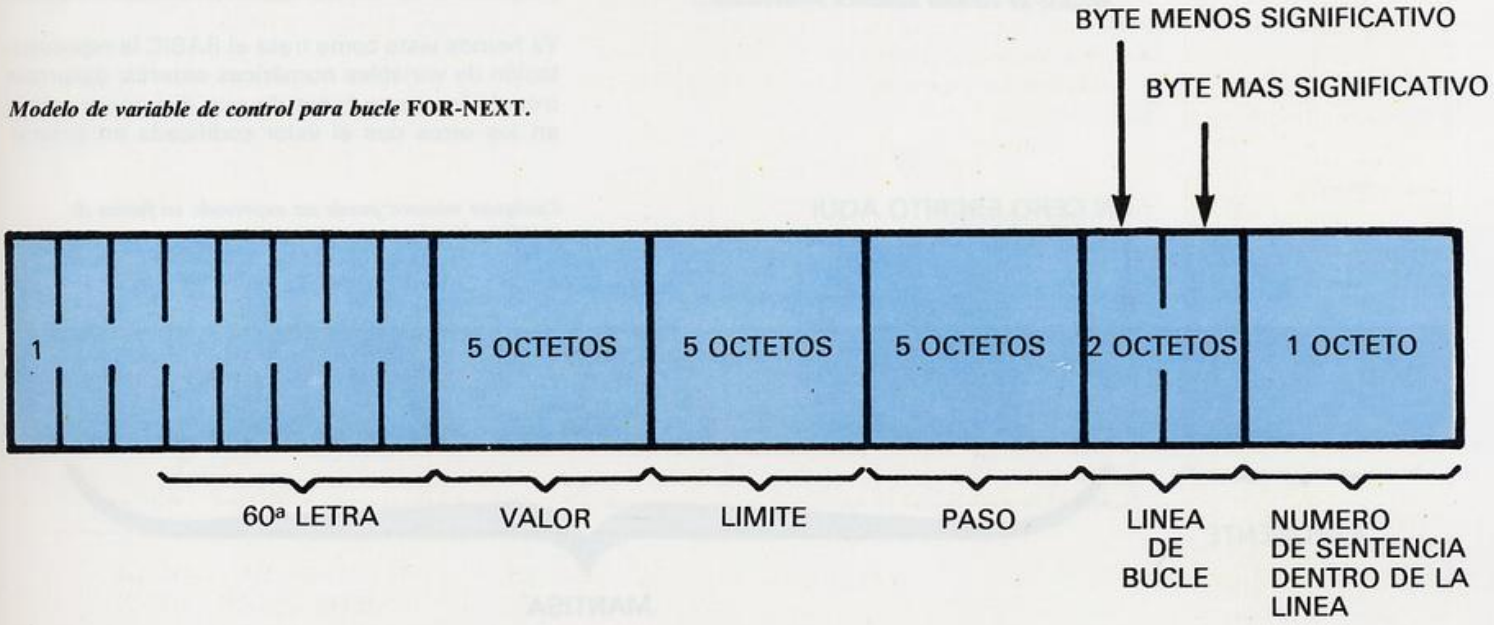
EL AREA DE VARIABLES

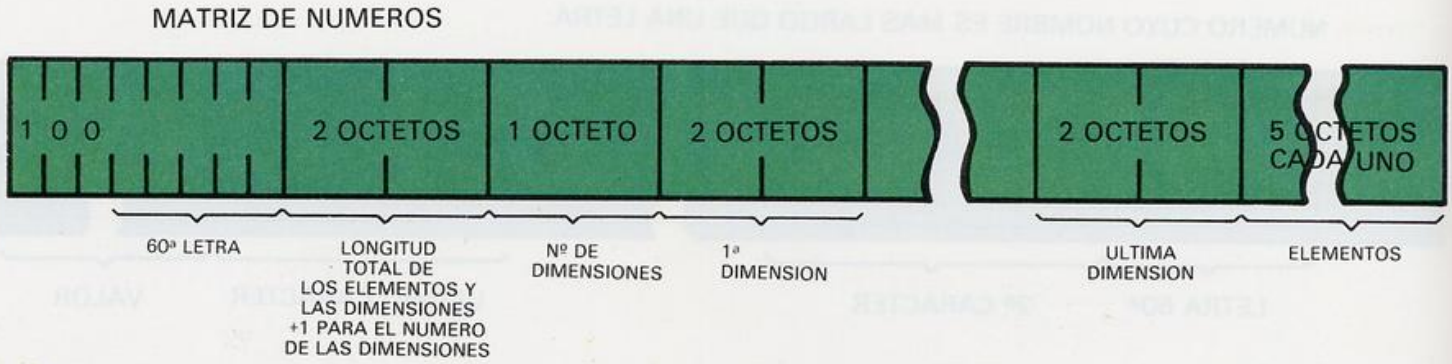
El área de variables se encuentra inmediatamente después de concluir la zona de texto BASIC. Su dirección de comienzo se almacena en una variable del Sistema denominada **VARS,** en las posiciones decimales 23627 y 23628. A partir de esta dirección, dependiente de la longitud del programa, van almacenándose los contenidos de las variables y matrices utilizadas por el BASIC. Un dato importante que debemos conocer en cuanto se refiere al nombre de las variables, y que recordaremos por capítulos anteriores, es que son consideradas iguales por el Sistema las escritas en mayúsculas o minúsculas. La varia-



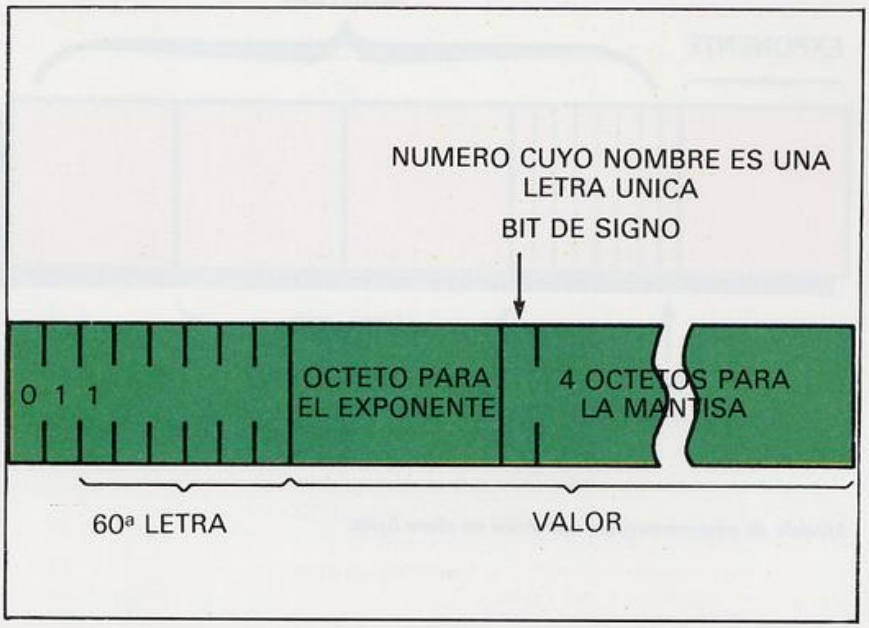
Modelo de almacenamiento numérico en cinco bytes.

Modelo de variable de control para bucle **FOR-NEXT.**





Modelo de matriz numérica.



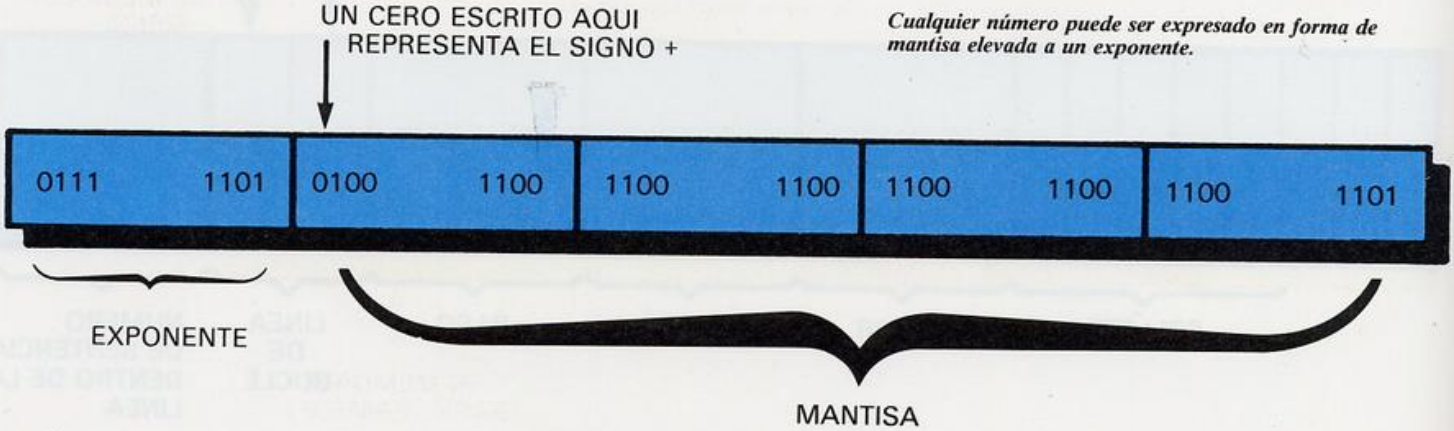
Modelo de variable numérica monocarácter.

ble X es pues idénticamente igual a x. Este hecho es importante si consideramos que permite emplear un total de 5 bits para la representación de los caracteres que componen su nombre.
Otro dato importante que debemos recordar es que el Sistema admite nombres de variables compuestos por más de un carácter para las variables numéricas. En cualquier caso, se modifica el contenido del primero del nombre de ésta forzando en sus tres bits más significativos ciertos contenidos, así como se modifica también el más significativo de los siguientes caracteres, si el nombre de la variable contiene más de uno. Estos bits son:

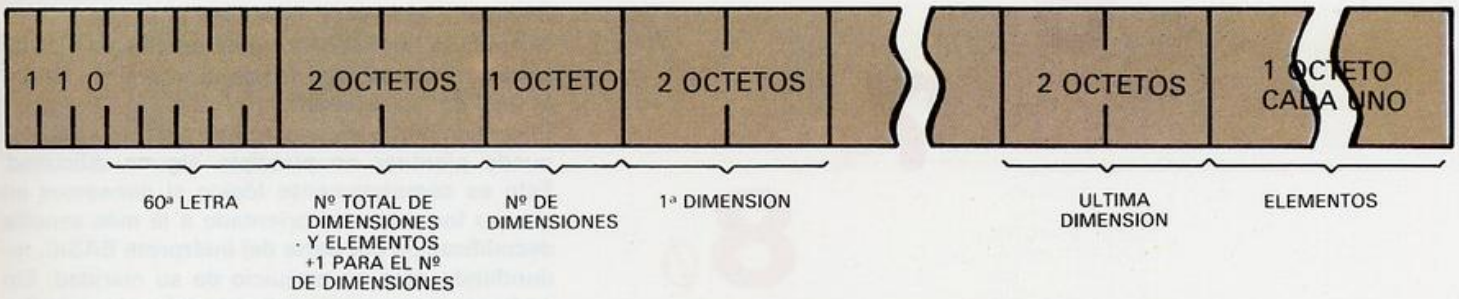
TIPO DE VARIABLE	BITS
Numérica (un carácter)	0 1 1
Numérica (más de uno)	1 0 1
Matriz numérica	1 0 0
Bucle **FOR NEXT**	1 1 1
Cadena	0 1 0
Matriz de cadena	1 1 0

Ya hemos visto como trata el BASIC la representación de variables numéricas enteras: desprecia tres de los cinco bytes de que dispone y coloca en los otros dos el valor codificado en binario.

Cualquier número puede ser expresado en forma de mantisa elevada a un exponente.



MATRIZ DE CARACTERES:



Modelo de matriz de caracteres.

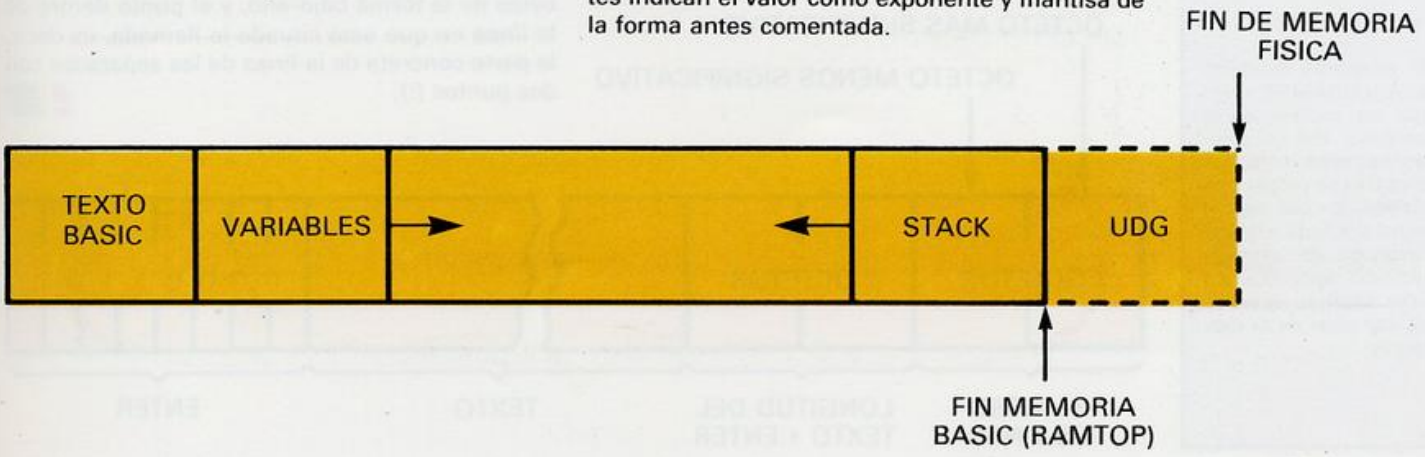
Pues bien, este es sólo un caso particular. El sistema general de codificación de este tipo de datos es el siguiente: en el primer byte se expresa el exponente y en los sucesivos la mantisa del número a almacenar, teniendo en cuenta que el primer bit del segundo byte, se emplea para indicar el signo, que será un cero para los números positivos y un uno para los negativos. Veamos unos ejemplos con el siguiente programa.

```
  10 REM - VARIABLES BASIC (C) 1985 LOPE
Z MARTINEZ
  20 DEF FN P(X)=PEEK X+256*PEEK (X+1)
  30 LET X=23627: LET W=123.45: LET VALO
R=X/3: DIM Y(2,2): LET Y(1,1)=W: LET X$=
"ABCDE": DIM Z$(3): LET Z$(2)=X$
  40 FOR I=FN P(X) TO FN P(X)+86
  50 PRINT PEEK I,(CHR$ PEEK I AND PEEK
I>31)
  60 NEXT I
```

El sistema de crecimiento del stack funciona en sentido opuesto al del área de texto y variables BASIC.

VARIABLES NUMERICAS

Los seis primeros bytes del área de variables contienen el valor de **X**. Su primer byte es un 120 por ser el código ASCII de la letra x minúscula. A continuación, en cinco bytes la representación de su valor entero de la forma que ya hemos comentado.
Los seis siguientes bytes representan a la variable real **W**. Su primer byte adopta la forma anterior. Los cinco que figuran a continuación se distribuyen de la siguiente manera: el primero indica el exponente, teniendo en cuenta que su primer bit corresponde al signo, y los otros cuatro componen la mantisa del número.
La siguiente variable **(VALOR)** es más larga, ocupando un total de 10 bytes. El primero de ellos es la inicial del nombre de la variable con sus tres primeros bits en la forma 1 0 1, a continuación los bytes de las letras A a la O con su bit más significativo a **0** y, por último, la letra R con el bit de control a 1. Las cinco posiciones siguientes indican el valor como exponente y mantisa de la forma antes comentada.



63 74 80

Cada línea de programa del texto BASIC comienza por un número binario codificado en dos bytes (alto-bajo).

¡!

El conocimiento de la estructura del área de variables puede plantear en principio alguna dificultad. No nos desanimemos por ello.

*

Un dato importante que debemos recordar es que los nombres de variables son considerados iguales, tanto si se escriben en mayúsculas como minúsculas, o combinaciones de ambas.

*

El programa de ejemplos VARIABLES efectúa un vaciado en la pantalla, del contenido del área de variables tras su propia ejecución, lo cual contribuirá sin duda a la clarificación de la información aportada en este capítulo a cerca de esa zona de la memoria.

La siguiente variable es la matriz numérica **Y(2,2)** que ocupa un total de 33 bytes. Su configuración es un poco más complicada. El primer byte comienza por los bits 1 0 0. Los dos siguientes bytes indican la longitud total de los elementos y las dimensiones, más 1 para las dimensiones. A continuación se encuentra un byte conteniendo el número de dimensiones (2), dos bytes para la primera dimensión, otros dos para la segunda y, por último, la representación ordenada de cada uno de los elementos en cinco bytes como es habitual.

A continuación encontramos la variable de cadena **X$**. Su primer carácter es el nombre, la letra x minúscula. A continuación, dos bytes contienen su número de caracteres (5), y seguidamente el contenido de la cadena formado por los caracteres de la A a la E.

Los 28 siguientes caracteres corresponden a la estructura de la matriz de cadena de caracteres **Z$**. Los tres primeros bits del nombre son 1 1 0, a continuación encontramos una serie de bytes de idéntica configuración a la descrita para las matrices numéricas y, por último, la lista de elementos ocupando un byte cada uno.

Como byte final encontramos un código 128 (80 hexadecimal), el cual indica al intérprete BASIC el final de las variables.

El conocimiento de la estructura de las variables puede plantear en principio alguna dificultad. Esto es completamente lógico si pensamos en que su formato está orientado a la más sencilla decodificación por parte del intérprete BASIC, redundando esto en perjuicio de su claridad. Sin embargo, nos permite integrar subrutinas en Ensamblador en los programas BASIC de forma que pueden transferirse variables de un lenguaje a otro, cosa muy importante para confeccionar determinados programas que requieren de la velocidad del código máquina.

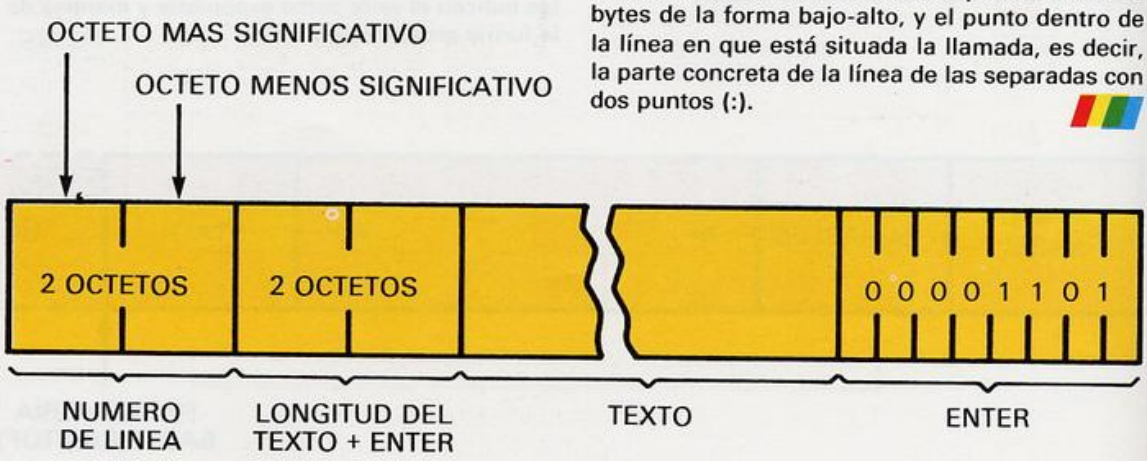
EL STACK DE SUBRUTINAS

Finalmente, vamos a comentar la estructura de una última zona de memoria utilizada por el BASIC: el *stack* de subrutinas. En él es en el que se almacenan las direcciones a las cuales debe retornar el BASIC al encontrar un **RETURN**, según el último **GO SUB** efectuado.

En primer lugar, hemos de decir que este área se encuentra ajustada al final de la memoria, y que su crecimiento es por tanto inverso al del área del programa y área de variables; es decir, los nuevos datos que se incorporan a la pila de retornos *(stack)* van ocupando direcciones de memoria cada vez más bajas.

Estos datos son muy simples: únicamente tres bytes, que contienen el número de línea desde el cual se efectuó el **GO SUB**, expresado en dos bytes de la forma bajo-alto, y el punto dentro de la línea en que está situada la llamada, es decir, la parte concreta de la línea de las separadas con dos puntos (:).

Modelo de línea BASIC.



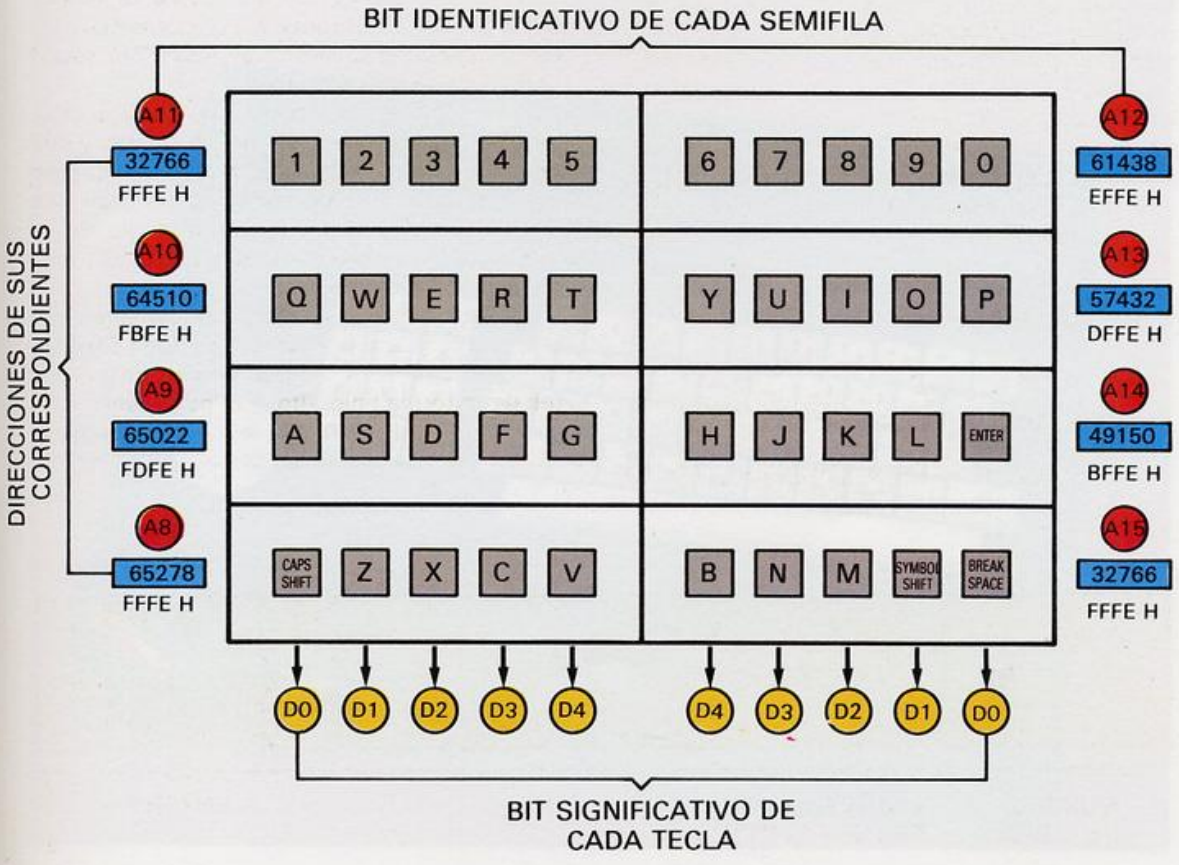
TECLA A TECLA

UNA de las razones fundamentales del éxito alcanzado por el ZX Spectrum es la pequeña inversión económica que el usuario debe realizar al adquirirlo. Hoy en día, por un precio realmente asequible, cualquiera puede tener acceso a uno de estos ordenadores, olvidando que hasta hace no muchos años, la informática quedaba reservada a un reducido grupo de técnicos especialistas en el tema.

Sin embargo, este abaratamiento lleva consigo algunos inconvenientes, como por ejemplo, el prescindir de detalles presentes en otro ordenador de mayor precio. Quizá el más destacado sea la ausencia de un teclado profesional, aunque como veremos a continuación, esta pequeña deficiencia puede ser suplida con algunos periféricos presentes en el mercado.

¿COMO FUNCIONAN?

La mayoría de los teclados que a continuación comentaremos, aparte de notables diferencias en cuanto a su forma de ser acoplados al Spectrum, siguen los mismos principios para que el microprocesador reconozca cual fue la tecla pulsada. Nuevamente, como ya es norma en el tratamiento de otros periféricos (casete, pantalla, etc.) es la U.L.A. la encargada de controlar las señales procedentes del teclado, pues lo considera como un equipo más, exterior al sistema. En su parte



Distribución lógica de las teclas en el teclado del Spectrum.

¡!

La U.L.A. es la encargada de controlar las señales procedentes del teclado, pues es considerado como un equipo exterior del Sistema.

*

En las direcciones que maneja el teclado, el primer bit (A0) es siempre cero y el resto unos, excepto los bits A8 y A15 cuyo valor variará al pulsar o no una tecla.

eléctrica, este consiste en una matriz de contactos de 5 filas por ocho columnas.
Es decir, cada vez que nosotros pulsamos una tecla cerramos uno de estos circuitos generándose dos señales, una correspondiente a la columna en la cual se encuentra la tecla, y otra a la fila, de manera que sea del todo imposible la aparición de errores a causa de una posible ambigüedad.
Esta señales llegan al circuito impreso del Spectrum a través de dos tiras plásticas que refuerzan la estructura de las pistas metálicas, yendo a acloparse en dos conectores preparados a tal efecto, uno para las filas (el de cinco terminales) y el otro para las columnas (el de ocho), según refleja la figura.
En otros teclados diferentes al de gomas de los primeros modelos de Spectrum, por ejemplo, en el del Spectrum +, las restantes teclas no son otra cosa que contactos en paralelo con los originales, con objeto de elevar la operatividad de algunas funciones de uso constante, como los cursores, el modo extendido o la edición **(EDIT)**.
Pero veamos cómo son interpretados los impulsos procedentes del teclado. En la figura se muestra la distribución lógica de las teclas, donde cada una de las filas se fracciona por la mitad. A cada semifila se le asigna una dirección en el bus del mismo nombre.
Como ya sabemos el bus de direcciones del Spectrum está formado por 16 líneas, circulando simultáneamente por cada una de ellas un bit de información. Las direcciones que maneja el teclado son aquellas cuyo primer bit A0 (el de menor peso), es siempre cero y el resto unos, excepto los bits A8 a A15 cuyo valor variará que pulsemos o no alguna tecla (pulsación=0, reposo=1).
Son algunas rutinas almacenadas en la ROM quienes constantemente (50 veces cada segundo), rastrean el teclado en busca de la información necesaria. Para ello, cuando detectan que alguno de los bits comprendidos entre el A8 y el A15 ha cambiado a 0, interpretan tal dirección, determinando en qué semifila se encuentra la tecla pulsada.
Ahora, es necesario descubrir cual o cuales de ellas en concreto están provocando tal alteración. A cada contacto correspondiente a una tecla le está asignado un bit, y puesto que cada semifila contiene cinco, los fabricantes les han asociado los bits D0 a D4 de un byte que contendrá la información precisa, y a través del bus de datos, llegará al microprocesador para ser interpretado.
El valor que tomen los tres restantes no nos interesa en absoluto, y según los modelos se obtienen resultados diferentes. Además cambian al ejecutar sentencias **BORDER** o **BEEP**. En la sec-

¡!

Los teclados LO-PROFILE y SAGA 1 EMPEROR proporcionan la alternativa de dotar a nuestro Spectrum de un aspecto profesional. Su conexión es sencilla, basta desprender la parte superior de nuestro micro y ajustar las tiras del nuevo teclado en el interior de éste.

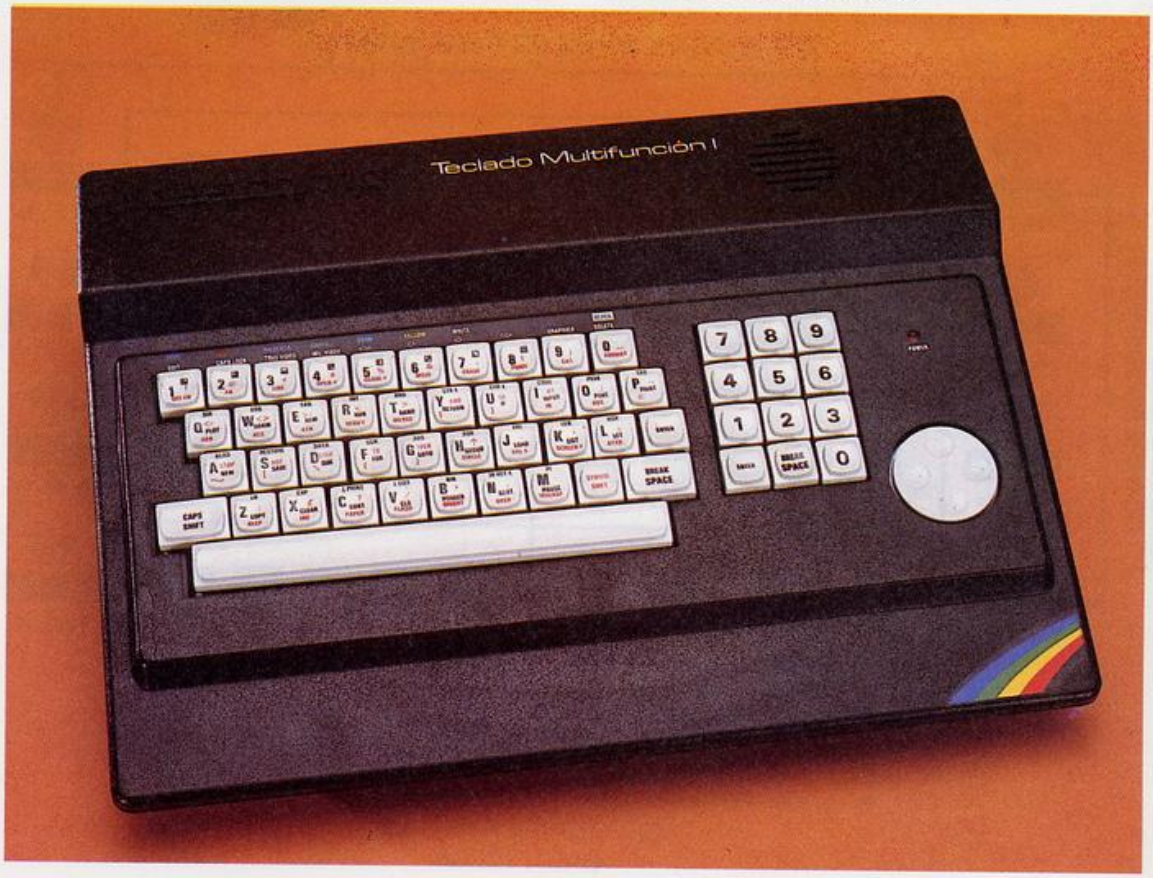
*

Para la conexión del nuevo teclado MULTIFUNCION 1 no es necesariio desmontar el Spectrum ni el INTERFACE 1 para alojarlos en su interior.

*

Las averías más frecuentes en lo referente al teclado son: el fallo de las teclas, y el borrado de las leyendas.

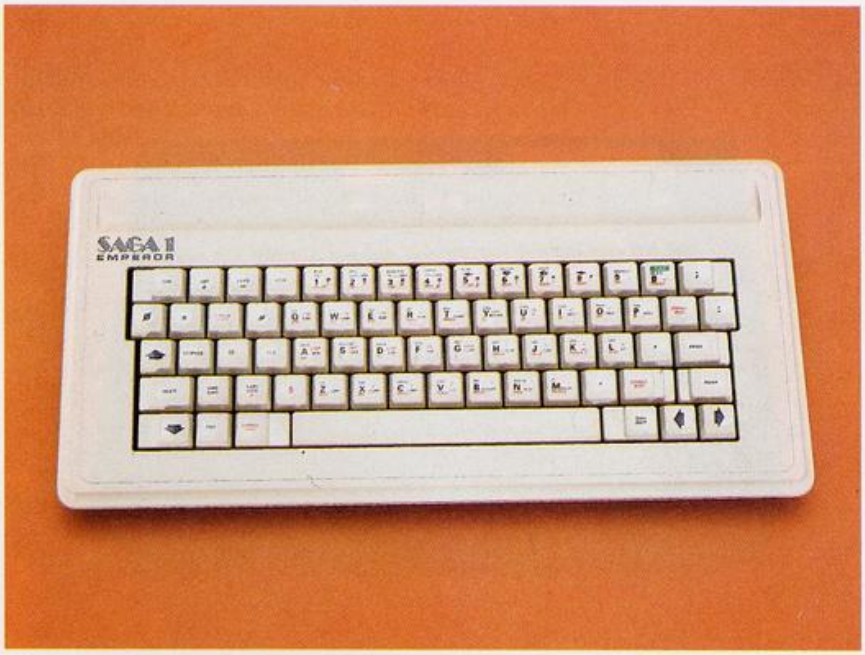
El nuevo MULTIFUNCION 1 subsana los errores cometidos en su anterior versión.



ción de BASIC podemos encontrar más información al respecto.
Una vez sentada esta base general, pasemos al análisis pormenorizado de los diversos teclados profesionales disponibles en el mercado.

SOLUCIONES ECONOMICAS

Dos son los tipos de averías más frecuentes entre los micromaníacos en lo referente a su teclado: las teclas comienzan a fallar, y las leyendas de éstas se borran de tal manera que es necesario convertirse en vidente para reconocerlas.
Si el problema es el segundo, y nos encontramos cortos de presupuesto, el ROYAL TOUCH, distribuido por INVESTRONICA, resulta ser una solución adecuada. El formato es idéntico al de el Spectrum de 40 teclas, con la salvedad de que éstas han sido fabricadas en un material más duro que las gomas primitivas.
No tiene membrana de contactos propia, por lo cual conviene asegurarse de si el defecto proviene de ella, pues en tal caso, la sustitución por este pseudo-teclado será completamente inútil. Entonces, encontraremos que el problema es el primero señalado anteriormente, y si andamos escasos de presupuesto, nos deberemos conformar con adquirir un teclado idéntico al averiado.



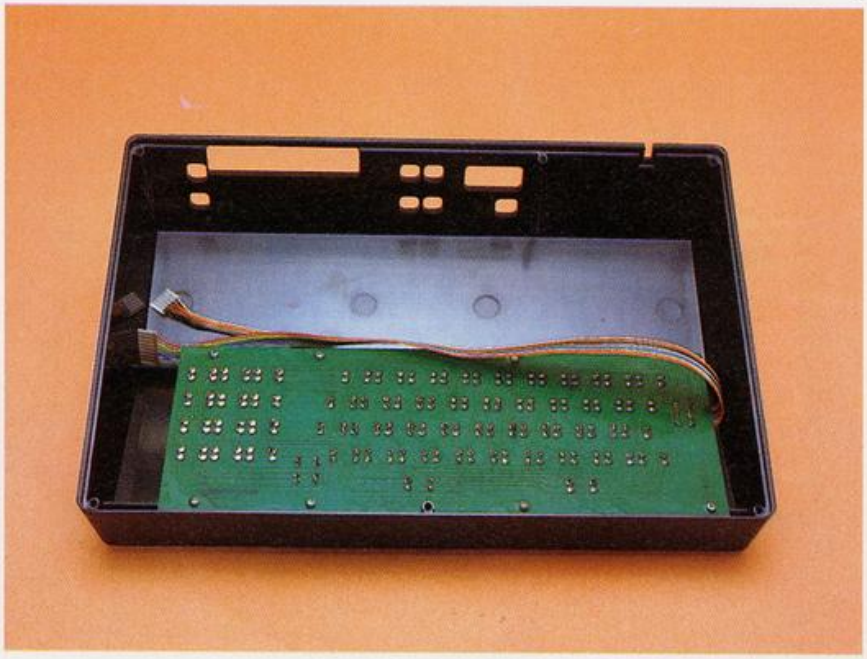
El SAGA 1 EMPEROR es posiblemente la mejor opción disponible actualmente en el mercado.

ciso desmontar el Spectrum completamente, extrayendo la placa del circuito impreso, y los usuarios del INTERFACE 1, deberán hacer lo mismo para ajustarla en su interior, donde una pequeña placa actúa de prolongador para acoplar el Microdrive.

El R. TOUCH es una solución económica a nuestros problemas de teclado.

DK'TRONICS

Distribuido por SILOG, este modelo es válido tanto para el Spectrum como para el ZX-81, y constituye una buena alternativa para aquel que desea sustituir su viejo teclado sin gastar mucho dinero, siendo además la relación precio/calidad, más que aceptable. Se encuentra dividido en dos zonas: en la primera se reproducen las funciones habituales del ordenador a las cuales se ha añadido la barra espaciadora duplicando la tecla **SPACE**. El segundo bloque es numérico, donde encontramos además, una tecla **DELETE** y el punto (.).
Para su conexionado y puesta en servicio es pre-



Las señales del teclado llegan al Spectrum a través de dos cables planos.

Un detalle importante consiste en poder adaptar en el interior la fuente de alimentación; si bien no están preparados anclajes especiales, un trozo de gomaespuma y algún adhesivo la fijarán firmemente.
Por otra parte, las teclas no están serigrafiadas, pero se suministra una plantilla de donde recoger las etiquetas de cada una de ellas, y posteriormente adherirlas en su lugar correspondiente. Mucho cuidado en este aspecto, pues no debemos guiarnos por la fotografía de la caja, sino por la distribución dada en el folleto de instrucciones.

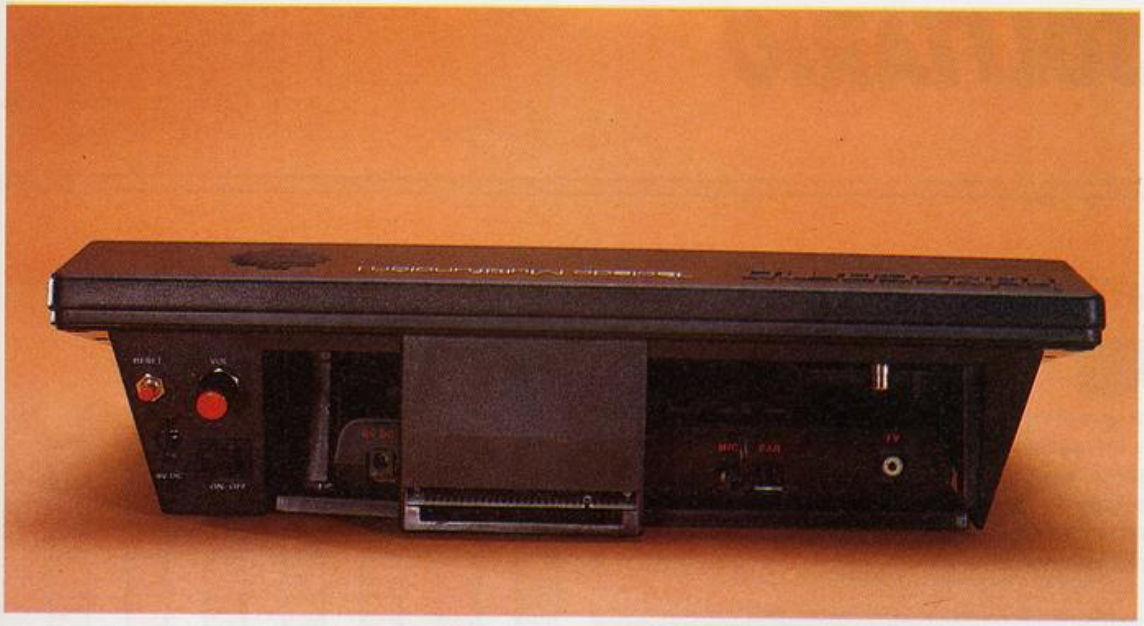
El teclado de DK'TRONICS es una opción a tener en cuenta si no se desea una gran inversión de dinero.

CONVERSION EN +

El teclado del ZX Spectrum +, incorpora notables ventajas sobre el de su predecesor, pues no en vano, ha sido diseñado teniendo en cuenta las deficiencias de éste. El número de teclas ha aumentado en 18, con el objeto de conseguir mediante una sola pulsación funciones para las que antes eran necesarias dos, como **EDIT, EXTEND MODE,** etc.
La ventilación ha sido mejorada, gracias al mayor tamaño de la carcasa y las nuevas rejillas a tal fin. Dos patas plegables optimizan la orientación del teclado y en uno de los latereales encontramos un pulsador de *RESET,* útil al menos para evitar el constante enchufar y desenchufar de la fuente de alimentación cada vez que perdemos el control sobre el ordenador.
En aproximadamente una hora, podemos transformar nuestro Spectrum en un plus, tan solo con seguir las sencillas instrucciones expuestas en el cuadernillo que acompaña al kit.

LA ALTERNATIVA PROFESIONAL

Dos teclados ante los cuales es imposible pasar con indiferencia, proporcionan la alternativa de dotar a nuestro Spectrum de un aspecto profesional. Su conexión es sencilla, bastando desprender la parte superior de nuestro micro y acoplar la restante, donde se ajustan las tiras del nuevo teclado, en el interior de éste.
El LO-PROFILE destaca sobre todo por la enorme superficie que ocupa (tres veces la del Spectrum), estando divididas sus 53 teclas en dos bloques: en el primero, la distribución de estas es idéntica a las del Spectrum, más una barra espaciadora y



A la hora de elegir un teclado profesional, debemos tener muy en cuenta la necesidad de que éste no impida la conexión de ningún otro periférico.

una supuesta **CAPS LOCK** duplicada que no es sino 2, pues habrá que pulsar ambas, **CAPS SHIFT + CAPS LOCK** para conseguir el modo mayúsculas. El segundo es numérico al que se suman otra tecla **CAPS LOCK** y el punto (.). Todas las teclas están serigrafiadas con sus leyendas correspondientes.

El SAGA 1 EMPEROR distribuido por Silog es posiblemente la mejor opción disponible hasta el momento, para todos aquellos que deseen utilizar su teclado en aplicaciones más serias que masacrar marcianitos.

Aunque forma un bloque compacto, en la zona central están distribuidas en idéntica disposición a las teclas del Spectrum, mientras que en los dos laterales se multiplican las funciones de uso más corriente como **RUN**, SAVE, etc. Además, otras pseudoteclas como **CAPS LOCK, DELETE, EDIT** y algunas más están duplicadas por el teclado, pero serán precisas en estos casos dos pulsaciones para conseguirlas.

Dos juegos de plantillas se suministran para ser adheridas sobre las teclas, pues las 67 de que consta no están serigrafiadas. La calidad del teclado es elevada, y aunque al principio puedan encontrarse algunas dificultades de adaptación, la suavidad y precisión con que responde justifican su elección.

Recientemente, el mismo fabricante ha anunciado, la próxima comercialización del SAGA-3 ELITE, teclado superprofesional compatible con el Spectrum. Funciones definidas y teclado numérico independientes hasta completar un total de 80 pulsadores, pueden convertirlo en breve en la más atrayente oferta que podamos encontrar en el mercado.

UNA NUEVA VERSION

Distribuido por LSB, el nuevo teclado MULTIFUNCION 1 de INDESCOMP, tiene poco que ver con su predecesor de igual nombre. Se han efectuado las mejoras pertinentes para que el teclado sea precisamente eso, un teclado, y no una espectacular caja de desagradables sorpresas. Por tanto, es necesario tener cuidado para no confundirlo con el modelo antiguo, pues como vemos se comercializa con el mismo nombre.

Su punto más fuerte quizá resida en que no es necesario desmontar el Spectrum ni el INTERFACE 1 para alojarlos en el interior del teclado, pues la conexión se realiza a través del bus de expansión, de donde la unidad toma los datos para su correcto funcionamiento.

Como ha quedado expuesto, podemos encontrar teclados para todos los gustos y economías. Pero la elección debe efectuarse cuidadosamente y tener presente las garantías ofertadas por los fabricantes, más que la propaganda que de los distintos modelos se realiza. Así evitaremos encontrar la desagradable sorpresa de que más que un teclado, hemos adquirido una bonita caja con las teclas saltarinas.

¡!

Podemos transformar nuestro Spectrum en un Plus, tan solo con seguir las sencillas instrucciones que acompañan al kit.

*

La mayoría de los teclados siguen los mismos principios para que el microprocesador reconozca cuál es la tecla pulsada.

SOLITARIO

¡!

Los caracteres subrayados en el listado corresponden a los gráficos definidos de usuario.

*

Para que tengamos noción del número de cartas que han sido barajadas disponemos de un contador de naipes.

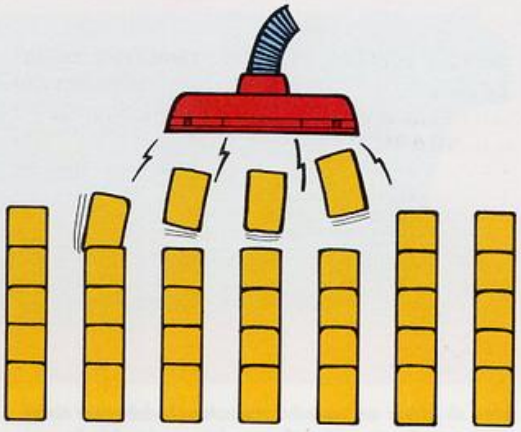
*

Si una pulsación de tecla (elección de carta) no es aceptable por las reglas del juego, el Spectrum responderá con un zumbido.

*

Para el almacenamiento del programa utilizaremos el siguiente comando directo: **SAVE "SOLITARIO"**. Si optamos por el modo de grabación con autoejecución teclearemos la siguiente orden: **SAVE "SOLITARIO" LINE 10.**

Es frecuente acudir al ordenador con la pretensión de que éste nos resuelva nuestros problemas de aburrimiento. La inmensa proliferación de programas en este sentido ha hecho abandonar a los usuarios de microordenadores los clásicos solitarios de cartas con que hasta entonces aplacaban su tedio. Este programa representa una atractiva combinación de los dos pasatiempos: el clásico y el informático.

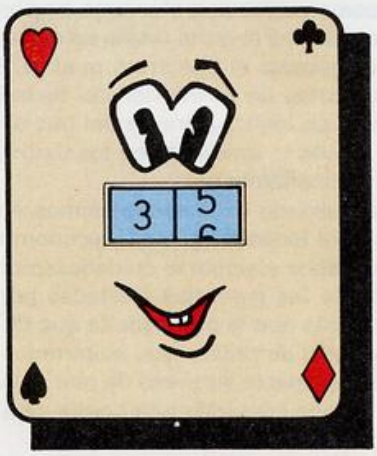


El juego consiste en retirar las siete columnas de cinco cartas que se distribuyen al principio.

EL PROGRAMA

Nuestro Spectrum utiliza la clásica baraja francesa de 52 cartas (picas, diamantes, corazones y tréboles). El juego consiste en retirar las siete columnas de cinco cartas que se distribuyen al principio, quitando los naipes de uno en uno, después de que nuestro Spectrum efectúe el barajado correspondiente.

El mezclado de las cartas nos lo muestra nuestro querido ordenador en la pantalla de un modo bastante peculiar. El dorso de los 52 naipes de la baraja irán surgiendo de uno en uno, a medida que el mazo de cartas se vaya desordenando. Para que tengamos noción del número exacto de cartas que han sido descolocadas, disponemos de un contador de naipes en el ángulo inferior derecho de la pantalla.



Una vez que el contador ha llegado a 52, automáticamente el Spectrum nos presentará en pantalla 7 columnas de 5 cartas cada una, y a la derecha de estas otra carta más y el resto del mazo, indicando en la parte inferior de la pantalla el número de naipes que hay en el mismo.

Para ir retirando los naipes sólo hay que tener en cuenta dos normas:

1. La carta a retirar es siempre la última de la columna que se ha escogido.
2. Para que el naipe pueda ser retirado, es necesario que sea el inmediatamente anterior o posterior a la carta descubierta que figura boca arriba en el lado derecho, encima del mazo, no influyendo para nada el palo al cual pertenezca. Es decir, la carta que se puede quitar es la que figura a la izquierda o derecha de la descubierta en la siguiente cadena: A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K, A.

Una vez retirada la carta, ésta pasa a ser la que se sitúa boca arriba encima del mazo; esto es: aquella con la cual hay que comparar la siguiente.

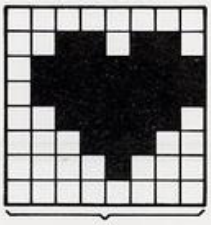
Si ninguno de los naipes últimos de columna puede ser retirado, hay que recurrir a extraer una carta del mazo, pulsando «0». Cuando en esa circunstancia no queden más cartas en el montón, pulsaremos de nuevo la tecla «0» y nuestro Spectrum informará del número de cartas que han quedado sin retirar.

Como es habitual, para la sustitución de los gráficos en el listado se ha empleado la notación de subrayado. El subrayado simple indica el gráfico

de la tecla, y el doble, el gráfico cambiado; es decir, el que se obtiene pulsando simultáneamente la tecla afectada y **CAPS SHIFT**, lógicamente en el modo **GRAPHICS** (cursor G).

```
 10 REM ************************
 20 REM * SOLITARIO      1985 *
 30 REM ************************
 40 REM  INICI. VARIABLES
 50 BORDER 1: PAPER 1: INK 9
 60 POKE 23658,8: CLS
 70 LET FILA=0
 80 LET COLUM=0
 90 LET SW=1
100 RANDOMIZE
110 LET TI=0
120 DIM A(6,7)
130 DIM b(17)
140 LET B(1)=17
150 LET B$=""
160 LET A$=""
170 FOR i=1 TO 52
180 LET a$=a$+CHR$ i
190 NEXT I
200 DATA 0,54,127,127,62,28,8,0
210 DATA 0,0,28,62,127,62,28,8
220 DATA 0,8,28,42,127,42,8,28
230 DATA 0,8,28,62,127,42,8,28
240 DATA 0,70,201,73,73,73,230,0
250 DATA 219,189,102,195
260 DATA 195,102,189,219
270 REM  GRAFICOS USUARIO
280 FOR I=144 TO 149
290 FOR J=0 TO 7
300 READ A
310 POKE USR CHR$ I+J,A
320 NEXT J
330 NEXT I
340 PAPER 1: BORDER 1: INK 9
350 CLS
360 REM  INICIO PANTALLA 1
370 PRINT #0; FLASH 1; INK 6; PAPER 2;"    "; FLASH
0; PAPER 1;"   "; FLASH 1; INK 2; PAPER 6;"    BARAJA
NDO    "; FLASH 0; PAPER 1;"   "; FLASH 1; PAPER 2; I
NK 6;"  "
380 PRINT  INK 5;AT 16,23;"CARTAS"
390 REM  BARAJADO CARTAS
```

CORAZON

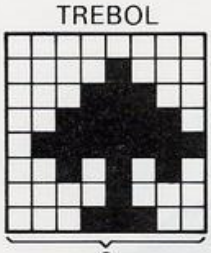


A

DIAMANTE

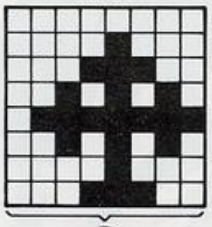
B

TREBOL



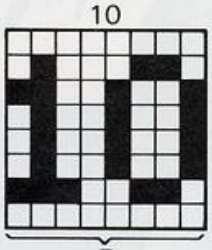
C

PICA



D

10



E

REVERSO CARTA

F

Representación de los seis gráficos definidos empleados para el programa.

```
400 FOR I=1 TO 52
410 LET W=INT (RND*52)+1
420 IF CODE A$(W)<>0 THEN  GO TO 470
430 LET W=W+1
440 IF W=53 THEN  LET W=1
450 BEEP .01,40
460 GO TO 420
470 LET B$=B$+CHR$ W
480 LET A$(W)=CHR$ 0
490 PRINT  INK 9;AT 18,25;I
500 BEEP .1,45
510 IF SW THEN  LET PA=2: LET SW=0: GO TO 540
520 LET SW=1
530 LET PA=4
540 GO SUB 2140
550 NEXT I
560 PRINT  FLASH 1; OVER 1;AT 18,25;"  "
570 REM  PROCESO TRANS
580 LET A=0
590 FOR I=1 TO 7
600 FOR J=2 TO 6
610 LET A=A+1
620 LET A(J,I)=CODE B$(A)
630 NEXT J
640 NEXT I
650 FOR I=1 TO 7
660 LET A(1,I)=6
670 NEXT I
680 LET A=A+1
690 LET B=CODE B$(A)
700 FOR I=2 TO 17
710 LET A=A+1
720 LET B(I)=CODE B$(A)
730 NEXT I
740 REM  EXPOSICION CARTAS
750 REM    (PANTALLA 2)
760 CLS : PRINT "  ";: FOR i=1 TO 7: PRINT STR$ i;"
  ";: NEXT i: PRINT
770 FOR I=2 TO 5
780 PRINT
790 FOR J=1 TO 7
800 LET A=A(I,J)
810 GO SUB 1840
820 PRINT " "; PAPER 6;b$;"  ";
830 NEXT j
840 PRINT
850 FOR J=1 TO 7
860 LET A=A(I,J)
870 GO SUB 1840
880 PRINT " "; PAPER 6;a$;"  ";
890 NEXT J
900 PRINT
910 FOR j=1 TO 7
920 PRINT  INK 0;" "; PAPER 6;"___";
930 NEXT J
940 NEXT i
950 PRINT
960 FOR J=1 TO 7
970 LET A=A(6,J)
980 GO SUB 1840
990 PRINT " "; PAPER 6;b$;"  ";
1000 NEXT J
1010 PRINT
1020 FOR J=1 TO 7
1030 LET A=A(6,J)
1040 GO SUB 1840
1050 PRINT " "; PAPER 6;a$;"  ";
1060 NEXT J
1070 PRINT
1080 FOR J=1 TO 7
1090 PRINT " "; PAPER 6;"    ";
1100 NEXT J
1110 PRINT
1120 FOR J=1 TO 7
1130 LET A=A(6,J)
1140 GO SUB 1840
1150 PRINT " "; PAPER 6;"  ";a$;
1160 NEXT J
1170 PRINT
1180 FOR J=1 TO 7
1190 LET A=A(6,J)
1200 GO SUB 1840
1210 PRINT " "; PAPER 6;b$;"  ";
1220 NEXT J
1230 PRINT AT 21,1; INK 6;"Cartas en el monton: 16";
PAPER 7
1240 GO SUB 1920
1250 GO SUB 1970: GO SUB 2030
1260 REM  CONTROL TECLADO
1270 IF INKEY$<>"" THEN  GO TO 1270
1280 LET A$=INKEY$
1290 IF A$<"0" OR A$>"7" THEN  GO TO 1280
1300 REM  PROGRAMA PRINCIPAL
1310 LET C=VAL A$
1320 IF NOT C THEN  GO TO 1680
1330 IF A(1,C)=1 THEN  BEEP 1,0: GO TO 1270
1340 LET A=A(A(1,C),C)
1350 GO SUB 1840
1360 LET D=W
1370 LET A=B
1380 GO SUB 1840
1390 IF D=13 AND W=1 THEN  GO TO 1420
1400 IF D=1 AND W=13 THEN  GO TO 1420
1410 IF ABS (D-W)<>1 THEN  BEEP 1,0: GO TO 1270
1420 BEEP .25,30
1430 LET B=A(A(1,C),C)
1440 GO SUB 1920
1450 LET A(1,C)=A(1,C)-1
1460 LET D=(A(1,C)-1)*3+1
1470 LET A=A(A(1,C),C)
1480 GO SUB 1840
1490 LET C=(C-1)*4+1
1500 IF D=1 THEN  FOR I=2 TO 6: PRINT AT I,C; PAPER 1
;"    ": NEXT I: GO TO 1540
1510 PRINT  PAPER 6;AT D,C;"   ";AT D+1,C;"  ";A$;AT
D+2,C;"  ";B$
1520 PAPER 1: PRINT AT D+3,C;"   ";AT D+4,C;"   ";AT
D+5,C;"  "
1530 PAPER 6
1540 FOR I=1 TO 7
1550 IF A(1,I)<>1 THEN  GO TO 1270
1560 NEXT I
1570 PAPER 4
1580 INK 0
1590 PRINT AT 11,3;"FELICIDADES, LO CONSIGUIO!"
1600 PRINT  PAPER 7; INK 2;AT 13,1;" QUIERES INTENTAR
LO OTRA VEZ? "
1610 PRINT  PAPER 1;AT 21,0;"
                             "
1620 LET A$=INKEY$
1630 IF A$="S" THEN  BEEP .1,40: RUN
1640 IF A$="N" THEN  GO TO 10000
1650 BEEP .01,35
1660 BEEP .005,40
1670 GO TO 1620
1680 BEEP .25,30
1690 IF B(1)=1 THEN  GO TO 1770
1700 LET B=B(B(1))
1710 GO SUB 1920
1720 LET B(1)=B(1)-1
1730 GO SUB 1970
1740 IF B(1))1 THEN  GO SUB 2030
1750 PRINT  PAPER 1;AT 21,22; INK 6;B(1)-1;" "
1760 GO TO 1270
1770 LET C=0
1780 FOR I=1 TO 7
1790 LET C=C+A(1,I)
1800 NEXT I
1810 LET C=C-7
1820 PAPER 2: INK 7: PRINT AT 11,2;"  Te han quedado
";c;" cartas  "
1830 GO TO 1600
1840 LET W=INT ((A-1)/13)
1850 LET A$=("A" AND NOT W)+("B" AND W=1)+("C" AND W=
2)+("D" AND W=3)
1860 INK 2
1870 IF W>1 THEN  INK 0
1880 LET W=A-W*13
1890 IF W<10 AND W<>1 THEN  LET B$=STR$ W: RETURN
1900 LET B$=("E" AND W=10)+("J" AND W=11)+("Q" AND W
=12)+("K" AND W=13)+("A" AND W=1)
1910 RETURN
1920 LET A=B
1930 GO SUB 1840
1940 PRINT AT 5,29; PAPER 1;"___"
1950 PRINT AT 6,29; PAPER 6; BRIGHT 1;b$;"  ";AT 7,29
;a$;"  ";
1960 PRINT  PAPER 6; BRIGHT 1;AT 8,29;"   ";AT 9,29;"
  ";a$;AT 10,29;"  ";b$: RETURN
1970 PAPER 1
1980 FOR I=12 TO 17
1990 PRINT AT I,28;"    "
2000 NEXT I
2010 PAPER 7
2020 RETURN
2030 IF SW THEN  LET PA=2: LET TI=0: LET SW=0: GO TO
2070
2040 LET SW=1
2050 LET PA=4
2060 LET TI=0
2070 PRINT  PAPER PA; INK TI; BRIGHT 1;AT 12,28;"4337
"
2080 PRINT  PAPER PA; INK TI; BRIGHT 1;AT 13,28;"5FF5
"
2090 PRINT  PAPER PA; INK TI; BRIGHT 1;AT 14,28;"5FF5
"
2100 PRINT  PAPER PA; INK TI; BRIGHT 1;AT 15,28;"5FF5
"
2110 PRINT  PAPER PA; INK TI; BRIGHT 1;AT 16,28;"5FF5
"
2120 PRINT  PAPER PA; INK TI; BRIGHT 1;AT 17,28;"1332
"
2130 RETURN
2140 REM  PRESENTACION BARAJADO
2150 LET W$="4337"
2160 LET Q$="5FF5"
2170 LET E$="1332"
2180 PRINT  PAPER PA; INK TI;AT FILA,COLUM;W$
2190 PRINT  PAPER PA; INK TI;AT FILA+1,COLUM;Q$
2200 PRINT  PAPER PA; INK TI;AT FILA+2,COLUM;Q$
2210 PRINT  PAPER PA; INK TI;AT FILA+3,COLUM;Q$
2220 PRINT  PAPER PA; INK TI;AT FILA+4,COLUM;E$
2230 LET COLUM=COLUM+2
2240 IF COLUM=28 THEN  LET FILA=FILA+5: LET COLUM=0
7810 RETURN
```